Nederlandstalige TEX Gebruikersgroep

Aan : Leden Nederlandstalige TEX Gebruikersgroep
Betreft : Verslag 10^e NTG bijeenkomst (MAPS #10)

Locatie : Grafisch Museum & Koninklijke Boom Pers te Meppel
Datum : 19 november 1992
Behandeling : bij de volgende bijeenkomst (10 juni 1993 te De Bilt)

Agenda:

Bijlagen:

De NTG vereniging

Voorzitter:	C.G. van der Laan, Internet: cgl@rug.nl
Secretaris:	G.J.H. van Nes, ECN, Service Unit Informatica, Petten. Internet: vannes@ecn.nl
Penningmeester:	J.L. Braams, PTT Dr Neher Laboratorium, Leidschendam. Internet: j.l.braams@research.ptt.nl
Bestuursleden:	T.A. Jurriens, RUG, Groningen. Internet: jurriens@fwn.rug.nl J.J. Winnink. Internet: jos.winnink@cpb.nl
Postadres:	Nederlandstalige TEX Gebruikersgroep, Postbus 394, 1740 AJ Schagen.
Postgiro:	1306238, t.n.v. Penningmeester NTG, Zoetermeer.
Internet:	ntg@nic.surfnet.nl

De Nederlandstalige TEX Gebruikersgroep (NTG) is een vereniging die tot doel heeft het bevorderen van de kennis en het gebruik van TEX.
De **NTG** tracht dat te bereiken door het uitwisselen van informatie, het organiseren van congressen, symposia en tentoonstellingen m.b.t. TEX en 'TEX-produkten', en door het onderzoeken en vergelijken van TEX met soortgelijke/aanverwante produkten, b.v. SGML.

De **NTG** biedt haar leden ondermeer het volgende:

- Tweemaal per jaar een **NTG**-bijeenkomst.
- Tweemaal per jaar de uitgebreide NTG MAPS (Minutes and APpendiceS).
- Speciale MAPS uitgaven (TEX cursus materiaal en PR set).
- Indien mogelijk eenmaal per jaar open '**NTG**-dagen', waar naast lezingen, ook cursussen (speciaal tarief voor leden) worden gegeven.
- De fileserver TEX-NL waarop algemeen te gebruiken 'TEX-produkten' staan. De meeste van deze TEX-produkten zijn, tegen geringe vergoeding, ook op diskette verkrijgbaar. Daaronder valt ook een volledige MS-DOS versie van TEX, LATEX, en een previewer.
- De discussielijst TEX-NL waarop vragen gesteld worden. Ook worden er via deze listserver ervaringen uitgewisseld.
- Korting op (buitenlandse) TEX congressen en cursussen en op het lidmaatschap van TUG.
- Eenmaal per jaar een ledenlijst met per lid informatie welke software en welke hardware, in relatie met TEX, wordt gebruikt.

Lid worden kan door overmaking aan de penningmeester van het verschuldigde contributie bedrag. Daarnaast dient een informatieformulier te worden ingevuld, welke laatste via het secretariaat te verkrijgen is.

De contributie voor een persoonlijk lidmaatschap bedraagt f 75,–, de contributie voor een instituutslidmaatschap bedraagt f 200,–. Een instituutslidmaatschap geeft het recht om drie personen aan te wijzen die informatie welke aan de leden wordt verstuurd, ontvangen. Van die drie personen dient één persoon te worden aangewezen als rechtsgeldige vertegenwoordiger van het bedrijf/instituut, een ander als vervangend vertegenwoordiger.
Indien meer leden per bedrijf/instituut lid willen worden, geldt als additioneel tarief f 50,– per persoon.
Voor studenten geldt eveneens een tarief van f 50,– (geen stemrecht). Voor afwijkende regelingen dient contact met het bestuur opgenomen te worden.
Een gecombineerd NTG/TUG lidmaatschap bedraagt f 162,- per jaar (i.p.v. f 75,- + $ 60).

De statuten van de Nederlandstalige TEX Gebruikersgroep zijn via het secretariaat of via de fileserver te verkrijgen.

Nederlandstalige TEX Gebruikersgroep

Aanwezig : A. Al-Dhahir (UT); E. Algera (EGD); A.W.W.M. Biegstraaten (TUD); P. Bloemen (TUE); E.P.M. Boets; T. Bloo (Jonge Onderzoekers); J. Braams (PTT); K. Claes (Drukkerij Mennen); L. de Coninck (de Kraal); N. Cox (KUN); M.M.M. Dings (AKZO); F. van Ditmarsch (EGD); W. Dolman (AHA); R. Doornebal (Wolters Kluwer); S. Eggermont (TUE); E. van Eynde (UR); E. Frambach (RUG); L. van Geest (CAWCS); M. van Geest (CAWCS); F. Goddijn; W.J. van de Guchte; M. Haenen (RUL); Y. Haralambous; S. van Harreveld (Actual Business Group); K. van 't Hoff (CWI); S.J. Hogeveen (RUU); A. de Jong (RUU); B.J. de Jong (Wolters Kluwer); H. Jurriens (UT); T.A. Jurriens (RUG); R. Kappert (KUN); W.J. Karman (KUN); C.H.A. Keijer (AHA); G. Kouijzer (NS); M.J. Krugers (Tech. Marketing Consulting); C.G. van der Laan (RUG); A. de Leeuw van Weenen (RUL/VTW); H.A.N. van Maanen; W.J. Maas (Elsevier); H. van der Meer (UvA); F. Mittelbach (Electronic Data Systems); J.C. de Moor (Theologische Univ.); H.P.A. Mulders (KUB); G.J.H. van Nes (ECN); G. Oomen (Wolters Kluwer); P. van Oostrum (RUU); N.A.F.M. Poppelier (Elsevier); J. Renkema (Theologische Univ.); R. Smedinga (RUG); A. Soos (UT); H. Trompert (AHA); P. Tutelaers (TUE); E. Ulijn (TUD); P. Vanoverbeke; M. van Veen; E.J. Vens (RUG/ICCE); J.E. van Weerden (RUG); F. van de Wiel (CWI); J.J. Winnink; R.V. van Zanten (SARA);

Notulisten : Gerard van Nes & Jos Winnink

1 Opening

In de ochtend was de NTG te gast bij het Grafisch museum te Meppel. Onder zeer veel belangstelling werd (vanwege de zeer grote opkomst) in een aantal groepen, de praktische aspecten van ondermeer de typografie op een zeer levendige wijze uiteengezet.

Rond 14:00 uur heet voorzitter Kees van der Laan een ieder van harte welkom en opent de bijeenkomst in een stampvolle zaal bij Koninklijke Boom Pers. Een korte uiteenzetting werd vervolgens gehouden door de gastheer M.C. Boom, directeur van het gelijknamige uitgeversbedrijf.

2 Verslag bijeenkomst 4 juni 1992

Bij dit verslag zijn geen opmerkingen en aldus wordt het goedgekeurd.

3 Ingekomen stukken en Mededelingen

De volgende mededelingen worden gedaan:

- Berichten van verhindering zijn ontvangen van de leden R.M. Berns, J.J. van Gorp, J.F. Krips, A.P. de Ruiter, A. Schuitman en W. Smit.
- Ontvangen zijn diverse tijdschriften van ondermeer de verschillende TEX zustergebruikersgroepen, TTN en SGML.
- Via e-mail (17-nov-92) heeft Victor Eijkhout het volgende medegedeeld:
  Nu Lollipop op drie verschillende plaatsen in de MAPS genoemd blijkt te worden, lijkt het me een goed idee om een aankondiging te doen.
  Lollipop is officieel op deze wereld losgelaten. Het valt op een paar adressen te ftp'en, en het is via email te krijgen.
  Eerst even over hoe goed Lollipop is: het is in staat om alle voorbeelden in de handleiding te zetten. Kijk maar op de leestafel als je wil zien wat dat inhoudt. Verder heb ik Lollipop gebruikt om mijn dissertatie en mijn boek te zetten.
  Ook is mij door een gebruiker gemeld dat het schrijven van een nieuwe stijl letterlijk een uur heeft gekost. Dat is mijn voornaamste claim betreffende Lollipop, dat het ontwikkelen van stijlen erg makkelijk is.
  Nu even over wat er nog mist: het belangrijkste is dat voetnoten en 'inserts' nog niet ondersteund worden. Dat moet verholpen zijn met de versie die ik in Januari zal openbaar maken. Verder ben ik van plan om bepaalde delen uit andere macro pakketten te gebruiken: van Karl Berry zal ik het interface naar BIBTEX gaan gebruiken, en ik ben van plan met de LATEX3 mensen te praten over gebruik van hun 'verbatim' en 'tabular' omgevingen. Ik weet nog niet of ik het NFSS zal gaan gebruiken.
  Ik hoop dat Lollipop ook in Nederland wat tevreden gebruikers zal vinden.
  En het doet me genoegen te zien hoe actief NTG is. Keep up the good works, en groetjes vanuit Knoxville, Tennessee!
  Aldus de e-mail van Victor Eijkhout.

Editor van deze MAPS is: G.J.H. van Nes.

==== Meppeler Courant ====

Meppeler Courant, Vrijdag 20 november 1992

M.b.t. de begroting 1993 waren er geen opmerkingen.

4 NTG presentaties: 'The future of TEX/LATEX'

Het middagprogramma bestond uit de volgende voordrachten:

- *'Virtual fonts, more fun not for Grand Wizards only'*, door Yannis Haralambous.
  Het principe van *Virtual fonts* is simpel. Het gaat om het combineren van (verschillende) reële fonts. Enkele toepassingen zijn
  - het combineren van characters,
  - het maken van ligaturen,
  - het maken van afbreekpatronen voor karakters uit verschillende fonts,
  - het combineren van meerdere fonts tot 1 font.

  Ook is het mogelijk om een compleet POSTSCRIPT programma op te nemen onder 1 teken (karakter) uit een *Virtual Font*.
  Een groot probleem is de *kerning* van tekens. Om een en ander te automatiseren heeft Eberhard Mattes (ja die van emTEX) QDTEXPL (Quick and Dirty TEX to PL) geschreven en is de source beschikbaar. Een probleem is dat het plaatsen van accenten gebeurt op de manier dat TEX denkt dat het correct is. Dit is niet altijd wenselijk en vaak moeten er met de hand correcties worden aangebracht.
  Yannis Haralambous heeft een programma geschreven, MakeKern, om de kerning te automatiseren. POSTSCRIPT gebruikt in feite *virtuele fonts* wanneer het gaat om samengestelde tekens.
  Waarom wordt MakeKern in Frankrijk gebruikt?
  - Yannis Haralambous heeft het programma geschreven.
  - Het programma bestond reeds voordat TEX 3.0 bestond. Nu TEX 3.0 bestaat zal het gebruik vermoedelijk wel verder afnemen.

  Op de vraag of er in een POSTSCRIPT-font kerning informatie zit, is het antwoord dat dat in elk geval wel zo zou moeten zijn.

- *'The pursuit of quality — thoughts and comments about future typesetting systems'*, door Frank Mittelbach.
  Knuth heeft besloten om na TEX3, TEX te bevriezen. Het is stabiel en nagenoeg vrij van fouten. Desondanks blijven er wensen over. Hoewel TEX op dit moment *state off the art* te noemen is, dreigt het langzaam maar zeker te worden ingehaald door andere programma's.
  Om te komen tot een *New Typesetting System* heeft **DANTE** een discussielijst opgezet. Het is niet duidelijk of TEX uitgebreid zal worden of dat er een compleet ander systeem zal komen. Te bedenken is wel dat de interne consistentie van TEX het gevolg is van de ontwikkeling door één persoon.
  We zouden met de toestemming van Knuth kunnen komen tot ETEX. Dit zou dan kunnen staan voor:
  - Enhanced TEX,
  - Extended TEX,
  - Extraodinary TEX,
  - Execution of TEX.

  Als voorbeeld kan gebruikt worden TeXeT. Er is op dit moment een wijziging voor TEX beschikbaar waarmee TeXeT ingebouwd kan worden in standaard TEX. Het resultaat is dat er een standaard DVI-file wordt afgeleverd. Het is evenwel een uitbreiding op het niveau van de source (WEB-file).
  Er moet geprobeerd worden om iedereen op één lijn te houden anders is het over enkele jaren afgelopen met TEX.
  Wat zijn de kwaliteiten van TEX?
  - Portabiliteit.
  - Beschikbaarheid.
  - Stabiliteit.
  - Het algorithme waarmee paragrafen worden afgebroken op basis van optimalisatie.
  - Krachtige mathematische typesetting.
  - Krachtig tabelopmaak mechanisme.

  Problemen:
  1. Het is onmogelijkheid om regels in een paragraaf na te bewerken. Denk bijvoorbeeld aan het toevoegen van regelnummers.
  2. Het is niet mogelijk om de (vorm) van een paragraaf te bepalen in relatie tot zijn positie op de pagina, dit omdat TEX de paragraaf als één geheel beschouwt.
  3. Er is geen mechanisme om zogenaamde *rivieren van wit* in een tekst te voorkomen. Idem voor dezelfde woorden die in een paragraaf toevallig precies onder elkaar staan.
  4. Er is geen ondersteuning voor zogenaamde *hanging punctuation*.

  Mogelijke oplossingen[1]:
  1. (*) Ontwikkel een soort **everyline**.
  2. (**) Ontwikkel een nieuwe methode voor het pagineren.
  3. (***) Hiervoor is nog geen oplossing.
  4. (*) Voorzien in tabellen waarin de hoeveelheid *overhang* zowel naar links als naar rechts wordt gespecificeerd. Op dit moment zijn er enkele systemen op de markt die gebruik maken van polygonen in plaats van rechthoeken om de dimensies van tekens te beschrijven.
     (**) Breid het **tfm** formaat uit zodat deze informatie wordt opgenomen.
     Het is dus aantrekkelijk om het model waarmee een teken wordt beschreven te wijzigen, maar het is niet triviaal en kost derhalve veel tijd.

  Het algorithme dat zorgt voor de paginering en nu een locale optimalisatie strategie volgt, zou moeten

[1] Moeilijkheidsgraad van eenvoudig (*) tot moeilijk (***).

worden uitgebreid met globale strategieën.
Afbreekalgorithme: het zou mogelijk moeten zijn om (ongebruikte) gedeelten van een paragraaf te verplaatsen naar een volgende pagina.
Het probleem van de items die niet uit de *main vertical list* te verwijderen zijn, zou kunnen worden opgelost door, of het verwijderen van items toe te staan, of een *look ahead* algorithme te implementeren.
In TEX kan geen *grid* worden gespecificeerd. Hierdoor is het moeilijk om de *baseline to baseline spacing* goed te krijgen in een document. Er zou een **topinsert** moeten worden geïmplementeerd.
Output routines bezitten:
1. een ontoereikend markerings mechanisme,
2. geen ondersteuning voor meerkoloms zetsel.

Oplossingen zouden zijn:
1. het toevoegen van een \newmark concept,
2. een *float* mechanisme.

Beide problemen zijn niet op te lossen binnen de huidige versie van TEX.
Oplossing: ontwerp een nieuw pagineringsmodel dat het huidige *box — glue — penalty* model vervangt.
Er zijn drie strategieën die tot een NTS leiden:
1. Behoud TEX en wacht tot andere systemen TEX inhalen en TEX daarmee overbodig maken.
2. Breid TEX zoveel mogelijk uit op een manier die compatibel is met het verleden.
3. Initieer en ondersteun onderzoek met als doel om tot een New Typesetting System te komen.

- *'Current work on LATEX 3.0'*, door Frank Mittelbach.
  Het werk aan LATEX 3.0 is begonnen als idee om de problemen in LATEX 2.09 op te lossen.
  De sterke punten van LATEX 2.09 zijn:
  - scheiding van inhoud en vorm,
  - *markup* taal die vergelijkbaar is met SGML,
  - de auteur is bevrijd van de zorg omtrent het ontwerpen van de *layout*.

  Problemen met LATEX 2.09 zijn:
  - er zitten problemen (fouten) in de implementatie,
  - er zijn (ontwerp)beperkingen in de implementatie,
  - de interne werking is slecht gedocumenteerd zodat het nauwlijks doenlijk is om een nieuwe stijl te ontwerpen.

  Noodzakelijke uitbreidingen:
  - de syntaxis van de commando's,
  - de *interface* met de specificaties van de layout,
  - de robuustheid moet worden verbeterd,
  - de uitbreidbaarheid moet worden vergroot,
  - de specificatie van de layout van materiaal in tabellen behoeft verbetering,
  - de specificatie en het opnemen van grafische materiaal,
  - de plaatsing van *floating* elementen.

  Wat is er al tot stand gekomen?
  - het nieuwe font selectie schema,
  - de DOC en DOCSTRIP gereedschappen,
  - de AMS-LATEX style en -optie,
  - het prototype van de nieuwe kern is in kleine kring verspreid,
  - er zijn nieuwe implementaties van de *array* en *tabular* omgeving.

  Op dit moment wordt er gewerkt aan een validatiesysteem voor LATEX 2.09 dat hopelijk ook voor LATEX 3.0 bruikbaar zal zijn.
  Wanneer is LATEX 3.0 beschikbaar? Een redelijke schatting lijkt dat inclusief documentatie het over 2 jaar beschikbaar zou moeten kunnen zijn.

- *'Recent Developments in ScholarTEX'*, door Yannis Haralambous.
  Het doel is om (La)TEX te gebruiken voor andere schriften dan het latijnse. Zo zou het bruikbaar moeten zijn voor schriften als het: Koreaans, Cambodjaans, Slovaaks, Mongools, Arabisch en Hiëroglyfen-schrift.
  Problemen die zich hierbij voordoen zijn ondermeer:
  - het grote aantal ligaturen,
  - het feit dat ligaturen niet slechts horizontaal maar ook verticaal geordend kunnen zijn. Dit zijn zogenaamde 2 dimensionale ligaturen en deze komen ondermeer in het Arabisch voor,
  - het ontbreken van *fonts*.

De inhoud van bovengenoemde lezingen is opgenomen in MAPS 92.2 en MAPS 93.1

5 Rondvraag en Sluiting

E.J. Vens deelt mede dat in vervolg op de TEX-NL discussie van afgelopen october betreffende 'Software patenten en League of Programming Freedom', nu een Nederlandse afdeling is opgericht.
De voorzitter vraagt naar vrijwilligers. Dit om het aanwezige werk zoveel als mogelijk te kunnen spreiden. Verwezen wordt ook naar de oproep welke bij de uitnodiging van deze vergadering was bijgesloten.

Om ruim 18:00 uur wordt de vergadering door de voorzitter gesloten. De gastheer wordt bedankt voor de genoten gastvrijheid. De aanwezigen en in het bijzonder de buitenlandse gasten worden bedankt voor hun bijdragen.

Een deel van de aanwezigen gaat het, inmiddels tot een traditie uitgegroeide, gezamenlijke diner genieten.

De volgende vergadering is op:

donderdag 10 juni 1993

te De Bilt. Het thema: 'Van font tot boek'.

1 TEX kalender 1993

26–29	jul	TUG '93	Aston, England
18	nov	NTG (12^{e})	Océ, Den Bosch

2 Glossary

Gebruikersgroepen

TUG	:	TEX Users Group
LUG	:	Local Users Group
CSTUG	:	LUG Tsjecho Slowakije
CyrTUG	:	LUG USSR (het Cyrillisch taalgebied)
DANTE	:	LUG Duitsland (het Duits taalgebied)
GUTenberg	:	LUG Frankrijk (het Frans taalgebied)
HunTUG	:	LUG Hongarije
ITALIC	:	LUG Ierland
JTUG	:	LUG Japan
Nordic	:	LUG Scandinavië, Denemarken, en IJsland
NTG	:	LUG Nederland en Belgie
SibTUG	:	LUG Siberië
UKTUG	:	LUG Engeland
YUNUS	:	LUG Turkije (feitelijk een discussielijst)
GUST	:	LUG Polen

Bulletins/journals

Baskerville	:	UKTUG
Cahiers GUTenberg	:	GUTenberg
TeX Bulletin	:	CSTUG
TeXnische Komödie	:	DANTE
TeXline	:	Malcolm Clark; UK
GUST bulletin	:	GUST
TTN	:	TEX and TUG News; TUG
TUGboat	:	TUG
MAPS	:	Minutes and APpendiceS; NTG

Diversen

AMS	:	American Mathematical Society
BoD	:	Board of Directors
SGML	:	Standard Generalized Markup Language
ltxiii	:	LATEX 3.0

3 NTG's TEX Bulletin Board Systeem

Nieuw voor de Nederlandstalige TEX Gebruikersgroep: een TEX Bulletin Board speciaal voor diegenen die niet op het Internet zijn aangesloten. De naam: **FGBBS**.
Op FGBBS is sinds kort een zo volledig en actueel mogelijke TEX, emTEX, LATEX, TEX-NL en MusicTEX collectie beschikbaar voor alle bezitters van een modem. Het BBS is kosteloos toegankelijk voor iedereen en er zijn geen beperkingen aan de hoeveelheid bestanden die kunnen worden opgevraagd. Het systeem is aangesloten op een High Speed modem, vergeleken met de transmissiesnelheid die een directe Internet link biedt misschien niet geweldig, maar veel beter kan het niet over de gewone huis- tuin- en keukenPTTlijn. De beheerders zijn Frans Goddijn en Henk de Haan.
FGBBS is te bellen op 085-217041.
Zie ook *Bijlage H*.

4 NTG's winkel

Via de NTG is beschikbaar:

- **Syllabus Advanced TEX course:**
  Insights and Hindsights, David Salomon (*revised*; ruim 500 pagina's).
  MAPS'92 speciale uitgave.
  Kosten f 50,– voor leden en f 60,– voor niet-leden (extra verzendkosten: f 10,–).
- **PR set MAPS'93 speciale uitgave:**
  Ruim 25 pagina's; 1 exemplaar gratis voor leden; extra exemplaren: f 2.50; niet-leden: f 5,– (extra verzendkosten: f 5,–).

Bestellingen kunnen gedaan worden door overmaking van het verschuldigd bedrag (plus verzendkosten) op de postgiro van NTG (1306238) t.n.v. de penningmeester Johannes Braams te Zoetermeer, met vermelding van hetgeen gewenst is.

Inhoud NTG's PR set:

- Preface
- Contents
- NTG's shop
- Wat is TeX?
- TEX on MS-DOS PC's: emTEX and 4TEX?
- An Introduction to TEX for New Users
- Mathematics & TEX
- Tables & TEX
- TEX and Metafont; Horak's examples
- Recreational
  Music; Schaaktest; typesetting Bridge via TEX
- Subscription form

5 MAPS 93.2

Sluitingsdatum voor het inleveren van artikelen, bijlagen, en/of mededelingen voor de volgende MAPS uitgaven is:
1 oktober '93 (MAPS 93.2) en 1 april '94 (MAPS 94.1).

Nadere richtlijnen voor auteurs zijn op te vragen bij de redactie. Zie ook *Bijlage B*.

Niet Internet-gebruikers kunnen hun bijdrage via modem/PTT lijn direct naar de redactie sturen. Gaarne hiervoor eerst kontakt opnemen met Gerard van Nes.

In Memoriam

Huub Mulders was lid van het oprichtingsbestuur van NTG.
Als stille, gestage werker heeft hij menigeen met (La)TEX op weg geholpen.
Huub, hartelijk dank voor wat je voor, en met, ons gedaan hebt.
Je ruste in vrede.

Van uw MAPS Editor
'5 jaar MAPS!'

Gerard van Nes

Mei 1993

Het heeft weer enige moeite en veel vrije tijd gekost, doch: er is wéér een nieuwe MAPS gereed gekomen! De *tiende* alweer. En een hele dikke. Voor de beginner, de gemiddelde gebruiker én de goeroe. Voor de Internetter én de niet-Internetter (jaja er is een NTG-TEX Bulletin Boad Systeem). Voor de TEX- en de LATEX aanhangers. Voor de PostScript en de Fonts fanaten. Voor de PC tot de BIGsystem gebruikers. En voor de toekomstkijkers. Kortom een MAPS met een zeer breed scala aan onderwerpen.
In zijn totaliteit een maximum aan bijdragen en een record (het is nu eenmaal een 'jubileum MAPS') aan pagina's. Naast veel originele bijdragen, ook wederom enkele herpublicaties van onderwerpen die zeker een brede verspreiding verdienen.
Opvallend deze keer vele 'nieuwe auteurs'. Hopelijk schrijven ze de volgende MAPS uitgaven ook weer vol!

In deze MAPS zeer veel interessant nieuws. Om te beginnen een nieuw, en direkt al volwassen Bulletin Board (wordt momenteel al veel gebruik van gemaakt). Verder een uitgebreide TEX cursus (en niet alleen voor Atari ST gebruikers!), een reeds eerste blik in het (zeer uitgebreide) *LATEX Companion* boekwerk welke na de zomer verschijnt (sterk aanbevolen), natuurlijk de proceedings van de NTG najaarsbijeenkomst en enkele bijdragen van de afgelopen EuroTEX conferentie en veel PC nieuws. Daarnaast vele andere zeer lezenswaardige bijdragen. Kortom een MAPS om er weer een half jaar 'tegen aan te gaan'!

Nogmaals dank aan de vele auteurs. Enkele hebben zelfs een 'eigen plaats' verworven in de MAPS. Bijdragen werden ontvangen uit binnen- én buitenland i.h.a. via e-mail (van internetters), doch bij deze MAPS nu ook via modem-en-PTT-telefoonlijn (van de niet-internetters). Eén bijdrage werd zelfs van het onlangs geopende Nederlandse TEX Bulletin Board geplukt (zie de bijlage H over FGBBS).

Slechts drie auteurs dienden hun bijdragen in *plain* TEX aan. Verder kon één bijdrage zowel in plain TEX als in LATEX verwerkt worden. Omzetting van plain TEX naar LATEX vond plaats met de hulp van Ronald Kappert en Jos Winnink.
Zoals u ook in de vorige MAPS (bijlage C) heeft kunnen lezen, heeft de MAPS redactie *een zeer duidelijke voorkeur* voor een aanbod van bijdragen in het min of meer *'standaard' LATEX formaat*. Logisch en vanzelfsprekend. Zien we ook bij vele andere tijdschriften en proceedings. Echter, natuurlijk blijft ook 'aanvoer' in zowel (plain) TEX TUGboat en ASCII formaat zeer welkom!

Het geheel aan bijdragen werd vervolgens verwerkt met de nog steeds *in de steigers staande MAPS stijlfile* met uitzondering van de E-TEX bijdragen van Frank Mittelbach welke vanwege het technische karakter geheel ongewijzigd is gelaten (inclusief fonts en bladspiegel). Met wat vallen en opstaan (u heeft de discussie over de leftquotes problemen bij het teletype font Courier via TEX-NL kunnen volgen), ging uiteindelijk het geheel naar een 600dpi (hopelijk ziet u het verschil!) PostScript laserprinter, waarna tenslotte Jos Winnink het vermenigvuldigen en het binden ter hand nam.

We zeiden het al: de MAPS stijl staat nog steeds in de steigers. Een goede robuuste stijlfile ten behoeve van zowel de LATEX als de plain TEX gemeenschap kost de nodige arbeid. Vandaar *een oproep voor vrijwilligers* voor deze taken:

- Wie wil medewerking verlenen tot het samenstellen van een LATEX stijl file voor de MAPS (in eerste instantie uitgaande van de huidige MAPS layout)?
- Wie wil medewerking verlenen tot het omzetten van (ondermeer) de (*plain*) tugboat stijl naar het uiterlijk van de huidige MAPS?

Neem contact op met de redactie of met het NTG bestuur! TEX en ook NTG (inclusief de MAPS) is gebouwd door vrijwilligers waar U als lezer gebruik van maakt. Draag daarom ook een steentje bij!

Vijf jaar MAPS. Dat betekent dat er (inclusief deze) tien MAPS uitgaven zijn verschenen. Beginnend met uitgaven bevattende het verslag van de NTG bijeenkomst met slechts enkele bladzijden bijlagen, al gauw overgaand tot een MAPS met bijna alleen maar bijlagen.

# 1	MAPS 88.2	30pp
# 2	MAPS 89.1	33pp
# 3	MAPS 89.2	68pp
# 4	MAPS 90.1	87pp
# 5	MAPS 90.2	156pp
# 6	MAPS 91.1	128pp
# 7	MAPS 91.2	146pp
# 8	MAPS 92.1	162pp
# 9	MAPS 92.2	176pp
# 10	MAPS 93.1	234pp

Op deze en volgende bladzijden treft u een overzicht aan van de belangrijkste bijdragen uit de eerste tien MAPS uitgaven, verdeeld over een aantal (willekeurige) rubrieken.
Geïnteresseerd in overdrukjes? We zoeken nog naar mogelijkheden (én vrijwilligers!). Ideeën? Laat het dan even weten (U bereikt het gehele NTG bestuur plus MAPS redactie via `ntg@nic.surfnet.nl`*, via de NTG Postbus, of via de telefoon (voor nummers zie de ledenlijst).*

– – * – –

General

Meeting reports

Education

Publishing

Graphics

Fonts

TEX

LATEX

Future of TEX/LATEX

The use of TEX/LATEX

TEX/SGML

PC systems

Games

Products

TeX/LaTeXbooks

Miscellaneous

Van de Voorzitter. . .

1 NTG's lustrum

NTG bestaat 5 jaar! De NTG is opgestart in 1988 door een handjevol enthousiaste gebruikers, met Gerard en ondergetekende als kern. Degenen die hun mond opendeden waren prompt kandidaat voor het oprichtingsbestuur. . .

Het aantal leden is inmiddels ruim 3-maal zoveel als verwacht. Het aantal gebruikers van (La)TEX, en Metafont is misschien wel het 10-voudige. Het aantal gecombineerde NTG-TUG lidmaatschappen is ook boven verwachting.

De bijeenkomsten genieten een royale opkomst, $\approx 20\%$. Met SGML Holland hadden wij een interessante SGML-TEX-bijeenkomst in 1990 gehad, met vanuit de TEX gemeenschap: Barbara Beeton, Malcolm Clark, Amy Hendrickson, Frank Mittelbach, en Richard Southall. Als waarnemer vanuit de UKTUG was er David Osborne. Op latere bijeenkomsten waren te gast: Anatoly Urvantsev, Daniel Taupin, Hanna Kołodziejska, David Salomon, Ralph Youngen, Frank Mittelbach, en Yannis Haralambous. Op de aanstaande bijeenkomst zijn er als speciale gasten: Huub van Krimpen (typograaf), en Frank Blokland (letter ontwerper). In het najaar komt Michel Lavaud.

Naar de MAPS wordt uitgekeken. Diverse goede publikaties van elders worden erin herverspreid; verder biedt het ruimte om ons eigen werk te tonen. De opzet is bijzonder, en eenvoudig. De halfjaarlijkse bijeenkomsten waar wij elkaar ontmoeten en over onze ervaringen vertellen zijn de 'levensader' van de vereniging. Een verslag ervan, een beetje uitgewerkt, vormt onze MAPS (Minutes and APpendixeS). Er zijn twee MAPS specials: Salomon's cursus en onze PR-set.

TEX-NL wordt effectief gebruikt.[1] Een PC-distributie set—een instapversie en een volledige set—is beschikbaar 'off-the-shelf.'

Een aantal cursussen zijn georganiseerd, en verder zijn diverse NTG leden bereid tot het geven van cursussen.

Als hoogtepunten kunnen op het cursus gebied genoemd worden: Amy's en Victor's cursus in 1990, en afgelopen jaar de 'Insights and Hindsights TEX' cursus van David Salomon. De opbrengst van deze cursussen gaf de NTG zijn financiële reserves. Daarnaast was de TUE actief en stelde hun (TEX) cursussen open voor niet-TUE medewerkers. Theo Jurriens gaf een aantal cursussen voor secretariaatsmedewerkers. Dit laatste was aanleiding voor zijn Ladies en LATEX artikelen.

Goede contacten zijn gelegd met het Wiskundig Genootschap en het CWI, en de diverse zustergebruikersgroepen: TUG en de LUGs, in het bijzonder GUST. Diverse instituten in Nederland bieden de NTG gastvrijheid voor het houden van de bijeenkomsten (waarvoor onze dank).

Aan de diverse TUG annual meetings en aan de EuroTEX-s wordt redelijk deelgenomen. Het LATEX 3.0 project wordt op verschillende manieren gesteund. Diverse NTG-ers zijn actief, . . . zoals moge blijken uit de MAPS van de afgelopen jaren. Ik hoop dat zij ons deelgenoot laten blijven van hun activiteiten.

2 Activiteiten

Een activiteit was een volledige set TUGboats in Nederland publiekelijk beschikbaar te hebben. TUG heeft ons hierin geholpen door de ontbrekende TUGboat nummers aan de RUG-WSN bibliotheek te schenken. De bibliothecaresse heeft toegestemd NTG-ers behulpzaam te zijn bij het lenen van oude TUGboat artikelen. Zij die geïnteresseerd zijn in TUGboat artikelen kunnen een kopie van het artikel, c.q. het desbetreffende nummer, opvragen, hetzij direct hetzij via het openbare bibliotheek circuit.[2]

Een andere activiteit is het overleggen met de AMS om hun fonts en materiaal via NTG beschikbaar te stellen.[3]

Op handen is het overleg met SURF, waarbij de PC-set wordt aangeboden ter verspreiding, en de mogelijkheden t.a.v. collectieve aanschaf van MathTime besproken zullen worden.

Voor de rest zijn er de lopende zaken—de halfjaarlijkse bijeenkomsten, de MAPS-en, het bijhouden van het ledenbestand, de financiële zaken, de fileserver, en de discussielijst—en het aan elkaar knopen van de losse eindjes, vooral het stimuleren dat de activiteiten, die door leden opgestart zijn en gedragen worden, worden afgerond.

3 NTG–TUG

Victor Eijkhout is associate editor van TUGboat. Nico Poppelier is gekozen lid van de BoD van TUG, en liaison naar de BoD voor de Knuth Scholarship commissie. Ondergetekende is vanuit zijn functie om historische en

[1] Dank, dank, beantwoorders van de vele vraagjes.

[2] WSN Bibliothecaresse: Mevr. J. Bulthuis; tel. 050-634001.

[3] Men kan het nu al van de file servers plukken en de diverse Guides bij AMS opvragen, c.q. zelf afdrukken.

pragmatische redenen lid van de BoD.

4 TUG

Christina Thiele is voorzitster. Het kantoor is verhuisd naar Santa Barabara, met Pat Monohan als directrice.[4]

Christina streeft goede contacten na met de LUG-s, en heeft de de werkgroepen opgepord om vóór de jaarlijkse bijeenkomst, ditmaal in Aston, Juli 1993, (thema: 'windowing on TEX'), met resultaten te komen. De TUGboat's en TTN's verschijnen regelmatig en zien er zeer goed uit.

DC good, CM bad

Op het inhoudelijke vlak is er hard gewerkt aan de 8-bits fonts. Wij worden dan ook uitgenodigd door Yannis Haralambous (zie TTN 1.4), om over te gaan van de 7-bits CM fonts naar 8-bits DC fonts—in het bijzonder voor publikaties die niet in het Engels gesteld zijn—met een plaats voor onze ij.

5 TEX in Europe

Bernard Gaulle en Malcolm Clark hebben zich teruggetrokken. De interim voorzitter van GUTenberg is Alain Cousquer.[5] Het laatste nummer van TEXline's is no 14. De GUTenberg cahiers lijden aan (vrijwillige) redacteuren. Het laatste nummer bestaat derhalve uit de proceedings van EuroTEX'92.

DANTE heeft het voortouw genomen met hun frequente en decentrale TEX-Stammtisch bijeenkomsten. Werkelijk een simpele, en doeltreffende gedachte om leden regelmatig met elkaar in contact te laten komen binnen een gebruikersgroep van een dergelijke omvang.

In Oost-Europa zijn er diverse gebruikersgroepen opgestart. Irina Makhovaya is de directrice van CyrTUG. Er wordt gefluisterd dat de Russen hun eigen low-budget TUG bijeenkomst gaan organiseren en dat het GUST bulletin uit is.

Metafoundry

Er is een nieuwe—jawel, het houdt niet op—discussielijst geopend speciaal voor Metafont zaken. Inhaken kan bij `listserv@ens.fr` via:

```
subscribe metafont <naam en adres>
setmail metafont mail ack .
```

6 Vorm vs. inhoud

Een soortgelijke titel is ooit door Marvin Minsky gebruikt voor zijn Turing award lecture. Voor het computer-ondersteunde publiceren is het motto ook toepasselijk, misschien nog wel meer.

> De motivatie om TEX te ontwerpen was de slechte kwaliteit van de 'The Art of Computer Programming,' toen de uitgever overging op computer-ondersteund zetten.

Nu wij beschikken over de TEX-Metafont-gereedschappen als basis, naast de alom aanwezige PC's, en de betaalbare inkjet printers (near laser), spelling- en stijl-checkers, treden er—paradoxaal genoeg—de volgende verschijnselen van achteruitgang in de kwaliteit op:

- de typografische traditie wordt genegeerd (wij kunnen het allemaal zelf wel),
- de vele, helaas niet foolproof, automatismen introduceren fouten, die het vroegere normale procédé niet doorstaan zouden hebben,
- de foutieve spelling (wij vertrouwen teveel op de checkers),
- de balans tussen inhoud en de typografische kwaliteit is verstoord (teveel aandacht voor een aantal layout aspecten, gegeven de doelgroep).

Draven wij niet te ver door over een aantal computer-ondersteunde typografische zaken, terwijl met een wat bescheidener gebruik van de middelen het doel even goed bereikt kan worden? Ralph Youngen, bijvoorbeeld in zijn TEX-based production at the AMS (MAPS92.2), gebruikt een heel eenvoudige mark-up: titel (met goede font en magstep), `\head`-s en `\subhead`-s, `\item`, en een `\halign`. Dat is alles.[6]

En hoe zit het met suboptimalisatie? Gaat het niet om de eeuwige invarianten: *productie en consumptie*, in dit geval van informatie? De typografische kwaliteit speelt hierin een belangrijke rol naast de vanzelfsprekende inhoudelijke kwaliteit. Maar binnen dit algemene kader hebben wij ook te maken met de kosteneffectieve:

- (productie),
- opslag, transformatie en transport,
- (consumptie).

Dit relativeert en stelt andere prioriteiten.

7 LATEX 3.0

Al gaande sinds 1989 en nu in een fase waarin er een fonds bestaat, en er een kader[7] is gecreëerd voor vrijwilligers om binnen te werken.

Waar gaat het eigenlijk om?

Om te beginnen, LATEX is de enige algemeen geaccepteerde en wijdverspreide collectie van stijlen, met een niet te moeilijke gebruikersgids.

Maar, . . . LATEX heeft zo zijn problemen *geintroduceerd*, vooral als men het wil gebruiken voor een an-

[4] Adres: TUG P.O. Box 869, Santa Barbara CA 93102 USA. Tel: 805-963-1338. FAX: 805-963-8358. email: tug@math.ams.org.

[5] Email: cousquer@lifl.fr.

[6] Hmmm, met stilzwijgende automatische hyphenation en beschikbare fonts.

[7] Zie TUGboat, 13, 4.

dere taal dan Engels (titel woorden zijn niet geparameteriseerd), en in het algemeen als men een stijl wil aanpassen.

Er zijn verbeteringsvoorstellen gepubliceerd: andere talen (Babel), soberder witruimtes, (partiële) meerkolomsuitvoer, A4, A5, . . . pagina opmaak, tabellen, wiskunde (theorem, array), verbatim (algemener, en inclusie van files), blokcomment, . . . en niet te vergeten de fontselectie via NFSS, of moeten wij tegenwoordig denken aan het gebruik van virtuele fonts?

Binnen het project is ook het spanningsveld tussen een zo'n eenvoudig mogelijke gebruikersinterface, en een zo flexibel mogelijke designersinterface, aan de orde.

Het grote probleem is echter dat LaTeX zweeft, het hoort nergens bij. De stijlen zijn aardig, maar zij representeren geen enkele bestaande publikatie-reeks, noch zijn zij generiek. Er is geen bedrijf (en in mindere mate een gebruikersgroep) dat LaTeX voor productie geadopteerd heeft. De reden hiervoor is dat de stijlen aangepast moeten worden om het reeds bestaande resultaat-in-druk van de publikatie-reeksen te verkrijgen. En hier zit'm de kneep. Dat is te moeilijk, ergo te duur. Zelfs de AMS is er ondanks het inhuren van Mittelbach en Schöpf, en de professionele ondersteuning, slechts ten dele in geslaagd. Anderzijds is er nog veel research nodig en is het kennelijk te moeilijk om onderzoek van ontwikkeling te scheiden, te meer als er in gefaseerde projecten op basis van vrijwilligers gewerkt moet worden.

Als men mij op de man af zou vragen hoe realistisch dit project is, dan zou ik zeggen[8]: 'dat het te ambitieus, en te groot is, en dat er een duidelijke specificatie van wat te doen ontbreekt, om nog maar te zwijgen over een tijdpad.'

Er is geen bedrijf of organisatie, dat het echt nodig heeft en er zich sterk voor maakt. Het adopteert. Dit is wat anders dan het gebruiken zo gauw het beschikbaar komt.

8 Alternatief

Naar mijn smaak is er teveel en te lang in een verkeerde richting gewerkt, met het corrigeren van een verkeerde uitwerking. Wij kunnen beter teruggaan naar af, naar de wortels, naar manmac, of de generieke amsppt.sty, c.q. tugboat.sty.

> De (La)TeX wereld zou er heel anders uitgezien hebben als er een goede gebruikershandleiding van Knuth's voorbeeldstijlen zou zijn geweest, te beginnen met een ietsje aangepaste Manmac.

Van daaruit kan men een bescheiden en flexibele stijl opbouwen, met een eenvoudige maar doeltreffende gebruikersgids, waar benodigde zaken zoals inhoudsopgave, kruisverwijzingen en symbolische verwijzingen, illustraties maken, index preparatie, meerkolomsuitvoer, tabellen opmaken, wiskunde opmaken, gebruiken van speciale fonts, en het opstellen van de parameters voor de pagina opmaak (lopende kop- en voetteksten, grootte van de pagina), . . . behandeld worden. Dit alles binnen de context van een duidelijke doelgroep:

> Uitgever zus-en-zo produceert via TeX zijn xyz-reeks, met de gereedschappen zoals vermeld in de gebruikershandleiding, annex de uitgewerkte voorbeelden en het sjabloon als 'invulformulier' voor de publikatie.

Enfin, . . . eenieder moet het voor zichzelf maar uitmaken.

9 De TeX-werkomgeving

De afgelopen maanden heb ik mij georiënteerd t.a.v. een nieuwe TeX-werkomgeving.[9] Zoals bekend dalen de hardware prijzen en gaan de ontwikkelingen door. TeX draait op elk systeem, en zijn er de diverse driver families, dus dat is het probleem niet.
Mijn wensenlijstje:[10]

- multi-tasking (in de sfeer van lightning TeX: edit window en preview window gekoppeld via een 'pipe')
- interproces communicatie (editen van meerdere documenten tegelijk met uitwisseling van delen, algemene documentstukken combineren, automatisch bijwerken van gekoppelde bestanden).

Het gonst alom van de 486-s. Lavaud heeft een aardige werkomgeving gebaseerd op FRAMEWORK. De Groningers Dol en Frambach et al, hebben 4TeX ontwikkeld, gebaseerd op freeware en shareware. De prijs van een 486 systeem ligt nog rond de *f* 3000,–, en dan is Desqview, OS/2, of een soortgelijk multi-tasking systeem nog nodig, dat nogal wat resources vraagt.

Daarnaast had ik TeX op de Mac al ervaren, en onveranderlijk zie ik, en hoor ik, dat TeX-en op een Apple zo makkelijk gaat.[11] Welnu een Mac Classical II 4/40 (ik heb geen kleur nodig, noch veel ruimte) is nu onder de

[8] Ik weet wel dat velen mij dit niet in dank zullen afnemen, maar het geeft in ieder geval te denken. Ook weet ik dat juist niet-NTG leden die publikaties gewoon TeX-en, het heel vanzelfsprekend vinden. Neem nou de LaTeX-companion in-the-making, hoe interessant ook voor ons hobby-isten, dat is toch veel te dik voor iemand die gewoon zijn publikatie wil TeX-en. Nee wij hebben gebruikersgidsen à la AMS-TeX nodig. Zeer handzame en to-the-point boekjes, met een uitgever die er borg voor staat.

[9] Jawel, mijn hoofdtoepassing is documenten maken via TeX, met in het verschiet het ontwerpen via Metafont.

[10] Alhoewel ik weet van de CD ROM mogelijkheden, en beeld-enhancements op de Apple, en dat o.a. Prime Time Freeware een CD ROM levert voor zo'n $60,– met het hele UK-TeX-archive erop, naast veel, teveel andere zaken, is een dergelijk systeem nog te ver van mijn dagelijks werk af. Hypertextachtige toepassingen appeleren echter wel!

[11] Hetzelfde geldt voor andere grafisch georienteerde sytemen zoals de Atari, als ik Robert Best en Theo Jurriens mag geloven.

ƒ 1900,–, alles incluis! Intern wordt zelfs gebruik gemaakt van PostScript. Het is werkelijk een verademing na mijn 8086-stoom-PC, en een prima leeromgeving.

Zaken die ik nog in orde moet brengen zijn

- 'kabeltje' van 8086 naar de Mac voor bestandsuitwisseling (via ...Kermit? Apple voert commerciëel Maclink voor zo'n ƒ 700,–.)
- bestandsuitwisseling direct met de RUG mail machine (VAX8650, wederom via Kermit, zoals ik nu al doe met de 8086.)
- selectie van een (laser) printer (PostScript, $\geq$ 300dpi), c.q. TEX driver voor de bescheiden en betaalbare stylewriter (360dpi,[12] 1 pagina per 2 minuten, sheetfeeder met 50p.).

Als alles naar wens verloopt dan ligt de aanschaf van een powerbook, tijdens mijn deelname aan TUG'94 in de US, voor de hand. Bij het mij oriënteren had ik veel aan het boek:

Aston-Tate's Quick and
Easy Macintosh Guide for the MS-DOS users.

Hierin worden de functionaliteiten goed naast elkaar gezet. De TEX-werkomgeving komt in de najaarsbijeenkomst uitgebreid aan de orde, nadat het in Aston al worldwide belicht is geworden.

[12] Bloob, onzin natuurlijk de fonts zijn 300dpi.

Jaarverslag NTG

jan–dec 1992

Gerard van Nes

1 Algemeen

In 1992 ging de NTG zijn 5^e jaar van bestaan in. Twee NTG bijeenkomsten vonden plaats met vele goede lezingen, naast een succesvolle en goed bezette 5-daagse Advanced TEX cursus (low budget; met het cursusmateriaal als MAPS-Special). Er verschenen wederom een tweetal MAPS uitgaven (Minutes & APpendiceS). Medewerking werd tevens verleend aan het LATEX3 projekt.
En voor de rest is er het diverse door de leden gedaan waaronder een Public Domain MS-DOS set (instap en volledige versie), proeflezen van de MAPS, en de hulpverlening bij vragen op de TEX-NL discussielijst.

2 Het NTG bestuur

Het NTG bestuur bestond uit de volgende personen:

- C.G. van der Laan, voorzitter,
- G.J.H. van Nes, secretaris,
- J.L. Braams, penningmeester,
- J.J. Winnink, bestuurslid,
- H.P.A. Mulders, bestuurslid (tot juni 1992),
- T.A. Jurriens, bestuurslid (vanaf juni 1992).

Op de NTG bijeenkomst in juni vonden wederom bestuursverkiezingen plaats. H.P.A. Mulders en G.J.H. van Nes traden reglementair af, waarbij laatstgenoemde zich herkiesbaar stelde. De door het bestuur voorgestelde twee kandidaten, T.A. Jurriens en G.J.H. van Nes, werden bij acclamatie gekozen.

3 Het NTG ledenbestand

Het ledenaantal bleef ook in 1992 duidelijk stijgen. Eind 1992 telde de NTG 193 leden waarvan 30 instituutsleden. T.o.v. eind 1991 betekent dit een stijging met 34 leden (waaronder 2 instituutsleden). De contributie bleef in 1992 ongewijzigd.
De duidelijke samenwerking van NTG met TUG, waarbij het lidmaatschap gecombineerd kon worden overgemaakt, bleek te resulteren in 48 NTG/TUG leden eind 1992.
De volledige NTG ledenlijst met aanvullende hardware- en software informatie werd wederom gepubliceerd in de beide verschenen MAPS uitgaven.

4 De NTG bijeenkomsten

Er zijn in 1992 twee NTG bijeenkomsten georganiseerd welke gekenmerkt werden door een levendige discussie en een hoge mate van informatie-uitwisseling:

1. Op 4 juni 1992 bij het Centrum voor Wiskunde en Informatica te Amsterdam.
   Aanwezig waren 32 leden.
2. Op 19 november 1992 bij het Grafische Museum & Koninklijke Boom Pers te Meppel.
   Aanwezig waren 59 leden.

Op deze bijeenkomsten, met als thema respectievelijk: 'TEX and Scientific Publishing' en 'The Future of TEX/LATEX' vonden de volgende lezingen plaats:

- 'From Observation to Publication', door Theo Jurriens;
- 'On Standard Notations in Mathematics', door Nico Poppelier;
- 'AMS-TEX-based Production', door Ralph Youngen;
- 'AMS-TEX-related Services', eveneens door Ralph Youngen;
- 'Math into BLUes', door Kees van der Laan;
- 'Automatic Index Generation', door David Salomon;
- 'Virtual Fonts, more fun not for Grand Wizards only', door Yannis Haralambous;
- 'The Pursuit of Quality — thoughts and comments about future typesetting systems', door Frank Mittelbach;
- 'Current work on LATEX 3.0', door Frank Mittelbach;
- 'Recent Developments in ScholarTEX', door Yannis Haralambous.

Van bovengenoemde twee vergaderingen (en lezingen) verschenen verslagen met bijlagen (zie MAPS 92.1, 92.2 en 93.1).

Een belangrijk deel van de lezingen werden gepresenteerd door gerenommeerde buitenlandse sprekers, die door de NTG speciaal voor deze gelegenheden waren uitgenodigd. De reis van Ralph Youngen was gesubsidieerd door het Wiskundig Genootschap.
De leestafels met TEX-achtige boeken, tijdschriften, brochures en andere aan TEX verwante documenten, trok wederom de nodige belangstelling.

Het Grafisch Museum te Meppel werd, als voorafje aan de najaarsbijeenkomst, met veel belangstelling bezocht.

5 De NTG MAPS

Ook in 1992 verschenen er twee MAPS uitgaven met ongeveer 340 goed gevulde bladzijden (in 1991: ongeveer 275) met ondermeer als inhoud: verslagen van nationale en internationale bijeenkomsten, book- en software reviews, artikelen m.b.t. de NTG lezingen, cursus- en educatiemateriaal, TEX/LATEX & PostScript bijdragen, font artikelen, de laatste informatie over LATEX3, algemene- en technische TEX en LATEX bijdragen.

6 De NTG werkgroepen

De functie van de NTG werkgroepen blijkt toch wat op zijn retour te zijn. Bijdragen komen hoofdzakelijk van individuele leden. De TEX-NL discussielijst zorgt dikwijls voor een vorm van 'overleg'.

7 Bestuursactiviteiten

- De contacten met de Nederlandse Wiskundigen hebben in 1992 geresulteerd in een aantal TEX artikelen in het tijdschrift 'Mededelingen van het Wiskundig Genootschap', in een samenwerking bij de NTG voorjaarsbijeenkomst (subsidiëring reis Ralph Youngen), én in een TEX-PR lezing op het Nederlands Wiskundig congres te Delft.
- Van 15 tot 19 juni vond in Groningen een zeer geslaagde TEX cursus plaats. Docent was de Amerikaan David Salomon. De cursus werd bijgewoond door een veertigtal deelnemers. Een voor NTG speciaal gemaakt cursusboek verscheen als 'MAPS special'.
- Er werden vrijwilligers gevonden voor ondermeer het proeflezen van de bijdragen aan de MAPS, en het beschikbaar maken van de PD MS-DOS set.
- De voorzitter had vanuit zijn functie zitting in de Boards of Directors van TUG. Tevens werden door hem functies vervuld binnen de TUG 'Long-Range Planning committee', de 'Education committee' en de 'Publications committee' (bij de laatste als voorzitter). Bij een reorganisatie van de commissies is hij bij verstek buiten de nieuwe commissies gehouden.
- De NTG wisselde informatie uit met de bekende LUGs (exemplaren van de MAPS werden verstuurd).
- Namens de NTG werden in 1992 de volgende bijeenkomsten bezocht:
  - TUG'92 te Portland, Oregon/USA:
    Van 27 tot 31 juli werd door de penningmeester deelgenomen aan deze jaarlijkse TUG meeting.
  - EuroTeX'92 te Praag:
    Van 14 tot 18 september werd ondermeer door de voorzitter en de penningmeester deelgenomen aan deze Europese TEX gebruikersbijeenkomst. Vooraf werd GUST bezocht, Hanna Kołodziejska en Wlodek Bzyl.

  Van beide bezoeken is verslag gedaan in MAPS 92.2.
- Op 21 januari en 27 augustus vonden telefonische NTG bestuursvergaderingen plaats waarin naast de algemene gang van zaken binnen de NTG, de organisatie van de NTG bijeenkomsten, de TEX cursus van David Salomon, Public Relation activiteiten, en TUG contacten ter sprake kwamen. Daarnaast werd aandacht besteed aan ondermeer bestuursverkiezingen en NTG's toekomst.

8 Diversen

- Van zowel de TEX-NL fileserver als de TEX-NL listserver werd in 1992 wederom veel gebruik gemaakt. Het aantal abonnees van de listserver schommelde tussen de 160 en 170. Gedurende 1992 werden in totaal 1097 (in 1991: 846 mails) verstuurd. Daarnaast werd via de NTG diverse bestanden op de TEX-NL fileserver geplaatst.
- Vele leden maakten via e-mail dan wel via ftp dankbaar gebruik van de RUU fileserver van Piet van Oostrum voor het verkrijgen van diverse soorten TEX materiaal. Daarnaast kwam ook de ftp-server van de Piet Tutelaers (TUE) en die van Erik-Jan Vens (RUG; 4TEX en fontzaken) binnen de belangstelling.
- Indirect werd medewerking verleend aan het LATEX3 project van Frank Mittelbach.
- Door NTG leden werd een belangrijk aantal artikelen gepubliceerd in TUGboat, het tijdschrift van de internationale TEX Users Group:
  - Victor Eijkhout (1992): New books on TEX, TUGboat 13#1, 57-58.
  - Nico Poppelier (1992): New books on LATEX, TUGboat 13#1, 58-59.
  - Victor Eijkhout (1992): The bag of tricks, TUGboat 13#1, 74.
  - Victor Eijkhout (1992): Oral TEX: Erratum, TUGboat 13#1, 75.
  - Victor Eijkhout & Ron Sommeling (1992): Self-replicating macros, TUGboat 13#1, 84.
  - Piet Tutelaers (1992): A font and a style for typesetting chess using LATEX or TEX, TUGboat 13#1, 85-90.
  - Kees van der Laan (1992): Tower of Hanoi, revisited, TUGboat 13#1, 91-94.
  - Nico Poppelier (1992): Book reviews, TUGboat 13#2, 182-185.
  - Victor Eijkhout (1992): Book review, TUGboat 13#2, 185-188.
  - Victor Eijkhout (1992): Names of control sequences, TUGboat 13#2, 189-190.
  - Victor Eijkhout (1992): The bag of tricks, TUGboat 13#2, 191.
  - Walter van der Laan & Johannes Braams (1992): Writing reports with more than a hundred people, TUGboat 13#3, 309-314.
  - Victor Eijkhout (1992): Just give me a lollipop (it makes my heart go giddy-up), TUGboat 13#3, 341-346.

  - Victor Eijkhout (1992): Book review, TUGboat 13#4, 486-188.
  - Victor Eijkhout (1992): The bag of tricks, TUGboat 13#4, 494-495.
  - Victor Eijkhout (1992): One error less, TUGboat 13#4, 496-497.
- Victor Eijkhout was editor van de macro column van de TUGboat.
- Nico Poppelier was lid van het 'Knuth scholarship committee'.
- Door Theo Jurriens, Kees van der Laan en Erik-Jan Vens zijn lezingen gegeven op de EuroTEX'92 conferentie te Praag.
  Door Johannes Braams en Walter van der Laan werd op de TUG93 conferentie te Portland een paper gepresenteerd.
  Kees van der Laan hield een TEX-PR verhaal op het Nederlands wiskundig congres in Delft.

Financieel verslag 1992
Nederlandstalige TEX Gebruikersgroep

Johannes Braams

April 1993

1 Inleiding

Voor U ligt het financieel jaarverslag van de NTG vereniging over het jaar 1992. De financiële ontwikkeling van de vereniging wordt aan de hand van de winst en verliesrekening en de balans besproken.

2 De winst en verliesrekening

In tabel 1 is de winst- en verliesrekening over 1992 weergegeven. De diverse posten worden in de volgende paragrafen kort toegelicht.

Dankzij het feit dat in 1992 de cursus „Insights and Hindsights" is georganiseerd wordt het jaar afgesloten met een positief saldo.

Bedragen in guldens	**Debet**	**Credit**
Contributie		13.477,50
Sponsoring		
Rente		1.163,59
Evenementen		1.227,38
Administratie	658,15	
KvK en notaris	61,00	
Bestuurskosten	335,00	
Computer faciliteiten		
Mededelingen a/d leden	9.510,99	
Reisbijdragen	3.609,84	
Representatie	136,00	
Diversen	999,02	
Saldo	1.558,47	
	15.868,47	15.868,47

Tabel 1: *De winst en verliesrekening over 1992*

2.1 Inkomsten

- **Contributies**
  De groei van het aantal leden is in 1992 doorgegegaan, waardoor de contributie-inkomsten dit jaar ongeveer f 2700,– hoger zijn dan begroot. Een belangrijke oorzaak hiervan is de in 1992 geïntroduceerde mogelijkheid om met korting lid te worden van zowel de TEX User Group als van de NTG.
- **Rente**
  De inkomsten uit rente zijn hoger dan begroot. De inschatting van de hoeveelheid rente-inkomsten van de Leeuwrekening was te laag.
- **Evenementen**
  In 1992 is door de NTG een cursus georganiseerd. Dit was mogelijk gemaakt daar de in de afgelopen jaren opgebouwde financiële reserve het bestuur in staat stelde garant te staan voor de kosten. Achteraf bleek de cursus een dermate succes dat van een positief saldo sprake is. Voor het cursusboek „Insights & Hindsights" was veel belangstelling, ook van niet-cursisten.

2.2 Uitgaven

- **Administratie**
  Deze post bevat uitgaven voor het voeren van de administratie zoals portokosten, enveloppen, ordners. Ook de kosten van het verzenden van informatiemateriaal zijn in deze post opgenomen.
- **KvK en notaris**
  Deze post omvat de kosten van de inschrijving bij de kamer van koophandel in Groningen.
- **Bestuurskosten**
  Deze post bestaat uit de kosten van telefonische bestuursvergaderingen. De kosten blijken lager te zijn dan begroot was doordat minder telefonisch vergaderd werd dan verwacht.
- **Mededelingen aan de leden**
  Deze post bevat de uitgaven gedaan voor het maken, reproduceren en verzenden van het 'verslag met bijlage'. Door het ruim hogere ledental zijn de kosten hiervan belangrijk hoger dan verwacht.
- **Reisbijdragen**
  In 1992 zijn bijdragen gegeven voor het bijwonen van de conferentie in Portland door Johannes Braams en het bijwonen van de EuroTEX conferentie in Praag door Kees van der Laan.
- **Representatie**
  In deze post zijn de kosten van de gemaakte informatiemap en het bijwonen door de voorzitter van het wiskundig congres opgenomen.
- **Diversen**
  Het bestuur heeft in 1992 besloten om f 1000,– beschikbaar te stellen voor het LATEX3 project. Dit bedrag is overgeboekt naar de girorekening die voor

TUG wordt beheerd. Het bedrag dat in de tabel staat is iets lager omdat de rekening van het CWI voor de bij de bijeenkomst verkochte boeken iets lager was dan de opbrengst van de verkoop.

- **Saldo**
  Deze post is geen uitgave, maar het bedrag dat de vereniging in 1992 heeft overgehouden.

3 De balans

In tabel 2 is de balans per 1 januari 1992 weergegeven. Daarin is te zien dat een groot aantal leden de contributie voor 1992 al in 1991 hadden voldaan. De reden hiervoor was dat de voor 1992 voor het eerst met acceptgiro formulieren is gewerkt die al in december zijn verstuurd. Het is gebleken dat het betalen van de contributie nu soepeler verloopt dan voorheen.

Bedragen in guldens	**Aktiva**	**Passiva**
Giro	19.703,23	
Kas	1,10	
Contributies		3.059,00
Cursusgeld		200,00
Reserveringen		446,25
Kapitaal		15.999,08
	19.704,33	19.704,33

Tabel 2: *De balans per 1 januari 1992*

Bedragen in guldens	**Aktiva**	**Passiva**
Giro	23.256,16	
Kas		
Contributies	222,00	492,50
Debiteuren	90,00	
Crediteuren		6.518,11
Kapitaal		16.557,55
	23.568,16	23.568,16

Tabel 3: *De balans per 31 december 1992*

In tabel 3 is de balans per 31 december 1992 weergegeven. Uit de balans blijkt dat de vereniging een financiëel gezond jaar achter de rug heeft. Ook is te zien dat een aantal leden nog geen contributie heeft betaald. Degene(n) die op de ledenvergadering nog steeds niet aan hun verplichtingen over 1992 hebben voldaan zullen voor royement worden voorgedragen.

Het kapitaal is in 1992 toegenomen met f 558,47; dat is iets meer dan begroot.

De post 'Debiteuren' omvat bestelde en geleverde, maar nog niet betaalde exemplaren van het cursusboek. De post 'Crediteuren' omvat op 1 januari nog niet betaalde rekeningen en declaraties. Op het moment van het maken van dit verslag zijn deze allen voldaan.

4 Samenwerking met TUG

Eind 1991 is een overeenkomst met TUG gesloten die NTG leden 10% korting geeft op het lidmaatschap van beide verenigingen. Daar is door ongeveer de helft van de persoonlijke leden van de NTG gebruik van gemaakt. Vanwege het succes van de regeling is deze in 1992 gecontinueerd en zelfs uitgebreid voor 1992. De regeling geldt nu ook voor mensen die als instituutsvertegenwoordiger lid zijn van NTG zij ontvangen 10% korting op het TUG lidmaatschap. Daarnaast is het zo dat *alle* TUG lidmaatschapsbetalingen van mensen in Nederland nu via de NTG lopen. Ook studenten kunnen van deze regeling gebruik maken, zij krijgen echter geen extra korting vanwege het scherpe studententarief.

Is de overeenkomst vanwege de financiële voordelen een succes, het is gebleken dat de verzending van TUGboat en ttn naar mensen die op deze wijze in 1992 lid zijn geworden van TUG niet geheel vlekkeloos is verlopen. Een aantal leden heeft niet alle numers of zelfs geen enkel nummer ontvangen. Dit moet achteraf gecorrigeerd worden hetgeen tot extra werk heeft geleid. Hopelijk verloopt een en ander in 1993 beter.

5 Conclusie

De vereniging heeft andermaal een financieel gezond jaar achter de rug. Dank zij de goed bezochte cursus is een batig saldo het resultaat. Het blijkt dat de overeenkomst die is gesloten met TUG om korting te geven op het gecombineerde lidmaatschap van beide verenigingen veel weerklank heeft gevonden bij de leden.

NTG's Listserver TEX-NL

9 mei 1993

TEX-NL is de Nederlandstalige TEX-informatie distributielijst (ook wel discussielijst genoemd). Het adres is:

```
tex-nl@nic.surfnet.nl
```

Men kan zich op deze TeX-NL discussielijst abonneren (TEX-NL mails ontvangen en versturen) door het versturen van de volgende één-regelige e-mail:

```
to      : listserv@nic.surfnet.nl
subject :'any'
SUBSCRIBE TEX-NL your_name
```

Een lijst van deelnemers is te verkrijgen door het versturen van de volgende één-regelige e-mail:

```
to      : listserv@nic.surfnet.nl
subject : 'any'
REVIEW TEX-NL
```

Met als resultaat:

```
*
*  TEX-NL
*
*  Review=       Public
*  Subscription= Open
*  Send=         Public
*  Notify=       Yes
*  Reply-to=     List,Ignore
*  Files=        Yes
*  Validate=     Store only
*  Errors-To=    Owners
*  X-Tags=       Comment
*  Stats=        None,Private
*  Confidential= No
*
*  owner=  Quiet:,N.Cox@UCI.KUN.NL  (Niek Cox)
*  owner=  Quiet:,BRAAMS@HLSDNL5    (Johannes Braams)
*  owner=  EVERS@HUTRUU53           (Evert Jan Evers)
*
VDBERG@ALF.LET.UVA.NL                    Martin H. vdBERG
KROPVELD@AMC.UVA.NL                      Dani"el Kropveld
CI@ANALYSIS.RUG.AC.BE                    Chris Impens
raichle@AZU.INFORMATIK.UNI-STUTTGART.DE  Bernd Raichle
R.Mahieu@BE.RULIMBURG.NL                 : Ronald Mahieu
saskia@BGUMAIL.BGU.AC.IL                 Saskia Beeser
LAAAA18@BLEKUL11                         Erik van Eynde
HJBORTOL@BRLNCC                          Humberto Jose Bortolossi
FDC@CAGE.RUG.AC.BE                       "F. De Clerck"
steiner@CLIO.RZ.UNI-DUESSELDORF.DE       Rene Steiner
graf@CONMUC.DE.CONVEX.COM                Ingbert graf
jjw@CPB.NL                               jos winnink
piet@CS.RUU.NL                           Piet van Oostrum
eijkhout@CS.UTK.EDU                      Victor Eijkhout
BOLDY@CS.UTWENTE.NL                      Mike Boldy
vansoest@CS.UTWENTE.NL                   Dick C. "van Soest"
A.G.Geraets@CTRL.PHYS.TUE.NL             Tonnie Geraets
frankw@CWI.NL                            Frank van de Wiel
Kees.van-t.Hoff@CWI.NL                   Kees van 't Hoff
rvdh@CWI.NL                              Rob van der Horst
ernst@DCMR1.UUCP                         E.R. de Vreede
Leendert.Combee@DELFT.GECO.SLB.COM       leendert combee
combee@DELGEO.NL                         leendert combee
X33@DHDURZ1                              Joachim Lammarsch
nust@DUTENTB.ET.TUDELFT.NL               Jan H Nusteling
abi@DUTIAA.TUDELFT.NL                    Ton Biegstraaten
wim@DUTIOSA.TUDELFT.NL                   Wim Penninx
witajgb@DUTISTA.TUDELFT.NL               Hans Braker
wiorst5@DUTIWS.TUDELFT.NL                Bert van Zomeren
ewout@DUTMPW1.TUDELFT.NL                 EWOUT BIEZEN
mkmfhuy@DUTREX.TUDELFT.NL                Tom Huijgen
wbtrvos@DUTREX.TUDELFT.NL                Ron v. Ostayen
robk@DUTTWTA.TUDELFT.NL                  Rob Kuyper
andries@DUTW6.TUDELFT.NL                 jans andries
gerard@DUTW9.TUDELFT.NL                  Gerard Kuiken
```

jaap@DUTW9.TUDELFT.NL	Jaap van der Zanden
martien@DUTW9.TUDELFT.NL	Martien Hulsen
eikelboom@ECN.NL	Jaap Eikelboom
hogenbirk@ECN.NL	Alfred Hogenbirk
vanderstad@ECN.NL	Rob C. L. van der Stad.
vannes@ECN.NL	Gerard van Nes
verhoef@ECN.NL	Hans Verhoef
DOLLY@ECO.RUG.NL	Wietse Dol
FRAMBACH@ECO.RUG.NL	"Erik Frambach"
KONING@ECO.RUG.NL	Ruud H. Koning
SIEPO@ECO.RUG.NL	N.S. Kroonenberg
phons@EI.ELE.TUE.NL	Phons Bloemen
N.POPPELIER@ELSEVIER.NL	"Nico Poppelier"
alex@ET.KULEUVEN.AC.BE	Alex Schoenmakers
bruno@ET.KULEUVEN.AC.BE	Bruno Tersago
ludo@ET.KULEUVEN.AC.BE	Vangilbergen Ludo
AJKRIJGSMAN@ET.TUDELFT.NL	Ardjan Krijgsman
COMBEE@ET.TUDELFT.NL	leendert combee
rafel@FENK.WAU.NL	Rafel Israels
huygen@FGG.EUR.NL	Paul E.M. Huygen
vdende@FGG.EUR.NL	Jan van der Ende
lieven@FLAND.RUG.AC.BE	L. Van Vooren
ekoster@FREYA.LET.RUG.NL	Elwin Koster
luijten@FYS.RUU.NL	Erik Luijten
BOLDY@F2.NHL.NL	Mike Boldy
INEKE_VAN@GCRC2.WUSTL.EDU	ineke vandermeulen
roosmalen@GLAS06.DECNET.PHILIPS.NL	Martin van Roosmalen
HELLINGS@HASAMC51	JAN HELLINGS
A3530004@HASARA11	Hans Verhey.
A401INEK@HASARA11	ineke weijer
A421FLUE@HASARA11	Pim Coenradie
A471BERN@HASARA11	Bernard R. Bollegraaf
A471HANS@HASARA11	hans van der meer
SOND0016@HASARA11	R Veldhuyzen van Zanten
EMMEN@HASARA5	Ad Emmen
DENHAAN@HDETUD5	Jack den Haan
RCRONH@HEITUE5	Ron Helwig
ELEICZ@HEITUE51	C. van Zwijnsvoorde
ALDHAHIR@HENUT5	Alaaddin Al-Dhahir
joost@HFWORK1.TN.TUDELFT.NL	Joost Dijkstra
CGL@HGRRUG5	CG VAN DER LAAN
DRUNEN@HGRRUG5	Rudi van Drunen
BOSVELD@HGRRUG51	"Gerard Bosveld"
STOOP@HGRRUG51	"Paul Stoop"
KANABY@HHEOUH51	Abdy Jooya
APPRMB@HHEOUH53	Rut Berns
LETTVA@HLERUL2	Andrea de Leeuw van Weenen
BORSBOOM@HLERUL53	G.J.J.M. Borsboom
FORCHK@HLERUL53	Jan Joris Vereijken
VDSCHOOT@HLERUL53	Jan Vanderschoot
U001290@HNYKUN11	Niek Cox
U001310@HNYKUN11	Ronald Kappert
U070040@HNYKUN11	Patrick Wever
U212757@HNYKUN11	Mathieu Koppen
U216002@HNYKUN11	Paul Wackers
U250005@HNYKUN11	Peter-Arno Coppen
U251006@HNYKUN11	Hans Stoks
U253002@HNYKUN11	Constant Cuypers
U439019@HNYKUN11	Ton de Haan
U605005@HNYKUN11	Willem Jan Karman
U605008@HNYKUN11	Rik Fleuren
U641012@HNYKUN11	Rini van Doorn
BISON@HNYKUN52	PIETER BISON
CAOS@HNYKUN52	HENS BORKENT
SYLVIA@HNYMPI51	"Sylvia Aal"
GPTEX@HRZ.UNI-GIESSEN.DBP.DE	TeX-Inst., HRZ Univ. Giessen, F.R.G.
Guenter.Partosch@HRZ.UNI-GIESSEN.DBP.DE	Guenter Partosch, HRZ Univ. Giessen, F.
SURF083@HTIKUB5	Johannes de Moor
S172HMUL@HTIKUB5	Huub Mulders
EVERS@HUTRUU53	Evert Jan Evers / Rijksuniv. Utrecht
KETTENIS@HWALHW5	"Di(r)k Kettenis"
VDVELDEN@HWALHW5	Mark van der Velden
erikjan@ICCE.RUG.NL	Erik-Jan Vens
pfuetz@IGD.FHG.DE	Matthias Pfuetzner, ZGDV Darmstadt
stokhof@ILLC.UVA.NL	Martin Stokhof
ITALIANO@IMEUNIV	Antonio ITALIANO
Uucp@INTERFHQ.HACKTIC.NL	Henk de Haan
E.H.M.Ulijn@IO.TUDELFT.NL	Erik H.M. Ulijn
E.W.G.Zweers@IO.TUDELFT.NL	Erwin Zweers
HAAN@IRIVAX.TUDELFT.NL	Henk de Haan
devries@KNMI.NL	Hans de Vries
Lex.L.Sijtsma@KONBIB.NL	Lex Sijtsma
Lex.Sijtsma@KONBIB.NL	Lex Sijtsma
wessel@KUB.NL	Wessel Kraaij
POL@KVI.NL	"John van Pol"
STAPEL@KVI.NL	"Kees Stapel"
AnneMarie.Mineur@LET.RUU.NL	Anne-Marie Mineur
Jules.vanWeerden@LET.RUU.NL	Jules van Weerden, RUU
andre@MAESTRO.HTSA.AHA.NL	Andre v.d. Vlies
NSEV@MARIN.NL	<E.F.G. van Daalen>
R.H.M.Huijsmans@MARIN.NL	rene huijsmans
s812726@MARS.YZIT.EDU.TW	Liang-Chia Chao
bnb@MATH.AMS.COM	Barbara Beeton
demeijer@MATH.RUU.NL	Andre de Meijer
hvdberg@MATH.UTWENTE.NL	Harmen van den Berg
soos@MATH.UTWENTE.NL	Adwin Soos
twpolder@MATH.UTWENTE.NL	Jan Willem Polderman
BV@MECANO.RUG.AC.BE	Benedict R. Verhegghe
bob@MPI.KUN.NL	Bob Boelhouwer
R.Pauly%KE%RuLimburg.NL@MSN.RULIMBURG.NL	Rob Pauly
hdavids@MSWE.DNET.MS.PHILIPS.NL	Henk Davids

```
tex-nl@NESSY.RUG.AC.BE                  Walter Bossaert
MBR@OCE.NL                              Marius Broeren
colpa@PHYS.UVA.NL                       Jaap Colpa
mwijz@PHYS.UVA.NL                       Maurits Wijzenbeek
SZAJOW@PLWRTU11                         Krzysztof Szajowski
vdkoijk@RADTH.RUU.NL                    John van der Koijk
MOUCHE@RCL.WAU.NL                       Pierre von Mouche
suykerb@REKS.UIA.AC.BE                  Benoit Suykerbuyk
nibr!mark@RELAY.NLUUG.NL                Mark van Veen
J.L.Braams@RESEARCH.PTT.NL              Johannes L. Braams
egdnt01hbtex@RUG.NL                     Henk Brouwer
DINGS@RUGR86.RUG.NL                     Marcel Dings
nijhof@RUGTH4.TH.RUG.NL                 Jeroen Nijhof
rein@RUG4.CS.RUG.NL                     Rein Smedinga
david@RULGM0.LEIDENUNIV.NL              David van Leeuwen
VDGRIEND@RULWINW.LEIDENUNIV.NL          J.A. van de Griend
ELBERS@SARA.NL                          Chris Elbers
A.W.J.HEIJS@SC.AGRO.NL                  Anton Heijs
gert@SG.TN.TUDELFT.NL                   Gert Rietveld
lenssen@SG.TN.TUDELFT.NL                K.-M. Lenssen
v912182@SI.HHS.NL                       Eric Veldhuyzen
marks@STACK.URC.TUE.NL                  Mark J. Sinke
toin@STACK.URC.TUE.NL                   Toin Bloo
POPPE@SWOV.NL                           "Frank Poppe"
roorda@TIR.FEW.EUR.NL                   Berend Roorda
HOEB@TUDW02.TUDELFT.NL                  J.B.W. HOEBEEK
HUISMAN@TUDW03.TUDELFT.NL               H. Huisman
rcpt@URC.TUE.NL                         Piet Tutelaers
KALPAKLI@UWAV1.U.WASHINGTON.EDU         Mehmet Kalpakli   Kalpakli@uwav1.u.wash
Wilfred=Zegwaard%Alg%AAE.WAU@VINES.WAU.NL Wilfred Zegwaard
KNAPPEN@VKPMZD.KPH.UNI-MAINZ.DE         Joerg Knappen Uni-Mainz
KNAPPEN@VKPMZD.PHYSIK.UNI-MAINZ.DE      J"ORG KNAPPEN
SPIT@VM.CI.UV.ES                        Werenfried Spit
francstr@WI.LEIDENUNIV.NL               Franc A.J. Straetemans
dee@WLDELFT.NL                          Dick Dee
Marc.Kool@WLDELFT.NL                    Marc H. Kool
vdvoorn@WLDELFT.NL                      Marjan v\d Vooren
best@ZEUS.RIJNH.NL                      Robert W. Best
*
*
* Total number of "concealed" subscribers:        3
* Total number of users subscribed to the list:  174  (non-"concealed" only)
* Total number of local node users on the list:    0  (non-"concealed" only)
*
```

Opmerkingen:

- Verzocht wordt om de TEX-NL listserver niet te gebruiken voor het versturen van grote bestanden (programma's) indien van het alternatief: **de TEX-NL fileserver** (*zie bijlage G*), gebruik gemaakt kan worden.
- Daar ook enkele buitenlanders meeluisteren, wordt men verzocht de 'subject' van de mail in het Engels op te geven.
- De TEX-NL listserver is bij uitstek geschikt voor ondermeer een ondersteuningsverzoek bij een TEX/LATEX/driver probleem, voor vragen over beschikbaarheid van bepaalde software modulen, voor aankondigingen van bijeenkomsten en/of cursussen, voor het attenderen op bepaalde publicaties, voor het attenderen op bepaalde produkten en voor een mededeling die ook voor een grotere groep interessant is.
- Daar het versturen van e-mail's zowel voor het netwerk als voor de ontvanger een duidelijke belasting is, wordt men verzocht geen 'overbodige' boodschappen te versturen zoals bijvoorbeeld 'bedankt', 'geheel mee eens' en dergelijk.
- Ondanks het feit dat het opnemen van een vraag bij de beantwoording dikwijls verhelderend kan werken, dient de verhouding 'antwoord' tot 'vraag' binnen de redelijke proporties te blijven.
- Indien problemen optreden bij het opzeggen dan wel het wijzigen van het eigen e-mail adres op de listserver, wordt men verzocht contact op te nemen met de beheerder E.J. Evers.

NTG's Fileserver TEX-NL

9 mei 1993

Sinds mei 1989 heeft NTG de TEX-NL fileserver. Voor leden interessante files worden daarbij centraal beschikbaar gesteld.
Men kan files van deze fileserver betrekken door het sturen van een volgende e-mail:

```
to       : listserv@nic.surfnet.nl
subject  : 'any'
GET filename1 filetype1
GET filename2 filetype2
GET filename3 filetype3
etc
```

Waarbij de mogelijke *filenames* en *filetypes* in de hieronder getoonde listing zijn opgenomen.

De lijst van alle aanwezige files is te verkrijgen door het sturen van de volgende e-mail:

```
to       : listserv@nic.surfnet.nl
subject  : 'any'
GET TEX-NL FILELIST
```

Door het versturen van bovenstaande één-regelige boodschap ontvangt men de volgende informatie terug:

```
*  TEX-NL FILELIST for LISTSERV@HEARN.
*  TeX-NL Filelist
*
*  Contains
*      -- general TeX stuff (implementations for micros, graphical
*         shells, printer drivers, etc.)
*      -- specifically Dutch stuff (styles and options, hyphenation
*         patterns)
*      -- Dutch TeX Users Group (NTG) stuff
*
* Please Note:
*         To prevent having to send large files across the networks,
*         the uuencoded zoo archives will be split if they are larger
*         then 1024 records. In these cases the command
*         GET <name> PACKAGE will send all the parts to the requestor.
*
*  ****************************************************************
*
*  This file lists the programs that are stored on LISTSERV and can be
*  retrieved by network users.
*
*  If an entry shows nrecs=0 the file is not available.
*
*  This filelist may be sorted in columns 47 to 63 to get a list of
*  files in the order of their updates. Sorting in descending order
*  shows the most recently updated files at the top.
*
*
*  NTG= U641000@HNYKUN11 Victor Eijkhout
*  NTG= U641001@HNYKUN11 Victor Eijkhout
* ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
*
*  The GET/PUT authorization codes shown with each file entry describe
*  who is authorized to GET or PUT the file:
*
*     ALL = Everybody
*
*:    NTG = 'BRAAMS@HLSDNL5',   /*  Johannes Braams                  */
*:          'BRAAMS@HLSDNL50',  /* Johannes Braams                   */
*:          'BRAAMS@HLSDNL51',  /* Johannes Braams                   */
*:          'BRAAMS@HLSDNL52',  /* Johannes Braams                   */
*:          'EVERS@HUTRUU53'    /* Evert Jan Evers                   */
*
* ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

**************************************************************************
*
*  Dutch hyphenation patterns
*
*  Hyphen1 TeX : shortened Celex-list, all lines with a 5 in them removed,
*                in order to be able to load it when you can't stretch
*                the 'triesize'
*  Hyphen2 TeX : long and powerful (author: Celex, Nijmegen)
*                Note that this requires stretching the 'triesize'
*                of both TeX and IniTeX!
*  Hyphen3 TeX : The (very short) patterns for Dutch created by Peter Vanroose
```

```
*  UShyphen ADD: extra patterns to handle the Tugboat exception log
*                (author: Gerard Kuiken)
*
************************************************************************
*                              rec               last - change
* filename filetype   GET PUT -fm lrecl nrecs   date     time   File description
* -------- --------   --- --- --- ----- ----- -------- -------- ----------------
  HYPHEN1  TEX        ALL NTG V      80  6122 91/05/03 20:00:23
  HYPHEN2  TEX        ALL NTG V      80  7945 91/05/04 10:07:40
  HYPHEN3  TEX        ALL NTG V      80   338 91/05/03 19:56:55
  USHYPHEN ADD        ALL NTG V      73   378 90/05/14 13:20:11

************************************************************************
*
*  Options for Dutch
*
*  A4 STY    : A4-paper width and height
*              by Nico Poppelier and Johannes Braams (historical order)
*              Note that this is not the A4 option of John Pavel.
*  A4 TeX and A4 DOC: Accompanying documentation for A4.STY
*  Dutch old : Redefines captions and does other useful things for
*              all standard document styles. (author: Johannes Braams)
*              This is really an international option.
*              This file has been superseded by the dutch.sty in the
*              BABEL system (See further on)
*  German STY: The style on which 'Dutch' was based. The two are
*              compatible. (author: Hubert Partl) version 2.3e
*  Sober STY : Reduces section headings and white spaces a bit;
*              this is only repair for the standard styles. The official
*              NTG styles (below) can do without. (author: Nico Poppelier)

************************************************************************
*                              rec               last - change
* filename filetype   GET PUT -fm lrecl nrecs   date     time   File description
* -------- --------   --- --- --- ----- ----- -------- -------- ----------------
  A4       STY        ALL NTG V      80   135 91/02/13 10:50:05
  A4       DOC        ALL NTG V      80   511 91/02/13 13:58:32
  A4       TEX        ALL NTG V      80    57 92/08/26 11:09:48
  DUTCH    OLD        ALL NTG V      80   397 90/12/20 18:45:23
  GERMAN   STY        ALL NTG V      80   627 91/11/06 11:17:50
  SOBER    STY        ALL NTG V      77   147 89/06/24 16:06:16

************************************************************************
*
*  The BABEL system
*
*      This is the BABEL system as it is described in TUGboat.
*
*      See the file BABEL README for further instructions
*      The file BABEL BUG lists bugreports and comments since 8/7/91
*      The files BABEL UA?ZOO contain all files.
*   (they can be ordered by sending "GET BABEL PACKAGE" to LISTSERV)
*
************************************************************************
  BABEL    README     ALL NTG V      80   128 92/01/20 18:33:56
  BABEL    $PACKAGE   ALL NTG V      80     6 93/01/29 20:07:30
  BABEL    UAAZOO     ALL NTG V      80  1000 93/04/01 10:52:24
  BABEL    UABZOO     ALL NTG V      80  1000 93/04/01 11:48:36
  BABEL    UACZOO     ALL NTG V      80  1000 93/04/01 11:49:20
  BABEL    UADZOO     ALL NTG V      80  1000 93/04/01 11:50:36
  BABEL    UAEZOO     ALL NTG V      80  1000 93/04/01 11:52:20
  BABEL    UAFZOO     ALL NTG V      80   338 93/04/01 11:52:53
  BABEL    BUG        ALL NTG V      80   165 93/03/31 14:04:55
  BABEL    TEX        ALL NTG V      80    52 93/01/29 20:08:12
  BABEL    DOC        ALL NTG V      80   763 93/01/29 20:08:57
  BABEL    COM        ALL NTG V      80   235 93/01/29 20:09:23
  HYPHEN   DOC        ALL NTG V      80   405 93/03/31 14:07:54
  BABEL    HYPHEN     ALL NTG V      80   117 93/03/31 14:08:19
  BABEL    SWITCH     ALL NTG V      80    85 93/03/31 14:08:34
  BABEL22  SWITCH     ALL NTG V      80    93 93/03/31 14:08:49
  BABEL32  SWITCH     ALL NTG V      80    92 93/03/31 14:09:11
  LANGUAGE DAT        ALL NTG V      80     6 91/05/22 01:52:28
  ESPERANT DOC        ALL NTG V      80   199 93/03/31 14:06:23
  ESPERANT STY        ALL NTG V      80    96 93/04/01 11:44:35
  DUTCH    DOC        ALL NTG V      80   501 93/01/29 20:35:40
  DUTCH    STY        ALL NTG V      80   154 93/01/29 20:36:01
  ENGLISH  DOC        ALL NTG V      80   244 93/01/29 20:36:20
  ENGLISH  STY        ALL NTG V      80   111 93/01/29 20:36:48
  GERMANB  DOC        ALL NTG V      80   692 93/03/31 14:07:19
  GERMANB  STY        ALL NTG V      80   255 93/03/31 14:06:40
  FRANCAIS DOC        ALL NTG V      80   595 93/03/31 14:05:58
  FRANCAIS STY        ALL NTG V      80   248 93/04/01 11:09:49
  ITALIAN  DOC        ALL NTG V      80   211 93/01/29 20:38:39
  ITALIAN  STY        ALL NTG V      80    95 93/01/29 20:38:54
  PORTUGES DOC        ALL NTG V      80   220 93/01/29 20:39:11
  PORTUGES STY        ALL NTG V      80   104 93/01/29 20:39:31
  SPANISH  DOC        ALL NTG V      80   746 93/04/01 11:45:03
  SPANISH  STY        ALL NTG V      80   228 93/04/01 11:45:38
  DANISH   DOC        ALL NTG V      80   194 93/01/29 21:37:01
  DANISH   STY        ALL NTG V      80    94 93/01/29 21:37:19
  NORSK    DOC        ALL NTG V      80   240 93/01/29 21:37:37
  NORSK    STY        ALL NTG V      80   117 93/01/29 21:37:54
  SWEDISH  DOC        ALL NTG V      80   200 93/01/29 21:38:12
  SWEDISH  STY        ALL NTG V      80    95 93/01/29 21:38:33
  FINNISH  DOC        ALL NTG V      80   198 93/01/29 21:38:52
  FINNISH  STY        ALL NTG V      80    96 93/01/29 21:39:10
  MAGYAR   DOC        ALL NTG V      80   216 93/01/29 21:39:24
  MAGYAR   STY        ALL NTG V      80   104 93/01/29 21:39:39
  CROATIAN DOC        ALL NTG V      80   198 93/01/29 21:39:55
  CROATIAN STY        ALL NTG V      80    96 93/01/29 21:40:12
  CZECH    DOC        ALL NTG V      80   220 93/04/01 11:46:22
```

```
  CZECH    STY        ALL NTG V      80   104 93/04/01 11:45:57
  POLISH   DOC        ALL NTG .       .     0 ........ ........
  POLISH   STY        ALL NTG .       .     0 ........ ........
  ROMANIAN DOC        ALL NTG V      80   195 93/01/29 21:59:20
  ROMANIAN STY        ALL NTG V      80    95 93/01/29 21:59:35
  SLOVENE  DOC        ALL NTG V      80   198 93/01/29 21:59:53
  SLOVENE  STY        ALL NTG V      80    95 93/01/29 22:01:53
  RUSSIAN  DOC        ALL NTG V      80   423 93/01/29 22:02:12
  RUSSIAN  STY        ALL NTG V      80   163 93/01/29 22:02:32
  CYRILLIC DOC        ALL NTG V      80   301 93/01/29 22:02:50
  CYRILLIC STY        ALL NTG V      80   135 93/01/29 22:03:08
*************************************************************************
*
*  Dutch styles (author: Victor Eijkhout)
*
*  Completely compatible to 'article' and 'report', but improved layout;
*  these styles have as default language English,
*  for Dutch or German add corresponding style options
*
*  Artikel1 doc : Article-compatible, tight look, documented (somewhat)
*  Artikel1 sty : without documentation
*  Artikel2 doc : Article-compatible, heavily indented; quite something else
*  Artikel2 sty : without documentation
*  Artikel3 doc : Article-compatible; zero parindent, positive parskip;
*                 otherwise similar to Artikel1
*  Artikel3 sty : without documentation
*  Rapport1 doc : Report-compatible; looks like Artikel1
*  Rapport1 sty : without documentation
*  Rapport2 doc : will probably not come into being.
*  Rapport2 sty : without documentation
*  Rapport3 doc : Report-compatible; looks like Artikel3
*  Rapport3 sty : without documentation
*  Boek doc     : Book-compatible; artikel1 layout
*  Boek sty     : without documentation
*
*  Options for the Dutch styles
*
*  Ntg10 doc    : 10point option for all styles
*  Ntg10 sty    : without documentation
*  Ntg11 doc    : 11point option for all styles
*  Ntg11 sty    : without documentation
*  Ntg12 doc    : 12point option for all styles
*  Ntg12 sty    : without documentation
*  Voorwerk doc : Replaces Titlepage.STY for report styles
*  Voorwerk sty : without documentation
*
*  NTGstyle UA? : All in one buy; UUencoded ZOO archive (see below
*                   for ZOO)
*   (they can be ordered by sending "GET NTGSTYLE PACKAGE" to LISTSERV)
*************************************************************************
*                            rec               last - change
* filename filetype   GET PUT -fm lrecl nrecs   date     time   File description
* -------- --------   --- --- --- ----- ----- -------- -------- ----------------
  ARTIKEL1 DOC        ALL NTG V      80  1344 92/08/25 23:15:51
  ARTIKEL1 STY        ALL NTG V      80   712 92/08/25 23:20:17
  ARTIKEL2 DOC        ALL NTG V      80  1304 92/09/04 13:34:41
  ARTIKEL2 STY        ALL NTG V      80   675 92/09/04 13:35:52
  ARTIKEL3 DOC        ALL NTG V      80  1379 92/08/25 23:17:52
  ARTIKEL3 STY        ALL NTG V      80   722 92/08/25 23:18:40
  RAPPORT1 DOC        ALL NTG V      80  1668 92/08/25 23:25:12
  RAPPORT1 STY        ALL NTG V      80   854 92/08/25 23:21:09
  RAPPORT2 DOC        ALL NTG .       .     0 ........ ........
  RAPPORT3 DOC        ALL NTG V      80  1666 92/08/25 23:23:38
  RAPPORT3 STY        ALL NTG V      80   846 92/08/25 23:22:08
  BOEK     DOC        ALL NTG .       .     0 ........ ........
  BOEK     STY        ALL NTG V      80   682 91/02/21 11:24:43
  NTG10    DOC        ALL NTG V      80   193 92/01/15 23:22:17
  NTG10    STY        ALL NTG V      80   166 92/01/15 23:23:14
  NTG11    DOC        ALL NTG V      80   197 92/01/15 23:22:44
  NTG11    STY        ALL NTG V      80   169 92/01/15 23:23:28
  NTG12    DOC        ALL NTG V      80   196 92/01/15 23:22:58
  NTG12    STY        ALL NTG V      80   170 92/01/15 23:23:42
  VOORWERK DOC        ALL NTG .       .     0 ........ ........
  VOORWERK STY        ALL NTG V      80    78 92/02/07 00:07:44
  NTGSTYLE $PACKAGE   ALL NTG V      80     7 92/01/16 01:01:09
  NTGSTYLE UAA        ALL NTG V      80  1000 92/08/26 11:19:20
  NTGSTYLE UAB        ALL NTG V      80  1000 92/08/26 11:19:49
  NTGSTYLE UAC        ALL NTG V      80  1000 92/08/26 11:21:16
  NTGSTYLE UAD        ALL NTG V      80  1000 92/08/26 11:22:47
  NTGSTYLE UAE        ALL NTG V      80  1000 92/08/26 11:23:52
  NTGSTYLE UAF        ALL NTG V      80  1000 92/08/26 11:28:38
  NTGSTYLE UAG        ALL NTG V      80   709 92/08/26 11:29:22

*************************************************************************
*
*  The letter style according to Dutch NEN norms (by Victor Eijkhout)
*
*  BRIEF STY    : The style file
*  BRIEF TeX    : An example letter
*  BRIEFDOC TeX : Explanation of the options of the letter style
*
*************************************************************************
*                            rec               last - change
* filename filetype   GET PUT -fm lrecl nrecs   date     time   File description
* -------- --------   --- --- --- ----- ----- -------- -------- ----------------
  BRIEF    STY        ALL NTG V      80   709 92/03/31 21:07:15
  BRIEF    TEX        ALL NTG V      80   199 92/03/31 20:54:46
  BRIEFDOC TEX        ALL NTG V      80   294 92/03/31 20:55:06

*************************************************************************
*
```

```
*  The latest in TeXnology
*
*  ASCII TeX    : ASCII table (author: Victor Eijkhout)
*
*  BTXMAC.TEX   : BibTeX 0.99c macros for use with plain TeX.
*                 The file specifies that is meant for TeX 3.0 or later
*
*  DUTCH BST    : BibTeX style v 1.11 for Dutch by Werenfried Spit
*  DUTCH2 BST   : BibTeX style v 2.0 for Dutch by Werenfried Spit
*                 this needs the harvard files
*
*  HARVARD README : short description of what's in harvard.zoo and where
*                   it came from.
*  HARVARD UUE    : A uuencoded zoo archive containing 6 files
*
*  CHNGEBARS    : Michael Fine's changebar.sty, modified for use with plain
*                 TeX as well as with LaTeX. Also modified to support DVItoPS
*                 \specials as well as DVI2LN3 \specials
*
*  LOLLIPOP UA? : Victor Eijkhout's lollipop format.
*
**************************************************************************
*                              rec               last - change
* filename filetype   GET PUT -fm lrecl nrecs   date     time   File description
* -------- --------   --- --- --- ----- ----- -------- -------- ----------------
  ASCII    TEX        ALL NTG V      80   190 91/06/26 22:43:05
  BTXMAC   TEX        ALL NTG V      80   624 90/08/15 16:59:21
  DUTCH    BST        ALL NTG V      80  1413 91/11/14 14:19:43
  DUTCH2   BST        ALL NTG V      80  1459 92/03/17 22:53:35
  HARVARD  README     ALL NTG V      80    44 92/03/13 19:08:30
  HARVARD  UUE        ALL NTG V      80   730 92/03/13 19:10:28
  LOLLIPOP $PACKAGE   ALL NTG V      80     5 92/11/02 12:40:12
  LOLLIPOP README     ALL NTG V      80    51 92/10/23 14:04:11
  LOLLIPOP UAA        ALL NTG V      80   900 92/10/23 14:08:07
  LOLLIPOP UAB        ALL NTG V      80   900 92/10/23 14:11:17
  LOLLIPOP UAC        ALL NTG V      80   900 92/10/23 14:12:58
  LOLLIPOP UAD        ALL NTG V      80   355 92/10/23 14:13:42
*-------------------------------------------------------------------------------
*
*  TUGBOAT CMN  : Common commands for Tugboat styles
*  TUGBOAT STY  : Plain TeX style for Tugboat article
*  LTUGBOAT STY : LaTeX style for Tugboat articles
*  TUBGUIDE TEX : A guide for auhtors
*
*  TUGPROC STY  : Plain TeX style file for the proceedings of TuG meetings
*  LTUGPROC STY : LaTeX TeX style file for the proceedings of TuG meetings
*                 Both files need the Tugboat files
*  GUIDEPRO TEX : A guide for authors
*
*-------------------------------------------------------------------------------
  TUG      $PACKAGE   ALL NTG V      80     7 92/03/11 23:40:39
  TUGBOAT  CMN        ALL NTG V      80   894 92/03/11 23:11:19
  TUGBOAT  STY        ALL NTG V      80  2351 92/03/11 23:13:35
  LTUGBOAT STY        ALL NTG V      80   600 92/03/11 23:14:19
  TUBGUIDE TEX        ALL NTG V      80   844 92/03/11 23:38:18
  TUGPROC  STY        ALL NTG V      80   357 92/03/11 23:14:39
  LTUGPROC STY        ALL NTG V      80   193 92/03/11 23:14:50
  GUIDEPRO TEX        ALL NTG V      80   933 92/03/11 23:39:28
*-------------------------------------------------------------------------------
* CHANGEBAR     : Changebar style file for LaTeX 2.09
*                 Changebar V3.0
*                 Supports DVItoLN03, DVIps, DVItoPS, DVIdrv (v1.5+)
*                 Documentation uses doc.sty
*-------------------------------------------------------------------------------
  CHANGBAR $PACKAGE   ALL NTG V      80     4 92/01/15 01:38:21
  CHANGBAR BUG        ALL NTG V      80    79 92/01/16 00:03:27
  CHANGBAR DRV        ALL NTG V      80    76 92/03/13 12:41:47
  CHANGBAR DOC        ALL NTG V      80  1216 92/01/15 01:40:00
  CHANGBAR STY        ALL NTG V      80   333 92/01/15 01:40:22
*                 Changebar V2.? to be removed soon
  CHNGBARS STY        ALL NTG V      80   881 91/06/16 16:02:05
*-------------------------------------------------------------------------------
*  LATEX PACKAGE : Latest versions of all LaTeX materials;
*                  UUencoded ZOO archive
*                  Release april 1992
*-------------------------------------------------------------------------------
  LATEX    $PACKAGE   ALL NTG V      80    11 91/12/02 16:15:01
  LATEX    UAA        ALL NTG V      80  1010 92/04/21 13:48:03
  LATEX    UAB        ALL NTG V      80  1010 92/04/21 13:49:54
  LATEX    UAC        ALL NTG V      80  1010 92/04/21 13:53:02
  LATEX    UAD        ALL NTG V      80  1010 92/04/21 13:58:57
  LATEX    UAE        ALL NTG V      80  1010 92/04/21 14:05:09
  LATEX    UAF        ALL NTG V      80  1010 92/04/21 14:13:05
  LATEX    UAG        ALL NTG V      80  1010 92/04/21 14:15:16
  LATEX    UAH        ALL NTG V      80  1010 92/04/21 14:23:01
  LATEX    UAI        ALL NTG V      80  1010 92/04/21 14:27:33
  LATEX    UAJ        ALL NTG V      80  1010 92/04/21 14:31:03
  LATEX    UAK        ALL NTG V      80  1010 92/04/21 14:35:28
  LATEX    UAL        ALL NTG V      80  1010 92/04/21 14:43:38
  LATEX    UAM        ALL NTG V      80   947 92/04/21 14:55:01
*-------------------------------------------------------------------------------
*  LATEXFON PACKAGE: Latest versions of all LaTeX fonts;
*                    UUencoded ZOO archive
*                    Release februari 1992
*-------------------------------------------------------------------------------
  LATEXFON $PACKAGE   ALL NTG V      80     2 91/12/02 16:18:52
  LATEXFON UAA        ALL NTG V      80   640 92/02/05 12:56:35
  LATEXFON UAB        ALL NTG V      80   386 92/02/05 12:57:35
*-------------------------------------------------------------------------------
*  NFSS PACKAGE : The New Font Selection Scheme as published by
*                 Frank Mittelbach and Rainer Schoepf
```

```
*                Release: (temporarily removed)
*-------------------------------------------------------------------------------
  NFSS     $PACKAGE   ALL NTG V      80     4 91/12/02 16:19:52
*-------------------------------------------------------------------------------
*  MULTICOL     : The multicolumn package written by Frank Mittelbach and
*                 Rainer Schoepf, as published in TUGboat.
*                 The packacge includes DOC.STY. The package consists of three
*                 files, MULTICOL README, MULTICOL UAAZOO, MULTICOL UABZOO.
*                 These files must be distributed together.
*   (they can be ordered by sending "GET MULTICOL PACKAGE" to LISTSERV)
*                 Release: (temporarily removed)
*-------------------------------------------------------------------------------
  MULTICOL $PACKAGE   ALL NTG V      80     3 92/01/06 17:10:25
  MULTICOL README     ALL NTG V      80    82 91/11/06 11:18:05
*-------------------------------------------------------------------------------
*  SUPERTAB     : Theo Jurriens' supertabular.sty for creating tables longer
*                 than one page. Modified by Gabriele Kruljac and Johannes
*                 Braams. Now also supports different tablehead on first page
*                 and different tabletail on last page of the table.
*                 Note: supertabular.doc is *NOT* meant for FMi's doc option
*-------------------------------------------------------------------------------
  SUPERTAB DOC        ALL NTG V      80   506 92/07/10 10:24:13
  SUPERTAB STY        ALL NTG V      80   285 92/07/10 10:24:41
  SUPERTAB TEX        ALL NTG V      80   234 91/04/25 17:37:27
*-------------------------------------------------------------------------------
*  CMRULE       : "The TeX Ruler" by Victor Eykhout using cm-fonts
*  PSRULE       : "The TeX Ruler" by Victor Eykhout using PostScript fonts
*                 Both files contain uuencoded dvi-files
*-------------------------------------------------------------------------------
  CMRULE   UUE        ALL NTG V      80  1008 91/07/15 17:17:01
  PSRULE   UUE        ALL NTG V      80  1019 91/07/15 18:07:46
*-------------------------------------------------------------------------------
*  NASSFLOW UUE : A uuencoded ZOO archive containing style options for
*                 nassi-schneidermann diagrams or flow-diagrams.
*                 Man-pages are included in the archive.
*                 The file NASSFLOW README lists what is available.
*-------------------------------------------------------------------------------
  NASSFLOW README     ALL NTG V      80    46 93/04/15 00:09:43
  NASSFLOW UUE        ALL NTG V      62   607 93/04/15 00:10:43

**********************************************************************
*
*    'Fun with TeX'
*
*    The files below were collected at the NTG meeting in Eindhoven,
*    in november 1991. The meeting was devoted to 'Fun with TeX'
*    Hanna Ko{\l}odziejska presented her GO macros and fonts.
*    Daniel Taupin spoke about MusicTeX.
*    Both packages are provided here.
*
**********************************************************************
*-------------------------------------------------------------------------------
*    The MusicTeX package is stored as multi-part UUencoded ZOO archives
*    It contains the macros, the METAFONT files and the fonts.
*
*    The following files are provided:
*
*    MusicTeX README with a description of what is in the ZOO files and some
*                     comments on how to install everything
*
*    MusicTeX UA?    contains TeX and Metafont sources as well as examples
*    MusicPK  UA?    contains PK files
*    Recueil  UA?    contains a dvi file that can be printed when the fonts
*                    are installed.
*
*-------------------------------------------------------------------------------
*                              rec               last - change
* filename filetype   GET PUT -fm lrecl nrecs   date     time   File description
* -------- --------   --- --- --- ----- ----- -------- -------- ----------------
  MUSICTEX $PACKAGE   ALL NTG V      80    27 92/01/21 16:11:34
  MUSICTEX README     ALL NTG V      80    65 92/01/16 00:02:21
  MUSICTEX UAA        ALL NTG V      80  1000 92/01/15 23:30:30
  MUSICTEX UAB        ALL NTG V      80  1000 92/01/15 23:31:57
  MUSICTEX UAC        ALL NTG V      80  1000 92/01/15 23:33:54
  MUSICTEX UAD        ALL NTG V      80  1000 92/01/15 23:35:15
  MUSICTEX UAE        ALL NTG V      80  1000 92/01/15 23:36:40
  MUSICTEX UAF        ALL NTG V      80  1000 92/01/15 23:38:06
  MUSICTEX UAG        ALL NTG V      80  1000 92/01/15 23:39:33
  MUSICTEX UAH        ALL NTG V      80  1000 92/01/15 23:41:03
  MUSICTEX UAI        ALL NTG V      80    61 92/01/15 23:40:47
  MUSICPK  UAA        ALL NTG V      80  1000 92/01/15 23:43:04
  MUSICPK  UAB        ALL NTG V      80  1000 92/01/15 23:44:11
  MUSICPK  UAC        ALL NTG V      80  1000 92/01/15 23:45:37
  MUSICPK  UAD        ALL NTG V      80  1000 92/01/15 23:47:08
  MUSICPK  UAE        ALL NTG V      80  1000 92/01/15 23:48:39
  MUSICPK  UAF        ALL NTG V      80  1000 92/01/15 23:50:00
  MUSICPK  UAG        ALL NTG V      80   824 92/01/15 23:51:13
  RECUEIL  UAA        ALL NTG V      80  1000 92/01/15 23:53:03
  RECUEIL  UAB        ALL NTG V      80  1000 92/01/15 23:54:12
  RECUEIL  UAC        ALL NTG V      80  1000 92/01/15 23:55:36
  RECUEIL  UAD        ALL NTG V      80  1000 92/01/15 23:57:00
  RECUEIL  UAE        ALL NTG V      80  1000 92/01/15 23:58:39
  RECUEIL  UAF        ALL NTG V      80  1000 92/01/16 00:01:37
  RECUEIL  UAG        ALL NTG V      80  1000 92/01/16 00:03:10
  RECUEIL  UAH        ALL NTG V      80    11 92/01/16 00:02:07
*-------------------------------------------------------------------------------
*    The GO package is stored as a two-part UUencoded ZOO archive
*    It contains the macros, the METAFONT files and the source for
*    the article as it appeared in the MAPS 91.2.
*-------------------------------------------------------------------------------
  GO       $PACKAGE   ALL NTG V      80     3 91/11/28 09:54:19
```

```
  GO       README     ALL NTG V      80    35 91/11/28 09:54:56
  GO       UAA        ALL NTG V      80   768 91/11/28 10:03:36
  GO       UAB        ALL NTG V      80   349 91/11/28 10:04:05

************************************************************************
*
*  Pleasant reading material about TeX and its uses
*
*  NTGstyle TeX : Manual for the Dutch LaTeX styles
*  Layout TeX   : Article about documentstyle development in LaTeX
*                 Intended as supplement to chapter 5 LaTeX book
*  Layout2 TeX  : Goes with previous; in German (Hubert Partl)
*  Refman STY   : Needed for previous two
*  Bridge TeX   : About setting bridge games in LaTeX (Kees van der Laan)
*  Artdoc TeX   : The history of the 'Artikel' styles; almost a
*                 manual for document style development; in Dutch
*  Rapdoc TeX   : The same for the 'Rapport' styles (Victor Eijkhout)
*  Gentle TeX   : A Gentle Introduction to TeX (Michael Doob)
*  Gentrev TeX  : A review of "A Gentle..." by C.G. van der Laan with
*                 comments by Michael Doob
*
*  TTN00    TEX : Het eerste nummer van 'TeX and TUG News'' a prototype issue'
*  TTV1N1   TEX : Het eerste echte nummer van 'TeX and TUG News''
*  TUGNEWS  STY : De bijbehorende style file
*
************************************************************************
*                             rec              last - change
* filename filetype   GET PUT -fm lrecl nrecs   date     time   File description
* -------- --------   --- --- --- ----- -----  -------- -------- ----------------
  NTGSTYLE TEX        ALL NTG V      72   122 89/09/04 12:20:21
  LAYOUT   TEX        ALL NTG V      79  1090 89/06/26 12:12:34
  LAYOUT2  TEX        ALL NTG V      80  1011 89/06/26 12:03:15
  REFMAN   STY        ALL NTG V      79   492 90/03/26 18:12:17
  BRIDGE   TEX        ALL NTG V      75   415 89/06/26 11:54:36
  ARTDOC   TEX        ALL NTG V      71   708 89/09/04 12:21:36
  RAPDOC   TEX        ALL NTG V      75   735 89/09/04 12:22:13
  GENTLE   TEX        ALL NTG V      79  5341 90/01/09 13:56:01
  GENTREV  TEX        ALL NTG V      80   380 91/11/05 09:39:07
  TTN00    TEX        ALL NTG V      80  2047 91/06/04 16:54:07
  TTNV1N1  TEX        ALL NTG V      80  1969 92/02/26 00:46:24
  TUGNEWS  STY        ALL NTG V      80    92 92/02/26 00:43:10

************************************************************************
*
*  Nederlandstalige TeX Gebruikersgroep (Dutch TeX Users Group)
*
*  NTG     INFO : Enige informatie over de Nederlandstalige TeX Gebruikersgroep
*  Notuul1 TeX  : Vergadering 23 juni 1988
*  Notuul2 TeX  : Vergadering 24 november 1988
*  Notuul3 TeX  : Vergadering 11 mei 1989 (3 bestanden: notuul3a,b,c)
*  TeXdag89 TeX : Verslag eerste Nederlandse TeXdagen 29/30 juni 1989
*
*  Statuten TeX : De statuten van de vereniging NTG
*  Statuten sty : bijbehorende document stijl-optie
*
*  Maps bib     : Bibliographic database of articles that have appeared in
*                 NTG's Minutes and APendices.
*
************************************************************************
*                             rec              last - change
* filename filetype   GET PUT -fm lrecl nrecs   date     time   File description
* -------- --------   --- --- --- ----- -----  -------- -------- ----------------
  NTG      INFO       ALL NTG V      77    40 93/04/22 09:45:13
  NOTUUL1  TEX        ALL NTG V      80  1043 89/06/26 11:50:54
  NOTUUL2  TEX        ALL NTG V      80  1457 89/06/26 11:52:06
  NOTUUL3A TEX        ALL NTG V      80   108 89/10/30 10:12:47
  NOTUUL3B TEX        ALL NTG V      80  1941 89/10/30 10:13:27
  NOTUUL3C TEX        ALL NTG V      80  2634 89/10/30 10:14:05
  TEXDAG89 TEX        ALL NTG V      73   274 89/12/08 13:04:52
  STATUTEN TEX        ALL NTG V      80   532 91/03/04 20:55:17
  STATUTEN STY        ALL NTG V      80    94 91/03/04 14:14:21
  MAPS     BIB        ALL NTG V      80  1196 93/01/26 23:06:00

************************************************************************
*
*  TeX for micros
*
*  STZOO UUE    : UUencoded ARC archive with ZOO for the Atari ST
*  MSZOO UA?    : zoo.exe version 2.1 for MS-DOS, UUencoded
*  MSFIZ UUE    : fiz.exe version 2.1 for MS-DOS, UUencoded
*  ZOO2-1 UA?   : Sources for zoo version 2.1. A uuencoded zoo archive
*  WP2LATEX UUE : WordPerfect to LaTeX translator, ZOOed
*
************************************************************************
*                             rec              last - change
* filename filetype   GET PUT -fm lrecl nrecs   date     time   File description
* -------- --------   --- --- --- ----- -----  -------- -------- ----------------
  STZOO    UUE        ALL NTG F      61  2181 89/12/14 12:33:31
  MSZOO    UAA        ALL NTG V      80   900 92/07/09 13:10:31
  MSZOO    UAB        ALL NTG V      80   657 92/07/09 13:12:19
  MSFIZ    UUE        ALL NTG V      80   412 92/06/22 09:12:32
  ZOO2-1   UAA        ALL NTG V      80   900 92/06/22 09:17:14
  ZOO2-1   UAB        ALL NTG V      80   900 92/06/22 09:19:06
  ZOO2-1   UAC        ALL NTG V      80   900 92/06/22 09:21:13
  ZOO2-1   UAD        ALL NTG V      80   900 92/06/22 09:23:20
  ZOO2-1   UAE        ALL NTG V      80   900 92/06/22 09:24:55
  ZOO2-1   UAF        ALL NTG V      80   900 92/06/22 09:26:50
  ZOO2-1   UAG        ALL NTG V      80   900 92/06/22 09:28:42
  ZOO2-1   UAH        ALL NTG V      80   102 92/06/22 09:28:11
  WP2LATEX UUE        ALL NTG V      80  1229 90/03/12 15:58:03
```

```
************************************************************************
*
*    Utility programs
*
*    UUE    .C    : An (almost) foolproof uuencode program that protects
*                   against network character conversions. It can also
*                   split a large file in multiple parts
*                   This program is used in creating the uue-files in
*                   in this filelist.
*    UUD    .C    : An (almost) foolproof uudecode program that can
*                   correct chaacter conversions. It automagically glues
*                   the parts created by uue.c together.
*    UUX    .DOC  : (Some) documentation to the above programs.
*
************************************************************************
  UUE      C          ALL NTG V      80   298 92/01/24 11:30:21
  UUD      C          ALL NTG V      80   732 91/12/24 22:25:34
  UUX      DOC        ALL NTG V      80   134 91/12/24 22:25:50

*-----------------------------------------------------------------------
*    AMSPEL20 UA% : A Uuencoded *zip* archive that contains a
*                   spelling checker for pc's
*-----------------------------------------------------------------------
  AMSPEL20 UAA        ALL NTG V      80   900 92/08/10 12:48:10
  AMSPEL20 UAB        ALL NTG V      80   900 92/08/10 12:49:35
  AMSPEL20 UAC        ALL NTG V      80   900 92/08/10 12:51:29
  AMSPEL20 UAD        ALL NTG V      80   900 92/08/10 12:52:58
  AMSPEL20 UAE        ALL NTG V      80   420 92/08/10 12:53:31

************************************************************************
*
*    METAFONT sources
*
*    AMSREAD.ME   : A few notes about the contents of AMSFONTS.UUE
*    AMSFONTS.UUE : The AMS font collection UUencoded ZOO archive
*                   split in ten pieces of app. 100kByte
*
*    NOTE : This is still version 2.0 of the distribution !
*
************************************************************************
*                              rec               last - change
* filename filetype   GET PUT -fm lrecl nrecs   date     time   File description
* -------- --------   --- --- --- ----- -----  -------- -------- ----------------
  AMSREAD  ME         ALL NTG V      74    45 90/08/02 14:18:33
```

Behalve via de fileserver TEX-NL, zijn files ook te verkrijgen bij ondermeer de volgende ftp centra:

- `archive.cs.ruu.nl` (Rijksuniversiteit Utrecht; Piet van Oostrum)
- `obelix.icce.rug.nl` (LaTeX& fontzaken; Erik-Jan Vens)
- `ftp.win.tue.nl` (Technische Universiteit Eindhoven; Piet Tutelaers)
- `ftp.th-darmstadt.de`
- `labrea.stanford.edu`
- `math.utah.edu`
- `rusinfo.rus.uni-stuttgart.de`
- `tex.ac.uk`

of via e-mail bij:

- `mail-server@cs.ruu.nl`
- `mail-server@rusmv1.rus.uni-stuttgart.de`
- `LISTSERV@DHDURZ1`
- `texserver@tex.ac.uk`
- `tuglib@math.utah.edu`

Voor NTG leden die niet op een netwerk zijn aangesloten, kunnen eventueel de meeste files via diskettes verkregen worden. Nadere informatie hierover bij Gerard van Nes.

Ook kan gebruik worden gemaakt van het Bullitin Board System van Frans Goddijn: FGBBS (*tel. 085-217041; zie bijlage H*).

NTG's Bulletin Board FGBBS

Frans Goddijn

Steenstraat 78
6828 CN Arnhem

24 april 1993

Abstract

Nieuw voor de Nederlandstalige TEX Gebruikersgroep: een TEX Bulletin Board speciaal voor diegenen die niet op het Internet zijn aangesloten. De naam: **FGBBS**.
Op FGBBS is sinds kort een zo volledig en actueel mogelijke TEX, emTEX, LATEX en MusicTEX collectie beschikbaar voor alle bezitters van een modem. Het BBS is kosteloos toegankelijk voor iedereen en er zijn geen beperkingen aan de hoeveelheid bestanden die kunnen worden opgevraagd. Het systeem is aangesloten op een High Speed modem, vergeleken met de transmissiesnelheid die een directe Internet link biedt misschien niet geweldig, maar veel beter kan het niet over de gewone huis- tuin- en keukenPTTlijn.
FGBBS is te bellen op 085-217041.

Mijn eerste kennismaking met TEX

Ruim een jaar geleden duizelde het mij, toen een vriend in Aken het had over dat vreemde softwarepakket, van reusachtige omvang, dat geen tekstverwerker mocht heten maar wel méér kon dan WP. . . Ik kreeg van hem een stapel floppies mee, kocht bij het Computercollectief Leslie Lamports grappige boek over LATEX en begon aan het avontuur dat LATEX is voor wie het grotendeels in zijn eentje moet uitzoeken.

Toen ik in een afgeladen trein op de grond zat te lezen in Lamport, werd ik aangesproken door een HTS-docent die op zijn werk net zo'n LATEX-fan kende. Dit bleek Cees Fortuin te zijn, wiskundige uit een typografenfamilie, en aan zijn geduldige telefonische raad dank ik de eerste successen. Want het eerste printje in TEX is niet gewoon een print, het vormt een overwinning op zichzelf, al zagen mijn huisgenoten dat anders. . .

De kennismaking met het NTG was een volgende stap, en de diverse publicaties zoals de MAPS en de 'TEX and TUG News' zijn voor mij een meer dan welkome bron van kennis en inspiratie. Vooral het Engelse magazine is door zijn vormgeving een lust voor het oog. LATEX-gebruikers blijken over het algemeen ook zeer beminnelijke en zorgvuldige mensen te zijn. Zo genoeglijk en nauwgezet als de deskundigen zich over een misplaatste 'apostrophe' kunnen buigen, zo lang wordt er gewikt en gewogen eer er op de NTG jaarvergadering wordt besloten een wanbetaler te royeren!

Als lid van de NTG probeer ik al geruime tijd om een efficiënte weg te vinden naar de bron van actuele TEX-implementaties — wie zoals ik geen directe Internetlink heeft, is aangewezen op kopietjes van kopietjes van mensen die wel over zo een verbinding beschikken. Mijn ervaring leert dat zulke kopietjes slechts werken na behoorlijk wat gepriegel. Maar de beleefdheids-Wet van het Gegeven Paard gebiedt dat daarover niet wordt gemopperd. . .

De Internet gebruikers

Hoewel de landelijke ontmoetingsdagen stimulerend zijn, merk ik toch dat iemand die, zoals ik, buitenuniversitair is, en daardoor niet normaal kan beschikken over de verbindingslijnen van het Internet-systeem, verstoken blijft van deelname aan de dagelijkse gesprekken en de meest recente vernieuwingen.[1] Doordat NTG-leden over het algemeen hoog-opgeleide en drukbezette personen zijn, blijkt er niet gemakkelijk daartussen iemand te vinden die wil fungeren als floppyverzendbureau en telefoon-support bij installatievragen. Het gevolg is dat de verspreiding van TEX noodzakelijkerwijs beperkt blijft tot de groep gebruikers die deel uitmaken van een grotere organisatie, en daarbinnen beschikken over een of meerdere TEX-goeroe's die er lol in hebben hun collega's verder te helpen. Dat maakt het wel tot een uiterst sociaal software-pakket: binnen de verschillende groepen ontstaat een intensieve uitwisseling die kan leiden tot schitterende resultaten op

[1] Deze practische onbereikbaarheid van TEX-bronnen zou weleens één van de belangrijkste oorzaken kunnen zijn waardoor TEX buiten het universitaire circuit zo weinig gebruikers vindt.

typografisch gebied maar evenzeer op het gebied van team-vorming. *Vermoed ik* — zelf sta ik er goeddeels buiten.

Het FIDO-netwerk

Het FIDO-netwerk is voor mij wat Internet is voor de meeste TEX users.[2] Via PC en modem communiceer ik met computervrienden en -kennissen. Maar het FIDO echomail-berichtengebied gewijd aan TEX is dermate stil dat het meer een sluimer-gebied is. Sinds begin dit jaar lees ik wel via Interface, een Haags BBS, het oorspronkelijk van Internet afkomstige `COMP.TEXT.TEX`, maar de inhoud daarvan is dermate specialistisch, dat ik me er niet in durf te mengen. Wanneer twee Amerikanen met elkaar spreken over de progammeer-technische complicaties van een TEX-applicatie onder UNIX, kan ik moeilijk zeggen 'En heren, ik zie blokjes in plaats van letters, wat doe ik ook al weer fout?'

Op dit Interface ontmoette ik echter Henk de Haan, mede TEX gebruiker, die bezig is te promoveren op 'Muonen gekatalyseerde kernfusie', en die over muonen zowel als TEX in zeer begrijpelijke taal kan spreken. Sinds die ontmoeting — als je daarvan kunt spreken, want we corresponderen dagelijks langs electronische weg maar hebben elkaar nog nooit gezien — is mijn ontwikkeling in LATEX snel gegaan. Verschillende dozen vol floppies, door Henk voorzien van de laatste versies, zijn heen en weer gestuurd tussen onze woonplaatsen Delft en Arnhem en schier onvermoeibaar heeft Henk me terzijde gestaan bij het installeren van TEX voor OS/2.

FGBBS

Om het anderen gemakkelijker te maken dit pad te volgen, is er een bulletin-board in het leven geroepen. Op:

085-217041

vindt de beller FGBBS, een mede door Henk de Haan onderhouden systeem dat geheel is toegewijd aan het woord: twee file area's zijn er, één voor boeken, en één voor TEX. Daar is een voor DOS, Windows en OS/2 zo compleet mogelijke collectie TEX-programmatuur kosteloos op te halen, *alle dagen van de week op vrijwel elk uur van de dag*.

De capaciteit van het modem aan de kant van FGBBS maakt het mogelijk op hoge snelheden te werken, al zijn getallen als op directe `ftp`-lijnen niet te evenaren. Wie slechts een 2400 baud modem heeft, kan in 5 minuten 75 Kilobytes binnenhalen en dat schiet niet op wanneer je van voren af aan begint met TEX. Op 14K4, met een High-Speed modem, gaat het achtmaal sneller. Een megabyte kost dan tien minuten en wie zijn TEX-setup drastisch wil omgooien, komt met een uurtje 'pompen' al een heel eind in de buurt.

Het is verstandig een hedendaags communicatiepakket hiervoor te gebruiken. In de eerste plaats omdat deze doorgaans is uitgerust met Zmodem, het meest efficiënte transfer-protocol, en in de tweede plaats om het gebruik van 'Avatar', een variant op ANSI die de schermafhandeling belangrijk versnelt, hetgeen het bladeren van scherm naar scherm in FGBBS flitsend laat verlopen. Ook kan het nuttig zijn als je je toetsenbord tijdelijk kan omschakelen naar 'doorway mode', zodat PageDown niet, zoals gebruikelijk bij communicatie, direct het downloaden start, maar desgewenst ook weer 'volgende pagina' kan betekenen.

Beschikbare TEX files

Via de 'Delftsche Connection' van Henk de Haan wordt de voorraad files behoorlijk actueel gehouden. Er wordt nog gewerkt aan een goede presentatie, zodat de lijst van bestanden van boven naar beneden eerst de meest essentiële files laat zien. Onderaan staan talloze `.STY`-files. Daartussen zijn er die de gebruiker in staat stellen 'camera-ready' kopij in te leveren in standaard formaat van de huisstijl voor Elsevier of Kluwer, hetgeen van pas kan komen. Ook stijlen voor het maken van notulen van vergaderingen met Franse archeologen of Amerikaanse atoomdeeltjes-onderzoekers, wat misschien niet direct aan een grote behoefte zal voldoen. Voorts blijkt eruit dat de `verbatim`-omgeving en de `tabular` menig programmerend gebruiker te beperkt zijn: voor beide staat er een handvol alternatieven tussen.

Een extra specialisme kan nog ontstaan. Ik ben namelijk aangestoken door een groot enthousiasme voor het MusicTEX pakket van Taupin, en heb, zonder veel van muziek te weten (net mijn eerste klarinet gekocht) samen met de klarinetjuf een begin gemaakt met een geheel in TEX geschreven muziekleerboekje, getiteld 'Notenmousse, muziektheorie voor Onze Kleinen'. Aangezien de door Taupin zelf geschreven handleiding meer uitblinkt door vormgeving dan door duidelijkheid, zou medewerking van een ervaren gebruiker van MusicTEX kunnen uitgroeien tot een extra loot aan FGBBS!

Tot bellens.

[2] Inmiddels kan ik soms via een FIDO-Internet tussenstation naar de Internet-wereld communiceren, echter dit wisselend succes steunt nog op een aantal noodzakelijke toevalstreffers.

Lijst beschikbare software[3]

Main Directory FGBBS

0Fgbbs.Lst	1k	15-05-93*	ASCII versie van ALLE files ALLE area's
Wmealist.New	1k	15-05-93*	ASCII versie van NIEUWE files tot 1 week oud
0Fgbbs.Arj	12k	13-05-93*	ARJ compressed complete directory
Newfiles.Txt	2k	13-05-93*	identiek aan Wmealist.new
Newfiles.Arj	1k	13-05-93*	ARJ compressed version NEW files max 1 week

General Utilities

Hoe_Zip.Txt	8k	24-01-93	Wat is ZIP nu weer?
Qedit215.Zip	131k	24-01-93	Shareware editor
Shez83.Zip	206k	24-01-93	look into / manage compressed files
List77A.Zip	90k	24-01-93	LIST / look at files
Arj230.Exe	195k	25-01-93	ARJ compress utility
Pkz204G.Exe	197k	10-02-93	ZIP compress utility
Tm4001.Zip	191k	17-01-93	Telemate - in 4 stukken.
Tm4002.Zip	195k	17-01-93	Telemate
Tm4003.Zip	178k	17-01-93	Telemate
Tm4004.Zip	99k	17-01-93	Telemate
Sio062A.Zip	24k	11-04-93	Voor snellere communicatie per modem & OS/2
Ufm-540O.Zip	149k	09-05-93*	UFM voor OS/2
Ufm-540D.Zip	164k	09-05-93*	UFM voor DOS
Ufm-540U.Zip	25k	09-05-93*	UFM Utilities
Ufm-540W.Zip	193k	10-05-93*	UFM voor Windows

NTG

Whatis.Ntg	2k	24-04-93	Informatie over NTG (in ASCII)
Ntginfo.Tex	10k	02-05-93	TeX file folder/ aanmeldingsformulier NTG
Brief.Arj	12k	31-03-93	Dutch variety of LETTER
Ntg.Arj	18k	31-03-93	add-ons for ARTIKEL and RAPPORT STY
Voorwerk.Arj	1k	31-03-93	Dutch version of titlepage

TEX-NL

Texnl91A.Exe	139k	24-04-93	Tex-Nl Discussion List 1991 Mrt-Jun
Texnl91B.Exe	360k	24-04-93	Tex-Nl Discussion List 1991 Second Part
Texnl92A.Exe	349k	11-02-93	Tex-Nl Discussion List 1992 First Part
Texnl92B.Exe	338k	11-02-93	Tex-Nl Discussion List 1992 Second Part
Texnl01.Exe	46k	24-04-93	Tex-Nl Discussion List Jan 1993
Texnl02.Exe	68k	02-04-93	Tex-Nl Discussion List Febr 1993
Texnl03.Exe	46k	02-04-93	Tex-Nl Discussion List March 1993
Texnl93.Exe	6k	09-05-93*	Tex-Nl Discussion List Table Of Contents 1993
Texnl89A.Exe	209k	09-05-93*	TEX-NL jan/dec 1989
Texnl90A.Exe	260k	09-05-93*	TEX-NL jan/jun 1990
Texnl90B.Exe	219k	09-05-93*	TEX-NL jul/dec 1990
Texnl91.Exe	10k	09-05-93*	TEX-NL jan/feb 1991
Texnl04.Exe	74k	09-05-93*	TEX-NL april 1993
Tugboat.Exe	43k	09-05-93*	Bibliography file TUGboat 1981-93
Mapsbib.Exe	11k	09-05-93*	Bibliography file MAPS 1992
Texnl90.Exe	6k	09-05-93*	
Texnl91C.Exe	60k	09-05-93*	
Texnl92.Exe	15k	09-05-93*	

TEX, LATEX, emTEX standard files

Latex1.Zip	234k	01-02-92	LaTeX files
Latex2.Zip	215k	01-02-92	LaTeX files deel II
Tex1.Zip	382k	24-01-92	basis files TeX
Tex2.Zip	291k	24-01-92	basis files TeX
Latexdoc.Zip	106k	01-02-92	Doc file

TEX, 4TEX

386.Arj	1k	15-03-93	4TeX complete start pack I
4Tex.A01	1422k	15-03-93	pack III
4Tex.A02	1254k	15-03-93	pack IV
4Tex.Arj	1422k	15-03-93	pack II
4Tex.Ins	5k	08-04-93	install manual 4TeX
4Tex.Txt	29k	21-03-93	what's in these 4tex files?
4Texdoc.Arj	343k	15-03-93	many manuals of 4TeX
4Texinst.Txt	4k	15-03-93	First install advice
Dutch.Arj	466k	15-03-93	NL word file
English.Arj	311k	15-03-93	English word file
French.Arj	139k	08-04-93	French words for AMSpell spell check
German.Arj	4k	11-04-93	Deutsche word file
Source.Arj	97k	15-03-93	source of AMSpell
Spell.Arj	42k	15-03-93	Spell checker AMSpell
Usengl.Arj	242k	15-03-93	USA word file
Laser.A01	1422k	08-04-93	same
Laser.A02	1422k	08-04-93	same
Laser.A03	1422k	08-04-93	same
Laser.A04	1422k	08-04-93	same
Laser.A05	1422k	08-04-93	same
Laser.A06	193k	08-04-93	same
Laser.Arj	1422k	08-04-93	HPlaser/DeskJet/Kyocera... fonts for TeX
Matrix.A01	1422k	08-04-93	matrix printer fonts
Matrix.A02	340k	08-04-93	matrix printer fonts
Matrix.Arj	1423k	08-04-93	matrix printer fonts
Postscri.Arj	735k	08-04-93	PostScript TFM files
Graphics.A01	1421k	15-03-93	incorporate graphics in TeX
Graphics.A02	1228k	15-03-93	incorporate graphics in TeX
Graphics.Arj	1384k	15-03-93	incorporate graphics in TeX
Bigtex.A01	118k	15-03-93	BIG TeX file 2
Bigtex.Arj	1422k	15-03-93	BIG TeX file 1
Ams.Arj	251k	08-04-93	All the files for AMS-TeX & AMS-LaTeX
Metafont.A01	463k	08-04-93	Metafont, Automatic Font Creation
Metafont.Arj	1422k	08-04-93	same
Tex.Arj	1185k	15-03-93	Collection Standard TeX files
Texdoc.A01	882k	15-03-93	DOC files
Texdoc.Arj	1422k	15-03-93	DOC files

Tiny TEX

Tinytex.Zip	663k	13-05-93*	TinyTeX, een werkende demo-cocktail
Read.Me	6k	13-05-93*	manual file / info

TEX STYle Files

2Up.Arj	10k	31-03-93	macros for printing a doc two-up
A4.Arj	3k	31-03-93	page size a4
A5.Arj	1k	31-03-93	a5 style, only for 10 points
Aaai.Arj	7k	10-04-93	style for AAAI conference 1988
Agugrl.Arj	6k	10-04-93	AGU Geopgysical Research letter style
Agujgr.Arj	6k	10-04-93	same, journal style
Aip.Arj	3k	10-04-93	American Institute of Physics style
Album.Arj	7k	10-04-93	make CASSETTE labels
Algorith.Arj	1k	11-04-93	typeset algorithms
Alleqno.Arj	1k	10-04-93	number all equations
Alltt.Arj	1k	10-04-93	like verbatim, but much more
Altnline.Arj	2k	11-04-93	append line numbers to text
Amssymbo.Arj	2k	10-04-93	load ams font
Annotati.Arj	1k	11-04-93	print or ignore text
Answers.Arj	2k	11-04-93	for making exercises
Apalike.Arj	5k	10-04-93	American Psychological Association
Apl.Arj	60k	10-04-93	typeset apl programming language
Apsabstr.Arj	5k	10-04-93	handig voor leden van American Physical Soc.
Arabtex.Zip	348k	02-04-93	include arab writing
Array.Arj	71k	10-04-93	revamped array and tabular environment
Artikel.Arj	74k	31-03-93	Dutch variety of article
Arydshln.Arj	2k	13-04-93	dashed horizontal/vertical line for array/ta
Asaetr.Arj	10k	10-04-93	idem
Askinclu.Arj	1k	11-04-93	prompts user for files to \include
Astyped.Arj	1k	11-04-93	extended verbatim environment
At-Sty.Arj	1k	11-04-93	put text on absolute position
Autotab.Arj	2k	11-04-93	create tabs automatically reading seperate
Balanced.Arj	1k	11-04-93	like two-column environment
Biblist.Arj	21k	08-04-93	typeset large bibliography lists
Bibmods.Arj	1k	11-04-93	modified \thebibliography
Bibunits.Arj	2k	11-04-93	seperate bibliographies in doc
Bigbox.Arj	1k	11-04-93	all figures in a big box
Bigsign.Arj	6k	10-04-93	make VERY big signs like arrows
Bigtabul.Arj	1k	11-04-93	let large tables split across pages
Biihead.Arj	1k	10-04-93	underlined heading
Blackboa.Arj	1k	31-03-93	definitions for blackboard N, Q, R and Z

[3] Versie 15 mei 1993; files gemarkeerd met een '*' zijn minder dan 7 dagen oud.

Bnf.Arj	11k	11-04-93	typeset Backus-Naur Form Syntax notation
Boek.Arj	7k	31-03-93	DUTCH style book
Bookform.Arj	23k	10-04-93	mil-std-490 bookform docstyle
Boxedeps.Arj	20k	11-04-93	include epsf files in a doc in driver-indep
Boxedmin.Arj	1k	10-04-93	put box round minipage
Boxit.Arj	1k	11-04-93	draws lines around contents
Boxminip.Arj	1k	31-03-93	boxed-minipage
Breakcit.Arj	1k	10-04-93	allow citations to break across lines
Brief.Arj	12k	31-03-93	dutch variety of letter
Bsf.Arj	1k	10-04-93	Bold Sans Serif
Bsl.Arj	1k	10-04-93	Bold Slanted font
Buscard.Arj	3k	11-04-93	make business cards
Calendar.Arj	6k	11-04-93	typeset a calendar
Captcont.Arj	1k	10-04-93	captions in continuation of floats, erg goed
Card.Arj	3k	11-04-93	make card file
Catmac.Arj	11k	10-04-93	commutative diagram macros
Cd.Arj	3k	10-04-93	same
Cea.Arj	1k	11-04-93	Computers and Electronics in Agriculture
Changeba.Arj	16k	10-04-93	changebars for emTeX and DVIPS
Chappg.Arj	1k	11-04-93	output chapter-page, like 1--3, 5--2 etc
Chapref.Arj	1k	11-04-93	seperate reference sections for chapters
Chapterb.Arj	3k	10-04-93	seperate bibliography per chapter
Chbars.Arj	100k	10-04-93	same withoput \special 's
Chemtex.Arj	55k	10-04-93	Typeset chemical formula's
Chomsky.Arj	2k	11-04-93	typeset parse trees
Cite.Arj	3k	10-04-93	compress lists of numbers in citations to ra
Citesidx.Arj	4k	10-04-93	add reference-page-lists to bibliogr-items
Citesort.Arj	1k	11-04-93	sort and compress lists of \cite's
Colors.Arj	6k	11-04-93	add grey scales and colours for PS devices
Colortab.Arj	7k	11-04-93	add colours to tables etc
Colortex.Arj	41k	10-04-93	add color, postscript printers only
Comment.Arj	1k	11-04-93	comment out sections and parts
Covingtn.Arj	9k	11-04-93	for special notation in linguistics
Cprog.Arj	4k	11-04-93	tyeset C programs
Crosswor.Arj	7k	11-04-93	make crossword puzzles
Csart.Arj	3k	11-04-93	article style for short texts
Csquote.Arj	3k	11-04-93	replace " by `` and ''
Csty.Arj	4k	31-03-93	typeset c, c++ and the like
Cup.Arj	12k	10-04-93	Cambridge University Press bookstyle
Custombi.Arj	1k	11-04-93	customize section head
Cyrillic.Arj	1k	10-04-93	load cyrillic font
Dashline.Arj	3k	13-04-93	produce dashed lines in picture environment
Db-Sty.Arj	2k	11-04-93	generate listings, send letters use database
Decalign.Arj	2k	11-04-93	align in tables on decimal point
Dectab.Arj	1k	11-04-93	same
Deflist.Arj	1k	11-04-93	enhance list environment
Deproc.Arj	17k	10-04993	DECUS proceedings style
Dina4.Arj	1k	11-04-93	DIN A4 format
Doc.Arj	93k	10-04-93	mix doc and code
Doublesp.Arj	2k	10-04-93	double spacing in text
Dpccyr.Arj	2k	11-04-93	makes ams cyrillic fonts available underNFSS
Draft-Ma.Arj	1k	11-04-93	prints DRAFT in large grey pages diagonal
Draft.Arj	3k	10-04-93	draft option for docs - handy for debugging
Drafthea.Arj	1k	10-04-93	print draft in heading
Drafts.Arj	1k	11-04-93	put word DRAFT on top and bottom page
Drftcite.Arj	1k	11-04-93	modify \cite to print bibliography tags
Drop.Arj	1k	10-04-93	make dropped initials for paragraphs
Eclbip.Arj	4k	10-04-93	draw bipartite graphs with epic.sty
Eepic.Arj	17k	10-04-93	enhance EPIC USING TPIC \SPECIAL's
Elsevier.Arj	7k	31-03-93	Uitgeverij Elsevier camera-ready style
Endfloat.Arj	4k	11-04-93	put figures and tables at end of doc
Endnotes.Arj	3k	11-04-93	make endnotes instead of footnotes
Enumerat.Arj	7k	11-04-93	enhanced enumeration
Enumspec.Arj	1k	11-04-93	leading character on enumerations
Env.Arj	1k	11-04-93	print on envelopes
Epac.Arj	3k	11-04-93	for European Particle Accelerator Conference
Epsfig.Arj	10k	11-04-93	psfig merged with EPSF
Eqnarray.Arj	1k	11-04-93	refined eqnarray
Equation.Arj	3k	11-04-93	help construct display of equations
Eslides.Arj	5k	10-04-93	convert doc to slides
Espo.Arj	1k	11-04-93	esperanto
Exam.Arj	4k	11-04-93	typeset exams (multiple choice etc)
Fancybox.Arj	18k	11-04-93	tips and tricks several boxes
Fancycha.Arj	1k	11-04-93	fancy chapter headings
Fancyhea.Arj	4k	11-04-93	modify headers and footers
Farticle.Arj	2k	11-04-93	not that! French style Article
Fepsf.Arj	3k	11-04-93	fake-EPSF.TEX macro file
Feynman.Arj	91k	08-04-93	typeset Feynman diagrams (Physics)
Figures.Arj	1k	13-04-93	right and left side figures
Fillform.Arj	7k	10-04-93	helps fill forms with LaTeX
Fixhead.Arj	1k	11-04-93	handle marks correctly with 2 columns
Fixup.Arj	2k	11-04-93	fixup Plain's \bigl, etc to track size change
Float.Arj	11k	11-04-93	improved floating
Floatfig.Arj	8k	10-04-93	let text flow around figures
Floatnoh.Arj	1k	11-04-93	only floats, no headers and footers
Flow.Arj	1k	11-04-93	flow text round illustration
Flowchar.Arj	1k	11-04-93	write flow charts
Fnpara.Arj	2k	11-04-93	set footnotes as paragraphs
Foiltex.Arj	96k	10-04-93	make transparant foils etc
Fontsel.Arj	70k	07-04-93	Preview from LATEX3! New Font Selection Scheme
Footnpag.Arj	8k	08-04-93	enumerate footnotes on each page
Format.Arj	1k	11-04-93	print FP nrs in fixed format
Fpbox.Arj	1k	11-04-93	make framed parbox
Frenchpo.Arj	1k	11-04-93	french punctuation
Frontier.Arj	3k	11-04-93	for FRONTIER conferences
Ftnright.Arj	16k	07-04-93	foornote in multicolumn.sty (multicol.arj)
Fullpage.Arj	1k	11-04-93	get more into that page
Geom.Arj	150k	10-04-93	enhance Article&book styles incl PS support
Geophysi.Arj	1k	11-04-93	Geophysycs Journal style
Hackallo.Arj	1k	11-04-93	make allocation local
Hangcapt.Arj	1k	12-04-93	captions with hanging indentions
Harvard.Arj	4k	12-04-93	support Harvard bibliography styles
Headerfo.Arj	1k	11-04-93	underlinde heading
Here.Arj	3k	12-04-93	If you don't float you can't sink
Histogr.Arj	3k	12-04-93	draw histogram bars inside picture
Hvdashln.Arj	1k	12-04-93	dashed lines in tabulars
Icassp.Arj	3k	11-04-93	camera-ready copy for ICASSP '89 conference
Idx.Arj	1k	07-01-92	input file to print .IDX files
Ifthen-A.Arj	1k	12-04-93	extension to \ifthen
Ijcai91.Arj	8k	11-04-93	conference style
Indent.Arj	2k	26-04-93	Indentiation Environments For \Latex
Indentfi.Arj	1k	12-04-93	indent first paragraph of section
Index.Arj	7k	26-04-93	Re-Implement \Index Command In Latex
Inputfil.Arj	1k	12-04-93	keep track of current inputfile
Insertpl.Arj	1k	11-04-93	keep track of current \input file
Inwords.Arj	1k	26-04-93	Typeout A Number In Normal Words
Isf.Arj	1k	12-04-93	provide access rto italic san serif fonts
Iso.Arj	37k	11-04-93	to write ISO standards
Ist21.Arj	1k	11-04-93	IST21 doc style for cover page
Isucapti.Arj	1k	12-04-93	captions with hanging indentions
Italic.Arj	1k	12-04-93	typeset in italic and insert italic correction
Itcorrec.Arj	1k	31-03-93	smarter \it
Jbs.Arj	2k	11-04-93	Journal of Business Strategies style
Jeep.Arj	10k	11-04-93	useful changes in article style
Jmb.Arj	1k	11-04-93	Journal of Molecular Biology
Kluwer.Arj	34k	10-04-93	Kluwer Uitgeverij style file
Label.Arj	1k	31-03-93	alternative description environment
Labelfig.Arj	7k	12-04-93	put tex labels on imported graphs
Labels.Arj	6k	12-04-93	multiple address labels on one page
Lablst.Arj	1k	11-04-93	print label definitions end of doc
Laletter.Arj	81k	08-04-93	Los Alamos labs letter style
Lambda.Arj	14k	12-04-93	set of lambda-calculus and list-handling mac
Lamemo.Arj	61k	10-04-93	Los Alamos lab memo sty
Layout.Arj	25k	10-04-93	print diagram showing page layout
Layouttx.Arj	4k	07-01-92	shows several latex parameters
Lcircuit.Arj	68k	07-04-93	Electronic circuit symbols
Lcustom.Arj	3k	11-04-93	useful macros and definitions
Lfntsams.Arj	11k	11-04-93	lfonts_ams.tex use ams symbols
Listofth.Arj	1k	12-04-93	make list of theorems
Local-Su.Arj	1k	11-04-93	supplement to LOCAL GUIDE!
Loggates.Arj	7k	07-04-93	digital circuit symbols
Lollipop.Arj	98k	27-10-92	Victor Eijkhout's alternative to TeX-LaTeX
Lscape.Arj	2k	12-04-93	define landscape options (need dvips)
Lslide.Arj	13k	12-04-93	typeset slides
Ltgtsim.Arj	1k	12-04-93	combine <,~ and counterparts
Makerobu.Arj	1k	12-04-93	make commands so robust don't need \protect
Manpage.Arj	11k	12-04-93	make UNIX manual pages
Manual.Arj	10k	10-04-93	sty for manuals
Marginot.Arj	1k	12-04-93	marginal notes numbered like footnotes
Marnote.Arj	1k	12-04-93	short note vertically on all pages
Memo2.Arj	7k	10-04-93	for memos
Merge.Arj	2k	11-04-93	form letter option to

			letter sty
Mf-Sty.Arj	1k	11-04-93	Make metafont logo's at all sizes
Mitpress.Arj	1k	11-04-93	MIT PRESS format
Mitthesi.Arj	5k	11-04-93	Thesis style for MIT
Morefloa.Arj	1k	12-04-93	increase number of floats 18...36
Moreverb.Arj	2k	12-04-93	more verbatim tricks
Mrgnote.Arj	1k	12-04-93	put marginal notes all over the place
Multibox.Arj	1k	11-04-93	multiple boxes in pictures
Multicol.Arj	37k	08-04-93	multiple columns of text
Multido.Arj	6k	12-04-93	loop macro fixed point addition
Multirow.Arj	1k	12-04-93	tabular entries span multi rows
Nar.Arj	2k	11-04-93	nuclear acids research
Newequat.Arj	1k	12-04-93	define \newequation
Newthm.Arj	1k	12-04-93	customize theorem invironment
Nf-Sty.Arj	2k	12-04-93	Nuclear Fusion magazine style
Nl-Sty.Arj	1k	11-04-93	style file for DUTCH people
Nofm.Arj	1k	11-04-93	for "n of M" style pagination
Nopagenu.Arj	1k	11-04-93	remove page numbers
Nosecnum.Arj	1k	31-03-93	show headings leave out numbers
Ntg.Arj	18k	31-03-93	add-on options for ARTIKEL and RAPPORT sty
Numdef.Arj	1k	12-04-93	allow arrays of macros
Numline.Arj	3k	26-04-93	Type-Out Line Numbers
Outline.Arj	2k	10-04-93	enhance enumeration style
Oval.Arj	3k	10-04-93	put rounden boxes around text
Ovalfbox.Arj	1k	12-04-93	fbox with rounded corners
Overcite.Arj	3k	11-04-93	superscript numbers for citation etc
Oxford.Arj	4k	11-04-93	Oxford University style
Pagefoot.Arj	2k	11-04-93	put footnotes at bottom each page
Pagefram.Arj	16k	10-04-93	put frame or cropmarks at edges of page
Paper.Arj	29k	12-04-93	double-spaced college papers
Parskip.Arj	1k	11-04-93	set parindent to 0 and add glue
Path.Arj	4k	12-04-93	\verb like macros allows line breaks
Picinpar.Arj	8k	12-04-93	pictures in paragraphs
Picins.Arj	216k	10-04-93	Integrate pictures in docs
Pict.Arj	1k	31-03-93	several macros for picture environment
Portland.Arj	1k	11-04-93	NOT the state! PORTrait and LANDscape swits
Poster.Arj	7k	12-04-93	make posters and banners
Program.Arj	12k	07-04-93	algorithm typsettings
Proof.Arj	2k	12-04-93	typeset proof trees
Psbox.Arj	19k	26-01-93	incl postscript pictures
Psboxit.Arj	2k	12-04-93	put postscript drawing behing TeX box
Psfig.Arj	69k	31-03-93	incl encapsulated PostScript graphics
Psfrag.Arj	19k	08-04-93	overlay POSTSCRIPT figures
Pslatex.Arj	53k	10-04-93	user esident postscript fonts on PS printer
Pstricks.Arj	259k	26-01-93	PostScript macros
Qed.Arj	8k	12-04-93	little triumphant square after proof
Quote.Arj	1k	12-04-93	expand " into `` or ''
Rag.Arj	1k	31-03-93	alternative to \raggedright
Raggedta.Arj	1k	12-04-93	all parboxes raggedright
Rangecit.Arj	1k	12-04-93	compress \cites
Rapport.Arj	56k	31-03-93	DUTCH report.sty
Rcs.Arj	1k	12-04-93	interface to Revision Control System
Refman.Arj	6k	11-04-93	styl;e for reference manuals like PostScript
Remark.Arj	1k	11-04-93	newtheorem without it
Res.Arj	8k	11-04-93	format do resumes by Michael DeCorte
Resume.Arj	2k	11-04-93	same by Stephen Gildea
Revtex3.Arj	183k	10-04-93	American Physical Society periodicals
Roman.Msg	1k	22-04-93*	make roman numerals into arabic numbers
Romanneg.Arj	1k	11-04-93	roman numbered pages get negative numbers
Rotating.Arj	33k	10-04-93	for rotated objects, with DVIPS
Sc21.Arj	1k	11-04-93	iso/tc97/sc21 doc style for cover page
Sched.Arj	3k	12-04-93	draw schedule
Schedule.Arj	5k	11-04-93	generate schedule sheets
Screen.Arj	1k	11-04-93	helps create doc suitable for screen prevu
Seceqn.Arj	1k	12-04-93	cause equations to be numbered within section
Selectp.Arj	3k	26-04-93	Select Pages To Write From Within Source File
Seminar.Arj	133k	08-04-93	slides and notes
Semitic.Arj	1k	11-04-93	set semitic language
Setspace.Arj	3k	12-04-93	doubl;e and 1.5 spaces depending on size
Sfwmac.Arj	1k	11-04-93	macros for UNIX docs
Shading.Arj	5k	12-04-93	text on shaded background
Shadow.Arj	1k	11-04-93	shadowed background on text DVIPS
Showkeys.Arj	4k	12-04-93	modify \label \cite to print their key argument
Showlabe.Arj	1k	11-04-93	shows labels and references
Showtags.Arj	1k	12-04-93	show labels inline & print undefined ones
Siam.Arj	10k	10-04-93	produce articles for SIAM journals
Side.Arj	1k	12-04-93	include landscape figures and tables
Slem.Arj	1k	11-04-93	change \sl to \em
Slide20.Arj	6k	08-04-93	alternate slide making
Sober.Arj	1k	31-03-93	reduce space after section headings/lists
Spacecit.Arj	1k	11-04-93	give spaces between citations
Spie.Arj	1k	12-04-93	SPIE proceedings camera-ready copies
Springer.Arj	212k	10-04-93	Springer-Verlag periodicals layout
Sprite.Arj	1k	12-04-93	low resolution bitmap characters
Stickers.Arj	1k	12-04-93	make stickers, glue not included
Subeqn.Arj	2k	11-04-93	subnumbering in related equations 1a,1b etc
Subfigur.Arj	2k	11-04-93	same, related figures
Supertab.Arj	11k	31-03-93	enhanced tabular
Suthesis.Arj	3k	11-04-93	Stanford Univerity thesis style
Syllabus.Arj	3k	12-04-93	syllabus doc style
Symbols.Arj	4k	12-04-93	listing all standard math symbols
Tabextra.Arj	1k	12-04-93	extra tabular tricks
Tables.Arj	9k	11-04-93	easdy for ruled and unruled tables
Tabls.Arj	3k	11-04-93	simulate minimum-lineskip glue in tab envir
Tabverb.Arj	1k	12-04-93	verbatim extension, inc tab characters
Texnames.Arj	1k	11-04-93	define more TeX names
Textfit.Arj	1k	12-04-93	scale up text to desired size (need NFSS)
Textmerg.Arj	9k	12-04-93	like mail merge
Tgrind.Arj	2k	11-04-93	tgrind macros for LaTeX instead of TeX
Theorem.Arj	28k	10-04-93	reimplementation of \newtheorem command
Threecol.Arj	3k	12-04-93	typset in three columns
Threepar.Arj	1k	11-04-93	three part page headers
Threpart.Arj	1k	13-04-93	tabular environment notes
Trademar.Arj	1k	11-04-93	common trademarks
Tree-Sty.Arj	5k	12-04-93	draw binary trees with PICTeX
Tree.Arj	13k	08-04-93	draw binary trees with DVIPS
Truecols.Arj	6k	12-04-93	sub-style with slitex and colour printers
Tugboat.Arj	39k	31-03-93	TeX User Group magazine style
Twoup.Arj	1k	11-04-93	change page size make 2 fit on 1
Uga.Arj	10k	12-04-93	University of Georgia style
Ukdate.Arj	3k	11-04-93	change \today to UK style date
Ulem.Arj	5k	11-04-93	use Underline for \em
Underlin.Arj	2k	12-04-93	underline headings
Unixman.Arj	8k	12-04-93	produce UNIX manuals
Uscthesi.Arj	14k	12-04-93	Usenet Conference Proceedings
Usenix.Arj	1k	11-04-93	for usenix conference proceedings
Vdm.Arj	26k	11-04-93	Vienna Development Method
Verb-Fil.Arj	1k	11-04-93	include a file in verbatim mode
Verbatim.Arj	26k	10-04-93	improved verbatim and comment environments
Version.Arj	1k	12-04-93	define optionally ignored environments
Voorbeel.Arj	1k	12-04-93	define enumerate-like env for linguistics
Voorwerk.Arj	1k	31-03-93	dutch version titlepage
Vpage.Arj	2k	12-04-93	easy margins for given paper size
Vrbsubfi.Arj	1k	12-04-93	verbatim input to subsections of files
Window.Arj	3k	12-04-93	buils windows to typeset pictures in text
Wrapfig.Arj	2k	12-04-93	figure at side page, wrap text around
Xarticle.Arj	9k	10-04-93	7,8, and 9-points versions of article sty
Xcomment.Arj	10k	12-04-93	print only selected environments
Xrefwarn.Arj	1k	12-04-93	modify \cite command to warn undefineds
Xspace.Arj	2k	12-04-93	make space after abbreviation unless not
Xxxslide.Arj	2k	11-04-93	extra macros for slides
Yearcal.Arj	1k	12-04-93	print year calendar if not got from boss
Young.Arj	1k	12-04-93	draw Young tableaux (group theory)

BIG TEX

Blatex.Zip	224k	01-02-92	big latex
Tex386B8.Arj	182k	20-05-92	OS/2 en DOS BigTeX executable
Texb5.Zip	410k	11-03-92	idem
Btex1.Zip	259k	26-02-91	BigTeX oudere versie deel 1
Btex2.Zip	290k	24-01-92	BigTeX oudere versie deel 2
Bslitex.Zip	247k	01-02-92	BIG SliTeX

DVI things

Dvimsw26.Arj	27k	26-03-93	Dvi Previewer For Ms-Windows vers 2.6
Dvidvi.Zip	57k	22-02-93	Trick for pagination
Dvips54.Zip	308k	16-01-92	Files for PostScript needs DVPSLIB.ZIP

Dvidr14S.Zip	1009k	13-11-92	Beta versie 1.4s van drivers
Dvipslib.Zip	342k	28-11-91	with DVIPS54.ZIP
Dvi2Dvi.Zip	48k	14-05-93*	Print 2/4 DVI pages/sheet faster than DVIDVI
Dvivga2.Zip	125k	14-05-93*	TeX DVI Windows previewer fonts
Dvivga3.Zip	124k	14-05-93*	TeX DVI Windows previewer fonts
Dvivga4.Zip	120k	14-05-93*	TeX DVI Windows previewer fonts
Dvivga5.Zip	156k	15-05-93*	TeX DVI Windows previewer fonts
Dvivga6.Zip	92k	15-05-93*	TeX DVI Windows previewer fonts
Dvivga7.Zip	182k	15-05-93*	TeX DVI Windows previewer fonts
Dvivga8.Zip	182k	15-05-93*	TeX DVI Windows previewer fonts

TEX Graphics

Epic.Arj	27k	10-04-93	New commands for PICture environment
Texdraw.Arj	185k	10-04-93	Make PostScript drawings from within TeX

MetaFont

Bmf1.Zip	260k	28-11-91	BIG metafont
Bmf2.Zip	265k	28-11-91	big metafont
Mf1.Zip	244k	28-11-91	metafont
Mf2.Zip	354k	03-01-92	metafont
Mf3.Zip	272k	28-11-91	metafont
Mfjob11K.Zip	93k	25-04-92	MetaFont utils
Mfware1.Zip	320k	28-11-91	metafont utils deel I
Mfware2.Zip	137k	28-11-91	metafont utils deel II
Misc_Mf.Zip	35k	28-11-91	diverse metafont files

MusicTEX

M2Tex11.Arj	87k	16-03-93	Convert MIDI to MusicTeX
Musicexa.Zip	132k	23-02-93	Muziek Files bij MusicTeX
Musictex.Zip	318k	23-02-93	musicTeX
Recueil.Zip	122k	23-02-93	muziekje
Musicfaq.Arj	3k	04-05-93	ASCII medicine for MusicTeX despair
Taupin.Zip	1k	12-05-93*	Brief aan / briefje van TAUPIN himself!

TEX Fonts

Lj_0.Fli	580k	05-03-93	FLI font Library for Hewlett Packard part 0
Lj_1.Fli	539k	28-11-91	idem part 1
Lj_2.Fli	649k	28-11-91	idem part 2
Lj_3.Fli	787k	28-11-91	idem part 3
Lj_4.Fli	970k	28-11-91	idem part 4
Lj_5A.Fli	1162k	28-11-91	idem part 5a
Lj_5B.Fli	40k	28-11-91	idem part 5b
Lj_H.Fli	493k	28-11-91	idem part h
Lj_Sli.Fli	371k	28-11-91	idem part sli (for slides)

TEX Spell Checkers

Dvispell.Zip	51k	25-04-92	spell check dvi
Spell.Zip	378k	29-01-92	TeX sentitive spell check met NL woorden
Us.Zip	285k	29-01-92	usa word file
Texware.Zip	265k	26-02-91	Syntax checker for TeX (with few errors)

TEX Utilities

Babel.Zip	164k	23-02-93	Spraakverwarring in LaTeX
Emsy.Zip	7k	26-02-91	support files voor emTeX
Essentia.Zip	13k	23-02-93	Een paar basisbegrippen in TeX op een rij
Helppc21.Arj	263k	27-07-92	bij texhelp2.arj
Info.Zip	42k	24-01-92	docs
Makeindx.Zip	49k	26-02-91	maak index in LaTeX
Myfonts.Arj	929k	01-02-93	font verzameling Henk de Haan
Nfss.Zip	87k	03-01-92	Macros voor New Font Selection Scheme
Pmtex21E.Zip	53k	22-02-93	TeX shell voor OS/2 Presentation Manager
Tex-Faq.Zip	19k	23-02-93	Questions, questions... and answers!
Tex-Inde.Zip	84k	22-02-93	STY files catalogus uit Internet
Texbook.Arj	468k	13-02-93	DE manual voor TeX-neuten on disk
Texcad.Zip	113k	10-02-92	maak moeilijk eenvoudig, emTeX
Texedit.Arj	223k	16-04-91	Duitse TeX-minnende editor
Texhelp2.Arj	25k	20-02-93	TSR LaTeX Help van Han-Kwang Nienhuys
Wp2Latex.Zip	38k	23-04-90	strip wp nonsens van wp files
Bibdb10.Arj	126k	21-01-92	
Bibtex.Zip	116k	26-02-91	
Lkurz.Zip	33k	26-02-91	
Helptex.Arj	18k	05-02-92	
Texhelp1.Arj	15k	20-03-93	
Slitex.Zip	210k	01-02-92	
Tex-Nl.Arj	8k	21-04-93	
Files.Bbs	1k	15-05-93*	
Pictex.Zip	43k	26-02-91	
Pkedit.Zip	51k	26-02-91	
Macroind.Exe	96k	09-05-93*	A Catalogue of TeX macros (version 1.06)
Jemtex2.Zip	430k	15-05-93*	Japanese TeX Fonts & Macros - Version 2.00

BOOKS

...

A Catalogue of TEX Macros

version 1.06

David M. Jones

MIT Laboratory for Computer Science
Room NE43-316, 545 Technology Square, Cambridge, MA 02139, USA
dmjones@theory.lcs.mit.edu

22 December 1992

Keywords: TEX, LATEX, AMS-TEX, AMS-LATEX, macros,

Introduction

This is a catalogue of TEX macros. Its scope includes all macros that are available via anonymous ftp or mail-server or some similar mechanism. Commercial packages will be included only if a full Catalogue entry is supplied to me by the vendor.

In this, its first incarnation, the Catalogue is heavily slanted towards the ymir and SHSU archives, in the sense that most of the entries have been drawn from the packages at those sites. Although I have included a significant amount of material from the Stuttgart archive, I've fallen far short of exhausting its wealth of macros, and the excellent Aston archive is hardly represented at all. I hope to remedy both of these shortcomings in the near future.

In addition to the archives just mentioned, a number of smaller, more specialized archives are also represented. For example, e-math.ams.com is the home of all the official AMS macro packages, including AMS-TEX, AMS-LATEXand the AMSFonts package. Similarly, dhdspri6.bitnet is the home of all official Springer-Verlag macros.

Since the Catalogue is devoted to macros, fonts and special-purpose programs are mentioned only when they are necessary to explain the purpose of a set of macros. For a list of fonts, I recommend Liam R. E. Quin's list of metafonts [1]. For a list of TEX implementations and other programs, see Bobby Bodenheimer's frequently-asked questions list [2] and Guoying Chen's supplement [3].

Each entry is divided into a number of fields, with the following functions:

- **Name:**
  The name of the macro package, usually the same as the name of the file containing it.
- **Description:**
  A short (usually 1-3 line) description of what the package does.
- **Keywords:**
  A list of keywords to facilitate searching for special-purpose macros, as well as to help describe the macros. A glossary of keywords can be found at the end of the file.
- **Archives:**
  A list of archives where the package can be found. Whenever known, the primary distribution site is marked with an asterisk. This list is not meant to be comprehensive. Instead, priority is given to the ymir, Aston, SHSU, Stuttgart and Utrecht archives. When the archive is well-known and has a significant collection of macros, I refer to it by an abbreviated name and give complete information about the archive in the 'List of TEX Archives' at the end of the Catalogue. In many cases, however, an ftp site serves as the home of only one package, in which case I include the full name of the archive in the 'Archives' field and don't include a full listing for it below.
- **Author:**
  The name and address (preferably electronic) of the author of the package. All email addresses, including JANET addresses, have been normalized to little-endian order.
- **Bugs to:**
  The address where bug reports should be sent, if this is different from the address(es) listed in the Author field. Currently (version 1.06 of the TEX Macro Index), this is only used for files in the standard LATEX distribution. These files are being maintained by Frank Mittelbach and Rainer Schöpf, but all bug reports should be sent to latex-help@cs.stanford.edu.
- **Latest Version:**
  The date and/or version number of the latest release of the package.
- **Supported:**
  Whether or not the package is supported, that is, whether the author wants to receive bug reports and/or comments on the package. Note that in most cases the author of a 'supported' package does not promise to fix bugs or answer questions about the package. The only promise is that the author

won't get mad if you send a bug report. In addition, it means that the author would like to hear about any improvements or additions to the macros. In fact, some authors only want to hear about improvements: they will accept bug fixes but will not necessarily answer bug reports.

Needless to say, the Catalogue is bound to contain numerous small errors and omissions and, most likely, not a few large ones. I've tried to indicate some of the larger gaps that I am aware of, and I'll be working to close these gaps as time permits. However, I would appreciate hearing about any errors or omissions in the information below. I would especially appreciate it if authors of macros would send me full index entries for their packages. I'll continue creating entries on my own, but the task will be greatly simplified if I receive ready-made index entries, rather than having to retrieve the files and process them by myself. I'll attempt to answer all mail sent to me, but please keep in mind that it might take me some time to do so.

Whenever possible, I asked the authors these packages to verify the information listed. I'm grateful to those who responded. In a few cases I didn't have time to follow up on the information that authors supplied me, but, again, I'll do this as soon as time permits.

I'd like to thank the following people for reading a preliminary version of the Index and commenting on it:

Ravinder Bhumbla
David Carlisle
Steve Fisk
Graham O'Neil
Eric Schenk
Dave Steiner
Dominik Wujastyk

They provided me with numerous helpful comments and suggestions. Unfortunately, lack of time once again prevented me from implementing all of their ideas for this first release of the Catalogue, but I will be working on them for future versions of the Catalogue.

My thanks also goes out to George Greenwade and Nelson Beebe for helping me get connected to the community of TEX archivers and for answering numerous questions.

Finally, I extend my special thanks to Barbara Beeton for her constant support and encouragement. Without her always gentle nudges, this project would probably have become moribund on a number of occasions.

How to retrieve the TEX Macro catalogue

• `archive.cs.ruu.nl` (Netherlands)

How to get `TeX-index.Z` from the archive at Dept. of Computer Science, Utrecht University:
by FTP: (please restrict access to weekends or evening/night (i.e. between about 20.00 and 0900 UTC)).

```
ftp archive.cs.ruu.nl [131.211.80.5]
user name: anonymous or ftp
password: your own email address
  Don't forget to set binary mode if
  the file is a tar/arc/zoo archive,
  compressed or in any other way
  contains binary data.
get TEX/DOC/TeX-index.Z
```

by mail-server:

send the following message to
`mail-server@cs.ruu.nl`

```
begin
path john@highbrow.edu
       (PLEASE SUBSTITUTE
          *YOUR* ADDRESS)
send TEX/DOC/TeX-index.Z
end
```

The path command can be deleted if we receive a valid from address in your message.

• `theory.lcs.mit.edu` (USA)

The TEX Macro Catalogue is available via anonymous ftp and mail server from `theory.lcs.mit.edu` [18.52.0.92] in the file `TeX-index` in the directory `pub/tex`. To retrieve it by mail server, send a message to `archive-server@theory.lcs.mit.edu` containing the following line

```
send tex TeX-index
```

The Catalogue is also available in compressed, zip'ed and zoo'ed format in the files `TeX-index.Z`, `TeX-index.zip` and `TeX-index.zoo`, respectively. Note that if you want to request one of the compressed files by mail server, you'll have to specify a method of ASCII encoding by including one of the following lines in your mail message:

```
encoder btoa
encoder uuencode
encoder rscs
```

References

[1] Complete list of all metafont-format fonts in the world. Liam R. E. Quin <lee@sq.sq.com>. Posted bi-monthly to `comp.text.tex` and `comp.fonts`.

[2] TEX, LATEX, etc.: Frequently Asked Questions with Answers [Monthly]. Bobby Bodenheimer. Posted monthly to `comp.text.tex`. Also available on most of the major TEX archives.

[3] Supplement to the Frequently Asked Questions. Guoying Chen. Posted monthly to `comp.text.tex`. Also available by anonymous ftp from `cs.nyu.edu` (128.122.140.24) in the `pub/tex/tex.supplement`.

Table of Contents

The Catalogue is divided into a rough hierarchy based on macro package. I hope to refine the division in the future, but for now here is the list of headings:

Examples

```
Name: amstex.tex
Description: AMS-TeX is the American Mathematical
    Society's macros for simplifying the typesetting of
    complex mathematical material. AMS-TeX is meant to
    be used with plain TeX. For corresponding support
    for LaTeX, see AMS-LaTeX.
Keywords: AMS-TeX, math
Author: American Mathematical Society
        <Tech-Support@math.ams.org>
Supported: yes
Latest Version: v2.1, 5 Apr 1991
Archives: e-math*, ymir, utrecht, aston, stuttgart
Note: The AMS-TeX distribution includes a readme file,
      a user's guide (amsguide.tex) and installation
      notes (amstinst.tex).
See also: AMS-LaTeX

....
....
....

Name: artikel1.sty (.doc)
Description: LaTeX document style for producing an article
    compatible with Dutch typesetting conventions,
    design 1. Compatible with article.sty.
Keywords: LaTeX, article, Dutch, NTG
Author: Victor Eijkhout <eijkhout@cs.utk.edu>
Supported: yes
Latest Version: v1.18, 3 Dec 1991
Archives: tex-nl*, ymir
Note: Part of the Dutch national LaTeX effort. Requires
    ntg10.sty, ntg11.sty, ntg12.sty (q.v.).
See also: artikel2.sty, artikel3.sty, boek.sty, brief.sty,
    rapport1.sty, rapport3.sty

....
....
....

Name: badges.tex
Description: plain TeX program for making name tags.
Author: Piet Tutelaers <rcpt@urc.tue.nl>,
        <rcpt@heithe5.bitnet>
Supported: ???
Latest Version:
Archives: ymir
Note: Package includes badges.ps, badges.readme
      and addresses.tex.

....
....
....

Name: cassette.tex
Description: plain TeX macros to typeset labels
    for standard Phillips audio cassette boxes.
Keywords: plain TeX, cassette label
Author: David Strip <drstrip@isrc.sandia.gov>
Supported: yes
Latest Version: 1991
Archives: ymir
Note: Uses, but does not require, two special fonts,
      dolby and cdlogo.
See also: tape.sty

....
....
....

Name: comment.sty
Description: Macros for commenting out sections of
    documents in plain TeX or LaTeX.
Keywords: plain TeX, LaTeX, comment
Author: Victor Eijkhout <eijkhout@cs.utk.edu>
Supported: yes
Latest Version: v2.0, 19 Jun 1992
Archives: stuttgart, shsu
See also: comment.tex, verbatim.sty

....
....
....

Name: dcolumn.sty (.doc)
Description: LaTeX style option for producing tabular
    entries aligned on a decimal point.
Keywords: LaTeX, tabular, array, decimal
Author: David Carlisle <carlisle@cs.man.ac.uk>
Latest Version: 1.01, 12 June 1992
Archives: stuttgart*, shsu, aston
Supported: yes
Note: Requires array.sty.  Documentation requires
      Mittelbach's doc.sty.
See also: decalign.sty, dectab.sty

....
....
....

Name: Elsevier.txs
Description: TeXsis style file for North Holland
    publications.
Keywords: TeXsis, proceedings
Author: Eric Myers and Frank E. Paige
        <texsis@lifshitz.ph.utexas.edu>
Supported: yes
Latest Version: v2.14, 9 Jul 1991
Archives: lifshitz.ph.utexas.edu* [128.83.131.57]

....
....
....

Name: manpage.sty
Description: LaTeX document-style option for producing
    UNIX manual pages.
Keywords: LaTeX, manpage
Author: Rong Chen <rchen@cs.uiuc.edu>
Supported: yes
Latest Version: v1.03, 11 Jun 1992
Archives: shsu*, ymir
Note: Includes a sample file appended to the end of the
      style file.
See also: unixman.sty

....
....
....

Name: nederlands.bst
Description: BibTeX style based on the requirements
    of the department of dutch of the University of
    Nijmegen, being more or less the standard
    practice among (dutch) historians and linguists.
    It supports citations of the form "Jansen
```

```
    (1992)" and "(Jansen 1992)" depending on whether
    the citation is used as a noun or as a
    parenthetical note. Bibliography entries are
    sorted by authors name following Dutch (and
    german) rules (regarding the 'von' part of the
    name).
Keywords: BibTeX, bibliography
Author: Werenfried Spit <Spit@vm.ci.uv.es>,
        <Spit@ific.uv.es>
Supported: yes
Latest Version: v2.2, 22 Apr 1992
Archives: archsci.arch.su.oz.au* [129.78.66.1], shsu
Note: remarks regarding the layout are welcome.
    The main aim of the style is to treat names
    regarding to dutch (and german) rules.
    Variations of style could be provided. This is
    part of the Harvard family of bibliography
    styles, which is described in "The Harvard
    Family of Bibliography Styles," distributed with
    the package in the file harvard.tex.
See also: agsm.bst, dcu.bst, kluwer.bst, harvard.sty

....
....
....

Name: newsletr.tex
Description: plain TeX macros for typesetting a
    newsletter. Includes support for multiple
    columns.
Keywords: plain TeX, newsletter, multi-column
Author: Hunter Goatley <goathunter@wkuvx1.bitnet>
Supported: yes
Latest Version: v01-014, 3 Jun 1989
Archives: shsu*

....
....
....

Name: overcite.sty
Description: LaTeX document-style option for compressing
    lists of three or more consecutive numbers into one
    number range and displaying them as a superscripted
    citation. E.g., "foo [1,2,3]." becomes "foo.^{1-3}".
Keywords: LaTeX, citation, bibliography
Author: Donald Arseneau <asnd@jack.triumf.ca>,
        <asnd@triumfcl.bitnet>
Supported: yes
Latest Version: 1991
Archives: shsu*, utrecht
See also: citesort.sty, drftcite.sty, rangecite.sty

....
....
....

Name: psfig.tex
Description: TeX macros for including Encapsulated
    PostScript graphics in a TeX document.
Keywords: plain TeX, AMS-TeX, LaTeX, epsf,
    PostScript, dvips, dvi2ps
Author: Trevor J. Darrell <trevor@media.mit.edu>
Supported: yes
Latest Version: v1.9, 17 Oct 1991
Archives: whitechapel.media.mit.edu* [18.85.0.125],
          shsu, aston
Note: Also available as a LaTeX style file, psfig.sty.

....
....
....

Name: revtex.sty
Description: REVTeX is the official LaTeX macro package
    for preparing manuscripts for producing manuscripts
    for the American Physical Society's Physical Review
    A, B, C and D, as well as Physical Review Letters.
    (Many AIP journals also accept REVTEX manuscripts.)
Keywords: LaTeX, REVTeX, Physical Review, physics,
          journal, publisher style
Author: American Physical Society <mis@aps.org>,
        <mis@apsedoff.bitnet>
Supported: yes
Latest Version: v3.0, 27 Oct 1992
Archives: shsu*, stuttgart*, labrea*, ymir, utrecht, aston
Note: The REVTeX package includes a README file, a users
      manual (apguide.tex) and three sample files
      (smplea.tex, smpleb.tex and smplec.tex).
See also: aps.sty, aps10.sty, aps12.sty, eqsecnum.sty,
          preprint.sty

....
....
....

Name: scorecard.tex
Description: plain TeX macros for printing a baseball
    scorecard for one team, as well as summary
    statistics boxes (Pitcher's stats, Offense, Defense,
    and Pitching miscellanea, a Notes section, and a
    line score).
Keywords: plain TeX, baseball
Author: Matthew Wall <matt@cs.brandeis.edu>
Supported: ???
Latest Version: v1.1, 28 Apr 1989
Archives: ymir

....
....
....

Name: shadebox.tex
Description: Macros for printing text on top of shaded boxes
    on PostScript devices.
Keywords: generic, PostScript, dvips
Author: <Leo@vaxc.cc.monash.edu.au>
Supported: ???
Latest Version: 16 Jan 1992
Archives: shsu

....
....
....

Name: symbols.tex
Description: A listing of all the standard LaTeX math
    symbols and the AMS symbols.
Keywords: LaTeX, symbols, AMSFonts
Author: David Carlisle <carlisle@cs.man.ac.uk>
Latest Version: 2.00, 12 June 1992
Archives: shsu*, aston
Supported: yes
Note: If used with the NFSS, requires AMSFONTS2.1, and
      amssymb.sty. If used without the NFSS, does not show
      the AMS Fonts.

....
....
....

Name: testfont.tex
Description: TeX program for displaying font tables.
Keywords: plain TeX, fontchart
Author: Donald E. Knuth
Supported: ???
Latest Version:
Archives:
See also: fontchart.tex, fontbl.tex

....
....
....

Name: texsis.tex
Description: \TeX sis is a collection of \TeX\ macros
    for typesetting physics documents such as papers and
    preprints, conference proceedings, books, theses,
    referee reports, letters, and memos. \TeX sis macros
    provide automatic numbering of equations, automatic
    numbering and formatting of references, double
    column formatting, macros for making tables and
    figures, with or without captions, including tables
    with horizontal and vertical rules. \TeX sis
    supports a wide variety of type sizes and a number
    of specialized document formats, and it even
    includes macros for making form letters for job
    applications or letters of recommendation.
Keywords: TeXsis, physics, fmt
Author: Eric Myers and Frank E. Paige
        <texsis@lifshitz.ph.utexas.edu>
Supported: yes
Latest Version: v2.15, 3 Aug 1992
Archives: lifshitz.ph.utexas.edu* [128.83.131.57]
Note: Includes a user's manual (Manual.tex), a UNIX man page
      (texsis.1), installation notes (INSTALL and
      Install.tex), a README file, a GNU emacs mode
      (texsis.el) and an example paper (Example.tex).
      For support for an index, see index.tex.

....
....
....

Name: tugboat.sty
Description: plain TeX macros for preparing an article
    for TUGboat, the Communications of the TeX Users
    Group.
Keywords: plain TeX, TeX Users Group, TUGboat,
Author: TeX Users Group <TUGboat@math.ams.org>
Supported: yes
Latest Version: v1.11, 8 Mar 1992
Archives: labrea, stuttgart, shsu
Note: Requires tugboat.cmn
See also: ltugboat.sty, tugproc.sty, ltugproc.sty,
          tugboat.cmn
```

A way to ensure the future of TEX: make its use easier on low-cost machines*

Michel Lavaud

C.N.R.S.
GREMI, Université d'Orléans, 45067 Orléans Cedex, FRANCE
lavaud@avion.univ-orleans.fr

Abstract

The PC is the cheapest computer and the most widespread one in the scientific community. Faced with commercial scientific word-processors that are improving steadily in wrong directions, it is urgent to make the use of TEX easier on the PC, to ensure its future and avoid costly dead-ends to researchers. We have designed a program, AsTEX, that allows to create easily multi-author scientific documents in TEX or LATEX on PCs. It provides an on-line hypertext help and a multi-level assistance in typing LATEX code. It allows to display and modify very easily the structure of a document, to archive and retrieve files related to it, to perform numerical and formal computations from the document and include automatically the results, to create LATEX tables from worksheets or databases of formulas. It processes electronic mail and files sent by list servers for a better use of information and eases considerably the use of anonymous ftp and archie servers by local archiving of selected informations.

Keywords: TEX, LATEX, back-end, front-end, scientific word-processor, hypertext, tree, link, multi-author document, file manager, numerical computation, formal computation, worksheet, database, e-mail, PC, notebook, OS/2.

1 Introduction

During the last years, there appeared several WYSIWYG scientific word-processors on PCs that allow to type easily short texts including mathematical formulas, and can give very nice-looking outputs.

These programs are not adapted to scientific research. Indeed, formulas are stored either as proprietary code incompatible with other software, other machines and preceding versions of the program itself, or they are stored as bitmap images, which gives rise to prohibitively large files even for small articles. This makes collaboration by electronic means very difficult, maintenance of personal scientific documents over the life-cycle of a research project very problematic, coupling to scientific software such as Maple or Mathematica almost impossible and building of databases of scientific documents with formula searching capabilities completely hopeless.

Nevertheless, they have a high seducing power and it is very important and urgent to make the use of TEX on PCs as easy as possible. This is very important because the PC is the cheapest machine and, by far, the most widespread (80 millions in use today, 20 millions more in 1991). This is also very urgent because the other products are getting better and better, and it might happen that they become a de facto standard on small machines, despite their considerable long-term drawbacks, if *easy-to-use* TEX-based Scientific Word-Processors are not available rapidly on PCs.

In section 2, we define what we mean by a TEX-based SWP (Scientific Word-Processor). In section 3 we argue that, in the future, SWPs will be used by researchers, no more by secretaries, so that "easy-to-use" TEX-based SWPs will have to ease all the tasks required to *create* scientific documents, not only to *type* documents already created on paper. This will have deep consequences on the structure of data manipulated by these programs, on the structure of the programs themselves, and probably also on the way of entering formulas. In section 4, we argue further that they must run on PCs, to be accessible to everybody, and to be usable on future pocket-size notepads. We propose to call this new kind of programs adapted to research work, "Scientific Document-Processors".

We have designed a prototype of Scientific Document-Processor, AsTEX. Version 1 was presented at the preceding EuroTEX conference in Paris [5]. In section 5, we recall the main possibilities of version 1, then we describe in some detail improvements brought in by version 2.

The readers who want to have at once an *overview* of the possibilities of AsTEX, can go directly to section 5.2.

*Presented at EuroTEX '92, September 14–18, Prague, Czechoslovakia.

2 What is a TEX-based Scientific Word-Processor?

TEX is not a Scientific Word-Processor. It is a typesetting *language* plus a *compiler* of this language. In a much cited conference[12], Donald Knuth said:

> [...] I expect that there will be a continual development of "front ends" to TEX and Metafont, as well as a continual development of "back ends" of device drivers that operate on the output of the systems [...]

We define a *TEX-based SWP* as any set of these three ingredients (front-end + TEX compiler + back-end), made to cooperate together.

2.1 Back-ends to TEX

In the early eighties, it was almost impossible to use TEX if you were not close to a big computing center[1]. The problem of building back-ends to TEX for cheap machines was crucial.

Now there are several Public Domain back-ends to TEX of professional-quality that run on PC: emTEX by E. Mattes, dvips by T. Rockiki and Ghostscript by L.P. Deutsch, among others. With these programs, it is possible to create documents integrating mathematical and chemical formulas with graphics and to output them on low-cost devices (VGA screen, dot-matrix or ink printers and cheap laser printers). Color Postscript illustrations and color texts in TEX [2] can be included and output on a color screen or a color-ink printer.

The problem of building good back-ends to TEX on PC can be considered as largely solved now[2]: Anybody can produce scientific documents of *much better quality* with TEX than with ordinary word-processors such as Word or Wordperfect – and at a *much lower cost* since the same hardware can be used and the whole software is Public Domain.

2.2 Front-ends to TEX

The availability of powerful Public Domain back-ends to TEX on PC means that it is *possible* to produce high-quality documents with TEX on cheap systems. This does not mean however that it is *easy*. Making TEX easy to use is a problem of designing a good front-end, and coupling it to TEX compiler and to existing back-ends in an efficient way.

While it was *necessary* to build powerful back-ends to be able to use TEX, there is no such necessity for front-ends. Indeed, any text editor is a valid front-end, even the simplest ones (e.g. `edlin`, or the built-in editor of dBase 3, etc...). This has the well-known considerable advantage that it is possible to include mathematical formulas in any non-scientific software (e.g. database managers). But this has the disadvantage that there has been no strong pressure to build front-ends that ease the use of TEX, as long as existing commercial products gave so poor outputs as compared to TEX.

2.3 An example of a TEX-based SWP: emacs customized for TEX

The most well-known and widely used front-end to TEX on workstations is the Public Domain editor GNU `emacs` of R. Stallmann. The PD macro-package `auc-tex` of K. Thorup transforms `emacs` into a TEX-based SWP: It lets you compile a TEX or LATEX file and preview or print it from inside `emacs`. It let's you browse through the errors TEX reported, while it moves the cursor directly to the reported error, and displays some documentation for that particular error. This works even when the document is spread over several files. It automatically indents LATEX source when typing, or indent and format an entire document. It has also a special outline feature, which helps 'getting the overview' of a document.

`emacs` has been very recently adapted to run on the PC under OS/2. This restricts its availability to the newest 80386-based machines with lots of memory and harddisks with large capacity, but it is nevertheless a big step forward.

3 What must be required of the SWPs of to-morrow?

3.1 They must be designed for researchers

A few years ago, almost all researchers wrote their scientific articles by hand, and had them typed by secretaries. In the future, all researchers will type themselves their articles directly to screen, as soon as adequate SWPs will be available, because this will allow them dramatic gains in time.

This is obvious if we recall the evolution in numerical computations: about twenty years ago, researchers used to write their computer programs on cross-ruled sheets of paper and had them typed on punched cards by specialized typists. Then they gave the pack of punched cards to an operator and waited until the operator bring back the results. As soon as teletype terminals became available, each researcher typed himself his own programs directly to screen and submitted them himself to the computer: although he did alone the work of three persons, he was able to gain much time because typing and submitting his programs himself eliminated *waiting delays*, and typing programs directly to screen

[1] When I tested TEX for the first time, I had to submit files to a distant computing center through a modem, with no possibility to preview the result locally. The only way to track errors was to wait for two or three days until the printed output arrived by regular surface-mail...

[2] This does not mean that existing back-ends cannot be improved any further. It would be very nice, e.g., to have a utility to do scrolling and zooming on Postscript code. Up to now, this is possible only in the X-Window environment with `ghostview`.

(i.e. without writing them first by hand) avoided him useless *duplication of tasks*.

3.2 They must allow to CREATE – not only TYPE – scientific documents

Almost all scientific word-processors are oriented towards *typing* scientific texts, not *creating* them, i.e. they are adapted to the way of working of secretaries, not researchers. With Word and Wordperfect for example, the Table of Contents is produced *from* the typed document, as a by-product of the typing process. Modifying the structure of large multi-author documents is very long and must be done by loading many files, selecting blocks of text and moving them from one place to another by cut and paste.
This approach is quite acceptable if the researcher writes his document by hand and then gives it to a secretary for typing: in general, it is quite close to its final form and only few modifications of structure are necessary. But this approach is very clumsy if he types the document himself. Indeed, the correct way of writing a document from the scratch is to build *first* the table of contents and *then* fill in the different sections with text, not the reverse.

A SWP adapted to the way of working of a researcher must allow to *display* and *modify* the structure of the document very easily. This ensures that even very long multi-author documents will be coherent and logically organized, and that they can be easily reorganized at any time, to improve their clarity and coherence.
This has deep consequences on the data types that are manipulated: a document is no more a long sequence of (possibly meaningless) characters and formatting codes typed in one (or a few) window. It is a *tree* whose leaves are coherent blocks of information of various nature (paragraph of text, worksheet of numerical results...).

3.3 They must allow to perform research tasks from the document

The researcher has also many other tasks to perform, while creating his document, that a secretary would not have to do. For example, he may have to manage hundreds of files related to the document (chapters, computer programs, numerical results, e-mail, illustrations...), created by several people on different media. He may have to perform *numerical* or *formal computations* and merge results into his document. He may want to manage the results into *worksheets* or *databases*, and include excerpts into tables. This can be very long if it is done by cut and paste, and updating can be a nightmare if tables are interdependent.

A good SWP must ease all these tasks and allow the researcher to perform them *from the document* and *automatically*, because this can save him much time. On Unix workstations, a great deal of work has already been done to integrate tools useful for research into GNU `emacs`. For example, it has been customized to include automatically Mathematica outputs in LATEX into the text [3]. But much remains to be done, in particular as concerns integrating worksheets and databases into documents.

3.4 They must be TEX-based

Since future SWPs will be used mainly by researchers, they *must* be based on TEX, as suggested by the preceding remark about the possibility to get outputs of scientific software in LATEX. Indeed, because it is of highest quality and is in Public Domain, TEX is now *THE privileged common language* of the international scientific community, for communication of scientific results by electronic means, as *English* is the privileged language for communication of scientific results by usual means (voice and paper). Many other reasons can be given, and has been given elsewhere, that are more or less elaborations on or consequences of these two fundamental reasons. Let us enumerate some of them here:

- TEX is sufficiently *powerful* to deal with every complexity likely to be encountered in mathematical typesetting [10];
- It gives superb outputs;
- It is easily understandable by everybody (instructions are written in plain English, not undocumented numerical codes);
- It is *extensible*, as natural languages: missing concepts / words can be created at will. In computer science terms, it is *programmable*: what is not included can be added by writing new procedures / macros;
- It is in the *Public Domain*: you have not to pay hundreds of dollars to software corporations, as a preamble to "speak electronically" with your colleagues or write your scientific articles or read scientific files sent by colleagues: what research would become if researchers had to pay to English or US governments, to be able to speak in English? *Freedom of communication* is a sine qua non condition for research, as freedom of speach and writing is for democracy;
- It is *transportable*, i.e. independent of the platform and available on all of them;
- It is *stable*, i.e. it does not vary every six months, at the mercy of new versions;
- Many scientific programs can give *outputs in LATEX* (Maple, Mathematica, Macsyma, Reduce...);
- Huge *databases* of scientific articles exist in TEX: e.g. all of Mathematical Reviews going back to 1959 (about one million records) is stored online in TEX format in the MathSci database[11];
- It is used by the biggest scientific book editors;
- Many scientific journals accept compuscripts in TEX and provide style files;
- It travels very well on networks;
- Maintenance is *world-wide*. There are many discussion lists in many languages to help solving problems.

3.5 They ought to have a WYSIWYG equation editor capable of importing TEX formulas

Several commercial scientific word-processors and equation editors on PCs allow to type mathematical equations in WYSIWYG mode, and export them into TEX or LATEX files (Mathor, Mathdesign, PreTEX, Grif...).

They can ease typing *portions* of documents with many formulas in TEX. But they cannot be considered as satisfactory TEX-based SWPs, nor can they be used as WYSIWYG equation editors in a TEX-based SWP. Indeed, exporting TEX equations is the easiest part of the story. To be really satisfactory, a WYSIWYG scientific word-processor ought also to be able to *modify* – and thus to *import* – TEX formulas[3]. And this is very difficult because of the macro capacity of TEX: this implies that the import module contains most of the capabilities of a TEX compiler. To give a simple example, if the greek letter `\gamma` is redefined somewhere in the file or in anyone of the inputted files as the macro `\g`, the import module must be able to detect it. In fact, to be completely satisfactory, the import module ought to be able to import *any TEX and LATEX file*. Indeed, more and more scientific journals provide their own style files, with a bunch of special commands to write compuscripts that are submitted in TEX. For the compuscript to be consistent with this style file, the SWP must be able to "understand" the style file, i.e. to import it.

This is quite comparable to the situation for Postscript: virtually any software now is able to export in Postscript. But there are, up to now, only a few programs that can import *any* Postscript file. Indeed, building such a software amounts largely to build (the software part of) a Postscript printer. Moreover, no program is able to import any Postcript file *and* modify it interactively. The graphics programs that allow to modify interactively Postscript code, such as Adobe Illustrator or Corel Draw, do it on a very limited subset of Postscript commands.

For the time being, a satisfactory compromise would be an equation editor able to import TEX formulas containing only *standard* TEX, LATEX and AMSTEX instructions (i.e. `\g` would NOT represent the letter γ, but just the two characters \ and g). This would be a useful ingredient to build TEX-based SWP with WYSIWYG construction of formulas.

4 TEX-based SWPs must be available on PCs

4.1 Price

The PD and commercial implementations of TEX on workstations (GNU `emacs`, Publisher, Decwrite) are very nice. However, workstations are very expensive as compared to PCs and most students cannot afford buying color workstations and software to run on it[4]. Moreover, Unix is not for the freshman (except in computer science), DOS or OS/2 are more adapted.

Since students of today are the scientists of to-morrow, it is very important that TEX can be used in as good conditions as possible on the *cheapest machine* that is easily accessible to *all* of them, i.e. the PC. This is all the more important for East-European and Developing countries, because only this type of machine is available.

4.2 Transportability

Another considerable drawback with workstations is that most of them cannot be moved, either because of their large and heavy screen (although a Sparc notebook appeared recently), or because they are bound to a network (if you disconnect the machine you use at work and bring it home with a lorry, the odds are great that you will be flamed by colleagues who use a software installed on your machine with a floating license...).

PCs are not subject to these constraints and evolution is towards *portability*. Market studies have shown that next year, about 70 millions of compatible PCs will sell, half of them being *notebooks* and *pocket-size* machines. Among these, many will have a *sensitive screen* with the capacity to recognize words hand-written on screen. It seems reasonable to think that, in the near future, hand-written formulas could also be recognized: problems are still tough but certainly not insurmountable. It would be highly desirable that formulas be translated into TEX and not into proprietary code of Word, Mathdesign or whatever commercial SWP in vogue at this time. This requires that TEX-based SWP must be available on these machines, to exploit this facility.

Within a few years, high capacity flash memories will replace hard disks and PCs will have the size and thickness of a few sheets of paper. Outputs will be done to printers or networks via infrared emitters plugged into parallel port or PCMCIA interface of the notepad. One can imagine it will be possible to hand-write mathematical formulas and scientific articles on the sensitive screen of the notepad, and to output them automatically in TEX, a few seconds or minutes later, either on a local

[3] Any SWP may modify its *own* outputs in TEX, by storing its internal description of a formula into TEX comments, and exporting the comments with the TEX formula. The modification is made by importing the commented part of the formula. But this trick does not allow to modify outputs of *other* SWPs.

[4] These last years, the prices of workstations had decreased so much that it seemed probable that cheap workstations would replace PCs very soon. However, prices of PCs have decreased even more (40% in 1991) and, since the outburst of cheap notebooks last year and the new release of OS/2 this year, it seems very likely that PCs and workstations will continue to develop in parallel for a long time, as complementary machines.

printer or on the one of a foreign colleague, thousands of miles away!

4.3 They must run under OS/2 for best performance

It has been advocated recently that
'*workstations of the* SUN SPARC *class are necessary for TEX. The speed is necessary to process text of more than a few pages and a large screen with good resolution* (1100×950) *is just sufficient for the previewer*' [1].

In fact, a workstation is indispensable only when pure computation power is required, as for imaging or intensive numerical computing. When doing word-processing, the machine is idle most of the time: it just loops, waiting for you to type characters. What is required is not the speed of a SPARC machine but its *multitasking* and *interprocess communication* capabilities, to make the three components of a TEX-based SWP (front-end + TEX compiler + back-end) run concurrently and cooperate, without having to reload them each time they are called to work. This is the real clue for cutting down the waiting delays for compiling, previewing and printing that plague the use of TEX on PCs running DOS[5]. With the new release of OS/2 2.0 and the new capabilities of the emTEX package, these problems will be solved very soon [9]). As concerns the size of the screen, what is required is not a large one, but a previewer with a *zoom* and *scroll* function (as in CAD software, for example). Indeed, text written in the smallest size printed by TEX is unreadable even with the resolution of a workstation, when a whole page is displayed. These functions exist in the emTEX previewers for DOS and OS/2.

In short, the problems that have been encountered in the past when using TEX on the PC came mainly from the limitations of DOS and old previewers, not from the hardware. Using TEX on a 80386 notebook with OS/2 and emTEX is certainly a better choice now than using it on a workstation, because this cuts price but not performance, and the machine can be carried anywhere.

5 AsTEX version 2

We have developed a program, AsTEX[6] that fulfills the requirements defined in the preceding sections (except for the coupling to a WYSIWYG equation editor, that do not exist up to now). It provides an easy-to-use TEX-based SWP, adapted to scientific research. Version 1 was presented at the preceding EuroTEX conference in Paris [5] and other places [6]–[7]. Version 2, that is currently in beta-test, brings in many improvements and adds new possibilities [8] that make it more powerful than Windows 3.1 and related programs (although their use is of course not excluded).

It is based on the kernel of public-domain back-ends to TEX already mentioned, and on a commercial program, *Framework* (from Borland) that is used as a basis to construct the front-end. On 80386 machines, it can be complemented by *Desqview* (from Quarterdeck) or *OS/2* 2.0 (from IBM).

In the sequel, we recall a few definitions and we give an overview of AsTEX's possibilities. Then, we describe the new possibilities of version 2. For a description of those already present in version 1 (in particular the hypertext file manager and the mechanism to export selectively subsets of objects), we refer to Refs. [5]–[8] for details.

5.1 Definitions

For AsTEX, a document is a *set of files related logically* and accessible from one of them (the master file) by loading into linked windows. The set of files has a *tree structure*, and each file is itself a tree whose leafs are *objects* of various nature (linked windows, texts, databases, worksheets, graphics, computer programs...). The files may be on different media and be created by various people on a network (see [5]).

External tasks required by TEX or by research work related to the document, can be run from a customizable toolbox, in windows that are created automatically. On 80386-based machines, windows are displayed *simultaneously*, each occupying a part of the screen ('space-windowing'). On 8088- or 80286-based machines, windows are displayed *alternatively*, each one occupying the whole screen ('time-windowing'). The important point is that AsTEX works on *all PCs*, in the same way modulo this distinction between space- and time-windowing.

5.2 Overview of AsTEX's possibilities

- *Hypertext file manager*:
  - Immediate access to thousands of files through hierarchy of explicit titles.
  - Easy modification of the structure of very big multi-author documents.
- *Scientific computations*:
  - Numerical (e.g. Fortran): compilation / execution run directly from text of the document.
  - Formal (e.g. Maple): results automatically included in text, worksheets, databases.
  - Live links to data files.
  - Interdependent worksheets.
- *Scientific text processing*:
  - Mathematical and chemical formulas displayed with a single keystroke.
  - Preprocessor of LATEX structure.
  - Creation of LATEX tables from worksheets or databases of formulas.

[5] Grouping the three components of a TEX-based SWP into one huge software would be unrealistic.
[6] AsTEX stands for Assisted TEX.

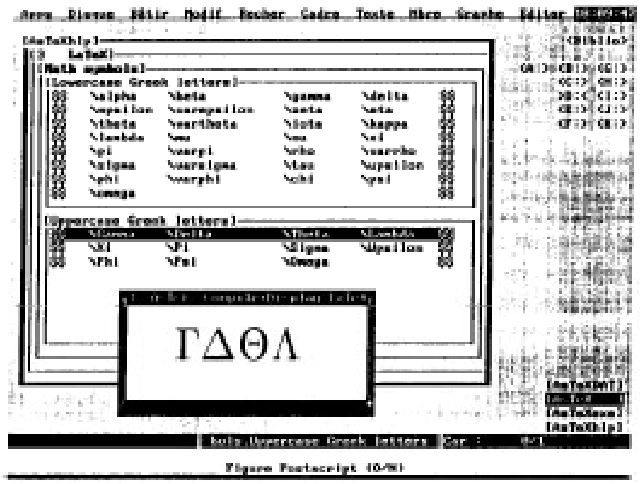


Figure 1: *LATEX symbols can be displayed with a keystroke*

- *Electronic mail*:
  - Hypertext archiving of messages.
  - Automatic extraction of messages from files issued from discussion lists.
  - Local archiving of informations about ftp and archie servers to speed up connections.
- *Hypertext help* for AsTEX, LATEX, emTEX, Ghostscript, graphs used in physics . . .

5.3 Hypertext file manager

AsTEX's hypertext file manager (cf. Ref. [5]) has been improved to include inheritance of relative and absolute links of parent windows. This allows easy transfer of complete sub-hierachies of objects from a machine to another (e.g. from a desktop to a notebook and reversely), for later improvement at home, in a public library etc. . .

5.4 Creating LATEX files

AsTEX helps creating LATEX files at *export* time and at *typing* time.

Sectioning commands

With AsTEX, no sectioning commands have to be written: they are generated automatically at export time. Any subset of selected objects is exported into a LATEX file with correct sectioning commands, according to the relative positions of the objects in the tree of the document. The export mechanism has been described in detail in [5]. As a complement to this facility, version 2 inhibits the typing of sectioning commands. For example, if `\chapter` is typed, it is automatically erased and the message "Instruction `\chapter` forbidden" is displayed at the bottom of the screen.

This function implements, at the front-end level, an important function of SGML parsers: eliminate ill-structured documents, with a `\chapter` command inside a `\paragraph` command for example. This kind of error is not detected by LATEX.

Hypertext help

A very common criticism addressed to TEX by new users is the difficulty to get informations on it [13]. Many recent word-processors have on-line help and even sometimes on-line tutorials.

AsTEX provides an on-line help for LATEX in *hypertext* form. The first module is done out of the latest original `latex.tex` file, so that it is up-to-date and *exhaustive*. Other modules are available for style files and special symbols. These can be *previewed* in a separate window and selected and *copied* into the text from the help window (see Fig.1). This is useful for complicated instructions such as `\Longleftrightarrow`, `\rightleftharpoons` etc. . .

AsTEX provides also an on-line hypertext help for itself and for several other useful programs: emTEX, Ghostscript, Maple, Kermit, programs to compress files and to create or manipulate graphics files. In all the help modules, searching can be done by *subject matter* (via the hypertext mode) or by *keyword* (via the "Search a string" function of Framework).

Generation of environments

Environments can be generated automatically, by typing the keyword "`@e` " in the text (this keyword can be customized to any other keyword, for example `\env`). This activates a hierarchical menu, from which the suitable environment can be selected. Automatic indentation at any level is provided. For example, typing `@e` and selecting item `List` then subitem `enumerate` generates:

```
\begin{enumerate}
  ...
\end{enumerate}
```

where the underscore character indicates the position of the cursor. By typing once again `@e` and selecting items `List` and `itemize`, we obtain:

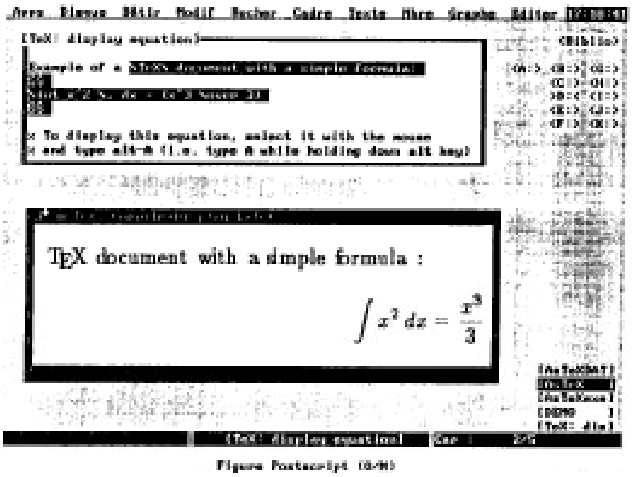


Figure 2: *Mathematical equations written in TEX can be previewed in a window.*

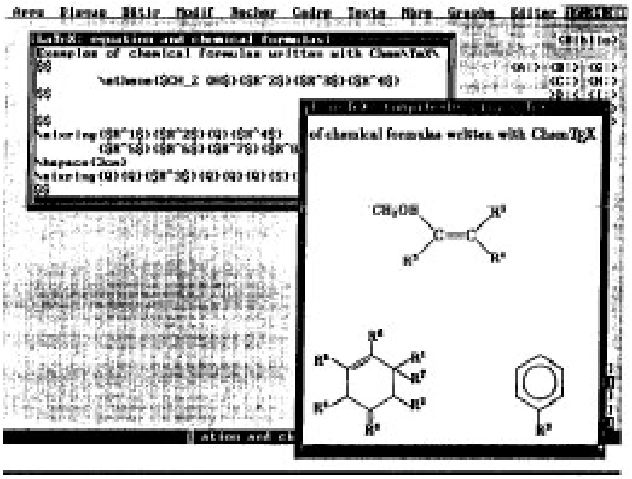


Figure 3: *Chemical formulas written with the ChemTEX package can be previewed as easily as mathematical formulas.*

```
\begin{enumerate}
   \begin{itemize}
     _
   \end{itemize}
\end{enumerate}
```

Transcoding of character attributes of Framework

AsTEX allows to transcode the attributes of characters of Framework (italic, bold, underlined, overstrike, indices and exponents). However, this facility is provided only to automatize the migration of files from Framework format to LATEX format and as an introduction to LATEX commands, for Framework users who are new to TEX. The transcoding is made interactively on the screen, in the subset of selected windows, and it ought to be done once and for all. Version 2 allows to performs the translation for selected, not-necessarily consecutive objects containing texts, worksheets or databases.

Displaying mathematical, chemical and formal equations

With AsTEX, mathematical equations are typed in native LATEX. They can be selected with the mouse or the key arrows and previewed by hitting a single key, in a window that is created automatically (see Fig. 2). Zooming can be done on the formula. In case of space-windowing with Desqview, the window can be moved and resized and the formula can be scrolled in the window with the mouse.

Chemical formulas can also be previewed in a window with a single keystroke (see Fig. 3). More generally, any portion of text can be debugged by selecting it and trying to preview it. A customizable prolog is automatically added, to allow using style files and personal commands defined in the current document.

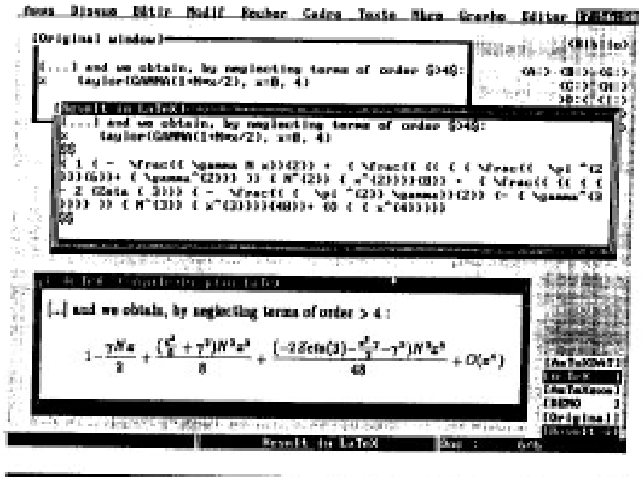


Figure 4: *Results of formal computations with Maple can be included as LATEX formulas in the text.*

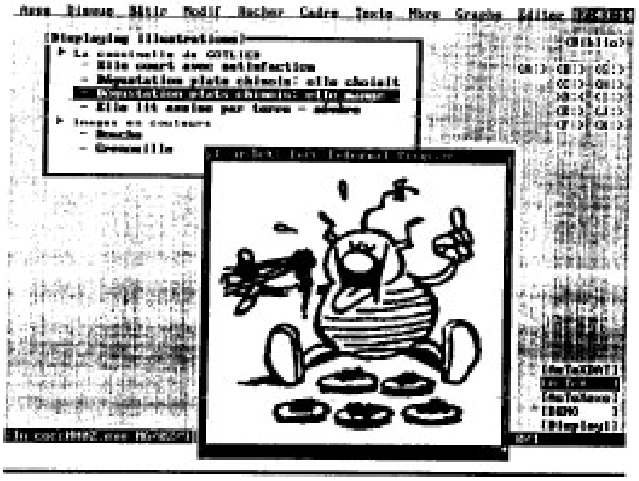


Figure 5: *Archiving and displaying external images.*

Equations written in the computer algebra language Maple can be *computed* and the result *previewed* in a separate window and included in the text as LATEX formulas, with a single keystroke (see Fig. 4).

It can be noted that the way used by ASTEX to preview mathematical formulas is analogous to the one used by Wordperfect 5.1. But Wordperfect uses `eqn` to describe formulas, which is much less powerful than TEX[7]. Moreover, Wordperfect does not know about chemical formulas nor computer algebra languages: Maple outputs must be retyped manually. This is time-consuming and very much error-prone, especially for long formulas.

5.5 Displaying illustrations and sounds

Internal images are created either with the drawing module of Framework or by a CAD program and imported as CGM files. They can be displayed in a window of Framework, and thus on any PC.

External images (i.e. images stored in any format other than CGM) can be previewed in a window[8] with a single keystroke (cf. Fig. 5).

Images (either internal or external) can be *archived hierarchically*. They can also be archived in databases or hierarchies of databases and displayed or previewed at any time from the document.

Musical sequences can also be archived and played from a document exactly as external images (cf. Fig. 6) and played at any time. Multimedia applications could also be run from a document in the same way, at least in principle (the real problem with multimedia is storage capacity and data compression).

[7] `eqn` is a system for typesetting mathematics devised by B. Kernigan and L. Cherry for Unix systems. Mathematical expressions are described in the `eqn` language and translated by the `eqn` program into `troff` commands, for final `troff` formatting.

[8] "Space-windowing" or "Time-windowing" is used, depending on the type of the machine (cf. section "Definitions" above).

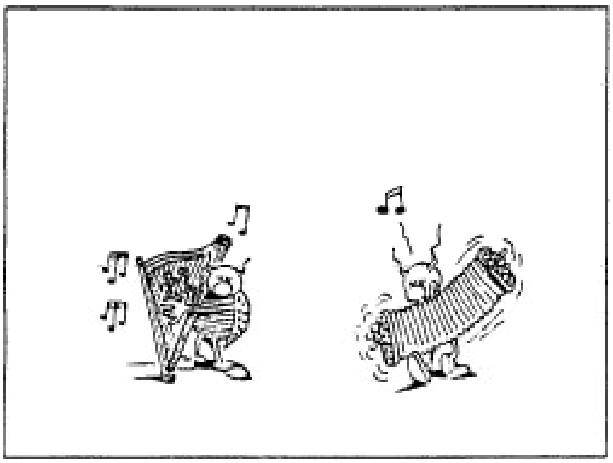


Figure 6: *Archiving and playing musical sequences.*

At last it is possible, on 80386-based machines, to run a computer program in a separate window from the document while displaying its code, as can be done on workstations or on Macintosh (this is impossible with Windows if the program output is graphical, as it is often the case in numerical simulations, studies about fractals etc. . .).

5.6 Electronic mail

Electronic mail is indispensable now for research. Messages related to a scientific document are part of it since they help creating it (even if they are not included into the final text in their raw form) and they must be managed by the Scientific Document Processor, as any other piece of text.

AsTeX's e-mail manager has been built by studying e-mail sent by a few selected discussion lists. Discussion lists can provide invaluable help. However, the back side of the medal is that you can be drowned into a flow of information and have no other possibility than reading your mail for the whole day and doing nothing else, or unsubscribe to interesting discussion lists and never see valuable information.

Moreover, if you decide to read mail and save it, you are very rapidly faced to the problem: "What is contained in such and such files" and the reverse one: "In which file did I store such and such information". Building a traditional database is too time-consuming, so you may once again have to choose between saving your mail (or at least the largest part of it) and being lost among thousands of files, and not saving it, so that useful informations may be irremediably lost.

AsTEX helps solving these problems in a very natural way by allowing to organize messages into a *hypertext database* inside the document. They are stored hierarchically, and organized as in a book. Information is retrieved either by navigating in the "book" by means of a hierarchy of titles or by using the "Search a String" function of Framework, over a subtree of messages (a "chapter" of the "book"). Any new message is archived at the most suitable place in the tree of messages, with a few keystrokes.

The advantage of this hypertext database is that you can add a new item in a blink of an eye, while it would be very time-consuming with a traditional database. Moreover, the hypertext database contains your expertise (you put the message at the correct place) and it can be queried efficiently by anybody. A traditional database requires no expertise to build (your secretary can do it), but the expertise has to be introduced at the questioning time (you must know what to ask, to get a pertinent information), so that it is useful mainly for experts.

AsTEX also provides a few functions to increase archiving speed: automatic *extraction of individual messages* from a file originating from a discussion list, and automatic *cleaning* of individual messages (only selected informations of the header are kept in the archived message).

A new function is under study, to help *regrouping related questions and answers* automatically, to optimize information extracted from discussion lists while minimizing the work (and time!) to do it. This could help solving the above-mentioned dilemma of receiving as much information as possible without being drowned into it.

5.7 Ftp and archie servers

Anonymous ftp provides an invaluable help to get various software and informations. However, using it must not overload the machines that provide this facility ("ftp is a privilege, not a right"). Moreover, international networks are very often overloaded and communications are very slow or impossible to get at rush hours.

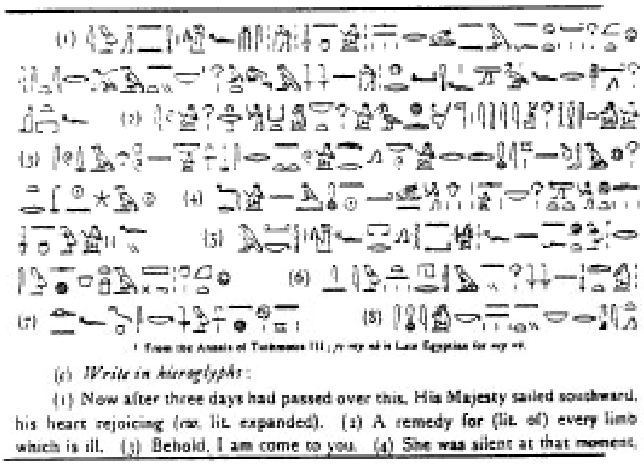


Figure 7: *A typical screen when creating large scientific documents with Windows 3 based programs. (Hummmm. . . did I mix some figures???)*

AS TEX relieves these servers by allowing to store customized information locally. Fast connections can be planned at slack hours and unsuccessful connections can be avoided. An *image* of the subtree of interesting directories of the ftp server is stored in a subtree of windows on the local machine. This local subtree can be consulted at any time, and enriched at each connection. These informations can be used, whenever needed, without having to reconnect to servers. For example, when a type of software is needed (e.g. graphics, network. . .), a list of possible programs can be obtained with the local "Search a String" function, and a list of servers where they are available.

The burden on archie servers can also be relieved by storing general informations locally (users' guide, list of servers. . .) and by storing selected parts of their answers into a customized hierarchy.

5.8 Comparison with Windows-based solutions

The multiwindowing system used by AS TEX (Framework + Desqview or OS/2) is *much more powerful* than that of Windows 3.1: a Windows 3 session occupies the whole screen and graphical DOS applications can be run only in full screen.

Dynamic Data Exchange and Object Linking and Embedding (DDE and OLE) can be done very easily with AS TEX and are not limited to only one level. For example, the bookkeeping of a whole laboratory can be done straightforwardly by importing linked files created at any lower levels (director, team managers, researchers, students). It would be limited to the team managers level with Windows 3.1.

5.9 Ease of learning

At last, AS TEX is *easy to learn*:

Icons are not used (although a complete set could be implemented very easily, for pictogram fans) because too many would be needed, which would requiretoo big memory efforts (see Fig. 7). Icons simplify simple tasks but complicates complicated ones, and writing a multi-author scientific book or even performing everyday research work are very complicated tasks[9].

Once you have learned a module of Framework, it is very easy to learn its other modules and those of AS TEX, since most commands are common. With heterogeneous solutions, simple functions (e.g. "Search a string") are accessed differently in each module. This results in an extra learning that is as enormous as useless (see Fig. 8).

5.10 Why use Framework, not GNU emacs?

Although GNU `emacs` is extremely powerful and very popular, it has nevertheless some shortcomings that made us choose the commercial program Framework as the basis for our prototype AS TEX:

- It is not available on *all* PCs in its complete version (i.e. with its programming language).
- It cannot deal, in its present stage, with the data structure provided by Framework and that is central to AS TEX, i.e. a tree of objects of various nature, in particular *worksheets* and *databases*.

However, future versions of AS TEX might be developed with `emacs` if it turns out that the required data types can be implemented by programming without too many problems and if PD spreadsheets and database managers, programmable with the same language

[9] Hieroglyphs are very well adapted to describe simple concepts such as cow, leg, house: even analphabets can understand them. But they are inadapted to describe complex abstract concepts. This is why alphabets were created out of small subsets of hieroglyphs (the letter "A" for example originates from a pictogram that describes a cow, and "B" from a pictogram describing a leg or a house).

as `emacs`, can be coupled to it.

Figure 8: *A researcher having learned Windows 3 +Word +Mathdesign +Lotus123 +Rbase +Imageworks +. . . +. . . to create a scientific book.*

Conclusion

In this article, we sketched the type of programs that researchers will use, in the future, to write their documents. Then, we described version 2 of an experimental program of this type, AsTEX, that we have designed. It is intended for creating easily multi-author scientific documents on PCs. It has many advantages as compared to existing commercial Scientific Word-Processors:

- It is based on TEX and LATEX, so that it shares all the advantages resulting from their use, without having any of the inconveniences since assistance in typing is provided at various levels, and a previewer very simple to use is available;
- It is thoroughly adapted to the whole creation process of scientific documents. This allows to save time at many stages of the process and results at the end in a *considerable gain in time*. In particular:
  - TEX and LATEX are interfaced with the other software used to create the document (Fortran compiler, computer algebra program, spreadsheet, database manager. . .), so that most laborious and error-prone cut and paste operations are eliminated;
  - Files and informations related to the document are managed efficiently in hypertext mode, so that time-consuming archiving and retrieving procedures are avoided.
- It can run on *any PC*, even the oldest 8088-based models, contrary to Windows 3-based SWPs (such as Word + Mathdesign). Therefore, Scientific Document-Processing with the highest quality of output provided by TEX, is accessible to everybody, not only to the happy owners of the latest PC models. This can be especially interesting for researchers of East-european and Developing countries, for students and for teachers of scientific disciplines in high schools;
- It preserves investment in DOS hardware and software: you have not to upgrade to a more powerful machine, or to buy a larger disk or the latest versions of all your programs, to have them work with Windows 3.1;
- It requires only a few megabytes, with its companion programs, and they can be installed on almost all notebooks, for those who need a transportable PC because they travel a lot, work outdoors or very often at home;
- It allows to use Unix software from a PC without going into the complications of the Unix system and its various idioms.

Acknowledgements

The drawings are taken from Marcel GOTTLIEB, *La Rubrique à Brac*, Dargaud.

References

[1] Paul BARTHOLDI, "Bittersweet experiences during 6 years using TEX and LATEX", in *Desktop Publishing in Astronomy & Space Sciences*, A. Heck Ed., World Scientific, 1992.

[2] Christophe CÉRIN, "Vers la construction de macros de mise en couleur pour TEX", in *Proceedings of the sixth EuroTEX Conference*, op.cit., pp. 197–206.

[3] D. DILL, "Interactive TEX/Mathematica documents", *TeXhax Digest*, March 10, 1991, vol.91 (no. 10).

[4] Leslie LAMPORT, *LATEX user's guide and reference manual*, Addison-Wesley, Reading, Mass., 1986.

[5] Michel LAVAUD, "AsTEX: an integrated and customizable multiwindow environment for sci-

entific research", in *Proceedings of the sixth EuroTEX Conference*, P. Louarn Ed., *Cahiers GUTenberg*, no. 10-11, September 91, pp. 93–116.

[6] Michel LAVAUD, "AsTEX: a software environment on PC adapted to scientific research", in *Computing Methods in Applied Sciences and Engineering*, R. Glowinski Ed., Nova Science Publ., pp. 779–788, 1991.

[7] Michel LAVAUD, "AsTEX: a software environment on PC for creating multi-author scientific books", in *Desktop Publishing in Astronomy & Space Sciences*, A. Heck Ed., World Scientific, Singapore, pp. 177–186, 1992.

[8] Michel LAVAUD, "AsTEX: a low-cost solution to create large multi-author scientific documents", posters presented at *EP'92 International Conference on Electronic Publishing, Document Manipulation and Typography*, 7–10 april 1992, Lausanne (Switzerland).

[9] Eberhard MATTES, *emtex-user discussion list*, Aug. 13 1992.

[10] Simon MITTON, "Desktop publishing applied to astronomical books and journals", in *Desktop Publishing in Astronomy & Space Sciences*, op. cit., pp. 67–76.

[11] Richard PALAIS, *NTS-L discussion list*, Jul. 13 1992.

[12] Donald KNUTH, "Remarks to Celebrate the Publication of *Computers & Typesetting*", TUGboat, vol. 7 (no. 2), 1986.

[13] Alexander SAMARIN and Anatoliy URVANTSEV, "CyrTUG – cyrillic branch of TEX world", in *Proceedings of the sixth EuroTEX Conference*, op.cit.

4TEX: a TEX Workbench for MS-DOS PC's

Wietse Dol, Erik Frambach, Maarten van der Vlerk

Department of Econometrics
University of Groningen
P.O. Box 800
9700 AV Groningen
W.Dol@eco.rug.nl, E.H.M.Frambach@eco.rug.nl, M.H.van.der.Vlerk@eco.rug.nl

Abstract

TEX and all its companions offer an enormous amount of possibilities. This is both an advantage and a disadvantage. The advantage is that almost anything is possible; the disadvantage is that you need detailed knowledge of all related programs to fully exploit the possibilities.
The MS-DOS program 4TEX is an attempt to integrate all major TEX related programs in a shell that shields you from the tedious and frustrating job of setting environment variables and program parameters.

1 Introduction

Preparing documents with TEX or LATEX is by no means a simple job. However, those who persevere are rewarded with a beautifully typeset document. There are at least two reasons why writing with the 'aid' of TEX may cause some problems.

First of all there is the enormous amount of possibilities that TEX offers. While this is the reason to use TEX in the first place, it has the disadvantage that many—at first puzzling and maybe hard to remember—commands have to be used. In principle this is a problem that you will have to live with. A thought to comfort you: many people have tried before you and most of them are by now enthusiastic TEX users.

The second reason why TEX may seem difficult to work with is of a more technical nature. Since TEX itself is only(!) a compiler, it needs several other programs and utilities to make a fully operational text preparation system. For example, at some time between conception and delivery of your document, you might need an editor, a previewer or a printing program. All these programs need setting of environment variables and/or some parameters to cooperate the way you want them to. This is of course just the type of thing you do not want to bother about.

So here is the good news: you don't have to! If you do not like to spend valuable time discovering all these technicalities, try 4TEX. While in its infant days, it did nothing more than constitute a simple menu for calling the TEX compiler or an editor and the like. Although it is a relatively young program and is still growing, we feel that by now it is powerful enough to justify the title *workbench*. In the next sections you find a description.

2 Principles of 4TEX

The 4TEX system consists of a host of separate programs, e.g. it uses emTEX as TEX implementation, which is public domain but state–of–the–art. It is named after its author, Eberhard Mattes, from the Stuttgart University. You must prepare your TEX input file with an ASCII editor (we suggest you use QEDIT); the actual TEX compiler converts this input file into a `.dvi` file (from DeVice Independent); separate programs generate printed output from the `.dvi` file and allow you to preview the typeset page on your screen. In addition, there are a great many add-on utilities: a spell-checker; a database program for maintaining bibliographic references; an index generator; a utility to remove TEX codes; a font generation program, extended graphics support etc.

Some of these programs, especially the various versions of the compiler and of the print and preview programs, require *lots* of parameters and/or environment variables. 4TEX is designed to shield you from the dirty bits.

As mentioned above, the first objective in creating 4TEX was to have some sort of integrating menu to call TEX and the other main programs without the fuss of setting, let alone remembering parameters.

One obvious way to do this is by a *batch file*. However, plain `command.com` batch files tend to be slow, since only one command is read at a time. A much bigger draw-back is its lack of even the most basic commands for an interactive system. A very attractive alternative is provided by 4DOS (shareware by JP Software Inc.), nowadays a well known replacement of `command.com`.

What once started as a small and simple batch file, grew into a collection of batch files that currently comprises over 7000 lines of sometimes fairly sophisticated code. The main reason to program 4TEX in the 4DOS batch language and not in a higher programming language (e.g. Pascal or C) is that by using 4DOS we could create an *open system*, i.e. everyone can modify the workbench to suit personal needs and taste without the need of special tools or extra compilers. Another reason to use the 4DOS batch language is the availability of environment variables and variable functions that enabled us to do things that would require very tedious programming in a higher programming language. Since it is also fast (the complete batch file is read into memory at once), it was an easy decision to implement 4TEX as a 4DOS batch file.

One might object that using 4DOS batch files deprives the old-fashioned `command.com` users from the benefits of 4TEX. We happen to think that this would only be a mild punishment for not recognizing how good 4DOS really is. However, for those who have a good reason not to use 4DOS we also implemented 4TEX to run under `command.com`. This is of course done by loading 4DOS as a secondary shell.

Though 4DOS is very powerful, some routines had to be written in other programming languages, e.g. to support mouse operation and to select files. Some *public domain* and *shareware* programs are used, e.g. MARKNET.EXE, RELNET.EXE and PKUNZIP.EXE.

2.1 Freeware, shareware and commercial software

4TEX uses several public domain (*freeware*) and *shareware* programs.

You may use a shareware copy for an evaluation period of 21 to 30 days. The purpose of this evaluation period is to allow you to determine whether the program meets your needs before purchasing it. Once the evaluation period ends, you agree to either purchase a registered copy of the program, or to stop using it.

Using freeware and shareware only, we are able to distribute 4TEX without violating any law or agreement. As a 4TEX *user* you are supposed to pay for the shareware programs that are listed in the documentation (see Chapter 13, page 79). Remember that 4TEX could never have been build without these programs.

Commercial software is not used, though interfaces to a few programs (e.g. EUROGLOT, VERTAAL! and WORDFINDER) are implemented. Naturally commercial fonts can be added. All DVI-drivers are installed in such a way that this should be easy.

2.2 Setting up 4TEX

After installing 4TEX from the distribution files, some customization is required before 4TEX will run.

The most important files are `texuser.set` and `system.set`. In these files all parameters that 4TEX needs are set. The file `system.set` contains values for the parameters that are needed for any user (e.g. what options does the compiler need, where are the fonts located), whereas `texuser.set` contains values for the parameters that are user specific (e.g. where TEX files are stored, what screen colors should be used).

As you might have guessed, 4TEX runs perfectly on a network. Because NOVELL NETWARE is the most popular type, 4TEX has an interface to run from such a network. However, only a few lines of code in one of the batch files need to be changed to make 4TEX run on any other type of network.

3 Features

In this section we will describe some features of the 4TEX workbench.

3.1 Add, delete, modify

4TEX aims to be an open system such that every user can add, delete or modify the 4TEX workbench to suit personal needs and taste. For instance 4TEX uses Babel: a simple way to generate TEX format files with multiple languages, and some control sequences to switch languages (and hyphenation patterns) within one TEX document. It is easy to adapt Babel for other/more languages than already used by 4TEX (i.e. English, German, French and Dutch). It is easy to add, modify or delete format files (e.g. 4TEX supports the New Font Selection Scheme).

Of course any user or department has specific printers. 4TEX currently supports matrix printers, LaserJets, DeskJets and PostScript printers. Adding or deleting a printer type should not cause problems. 4TEX lets you choose between local printers, network printers and printing to a file.

At this moment the spell-checker supports the languages Dutch, US English, English, French and German, but any other language can easily be added.

3.2 Help in 4TEX

There are several types of automated assistance available. For each item in every 4TEX menu there is a help screen.

The memory resident program TEXHELP is meant as partial replacement for the LATEX and TEX manuals. It is a hypertext system that can be called from the editor. For example, if the cursor is at the word `\documentstyle`, pressing the TEXHELP key results in a help screen that refers you to the topic `Document Styles`. Select this topic and you get a help screen that gives information about all valid LATEX document styles and options. Moreover, there are cross references to all options and, among others, the commands `\flushbottom` and `\twocolumn`. It is also possible to jump directly to the index of TEXHELP or to review the last help screen. For most topics, the text

is taken from the LATEX help system for MS-WINDOWS by Michael F. Reid, which is based on George Greenwade's help system for Vax-VMS computers. At this moment there are about 315 documented topics in the TEXHELP database.

Several *example* files exist, varying from a standard LATEX file to more complex subjects, such as tables with asymmetric columns or creating multiple indexes. Example files can be viewed from the editor and, if desired, be inserted in the current document.

Finally, there is on-line TEX documentation. This consists of documentation of e.g. style files. To save disk space, the documentation is kept in archives that are temporarily decompressed on selection.

3.3 Editing

We have chosen QEDIT (shareware by Semware) as the default editor in 4TEX. For this editor, we have developed many macros. For example, you can enter a LATEX environment defined by the commands `\begin{env}` ... `\end{env}` by picking it from a list; insert `\index{this}` behind the word `this` at the cursor position; or call the spell-checker (see below) to check the word at the cursor. Moreover, on loading a text file, the cursor will be placed at the exact location it was when you last edited the file (also available in the View-Block facility, see below).

3.4 Block-View: quick partial compilation

One of the most often heard objections to TEX is that it is not a so called WYSIWYG system. On the other hand, TEX users often reply that the acronym WYSIWYG is misleading: it should be WYSIAYG (What You See Is *All* You Get). We do not wish to solve this problem. Instead, 4TEX offers a feature that we think both sides may like.

Imagine that you have just typed a very complicated formula. The natural way to check whether you did not make a mistake, is to see what it looks like. Or maybe you are just curious. However, to see this single formula you would have to leave the editor, compile the entire document, start the previewer, and find the formula. Then you need to return to the editor, and find the exact position where you left it. If you made a mistake, you have to do this all over again.

In 4TEX this whole procedure can be performed by the touch of a few keys. Even better, only the preamble of your document and the formula are compiled, which makes it as fast as possible. To view a certain part of a document, mark it as a block in the editor and press the 'view-block' key. Alternatively, if you want to view an entire file, just press the 'view-file' key. If you are making slides, it makes sense to view a complete slide at a time. Therefore, in this case 4TEX automatically defines the before mentioned block to be the current slide. If TEX finds an error in the block, the user has the option to be returned to the editor at the line that contains the mistake.

3.5 Spell-checking

In 4TEX we use AMSPELL, a public domain program by A. Merckens, to check and correct spelling in TEX documents. AMSPELL is basically a spell-checker for plain ASCII files, with some special features for dealing with TEX files.

The basic idea behind this program is to make spell-checking easier by

- providing the context of the possibly misspelled word;
- offering alternatives;
- offering facilities for editing the word;
- automatically replacing misspelled words in your document;
- learning new words.

AMSPELL does not require TEX commands to be removed from your document (deTEXing). In fact, it will even interpret the standard accenting commands like `\"`, `\'`, `\'` and will automatically use them while replacing misspelled words. When checking a TEX file, AMSPELL will ignore all text between $'s and $$'s. Furthermore, AMSPELL will ignore parameters of the LATEX commands `\ref`, `\pageref`, `\cite`, `\nocite`, `\label`, and all text between `\begin{equation}`, `\begin{eqnarray}`, `\begin{eqnarray*}`, `\[` and their counterparts like `\]`. You can change or expand the lists by means of environment variables.

3.6 Extended graphics support

TEX has been developed with the idea that it should be possible to have a TEX implementation for every operating system (MS-DOS, VMS, UNIX etc.). Another important feature of TEX is that documents can be freely exchanged between operating systems (because documents are written in standard ASCII). Graphics, however, are machine dependent and the possibility to include graphics in TEX or LATEX depend on the operating system and the DVI-driver you are using.

A solution often used for including graphics in TEX documents is to insert PostScript pictures in the document through `\special` commands. The `\special` command is ignored/passed on by the TEX compiler but the PostScript DVI-driver knows how to insert the PostScript picture at the right place and in the right size in your document. The disadvantage of this method is of course that you can only include *PostScript* pictures in your document and that you need a PostScript printer to produce output.

The emTEX DVI-drivers support a `\special` command to include black-and-white bitmapped pictures. Both this feature and the PostScript possibilities are used by 4TEX to incorporate pictures in TEX documents.

Graphic files come in many flavors. 4TEX allows you to view, manipulate and include the following types of pictures in your TEX documents:

- bitmapped pictures: GIF, TIFF, PCX, BMP, IFF, LBM, IMG, CUT, and PCL;
- vector pictures: HPGL and PostScript.

This is done by using the following freeware software: BM2FONT, HP2XX, PCLTOMSP, and GHOSTSCRIPT.

4TEX automatically chooses the appropriate conversion program, depending on the type of graphic file, or rather the file extension. Any conversion that 4TEX performs will result in either TEX fonts (accepted by any DVI-driver) or both a PCX and EPS file (for emTEX and PostScript resp.). Furthermore, a small TEX file is produced that contains all the necessary commands to incorporate the picture in your document.

4 4TEX in practice

At the Department of Econometrics of the University of Groningen 4TEX is installed on a network, and its usage is logged. The time that 4TEX was running is logged and the number of calls to functions are counted.

To give you an idea of the activity we made a summary for the period of 21 January to 12 March.

# calls	*function*
17202	editor
4273	compiler
3984	block-view
204	4TEX help
942	LaserJet print
9	PostScript print
13	matrix print
2501	preview (not block-view)
230	view log-file
189	spell-check
26	oneword spell-check
689	shell to DOS
175	BibTEX
47	BiBDbase
216	view picture
6	bitmap to TEX font
47	HPGL to PCX + EPS
54	PS/EPS to PCX
0	PCL to PCX
5	GraphicsWork Shop
45	MakeIndex
1	DeTEX
0	UnTEX
5	Wordlist
4	TEXcheck
11	TEXcad
0	submit TEX batch
22	automatic font generation
196	user installed utilities

In principle, these figures speak for themselves. However, let us make a few comments.

The number of calls to the functions editor, compiler, print, and preview do not give information on how 4TEX is used in practice: they indicate actions that have to be taken whether you work in 4TEX or not. They do give information on the amount of TEX-ing that is going on at our department.

Among the other figures, the number of calls to the function block-view is rather striking. It indicates that this function is a feature indeed.

Another much used facility is shell-to-DOS. We suspect that many people start 4TEX only once per day, and from then on start any other program they need from the DOS-shell (at the expence of only 36K conventional memory). Note that the calls to user installed utilities represent activities that would have to be done in a DOS-shell in other programs.

Whereas without 4TEX most users would not even try to incorporate graphics in their document (other than by using scissors and glue), the total number of calls (328) to functions related to graphics bears witness to how easy this has become in 4TEX.

5 Availability and support

4TEX can be obtained through *anonymous ftp* from the file servers `obelix.icce.ruu.nl`, directory `/pub/erikjan/4tex`, and `ftp.cs.ruu.nl`, directory `/pub/TEX/4tex`. The system has a modular structure that allows you to install only those parts that you need. Therefore it is important that you read the file `install.txt` first. The installation files are compressed using ARJ.EXE and fit on 1.44 MB 3.5" high density diskettes. You may also take a look at the documentation file before you install anything. `4texdoc.dvi` and `4texdoc.ps` can be found in the same directory.

If you have trouble installing 4TEX or need more information you can send E-mail to

`4TeX-support@eco.rug.nl`.

However, don't expect an answer within the hour—we will try to help you as soon and as best as we can, but 4TEX is an after hours project.

4TEX users can join the 4TEX mailing list. On this list users can pose/answer questions regarding 4TEX usage. New or desired developments and features are also announced and discussed on this list.

TEX zonder omhaal voor Atari ST en andere PC's

Robert Best

best@zeus.rijnh.nl

Abstract

Deze cursus is bedoeld als eerste kennismaking met TEX op een eenvoudige PC. Een harde schijf is niet nodig. De cursus is gebaseerd op de PD-TEX van Christoph Strunk voor Atari ST. De installatie en de functie van de basis bestanden van TEX worden behandeld.
Deze cursus is een gecorrigeerde herdruk van een serie artikelen in het blad ST[1], uitgegeven door: Stichting ST [2] *Daar zijn de in de tekst genoemde schijfjes te verkrijgen.*

1 Inleiding

TEX automatiseert het werk van de zetter in een drukkerij. TEX software draait op vrijwel iedere computer. TEX kan simpel en goedkoop gebruikt worden; met de ca. 900 commando's zijn ook programmatische hoogstandjes te maken.
Dit artikel is een inleiding tot het gebruik van TEX. We beginnen zo eenvoudig mogelijk. Er hoort een schijf B118 bij[3] met een gebruiksklare selektie uit de PD TEX van Christoph Strunk.

1.1 Wat doet TEX?

In een ouderwetse drukkerij nam een zetter letterblokjes uit een bak en plaatste die op regels. Hij zette sommige letters in groepen (ligaturen) dicht tegen elkaar, b.v. ff fl fi. Sommige zetterijen hadden een schier eindeloze verzameling letters uit vele schriftsoorten en (wiskundige) tekens in vele formaten. De zetter verdeelde het 'wit' tussen de woorden zodat de regels even lang werden en een gelijkmatige verdeling van 'wit' in de alinea ontstond. Alleen als het niet anders kon, brak hij woorden af aan het eind van een regel.
Het pure handwerk bestaat nauwelijks meer. In alle commerciële drukkerijen is dit overgenomen door de computer. Bovendien gebeurt het zetten vaak helemaal niet meer op de drukkerij.

En dat is te merken. Sinds de meeste gebruikers hun materiaal 'camera ready' bij een drukker afleveren, is de kwaliteit van het zetwerk achteruit gegaan. Gewone tekstverwerkers leveren een zetsel op dat door de ouderwetse drukker afgekeurd zou worden: onjuiste woord- en letter afstand, incomplete fonts. Vooral voor wiskundige formules laat de tekstverwerker het afweten. Zelfs als de nodige tekens beschikbaar zijn, is het intikken moeizaam en het resultaat op papier abominabel.

TEX automatiseert het zetwerk met een resultaat dat vakkundig zetwerk benadert. Werken met TEX is moeilijker dan met een gewone tekstverwerker, maar het loont de moeite als je er oog voor hebt (gekregen). TEX is vooral geschikt om tekst met wiskundige formules mooi op papier te krijgen, maar met TEX kunnen ook teksten in b.v. Grieks, Hebreeuws of Russisch gezet worden.

TEX is géén direkt WYSIWYG (What You See Is What You Get) systeem. Je maakt een TEX-bestand, waarbij je op het scherm nog niet de opmaak in alinea's, de vreemde tekens, de verschillende schriften enz. ziet. TEX maakt daar een DVI-bestand van. 'DVI' staat voor 'device independent', oftewel uitvoerapparaat onafhankelijk. Deze uitvoer kan nu m.b.v. een DVI-programma bekeken of geprint worden. Dan zie je het resultaat pas. Je kunt een DVI-file ook van de ene computer naar de andere meenemen. Er is geen verschil tussen een Atari-, Mac-, UNIX- of DOS-DVI file.

TEX bestaat nu ca. 15 jaar. Al 10 jaar is TEX een voorbeeld van een vrijwel foutloos programma. In 1989 is een nieuwe versie verschenen met uitbreidingen maar geen wezenlijke wijzigingen. De schrijver van het pro-

[1] Uitgaven: ST37, ST39, ST41, ST42, ST43.

[2] Stichting ST, Postbus 11129, 2301 EC Leiden, tel. 071–130045. ST is een onafhankelijk tijdschrift van en voor gebruikers van Atari ST computers.

[3] Alle in dit artikel genoemde schijfjes zijn te verkrijgen via de Stichting ST.

gramma, Don. E. Knuth, hoogleraar informatica aan de Stanford Universiteit (USA), heeft het programma nu 'bevroren'.

Op schijf B118 staat de oude versie, omdat de nieuwe alleen compleet verspreid mag worden.

1.2 Wat heb je nodig voor TEX?

Minimaal een ST met 1 Megabyte RAM, dus een 520ST+ of een 1040ST. Beter is een ST met harde schijf, of 2 Mb RAM, zodat er ruimte is voor een 1 Mb RAM-disk om redelijk snel te kunnen wisselen tussen de editor en de TEX programma's. Verder een dubbelzijdige diskdrive en een SM124 of SM125 beeldscherm. Op een 9-naalds printer zijn al goede afdrukken te maken, maar de TEX kwaliteit komt beter tot zijn recht op een inkjet- of laser-printer.

En natuurlijk TEX software. Er is goede PD TEX voor de ST, maar het installeren is niet simpel zonder harddisk. Daarom heb ik uit de PD een schijfje samengesteld waarmee je zonder veel omhaal aan de slag kunt.

1.3 Installatie

Als er minstens 2 Mb RAM is, start de ST dan op met schijf B118 in drive A. Er wordt een RAM-disk van 1 Mb aangemaakt waarin je de hele schijf copiëert behalve de AUTO-folder. De RAM-disk is de werkschijf. Als er maar 1 Mb RAM is, copiëer dan de hele schijf behalve de AUTO-folder naar een lege schijf; dat wordt dan de werkschijf. Als er een harde schijf is, kan die als werkschijf gebruikt worden.

Open de folder FONT0115. Hier staan de 16 standaard fonts van TEX verpakt in 115.ARC (omdat ze zo veel beter te copiëren zijn). Start UNARC.TTP, tik

```
115.arc
```

en een `a` om alle fonts uit te pakken. Als je een Epson-compatibele printer hebt (b.v. een Star), pak dan ook 288.ARC uit in FONT0288. Verwijder de folders FONT0240 en FONT0288 en kijk onder het kopje 'Wat nu', als je niet zo'n printer hebt.

Zet (zo mogelijk) ook een editor op de werkschijf. Dat kan b.v. Tempus zijn of een tekstverwerker waarbij de wordprocessing uitgeschakeld is (dus geen haak voor WP in het menu 'Edit' van WordPlus).

1.4 Eerste stap

Start de editor en laad STAP_1.TEX of maak zelf een TEX-bestand op de werkschijf. Elke tekst die alleen de 26 letters van het alfabet (groot en klein), de cijfers en de gewone leestekens bevat, is goed voor een TEX-bestand. Breek de woorden niet af en gebruik een lege regel om alinea's te scheiden. Tik aan het eind van de tekst de 4 tekens `\bye` om TEX te laten weten dat er niets meer komt, en zet het bestand met extensie .TEX op de werkschijf.

Start het programma TEX.TTP en tik de naam van het TEX-bestand in; de extensie .TEX mag weggelaten worden. Zoals gezegd vertaalt TEX het bestand in een 'device independent' bestand en zet dit onder dezelfde naam met de extensie .DVI op schijf. (Als TEX dit niet automatisch doet, stopt en met onbegrijpelijke vragen komt, kijk dan onder het kopje 'TEX vraagt hulp'.)

Start het programma DVI_ST.TTP en tik weer de zelfde naam (zonder .DVI) om het 'zetwerk' op het scherm te krijgen. Er moet hier flink gerekend worden, dus het duurt even voor er iets te zien valt. Met de pijltjestoetsen kan je door de tekst wandelen. Probeer [Help], en tik [Esc] of [Undo] om de 'preview' op het scherm te beëindigen.

Met DVI_FXHD kan een afdruk op een Epson-compatibele printer gemaakt worden.

1.5 TEX vraagt hulp

TEX zet tijdens het werk altijd wat op het scherm. Als het goed gaat verdwijnt dat vanzelf weer; je kunt het nalezen in een bestand met de extensie .LOG. Als er iets mis gaat, stopt TEX en vraagt hulp. Een vreemd teken in het bestand, b.v. een ë, doet TEX stoppen. Je kunt TEX afschepen met [Return], vaak werkt TEX dan door. Je kunt ook TEX antwoorden met e voor exit. Dat werkt ook als TEX een bestand niet kan vinden (I can't find file $\cdots$) dank zij een bestandje E.TEX op de werkschijf.

TEX meldt 'Overfull \hbox $\cdots$' als een regel te lang wordt, omdat het laatste woord niet afgebroken kon worden. Vaak kan je TEX helpen door `\-` in te lassen op een geschikte plaats, b.v.

```
com\-mando
```

TEX kent nl. veel afbreekplaatsen, maar niet alle.

1.6 Fonts

Het font (tekenverzameling) dat TEX bij de eerste stap gebruikt heeft heet cmr10, afkorting voor 'computer modern roman 10 point'. Dit is het 'default' font dat TEX neemt als de gebruiker 'verzuimt' een font te specificeren. Dit font staat in de folder FONT0096 voor het scherm en in FONT0240 voor de printer.

10 punts fonts zijn geschikt voor boeken en tijdschriften. Ze zijn eigenlijk te klein voor brieven en rapporten. Daarom zullen we verder 20% grotere fonts gebruiken. (TEX maakt onderscheid tussen een 20% vergroot 10-punts font en een 12-punts font. Het laatste is iets dunner.) De volgende stap leert een paar commando's om TEX een grotere versie van het zelfde font of een ander font te laten nemen.

Een TEX commando bestaat uit een backslash \ en één of meer letters, of een \ en één teken dat geen letter is.

1.7 Stap 2

Start de editer en tik aan het begin van het TEX-bestand van stap 1:

```
\magnification 1200
```

om TEX te vertellen dat de tekens 1.2 maal groter moeten worden. Voor de preview neemt TEX dan cmr10 uit FONT0115. Kies een tekstdeel in het TEX-bestand om

schuin te drukken, b.v.

```
gelijkmatige verdeling
```

Verander dit tekstdeel als volgt:

```
{\it gelijkmatige verdeling}
```

`\it` betekent italic. De werking van dit commando wordt door de accolades begrensd. (Als je de {} weglaat, wordt *alle tekst* na `\it` cursief gezet.) Op de zelfde manier kan een tekstdeel vet gezet worden met het commando `\bf` voor boldface. TEX neemt dan voor de betreffende tekstdelen cmti10 (computer modern text italic) of cmbx10 (bold extended) i.p.v. cmr10 uit FONT0115. De tabel geeft nog een paar mogelijkheden.

roman	`\rm`	cmr10
italic	`\it`	cmti10
boldface	`\bf`	cmbx10
slanted	`\sl`	cmsl10
`typewriter type`	`\tt`	cmtt10

Bekijk het resultaat met TEX.TTP en DVI_ST.TTP.

1.8 Vreemde tekens

De volgende tabel toont enkele vreemde tekens die TEX kan maken, met de bijbehorende commando's. Let op de spatie tussen de 2 c's voor de cédille. Na een commando dat eindigt met een letter mag geen letter staan, want dan denkt TEX dat die letter ook nog tot het commando behoort.

trema, umlaut	ö	`\¨o`
accent grave	ò	`\`o`
accent aigu	ó	`\´o`
circonflexe	ô	`\^o`
cédille	ç	`\c c`
eszet	ß	`\ss`
hekje	#	`\#`
dollar	$	`\$`
procent	%	`\%`
ampersand	&	`\&`

Voor 'großartig' wordt `gro\ss artig` getikt met een 'loze' spatie. Maar om 'groß und großartig' te krijgen kan je

```
gro\ss\ und gro\ss artig
```

tikken; de backslash voor de spatie maakt er een 'echte' spatie van. Het kan ook met

```
gro{\ss} und gro{\ss}artig
```

ook niet echt handig. In stap 3 leren we TEX dat `\3` een ß is, zodat je dan kan volstaan met

```
gro\3 und gro\3artig
```

zonder echte of loze spaties (omdat `\3` niet op een letter eindigt).

De reden dat sommige tekens niet rechtstreeks ingetikt kunnen worden is dat TEX de tekens `\#$%&_^{}` reserveert voor speciale doeleinden. De volgende stap leert het gebruik van % voor commentaar en & voor een tabel, en we maken een 'macro' om een reeks toetsaanslagen te verkorten.

1.9 Stap 3

Start de editer en laad STAP_3.TEX. De regels die met een % beginnen zijn commentaar; TEX slaat ze over. Er staan wat voorbeelden van woorden met accenten en een tabel gemaakt met `\settabs`. Bekijk dit met TEX.TTP en DVI_ST.TTP.

Nu de macro voor ß. Zet ergens in het begin van STAP_3.TEX

```
\def\3{\ss}
```

en tik na deze definitie ergens `gro\3 und gro\3artig` of iets anders met ß om te zien of het werkt. Probeer ook de macro

```
\def\mvg{Met vriendelijke groeten,}
```

die `\mvg` definiëert. Na

```
\def\\{$\backslash$}
```

kan je met `\\`een backslash maken.

1.10 Wiskunde en Grieks

Veel tekens, Griekse letters en wiskundige symbolen, maakt TEX niet in 'text mode' maar in 'math mode', d.w.z. tussen dollartekens. Wiskundige formules kunnen in de tekstregel staan, b.v. $a^2 + b^2 = c_o^2$. Deze formule wordt met

```
$ a^2+b^2=c_o^2 $
```

gemaakt. Met `^` en `_` zet je super- en subscripts. Een formule tussen dubbele dollars wordt apart en gecentreerd gezet:

```
$$ y = {a/b \over \sqrt{1+c}}
    \int_0^x {p \over 1+z^2} dz $$
```

levert

$$y = \frac{a/b}{\sqrt{1+c}} \int_0^x \frac{p}{1+z^2} dz$$

Let op het gebruik van accolades en de automatische aanpassing van de hoogte van het integraalteken aan de integrand! Ook het wortelteken past automatisch bij hoogte en breedte van het argument.

Voor gewone letters in formules gebruikt TEX font cmmi10 (math italic), voor cijfers cmr10. Sub- en superscript worden in cmmi7 of cmr7 gezet en subsubscript enz. in cmmi5 of cmr5. De speciale wiskundige symbolen komen uit de drie cmsy fonts en cmex10.

In stap 3 zagen we dat { en } gemaakt worden met `$\{$` en `$\}$`. De tekens < en > krijg je met `$<$` en `$>$`. De Griekse letters $\alpha\beta\gamma\Gamma\ldots$ zijn te maken met `$\alpha\beta\gamma\Gamma$`, enz.

`$\cal ABC$` levert calligrafische hoofdletters $\mathcal{ABC}$.

1.11 Opmaak

De regellengte zet TEX op 6.5 inch en de teksthoogte op 8.9 inch, tenzij je iets anders voorschrijft. (De drukker rekent graag met inches en punten; 1 inch = 72 punt = 25.4 mm.) De commando's voor b.v. 160 mm regellengte en 250 mm teksthoogte (voor A4 papier) luiden

```
\hsize 160 true mm
\vsize 250 true mm
```

Een `magnification` moet in het TEX-bestand vooraf gaan aan de size-commando's. Bovendien zijn

dan de woordjes `true` nodig. Het woord `true` in een maataanduiding voorkomt dat die maat mee vergroot wordt.

TEX bepaalt zelf de afstand tussen woorden op een regel. Al zet je 10 spaties tussen twee woorden, het effekt is gelijk aan 1 spatie. Een afstand van b.v. 5 mm tussen twee woorden bereik je met het commando

```
\hskip 5 mm
```

TEX maakt ook zelf de vertikale afstand tussen regels. Al maak je met 10 Return-aanslagen een flinke ruimte tussen twee regels, het effekt is gelijk aan 1 lege regel. Een lege ruimte van b.v. 50 mm hoog wordt met

```
\vskip 50 mm
```

gemaakt.

Een alinea kan je laten inspringen met b.v. 10 mm extra kantlijnruimte met:

```
{\leftskip 10 mm
... alineatekst ....
\par}
```

In STAP_3.TEX staat een manier om een tabel te maken.

1.12 Initex, Metafont, LATEX

Een hulpbestand bij het TEX programma is PLAIN.FMT waarin o.a. de patronen verwerkt zijn die TEX gebruikt om woorden af te breken. Het bestand PLAIN.FMT op schijf B118 bevat Nederlandse afbreekpatronen, gemaakt door de Universiteit van Utrecht. Met onze taal werken die vrijwel foutloos, maar met een vreemde taal natuurlijk niet. INITEX.TTP dient o.a. om PLAIN.FMT te maken met andere afbreekpatronen.

Bij TEX wordt een selectie van fonts meegeleverd in vergrotingen die in stappen van 20% oplopen, dus voor magnification 1000, 1200, 1440, 1728, enz. Met METAFONT zijn fonts te maken in iedere gewenste grootte. Het heeft geen zin om de opdracht `\magnification 2000` te geven als je niet eerst de bijbehorende fonts hebt aangemaakt!

Er zijn vele macro's in omloop, b.v. om kopjes te maken, tekst in twee kolommen te zetten, enz. De meest bekende macroverzameling heet LATEX, naar de maker Leslie Lamport.

1.13 Wat nu?

In deze inleiding heb je kennis gemaakt met TEX. In vervolghoofdstukken zal dieper ingegaan worden op dit zetsysteem. De oude versie staat op de PD schijfjes B74–83. Daaruit is schijf B118 samengesteld. De nieuwe versie van TEX staat op B98–105. Deze is sneller, heeft mooiere fonts en kan afbreekpatronen voor meer talen tegelijk gebruiken. Zonder harde schijf is deze versie echter niet te installeren. Beide versies bevatten drivers voor vele printers, INITEX met afbreekpatronen voor Duits en Engels, METAFONT en LATEX.

1.14 Literatuur

Het standaardwerk is van de maker van TEX: Donald E. Knuth, *The TEXbook* (Addison-Wesley), een goudmijn voor gevorderde TEXers. Ruim voldoende voor gewoon gebruik. Meer geschikt om iets op te zoeken is: Norbert Schwarz, *Inleiding TEX* (Addison-Wesley), omdat dit boek een lijst van TEX-commando's bevat.

– – * – –

2 Boxen

Na de inleiding tot TEX in het vorige hoofdstuk, gaan we in dit hoofdstuk dieper in op het zetten van tekstblokken op een pagina. Als voorbeeld zetten we een brief en een tekst in twee kolommen.

2.1 Briefhoofd

De bedoeling is een bestand BRIEF.TEX te maken met een briefkader dat je eigen briefhoofd, de plaats voor het adres, de datum enz. bevat. Als je dan een brief aan P. Puk wilt schrijven, zet je BRIEF.TEX in de editor, je vult het adres in, schrijft de brieftekst en zet het bestand als PUK.TEX weer op de werkschijf. Alle namen en maten in dit verhaal zijn willekeurig.

BRIEF.TEX begint met een paar instellingen:

```
\magnification 1200
\vsize 250 true mm
\hsize 165 true mm
\nopagenumbers
\parindent 0 mm
```

De eerste drie commando's kennen we al uit de Inleiding (Stap 1 en Opmaak). Het vierde commando onderdrukt paginanummers; als je die in een lange brief toch wilt hebben, zet je een procentteken voor dit commando (Stap 3). Het laatste commando zorgt dat TEX niet bij elke nieuwe alinea inspringt (`\parindent 0` werkt niet, TEX wil een maat zien).

Vervolgens het briefhoofd, voorlopig in de linker bovenhoek. Andere plaatsen komen later aan de orde (onder 'Meer boxen').

```
% briefhoofd
   {\obeylines
   { \bf A. Texnicus }
   { \sevenrm Tolstr.\ 88
      1234 TT Assen
      Tel 099-99999 }}
```

Het commando `\obeylines` maakt dat TEX niet zelf de tekst gaat opmaken in regels van gelijke lengte. Zonder dit commando zouden naam, straat, enz. niet onder maar achter elkaar gezet worden. De werking van `\obeylines` wordt begrensd door de eerste en laatste accolade. De naam is vet gezet met `\bf`, zoals we in Stap 2 gezien hebben. Adres enz. zetten we in een klein lettertje. Met `\sevenrm` wordt het cmr7

font aangeroepen dat we al kennen als font voor sub- en superscripts in wiskundeformules. Achter de straatnaam staat een punt. Als die punt gevolgd wordt door een spatie, dan duidt dat meestal op het einde van een zin, en TEX zet daar wat extra wit achter. Om te vermijden dat het huisnummer ver achter de straatnaam komt zetten we een \achter de punt. Dat is niet nodig na `A.`, want TEX weet dat een zin niet eindigt met een hoofdletter.

2.2 Adres

We willen het adres zodanig plaatsen dat het b.v. onder het transparante raampje van een vensterenvelop terecht komt, als de brief daar opgevouwen in gedaan wordt. Dat kan als volgt:

```
\vskip 20 true mm
\moveright 90 true mm
% adres
\vbox to 30 true mm
{\vfil \obeylines
P. Puk
Pompoen 11
8888 PP Peek
\vfil}
```

Het `\vskip` commando (zie Inleiding onder 'Opmaak') maakt witruimte tot aan de bovenkant van het venster. De `\vbox to 30 mm` {...} opdracht laat TEX een (onzichtbare) rechthoekige ruimte van 30 mm hoog maken die opgevuld wordt met het materiaal tussen de accolades. Met `\moveright` wordt de box een eind naar rechts verschoven. De `\vfil`'s zorgen voor vulling: er wordt zoveel wit boven en onder het adres toegevoegd dat het precies in de 30 mm hoogte past. Zonder de eerste `\vfil` wordt het adres bovenin het venster gezet, zonder de tweede `\vfil` onderin, met beide gecentreerd.

Na het adres komt:

```
\vskip 20 true mm -
```

Dit zet een streepje op de plaats waar de brief gevouwen moet worden.

2.3 Brieftekst

De rest van het briefkader is nu snel gemaakt. Eerst de datum, rechts uitgelijnd:

```
\hfill Assen,
\the\day-\the\month-\the\year
```

TEX vult de datum in van het Control Panel. Een vulling `\hfil` is te zwak. TEX heeft ook sterkere vullingen met twee l's. Dat is hier nodig om de datum helemaal naar rechts te drukken.

Dan de brieftekst:

```
\parindent 10 mm
Beste Piet
\medskip
brieftekst brieftekst brieftekst
\bigskip
\hskip 80 true mm Groeten,
\bye
```

Met `\parindent` herstellen we het inspringen bij nieuwe alinea's. (Je kunt TEX ook de opdracht `\parskip 2 mm` geven. TEX scheidt dan de alinea's met een kleine vskip.) De `\medskip` en `\bigskip` commando's leveren vskip's ter waarde van een halve en een hele regelafstand. De brief is klaar.

2.4 Meer boxen

We gaan nu een andere indeling maken van het bovendeel van de brief, in twee kolommen. Daartoe zetten we twee vboxen naast elkaar. In de linker komt weer het briefhoofd en in de rechter het adres, maar dat kan evengoed andersom. Na de eerste vijf regels met instellingen komt:

```
\hbox
{\vbox to 80 true mm
{ \hsize 90 true mm
\vskip 10 true mm
% briefhoofd
...
\vfil }
\vbox to 80 true mm
{ \hsize 70 true mm
\vskip 30 true mm
% adres
...
\vfil }}
```

Op de eerste stippels komt weer het briefhoofd (5 regels) en op de tweede stippels het adres (6 regels in het voorbeeld). De vboxen worden naast elkaar gezet in een hbox. (Als je `\hbox` en de laatste accolade weglaat, zet TEX de vboxen onder elkaar.) Beide vboxen zijn 80 mm hoog. Hun breedte wordt bepaald door hsize. Deze maat is binnen de vboxen op 90 mm (links) en 70 mm (rechts) gezet. Buiten de vbox neemt TEX weer de oorspronkelijke instelling, 165 mm. Door de vskips (10 en 30 mm) te variëren kunnen de teksten op en neer geschoven worden in de kolommen.

2.5 Twee kolommen

Het opmaken van de brief gaat in WordPlus eenvoudiger (maar niet zo mooi). We gaan nu iets doen dat in WordPlus niet kan: tekst in kolommen zetten zoals in dit artikel. Stel nu dat we halverwege een pagina verder in twee kolommen willen werken. Daarvoor stoppen we de tekst bovenaan de pagina in een box, we laten TEX de verdere tekst in een lange kolom ter breedte `\colwidth` zetten, en van die kolom twee stukken afsplitsen van de juiste lengte. Eerst definiëren we een paar hulpvariabelen:

```
\newdimen \ohsize
\newdimen \ovsize
\newdimen \colwidth
\colwidth 80 true mm
```

Dan bewaren we de tekst bovenaan de pagina in een box. TEX heeft 256 boxen waarvan 0-9 vrij beschikbaar zijn voor de gebruiker. TEX gebruikt zelf box 255 voor het opmaken van een volle pagina. Die wordt dan doorgegeven naar het DVI-bestand volgens de output routine

```
\output {\plainoutput}.
```

Box 255 is dan leeg. Die output routine gaan we nu veranderen, zodat box 255 doorgegeven wordt naar box 0:

```
\output {\global\setbox0
\vbox{ \unvbox255 }}
\eject
```

Het `\eject` commando vult box 255 en roept de output routine aan. Vervolgens bewaren we hsize en vsize in de hulpvariabelen ohsize en ovsize, en dan veranderen we hsize in colwidth en we maken vsize ruim twee maal zo groot:

```
\ohsize \hsize    \ovsize \vsize
\hsize \colwidth
\multiply \vsize by 2
\advance \vsize by 5 mm
```

Voor we nu TEX box 255 met de nieuwe afmetingen hsize en vsize met tekst laten opvullen, definiëren we eerst een nieuwe output routine. Deze wordt pas gebruikt als TEX voldoende tekst heeft verwerkt om box 255 te vullen, of als TEX een eject of bye tegenkomt.

```
\output
{\dimen0 \ovsize
\advance \dimen0 by -\ht0
\setbox1 \vsplit255 to \dimen0
\setbox2 \vsplit255 to \dimen0
\setbox9 \vbox{ \unvbox255 }
\setbox255 \vbox to \ohsize
{ \unvbox0
\hbox to \ohsize
{ \box1\hfil\box2 } }
\hsize \ohsize    \vsize \ovsize
\plainoutput
\unvbox9}
```

Eerst wordt dimen0 (één van de 256 maatvariabelen van TEX) gelijk gemaakt aan de oude vsize verminderd met ht0, de hoogte van box 0: dat is juist de hoogte die nog beschikbaar is op de pagina. Dan worden er twee kolommen van die hoogte afgesplitst en in box 1 een 2 gestopt, en de rest wordt in box 9 opgeborgen. Box 255 is dan leeg. Vervolgens wordt box 255 weer opgebouwd met bovenin box 0 en daaronder naast elkaar box 1 en 2. Na het terugzetten van hsize en vsize op de oude waarden laten we plainoutput het karwei afmaken. Ten slotte halen we het restant uit box 9; dat wordt het begin van de volgende pagina. Als die vol is, komt onze output routine opnieuw in aktie, maar dan met een lege box 0.

2.6 Tips

Een paar tips tot slot. Eerst vskips die niet werken bovenaan de pagina. Ze worden meestal gebruikt om een tekstdeel duidelijk te scheiden van het volgende. Dat is aan het begin van een nieuwe pagina niet nodig, daarom negeert TEX vskips daar (ook `\medskip, \bigskip` enz.). Als je bovenaan de pagina ruimte wilt maken, gebruik dan `\vglue` in plaats van `\vskip`.

Bij het zetten van tekst in smalle kolommen wordt het voor TEX moeilijk om het wit tussen de woorden gelijkmatig te verdelen in een alinea. Je krijgt dan vaak de melding 'overfull \hbox'. Als dat te vaak gebeurt, verander dan de tolerance met b.v.

```
\tolerance 1000
```

De standaard waarde van deze parameter is 200, de grootste waarde 10000.

BRIEF.TEX en de twee kolom macro BICOL.TEX staan op disk ST39.

— — ∗ — —

3 METAFONT en PostScript

Met METAFONT maken we nieuwe fonts om tekst in grote letters te kunnen zetten. Er is weer een gebruiksklaar schijfje B144 gemaakt uit de PD TEX van Christoph Strunk. Ligaturen komen opnieuw aan de orde. We schuiven met letters: uit elkaar, in elkaar, iets omhoog of omlaag. En we laten TEX PostScript maken, de standaard taal voor computer-gestuurd drukken.

METAFONT is net als TEX een groot programma met veel hulpbestanden. Eigenlijk moet je een harddisk hebben om er goed mee te kunnen werken. Maar om de essentie te laten zien en om ook bezitters van een ST met 1 Mb RAM zonder harddisk de gelegenheid te geven met METAFONT kennis te maken, heb ik een selectie gemaakt uit PD disks B76 en B77. Daarmee zijn de fonts cmr10 (romein), cmbx10 (vet) en cmsl10 (schuin) willekeurig te vergroten. Alle andere TEX fonts (zie hoofdstuk 1) kunnen natuurlijk ook met METAFONT opgeblazen worden, maar daar zijn weer vele andere hulpbestanden voor nodig. Dat zou niet meer op één schijfje kunnen.

3.1 Installatie

Als er minstens 2 Mb RAM is, start de ST dan op met schijf B144 in drive A. Er wordt een RAM-disk van 1 Mb aangemaakt waarop je de folder METAFONT copieert (niet alleen de inhoud, maar folder mèt inhoud). De RAM-disk is de werkschijf. Als er maar 1 Mb RAM is (met minder gaat 't niet), copieer dan de folder METAFONT naar een lege schijf; dat wordt dan de werkschijf. Als er een harde schijf is, kan die als werkschijf gebruikt worden.

Open de folder METAFONT. Het hoofdprogramma is MF.TOS. Open de folder INPUTS. Hier staan 17 hulpbestanden verpakt in MF.ARC (omdat ze zo veel beter te copiëren zijn). Start UNARC.TTP, tik

```
mf.arc
```

en een `a` om alle files uit te pakken. Wis daarna eventueel MF.ARC om meer ruimte te maken op de werkschijf.

3.2 Toverspreuk ...

Als voorbeeld gaan we van het font cmr10.pk, dat op de TEX werkschijf in de folder FONT0096 staat, en 20% vergroot in de folder FONT0115, nog eens twee stappen van 20% verder opblazen. Start MF. TOS, op het scherm verschijnt dan

```
This is METAFONT, ......
**
```

METAFONT verwacht nu instructies over de base, de mode, de magnification en het font. Daartoe tik je achter de sterren de volgende toverspreuk:

```
&plain \mode=atari; mag=1.728;
                    input cmr10
```

(op één regel, dus [Return] na cmr10, niet vóór input). METAFONT gaat dan 128 tekens van het font cmr10 maken in de vergroting $1.2 \times 1.2 \times 1.2 = 1.728$ voor het Atari beeldscherm. Dat duurt ongeveer drie kwartier ...er moet heel wat gerekend worden! Op het scherm worden alle tekens en de gebruikte hulpbestanden vermeld. Het proces kan afgebroken worden met [Control] C of (voor oudere TOS-versies) met de Reset-knop (geen probleem, de RAM-disk is Reset-vast).

Na 3/4 uur staan er drie nieuwe bestanden op de werkschijf: CMR10.GF, CMR10.LOG en CMR10.TFM. In het LOG-bestand kan je nalezen wat op het scherm verschenen is. Het .GF font moet nog gecomprimeerd (packed) worden tot een .PK font. Start GFTOPK.TTP en tik

```
cmr10.gf cmr10.pk
```

Even later staat CMR10.PK op de werkschijf. CMR10.GF en CMR10.LOG mogen gewist worden, CMR10.TFM en CMR10.PK moeten gecopieerd worden naar de TEX-werkschijf. Zie onder het kopje 'Gebruik'.

3.3 ...verklaard

Eerst enige toelichting op de toverspreuk. `&plain` betekent dat het bestand plain.base (door TOS afgekort tot PLAIN.BAS) gebruikt moet worden. Dat geldt voor alle 16 standaard fonts van TEX.

`\mode` specificeert het scherm of de printer. In het bestand ATARI.MF staan achter mode_def de mogelijkheden, o.a. `atari` voor het scherm en `fx_eighty` voor een Epson-compatibele printer.

De vergroting wordt opgegeven met `mag`. Gebruikelijk zijn machten van 1.2, dus 1.2, 1.44, 1.728, ..., maar `mag=2` kan ook.

Na `input` komt het font. Door het beperkte aantal MF-bestanden op de werkschijf bestaat de keuze hier uit cmr10, cmbx10 en cmsl10.

3.4 Gebruik

Om het zojuist aangemaakte schermfont cmr10 in de vergroting 1.728 te gebruiken copieer je CMR10.TFM naar de TEX-werkschijf. Ook CMR10.PK moet op die werkschijf, en wel in een folder FONT0166. Het niet vergrote schermfont CMR10.PK staat in FONT0096, het 1.2 maal vergrote font in FONT0115 ($115 \approx 1.2 \times 96$), dus het 1.728 maal vergrote font moet in een folder FONT0166 omdat 96×1.728 afgerond 166 is. Het bij dit schermfont behorende Epson-printerfont moet in FONT0415 komen, reken maar na.

Het is zaak om een nieuw font direct in de juiste folder te zetten, omdat aan de naam van het font de mode en de magnification niet af te lezen zijn!

Neem nu een TEX-bestand, b.v. STAP_1.TEX, en tik aan het begin

```
\font\kop cmr10 at 17.28 true pt
```

Dit definieert het font 'kop' (willekeurige naam), dat je vervolgens kunt gebruiken om een kopje te maken met

```
{\kop Dit Is Een Titel}
```

Centreren gaat met

```
\centerline {\kop Titel}
```

Bekijk het resultaat op het scherm. Soms rondt TEX anders af bij het berekenen van de foldernaam; dan komt er een verkeerd font op scherm of printer. Experimenteer dan met de foldernaam, b.v. 0165 i.p.v. 0166.

Met magnification, dat maar éénmaal gebruikt mag worden in het begin van een TEX bestand, is de héle tekst in 17.28 punts font te zetten:

```
\magnification 1728
```

Als je over de nieuwe versie van DVI_ST.TTP beschikt (die heet DVI_VIEW .TTP en is verpakt in PREVIEW.LZH op PD disk B100), dan kan je een groot font ook gebruiken als loep om kleine lettertjes groot op het scherm te krijgen. Neem b.v. STAP_1.TEX en verwijder eventuele magnification commando's, zodat het in een 10 punts lettertje op het scherm komt. Start dan DVI_VIEW.TTP en tik

```
stap_1 -v=166
```

Dan is de preview 1.728 keer vergroot! De –v notatie is afkomstig uit de wereld van C en UNIX.

3.5 Regelafstand

Met het pas gemaakte 17.28 punts font kan natuurlijk ook een langere tekst gezet worden, maar dan moeten we de regelafstand, die normaal op 12 punt staat ingesteld, vergroten. Als dat niet gedaan wordt zet TEX de regels zo dicht mogelijk tegen elkaar.

```
{\kop \baselineskip 21 true pt
... lange tekst ...
}
```

Maak de baselineskip ongeveer 20% groter dan de font-afmeting.

Als je de hele tekst in `magnification 1728` zet wordt de regelafstand automatisch vergroot.

3.6 Een flikje aflikken

Als TEXnoloog zie je dat in 'flikje' de fl-ligatuur gebruikt is en in 'aflikken' niet. Zo hoort 't volgens sommigen en zo kan 't ook in TEX: `Een flikje af\/likken`.

Maar ... met het zojuist gemaakte font gaat er iets mis.

Vervang `Titel` maar door `fffiets`, je ziet dat 't niet werkt, er staat fffiets i.p.v. fffiets. Dat komt omdat Christoph Strunk van mening is dat de f-ligaturen in het Duits meestal storend zijn. Hij heeft de TFM-bestanden (TEX Font Metric) aangepast.

Als je de f-ligaturen wel wilt hebben, dan moet je het volgende doen: Vervang het zojuist gemaakte TFM-bestand en PLAIN.FMT op je TEX-werkschijf door de folder TFM en PLAIN.FMT op schijf B144. In de folder TFM staan de originele TFM-bestanden voor de 16 standaard fonts, verpakt in TFM.ARC ... uitpakken dus. Het nieuwe bestand PLAIN.FMT heb ik met deze TFM-bestanden gemaakt met INITEX. Gebruik de door METAFONT gemaakte TFM-bestanden niet. Probeer de fffiets nog eens.

3.7 Letters schuiven

Bij grote letters ontstaat de behoefte om sommige iets dichter tegen elkaar, of verder uit elkaar, te zetten dan TEX doet. Om 1 punt meer wit tussen twee letters te maken tik je daar het commando `\kern 1 pt` tussen, en met `\kern -1 pt` haal je 1 punt wit weg. Dus met

```
KOOS K\kern-2pt O\kern-5pt OS
```

krijg je KOOS KOOS.

Met raise en lower kunnen letters omhoog en omlaag gezet worden, maar dan moeten ze in een box zitten, b.v.

```
KO\lower2pt\hbox{O}S
```

levert KOOS.

Bij kern, raise en lower is het beter geen punten of millimeters te gebruiken maar maten die meeschalen met het font. Dat zijn de em, de breedte van de M, en de ex, de hoogte van de x. Met deze maten verschoven letters kunnnen ook in een ander font gebruikt worden.

Bekijk nu eens:

```
T\kern-.1667em\lower-.5ex
       \hbox{E}\kern-.125em X
```

Het TEX logo!

3.8 PostScript

PostScript (PS) is, net als TEX en METAFONT, een computertaal waarin beschreven wordt wat op een pagina afgedrukt moet worden. PS is ontwikkeld door de Amerikaanse firma Adobe. PS is nu een standaard voor professionele printers en zetmachines. Het is gemakkelijker een drukkerij te vinden die een floppie met een PS-bestand verwerken kan dan eentje die TEX- of DVI-files accepteert.

Geen nood, Tomas Rokicki e.a. hebben een programma geschreven dat DVI- in PS-bestanden omzet. De Strunk-versie van dit programma, DVI_PS.TTP staat in de folder PS op schijf B144. DVI_PS.TTP heeft het bestand DVI_PS.PS nodig, en fonts voor de PS printer, en véél ruimte omdat PS-bestanden groot zijn.

Het gebruik van DVI_PS is vrij simpel. Copieer de inhoud van de folder PS naar een aparte PS werkschijf, en zet daar het te converteren DVI-bestand bij, b.v. STAP_1.DVI. Maak de nodige fonts met METAFONT en zet die in de juiste folders op de PS-werkschijf. Als voorbeeld staat in FONT0300 het cmr10 font voor een 300 dpi laserprinter, gemaakt met `\mode=laserjet` (zie ATARI.MF in de folder INPUTS). Start DVI_PS.TTP en tik (`o` staat voor output)

```
stap_1 -o=.ps
```

Dat levert STAP_1.PS.

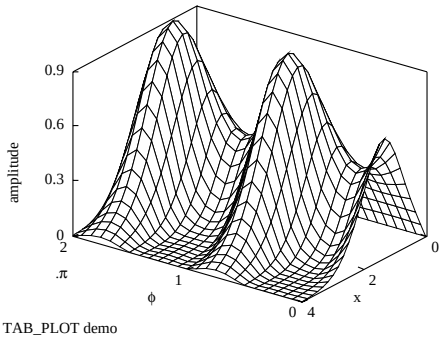
3.9 Figuren

Terwijl TEX sterk is in het zetten van tekst en formules, is PS goed voor tekenwerk. TEX en PS vullen elkaar uitstekend aan. Een tekening in PS kan makkelijk in een TEX-bestand verwerkt worden. Het hier afgebeelde grafiekje is zo in het TEX-bestand verwerkt: In het begin van het bestand staat

```
\input EPSF
```

en waar het grafiekje moet komen staat

```
\epsfxsize 58 true mm
\epsfbox{3D.PS}
```



Deze commando's maken een vbox van 58 mm breed en passen daar het grafiekje in, beschreven door 3D.PS. Let op het bijschrift langs de vertikale as: gedraaide tekst kan wel in PS maar niet in TEX. De macro EPSF.TEX staat ook in de folder PS; daarin worden epsfxsize en epsfbox gedefinieerd. De preview, die alleen laat zien wat in TEX beschreven is, toont lege ruimte waar het PS plaatje moet komen.

PS is een taal met vrij eenvoudige maar krachtige commando's. Een PS-bestand wordt meestal niet direct getikt maar automatisch gegenereerd door een tekenprogramma. Er zijn voor de ST vele tekenprogramma's, maar helaas maar enkele (dure CAD) die hun uitvoer als PS-bestand kunnen leveren i.p.v. direct naar scherm of printer. Het grafiekje is gemaakt met TAB_PLOT op PD disk A264, een programma dat ik heb geschreven om grafieken in (natuurkundige) artikelen te kunnen opnemen. In het volgende hoofdstuk zal ik terugkomen op het probleem om figuren te zetten met TEX.

3.10 Shell

Een veel voorkomend probleem met TEX en METAFONT is de 'onvindbaarheid' van hulpbestanden. Als je b.v. de naam van de folder METAFONT verandert,

en dan MF.TOS start, dan werkt 't niet, omdat METAFONT zijn hulpjes niet meer kan vinden. Christoph Strunk heeft programma's gemaakt die o.a. dit probleem oplossen. Op PD schijf B76 staat C_MF.PRG, een menu gestuurd programma dat je met muisklikken kunt leren waar alles te vinden is en wat METAFONT moet doen. C_MF zorgt dan dat METAFONT alles kan vinden en de juiste instructies krijgt zonder toverspreuken. Zo'n intermediair heet in computerees een shell of interface. C_MF stelt je ook in staat een aantal fonts na elkaar te laten maken, zodat de Atari 's nachts nuttig werk kan doen.

De nieuwe TEX versie op disk B98–105 wordt helemaal gestuurd vanuit de shell CTEX.PRG die, op toetsdruk of muisklik, de editor, TEX.TTP, de preview en printer programma's, of C_MF.PRG oproept. Van ontbrekende fonts voor een TEX-bestand wordt desgewenst een lijst gemaakt die C_MF automatisch afwerkt. De nieuwe versie van MF.TTP werkt drie maal zo snel en produceert TFM-bestanden mèt f-ligaturen. Allemaal heel mooi (hoewel ik soms in de vele menu-items van CTEX en C_MF het spoor bijster ben), maar zoals gezegd, dit werkt alleen met een harde schijf waarop al gauw 5 Mb ruimte nodig is.

3.11 Literatuur

Donald E. Knuth, *The METAFONTbook*.
Adobe Systems Inc., *PostScript Language*, 2 delen: Reference Manual en Tutorial and Cookbook.
Alles uitgegeven bij Addison-Wesley.

– – * – –

4 Figuren en een vreemd alphabet

Het zetten van figuren in TEX wordt vervolgd. We testen twee tekenprogramma's op hun bruikbaarheid om lijntekeningen voor een TEX-document te maken. En we doen TEX op z'n Russisch met Cyrillische letters, ook op het scherm.

4.1 Figuren

Via PostScript zijn we in hoofdstuk 3 begonnen met het zetten van figuren. TEX kan mooie figuurtjes zetten, als die met METAFONT gemaakt zijn: letters en zo. Voor andere figuren heeft TEX niet veel mogelijkheden. Je kunt natuurlijk met

```
\vskip 5 cm
```

ruimte maken om een plaatje in te plakken, maar het kan beter. TEX heeft een commando

```
\special{...}
```

waarmee ... in het DVI-bestand tussengevoegd kan worden. Dit commando wordt b.v. in EPSF.TEX toegepast om in het DVI-bestand de informatie ... door te geven waarmee DVI_PS de PS-tekening kan maken. Er kleeft een nadeel aan het special-commando. Het werkt met speciale DVI-programma's, dus het verstoort de apparaat-onafhankelijkheid. Het DVI-bestand werkt niet meer met een willekeurig DVI-programma op een willekeurige computer (zie inleiding in hoofdstuk 1).

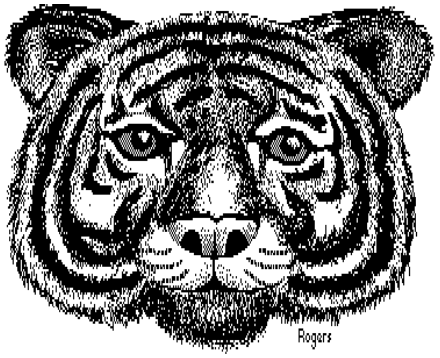
4.2 CS-grafiek

Met de DVI-programma's van Christoph Strunk zet je een IMG-tekening, b.v. de bekende tijger van WordPlus, als volgt:

```
\special {CS!r}
\vbox to 50 true mm {\vfil
\special {CS!g 0.45 TIGER.IMG}}
```

Dat levert de tijgerkop als het bestand TIGER.IMG op de werkschijf staat. Met CS! begint ieder CS-grafisch commando. Het eerste special-commando is de Reset voor CS-grafiek. Er wordt een box van 50 mm hoog voor de tijger gemaakt. De kop is met een factor 0.45 geschaald. De juiste waarden zijn makkelijk experimenteel te bepalen met de preview. Het werkt niet alleen met DVI_ST maar ook met DVI_PS, dus op deze manier zijn IMG-bestanden te converteren naar PS-bestanden!

IMG-figuren zijn met vele tekenprogramma's te maken en andere formaten (PIC, PI3, PC3, PAC, enz.) zijn te converteren naar IMG. Dit zijn allemaal rastertekeningen (bestaand uit losse punten), vaak goed genoeg, maar bij vergroting wordt het beeld grover (punten worden blokjes, letters worden lelijk). Voor goed drukwerk kan je beter uitgaan van lijntekeningen, maar er zijn niet veel lijntekenprogramma's voor de ST. Ik bespreek er hier kort twee.



4.3 Silhouette

Silhouette is geschreven door de Amerikaan Tim Reyes. Ik heb versie 1.40 (1992) gekocht voor £ 60.– bij Ladbroke Computing, (33 Ormskirk Road, Preston, Lancs. PR1 2QP, UK) omdat ik een tekenprogramma met PostScript uitvoer nodig heb. (Tussen haakjes, £ maak je in TEX met {\it\$}.) De handleiding van ca. 100 pagina's is gevat in een degelijke ringband. Het programma staat op één schijfje en werkt al op een ST met 1 Mb RAM zonder harddisk. GDOS en het nieuwere FONTGDOS worden bijgeleverd. Eén van de twee moet geïnstalleerd zijn voor lijntekenen. Silhouette kan rastertekeningen maken en omzetten in lijntekeningen, maar hier wordt alleen lijntekenen besproken.

Om een lijntekening te maken moet eerst het Vector

Window geopend worden. Het tekenen van rechte lijnen, cirkels, rechthoeken en ellipsen (met horizontale as) gaat probleemloos. Het vullen van veelhoeken met een vulpatroon gaat ook goed, maar de handleiding is onduidelijk over de ikonen voor transparant en dekkend vullen. Fraai is dat de grijswaarde van een vulling op de PostScript afdruk exact in te stellen is. Ellipsen kunnen gevuld worden na omzetting in een veelhoek. Draaien van een ellips is me pas gelukt na conversie in 4 punt Bézier krommen, wat niet vermeld is in de handleiding.

Elliptische bogen worden gedefinieerd door begin- en eindpunt en het middelpunt van de hele ellips met horizontale as. Het tekenen van een boog gaat mis als de rechte lijn van begin- naar eindpunt horizontaal of vertikaal loopt. Daar is de handleiding ook niet duidelijk over.

Onoverkomelijke problemen had ik met het selecteren van lijnen. Selecteren is nodig als je een b.v. een getrokken lijn gestippeld wil maken. Silhouette geeft selectie aan met een rechthoek die de geselecteerde lijn bevat. Maar die rechthoek is gelijk voor verschillende boogjes van dezelfde ellips!

Een lijntekening kan op schijf gezet worden o.a. als GEM-, EPS- (PostScript) of DXF-bestand (Data Exchange File voor CAD).

4.4 ZPCAD

ZPCAD heeft een uitvoer mogelijkheid in CS-grafiek. Versie 1.00 (1990) van ZPCAD staat op de PD-schijven B 114/115. Het is shareware: de auteur, Burkhard Strauß, vraagt DM 100.- aan serieuze gebruikers. Het is mogelijk ZPCAD met maar 1 Mb RAM en zonder harddisk te gebruiken, maar op de PD-disks staan zelfuitpakkende archief-bestanden die 3 Mb ruimte nodig hebben. Dat levert de handleiding van ca. 140 pagina's als DVI-bestand met de nodige fonts en printer drivers. Verder het hoofdprogramma ZPCAD.PRG en een reeks hulpbestanden. "ZPCAD ist das CAD Programm mit dem ultimativen Konzept für den ATARI ST" zegt de README. Niet onder de indruk, en niet gehinderd door enige kennis van Computer Aided Design, dus geheel onbevooroordeeld ben ik verder gaan testen.

ZPCAD start op met een helptekst, en door dubbelklikken op ikonen zijn tijdens het tekenen steeds helpteksten op te roepen. Aan de bovenste regel wordt steeds aangegeven welke muistoets gebruikt moet worden. Het tekenen van lijnen, cirkels, rechthoeken en ellipsen gaat makkelijk. Een gestippelde lijn maken lukte niet. Uiteindelijk bleek het wel gelukt, maar niet zichtbaar op het scherm. Als je wilt 'zien' of een lijn gestippeld is, dan moet je eerst de lijn selecteren, dan bij attributen op het knopje GET klikken, dat inverteert een genummerd knopje [3] onder TYP, en met nog een klik op een ikoon komt er een helptekst waarvan blad 4 vermeldt dat lijntype 3 gestippeld is.

Tekenen met ZPCAD kan ook door commando's te tikken op de onderste regel. Om b.v. een ellips te draaien selecteer je de ellips en het draai-middelpunt, en dan tik je

```
MOVE_ROTATE(30)
```

Dan draait de ellips 30 graden tegen de wijzers van de klok.

ZPCAD is bedoeld om plotpennen aan te sturen. Een vlak vullen met een grijswaarde kan dus niet. Gearceerd is het enige vulpatroon, maar arceren kan onder willekeurige hoek en met willekeurige lijnafstand. Het heeft mij wel hoofdbrekens bezorgd. Een tekening kan uit verschillende lagen opgebouwd worden, en ZPCAD reserveert een paar lagen voor arceren. Pas nadat de arceerlaag uitgeschakeld is met een muisklik kan in de 'gewone' tekenlaag gearceerd worden.

Een tekening opbergen als CSG-bestand (CS-Grafiek) is ook niet simpel. Eerst I/O aanklikken en het 'Plotter Format' TEX.ZPF laden. Dan de afmeting van de tekening in mm bepalen met 'Plotter Format Einstellen'. Om de tekening een rubberband trekken en (per muisklik) aanpassen aan de opgegeven afmeting. Weer I/O klikken en met 'Plotter Daten Speichern' de tekening op schijf zetten als b.v. ELLIPS.CSG. Het is verwarrend dat dit allemaal niets met plotters te maken heeft, maar ZPCAD beschouwt TEX als een plotter.

Nu kan je de tekening in TEX zetten met:

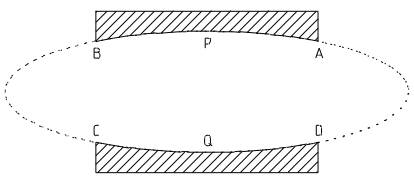
```
\special {CS!r}
\special {CS!u 1 mm}
\vbox to 30 true mm {\vfil
\special {CS!i ELLIPS.CSG}}
```

Dat levert de ellips-tekening. Het tweede special-commando stelt de lengte eenheid (unit) in. Het is te gebruiken voor schaling: 1.2 mm i.p.v. 1 mm vergroot de figuur met 20%.

ZPCAD kan ook figuren als GEM- of DXF-bestand lezen en schrijven, maar de benodigde ZPF-bestanden ontbreken op de PD-disks.

4.5 Test

Het bovenstaande is natuurlijk geen complete test van ZPCAD. Daarvoor is het programma te veelomvattend. De commandotaal is zeer uitgebreid en kan door de gebruiker verder ontwikkeld worden. Ik heb slechts getest op bruikbaarheid voor het opnemen van een lijntekeningetje in een TEX-document. Daarvoor is ZPCAD nogal groot en moeilijk te bedienen.



Silhouette is makkelijk in het gebruik, maar nogal beperkt. De toch niet zo ingewikkelde ellips-tekening is er niet goed mee te maken. De elliptische bogen AB en

CD moeten in delen gemaakt worden, dus AP, PB, CQ, QD. Als er dan aan één van de boogjes iets veranderd moet worden is het ondoenlijk om het juiste boogje te selecteren, omdat verschillende boogjes met dezelfde rechthoek aangeduid worden. In ZPCAD wordt een geselecteerd lijnstuk gestippeld, dus dat is eenduidig. Silhouette combineert slecht met Strunk-TEX dat GEM-bestanden niet goed kan lezen. Je moet dus de PostScript uitvoer gebruiken en dan zie je het plaatje in de tekst pas op papier. Met ZPCAD zie je het al in de preview.

Ik kies ZPCAD, een prachtig programma, al snak ik wel eens naar een eenvoudig voorbeeldje in de doorwrochte dubbelkoloms Duitse handleiding . . .

4.6 Cyrillisch

Nu iets heel anders. Hoe zet je tekst in een taal met een eigen alphabet, b.v. Grieks of Russisch? Elke letter intikken met commando's als `\alpha \beta` is te omslachtig. Als je een Russische tekst tikt, wil je ook Cyrillische letters zien op het scherm. Bovendien heeft het toetsenbord van een Russische typmachine een eigen indeling. Dus we hebben niet alleen Cyrillische TEX-fonts voor preview en printer nodig, maar ook een Cyrillisch schermfont voor de editor en een aanpassing van het toetsenbord. Wel, dat kan allemaal best op een ST! De nodige bestanden om TEX op z'n Russisch te doen staan op disk ST42 (verkrijgbaar via de Stichting ST) in de folder CYRILLIC.

TEX-fonts voor Cyrillische letters, die Russen en andere Oosteuropeanen schrijven, zijn gemaakt door Nana Glonti en Alexander Samarin. Beschikbaar zijn de MF-bestanden waarmee METAFONT Cyrillische fonts kan maken. Deze MF-files staan in een folder INPUTS die op de METAFONT werkschijf moet komen (zie hoofdstuk 3). Dan kan je in de 'toverspreuk' de opdracht `input cmcyr10` geven om een Cyrillisch font te maken. Of `cmcbx10` voor vette of `cmcsl10` voor schuine letters.

Omdat het Cyrillische alphabet 32 letters bevat worden in het Cyrillische font meer plaatsen voor letters gebruikt dan in een Latijns font. De extra plaatsen moet bij TEX aangemeld worden met het catcode-commando waarmee een fontplaats (ASCII) in een categorie ingedeeld wordt. Categorie 11 is letter. In de folder CYRILLIC staat CYR_TEST.TEX waarin de macro 'rusk' gedefinieerd wordt met de nodige catcodes. Daar staan ook de fonts cmr10 en cmcyr10 voor de preview van CYR_TEST. Helaas werken de catcodes niet goed met de oude TEX.TTP en PLAIN.FMT. Het werkt wel met de nieuwe TEX op PD-schijven B 98-101. Daarbij moet PLAIN.FMT met IniTEX gemaakt worden. Hoe dat gaat zien we in het volgende hoofdstuk.

4.7 Tempus

We besluiten nu met schermfont en toetsenbord. Ik heb die aangepast voor Tempus, maar er zijn andere goede editors op de ST waarmee het wellicht ook kan. Met WordPlus gaat het niet.

Tempus kan een ander font laden met '$8*16$er laden' in het menu Speziell. Met de fonteditor Olifont (geleverd bij STAD) heb ik CYRILLIC.FNT gemaakt. Met 'System-Zeichensatz' in hetzelfde menu is weer terug te schakelen.

Tempus heeft een hulpbestand QUELLDAT.INS dat je al hebt moeten aanpassen als je een Duitse Tempus op een Nederlandse Atari (met UK TOS) wilt gebruiken (Y en Z verwisselen enz.). Om de aangepaste indeling te installeren moet je QUELLDAT.INS inlezen en op 'Quelldaten übersetz.' klikken in het Parameter menu. Op de zelfde wijze is het toetsenbord Russisch te maken:

1 2 3 4 5 6 7 8 9 0 - = ‘
й ц у к е н г ш щ з х ъ
ф ы в а п р о л д ж э
я ч с м и т ь б ю /

En met Shift:

! " : ; % , . ( ) - +
Й Ц У К Е Н Г Ш Щ З Х Ъ
Ф Ы В А П Р О Л Д Ж Э
Я Ч С М И Т Ь Б Ю ?

Om terug te schakelen naar onze toetsenbordindeling moet je Tempus verlaten en weer starten.

Er ontbreken nu nog Cyrillische letters op de toetsen. Dat is een kwestie van plakkertjes maken. En Russische afbreekpatronen. Die heb ik niet kunnen vinden. Een oplossing is TEX met de opdracht

```
\hyphenpenalty 10000
```

het afbreken te verbieden, en het zelf te doen waar nodig.

– – * – –

5 INITEX en INIMF

We gaan 'updaten' naar TEX 3.1 en METAFONT 2.7. INITEX en INIMF worden behandeld. Daarmee komen we tot een afronding van plain-TEX.

5.1 Nieuwe TEX

In deze cursus is gebruik gemaakt van selecties uit de oude PD TEX van Christoph Strunk. Deze versie is echter traag, de preview heeft geen v-optie (zie hoofdstuk 3) en de catcodes werken niet goed (hoofdstuk 4). Het is mogelijk de bestanden uit de oude versie te vervangen door nieuwe op de PD-schijven B98–105. Wie een harddisk heeft, werkt waarschijnlijk al met de nieuwe versie.

Wie geen harddisk heeft, kan dat ook, en het is echt aan te bevelen. Het is wat lastiger, omdat het installa-

tieprogramma onbruikbaar is i.v.m. de beperkte schijfruimte. Je moet de bestanden stuk voor stuk uit de LZH-archieven halen zoals hieronder wordt beschreven, en op de plaats van de oude zetten. Omdat een mengsel van oude en nieuwe bestanden allerlei problemen kan geven, is het goed meteen alles te vervangen (behalve eigen TEX-bestanden natuurlijk).

Er zijn een paar verschillen in bestandsnamen tussen de oude en de nieuwe versie. DVI_ST.TTP heet DVI_VIEW.TTP in de nieuwe versie, en PLAIN.BAS is omgedoopt tot PLAIN.BSE. Het METAFONT programma MF.TTP heeft dezelfde naam op B76 en B102, maar het is handig om .TTP in .TOS te veranderen, omdat de 'toverspreuk' te lang is voor het paneel dat op het scherm komt bij het starten van een TTP-programma.

5.2 LZH

LZH-bestanden ontstaan bij comprimeren van files volgens methodes van Lempel, Ziv en Huffman, verder ontwikkeld in Japan tot een archiveringsprogramma. De Atari-versie van het programma (ook bekend als LHarc) is LZH.TTP op PD-schijf B116.

Laten we met dit programma eens kijken wat er in TEX31.LZH op B98 staat. Zet LZH.TTP met TEX31.LZH op een werkschijf, start LZH.TTP en tik

```
l tex31 -h
```

Dat levert een scherm vol bestanden. De `l` betekent list en de `h` staat voor hold; als je `-h` weglaat, krijg je geen tijd om de lijst te lezen. Bijna onderaan staat TEX.TTP. Om die uit het archief te halen (extraheren) verlaat je LZH.TTP, start hem opnieuw en tik

```
x tex31 tex.ttp
```

Na 'ontdooien' staat TEX.TTP op de schijf. Als je TEX.TTP weglaat, wordt het hele archief uitgepakt. Dat is misschien meer dan op de werkschijf kan.

5.3 INITEX

Het programma TEX.TTP heeft een format nodig, b.v. PLAIN.FMT.

INITEX is een programma dat een reeks commando's, font-structuren, parameterwaarden, enz. in PLAIN.TEX, en een lijst van afbreekpatronen in HYPHEN.TEX, comprimeert tot een format. INITEX kan desgewenst ook een ander format maken met andere afbreekpatronen of met andere commando's (b.v. voor LATEX).

INITEX werkt als volgt. Zet bij elkaar op een werkschijf:
INITEX.TTP, PLAIN.TEX, HYPHEN.TEX, de boodschappenlijst TEX.POO en de folder TFM met de 16 standaardfonts.
Start INITEX.TTP en tik

```
plain \dump
```

Dat levert PLAIN.FMT.

INITEX.TTP en TEX.POO zijn te vinden op schijf B98 in TEX31.LZH. Voor HYPHEN.TEX kan je kiezen uit EHYPHEN.TEX (English) en GHYPHEN.TEX (German) in INITEX.LZH op dezelfde schijf, en NLHYPHEN.TEX op B119.

PLAIN.TEX zit in INITEX.LZH op B98, maar Strunk heeft dit bestand aangepast. Ongeveer 15 regels voor het einde moet staan

```
\input hyphen
```

waarmee het bestand HYPHEN.TEX ingelezen wordt. Strunk heeft hier een % voor gezet om formats via de shell te kunnen maken (zie hoofdstuk 3). Als je dus PLAIN.FMT op de hier beschreven wijze wil maken, moet dat procentteken weer weg.

De TFM-bestanden in TFM.LZH op B98 zijn door Strunk aangepast om f-ligaturen te verwijderen (zie het 'flikje' in hoofdstuk 3). De originele files staan in TFM_ORI.LZH op B99.

5.4 Taalkeuze

Als je verschillende formats hebt gemaakt voor verschillende talen, dan heeft Strunk-TEX een eenvoudig keuze mechanisme. Zet de formats bij TEX.TTP b.v. als PLAIN.FMT voor Engels, PLAING.FMT voor Duits en PLAIND.FMT voor Nederlands. Zet als eerste regel in het TEX-bestand als het een Nederlands stuk betreft:

```
%format plaind
```

Dan wordt PLAIND.FMT i.p.v. PLAIN.FMT geladen.

5.5 INIMF

Voor METAFONT bestaat een programma INIMF, analoog aan INITEX voor TEX. Het programma MF.TOS heeft een base PLAIN.BSE nodig, die als volgt wordt gemaakt met INIMF.

Zet bij elkaar op een werkschijf:
INIMF.TTP, PLAIN.MF, ATARI.MF en MF.POO. Start INIMF en tik

```
plain \input atari; dump
```

Dat geeft PLAIN.BSE.

5.6 Wat nu?

We zijn gekomen tot een zekere afronding in deze cursus. Alle bestanden voor plain-TEX zijn behandeld. Ik heb laten zien hoe je deze basis-TEX zonder veel kosten kunt installeren op een eenvoudige computer. Het is een bijzonder krachtig zetsysteem, krachtiger dan menig duur DeskTop Publishing pakket. Het is niet makkelijk, maar die DTP's zijn ook niet simpel. Op de PD disks B98–105 staan veel meer bestanden, om meer fonts te maken, en de macropakketten LaTEX en $\mathcal{AMS}$-TEX om (wetenschappelijke) artikelen makkelijker te kunnen zetten.

Gezeefd uit de TEX-NL discussielijst

Philippe Vanoverbeke

Abstract

Gezeefd uit de NTG TEX-NL discussielijst van 1992 en begin 1993: een zestiental vragen én antwoorden. Niet alleen voor de niet-netwerkers doch ook voor de TEX-NL subscribers die de berichten te snel langs hun heen zagen gaan.
Keuze is gemaakt op persoonlijke titel, veel is dus ongetwijfeld missende. Echter de onderwerpen welke in deze bijdrage worden behandeld zijn zeker van algemeen belang.

Inhoud

1 De HP Laserjet III tekent te dunne lijntjes, wat is daar aan te doen?

Gebruikers van de Laserjet III en IIIP beklagen zich soms dat de afdrukkwaliteit niet voldoet. Meer bepaald wordt er geklaagd over (veel) te dunne lijnen. Oorzaak is de RET (**R**esolution **E**nhancement **T**echnology). Deze heeft evenwel vier modes : DARK? MEDIUM? LIGHT en OFF (wel, eigenlijk maar drie dus).

Na enig experimenteren is gebleken dat MEDIUM en LIGHT minder goede resultaten opleveren, DARK en OFF goede (aanvaardbare) daarentegen voldoen dan wel (met DARK zijn de lijntjes natuurlijk smoother en daardoor visueel ietsje dunner, maar het resultaat oogt mooier).

2 Hoe kan ik een dvi file afdrukken op een printer waar geen emTEX voorhanden is?

De dvi-drivers van emTEX kun je naar een file laten printen door de optie `/pofile.ext` te geven.

Met de optie `/po=myfile.dot` wordt al het nodige naar het bestand `myfile.dot` weggeschreven i.p.v. lpt1. Dat bestand zet je op een diskette waarna het afgedrukt kan worden met copy `/b myfile.dot prn` op een andere machine zonder dat daar emTEX geïnstalleerd is. Noch met DVIDOT als DVIHPLJ schept dit problemen.

3 Hoe krijg ik 2 EPS-plaatjes naast elkaar met een afzonderlijke figure-caption?

Dit kan op de volgende manier:

```
\begin{figure}[tbp]
   \begin{minipage}[t]{0.47\textwidth}
      \begin{center}
      \leavevmode
      \setlength{\epsfxsize}{%
                        0.8\textwidth}
      \epsfbox{fig1.eps}
      \caption{.....}
      \label{fig: ....}
      \end{center}
   \end{minipage}\hfill
   \begin{minipage}[t]{0.47\textwidth}
```

```
      \begin{center}
      \leavevmode
      \setlength{\epsfxsize}{%
                        0.8\textwidth}
      \epsfbox{fig2.eps}
      \caption{...}
      \label{fig: ...}
      \end{center}
  \end{minipage}
\end{figure}
```

4 Hoe maak ik een boekje in A5 formaat?

Dit gaat uitstekend met het programma `dvidvi`. Hiermee worden twee A5 pagina's op één A4 pagina afgedrukt (landscape). Het in de juiste volgorde zetten van de pagina's (b.v. pagina 1 naast pagina 80, pagina 2 naast pagina 79, enz) is tevens mogelijk.

Het programma `dvidvi` is te vinden vinden op `ftp.cs.ruu.nl`, in `[pub]/TEX/DVI`.

Een mogelijke pagina-layout is:

```
%
% De pagina layout, voor A5 paper,
% 10pt fontsize uit a5.sty van
% Mario Wolczko geknipt.
%
% Een sprekende kopregel hebben we
% niet:
\headheight=\z@
\headsep=\z@
%
% De afstand tussen de bovenkant van
% het papier en de eerste regel tekst
% moet wat kleiner worden dan voor A4
% (1 in=72pt)
\topmargin= -35.62truept
% De breedte van de text
\textwidth= 337truept
%
% De marges op even en oneven
% pagina's
\evensidemargin=-34.47truept
\oddsidemargin=-18.77truept
\advance\columnsep 5pt
%
\global\textheight 41\baselineskip
\global\advance\textheight by \topskip
%
% De ruimte voor de voetregel
\footskip=36truept
\footheight=12truept
%
```

En dan komt de grote truc:

```
dvidvi -m "4:-1,2(5.5in,0in)"
            <dvi-file> <dvi-deel1>
dvips -tlandscape -ta4 deel1
dvidvi -m "4:-3,0(5.5in,0in)"
            <dvi-file> <dvi-deel2>
dvips -tlandscape -ta4 deel2
```

Bovenstaande is een Unix shell script, maar het kan ook onder DOS of VMS. Het programma `dvidvi` is van Thomas Rockiki en verkrijgbaar op de meeste TEX-archieven.

5 HPGL plaatjes omzetten naar Metafont...

Er bestaat BM2FONT ofwel BitMap To Font, een MS-DOS programma waarmee verschillende grafische formaten (o.a. GIF, PCX) kunnen worden geconverteerd naar TFM- en PK-files. Het programma is te verkrijgen via de NTG.

Tevens bestaat er een programma 'Capture∩waarvan reeds melding werd gemaakt in een vorige MAPS (zie MAPS 91.1, blz. 94: 'CAPTURE turns HPGL files into PK format; a PC program, $130 from Micro Programs Inc., 251 Jackson Ave., Syosset, NY 11791').

6 Een (La)TEX file controleren op (Engelse) spelfouten...

Er bestaat een spellingchecker die door Arjen Merckens speciaal voor TEX-teksten is geschreven. Het is een Turbo-Pascal-programma en kan wat de auteur betreft best aan het public domain worden toegevoegd, c.q. door NTG verspreid worden.

Detex-en is niet nodig: hij 'kent' (La)TEX en verbetert fouten in je pure TEX-dokument. Overigens kunnen heel gemakkelijk talen toegevoegd of verwijderd worden.

De spelling checker is alleen voor MS-DOS computers, en de demo-versie verstaat alleen Nederlands.

7 MS-Windows schermen opnemen als tekening in TEX...

Er zijn diverse PD programma's voor windows beschikbaar (b.v. shoot) om een deel van je windows scherm op het clipboard te zetten. Maak vervolgens hier met paint een .PCX/.BMP file van. Vervolgens kun je hierop BM2FONT loslaten om een exacte kopie van een deel van je windows scherm in een (La)TEX document te includen.

Het werkt allemaal heel vlot en goed — zeer handig als je een handleiding bij een windows programma o.i.d. wil maken — maar dit dus voor enkel geldig met MS-Windows 3.0.

8 HPGL files opnemen met emTEX...

De oplossing is het converteren van de HPGL file naar een bitmap file. D.w.z. met behulp van een conversie programma (ik gebruik zelf het shareware programma PRINTGL) converteer je de file (zeg `plaatje.hpg`) naar een PCX bitmap file (`plaatje.pcx`). Heb je eenmaal de PCX file (stel dat het een plaatje is van 5cm breed en 3cm hoog) dan is deze op zeer eenvoudige wijze in je TEX tekst te voegen:

```
\begin{figure}[htbp]
  \setlength{\unitlength}{1cm}
  \begin{picture}(5,3)
    \put(0,3){\special{em:
               graph plaatje.pcx}}
  \end{picture}
  \caption{The caption of the figure}
\end{figure}
```

Het mooie van deze oplossing is dat je het plaatje kunt previewen en printen! Het nadeel is zeker het feit dat het resolutie afhankelijk is (d.w.z. op de matrix printer is het PCX plaatje groter dan op de laserprinter).

Wil je toch een device independent oplossing dan kun je m.b.v. het programma BM2FONT de bitmap file vertalen naar een heuze TEX font (met tfm en pk files).

9 Hoe krijg ik links en rechts een header en ook nog een footer in LATEX?

Gebruik fancyheadings.sty. Hartstikke makkelijk.

10 Hoe verkrijg ik het symbool 'groter of gelijk aan∩?

Dit en andere soortgelijke wiskundige symbolen staan in de fonts msam10 en msbm10. Deze fonts worden standaard geleverd bij $\mathcal{AMS}$-TEX.

11 Waar kan ik documentatie verkrijgen over het New Font Selection Scheme? Welke zijn de voordelen?

Daarvoor zijn twee mogelijkheden: er zijn in 1991 twee artikelen hierover gepubliceerd in TUGboat. De documentatie zit echter ook bij de distributie. Die kun je via ftp van één van de fileservers afhalen. (Utrecht, Stutgart, Aston).

Je hebt meer vrijheden om fonts te selecteren. Als je bij het oude schema zegt `\it\bf` krijg je een bold 'romein∩, terwijl bij gebruik van het NFSS dan een bold 'cursief∩font krijgt. NFSS onderscheidt 4, onafhankelijk in te stellen, grootheden:

```
\fontfamily{} (bijvoorbeeld cmr, cmtt,
                            Times, etc)
\fontshape{}  (bijvoorbeeld it, sl,
                               sc etc)
\fontseries{} (bijvoorbeeld m, b,
                             bx, etc)
\fontsize{}{} (bijvoorbeeld {10}{12pt}
                          of {12}{15pt})
```

een macro zoals `\tenbf` (die onder NFSS niet meer bestaat) kan geïmplementeerd worden als:

```
\def\tenbf{\fontfamily{\rmdefault}%
           \fontseries{\bfdefault}%
           \fontshape{n}%
           \fontsize{10}{12pt}%
           \selectfont}
```

Resteert het ophalen van de distributie; het in het TEXINPUTS pad zetten van de files, eventueel een aangepast `fontdef.local` maken en het bouwen van een format, nadat je `lfonts.tex` een andere naam (zoals `lfonts.old`) hebt gegeven. Je moet dan een paar maal filenaam opgeven tijdens het bouwen van het format. Hiervoor zit overigens ook documentatie in de distributie.

12 Hoe plaatst men het label van een item uit het description environment op een aparte regel?

Dus:

```
\begin{description}
\item[Aap] Dit is een beetje flauw.
\end{description}
```

zou niet tot gevolg moeten hebben (in grote lijnen):

Aap Dit is een beetje flauw.

maar wel:

Aap
Dit is een beetje flauw.

Dat kan door een `\hfill` tussen te voegen :

```
\begin{description}
     \item[Aap] \hfill \\
     Dit is een beetje flauw.
\end{description}
```

13 Ik wil in 'book'-style de referenties per hoofdstuk hebben en niet in één grote lijst op het eind. Kan dat in BibTEX?

Er zijn een aantal styles die dat doen: `chapterbib.sty` en `bibunits.sty` zijn de voor de hand liggende. `bibunits` kan ook per andere onderdelen dan hoofdstukken.

Verder is `bibperinclude.sty` er een die het per `\include` doet.

14 Weet iemand of er een programma dvi2fax bestaat?

HiJaak is een programma dat o.a. dvi files rechtstreeks kan faxen. Het is een commercieel programma.

15 Om uitdraaien op verschillende momenten van de dag te kunnen onderscheiden wil ik de tijd op elke pagina afdrukken. De tijd dient in standaardnotatie (uren en minuten) afgedrukt te worden. Wie weet hoe je dat doet?

```
%
% \begin{macro}{\now}
%    Hiertoe worden twee
%     \meta{count}-registers
%    gereserveerd om de uren en
%    de minuten in op te slaan.
```

```
%    \begin{macrocode}
\newcount\hh \newcount\mm
%    \end{macrocode}
%    In de \TeX-primitive \verb=\time=
% is het aantal minuten sinds
% middernacht opgeslagen. Hieruit
% wordt de tijd berekend.
%    \begin{macrocode}
\hh=\time \divide\hh by 60
\mm=\hh \multiply\mm by 60 \mm=-\mm
\advance\mm by \time
%    \end{macrocode}
%    Vervolgens kan de macro
%    \verb=\now= worden
%    gedefini"eerd.
%    \begin{macrocode}
\def\now{\number\hh:\ifnum\mm<10{}0%
                    \fi\number\mm}
%    \end{macrocode}
% \end{macro}
```

16 Weet iemand hoe in TEX een running head voor een woordenboek gemaakt kan worden? Dus collage — commentaar als collage het eerste en commentaar het laatste woord op de bladzijde is.

Hiervoor kan je het mark-mechanisme gebruiken. `\firstmark`, `\topmark` en `\bothmark` in plain TEX; `\markboth` en `\markright` in LATEX.

armTEX 3.14, een port van TEX voor de Archimedes

Mark J. Sinke

Mendelssohnstraat 5
5144 GD Waalwijk
marks@stack.urc.tue.nl

1 Inleiding

In dit artikeltje zal ik vertellen wat armTEX is en welke voor- en nadelen het systeem heeft. Het is geen uitvoerige opsomming van wat TEX kan, omdat de lezer geacht wordt daarvan (enigszins) op de hoogte te zijn. Ik zal me meer richten op de technische kanten van het werken met TEX op de Archimedes.

2 Specificaties en beperkingen

ArmTEX is een stabiele port van TEX 3.14 voor de Archimedes (alle modellen, waarschijnlijk inclusief 1 MB machines). De huidige versie heeft 832 kB geheugen nodig om in te runnen. Die versie is een zgn. `virtex`, dus een versie die alleen format files kan lezen. De `INITEX` heeft 1152 kB nodig, dus die draait zeker niet meer op een 1 MB machine. De `virtex` zou in principe (van de command line of zo) moeten kunnen draaien op een 1 MB machine. Ook zou `!Virtual`, een task window met virtual memory misschien uitkomst kunnen brengen. Ik heb nog niet de tijd gehad om dat uit te proberen. De huidige versie is een 16-bit TEX, i.e., een versie met 64 kB main memory. Ook de zgn. *triesize* is niet zó groot. Als `!Virtual` blijkt te werken, ligt ook een 32-bit versie binnen de mogelijkheden. Op het moment is het niet zo interessant om die te gebruiken, omdat hij ongeveer 3200 kB geheugen nodig heeft.

Ik heb TEX geport op een RISC OS 2 machine, waarop ik ook een aantal interessante `Alias$@RunType`s gedefinieerd had voor de filetypes die ik verzonnen had voor TEX, LATEX, GF, PK en TFM files. Deze blijken niet meer goed te werken onder RISC OS 3. Ik heb ook geen tijd gehad om dat uit te zoeken. Een vermoeden is dat het ligt aan een gewijzigd path-search mechanisme in RISC OS 3. Op het moment kan TEX gewoon in een task window worden opgestart in de goede directory, en dat werkt naar behoren.

3 METAFONT

METAFONT 2.7 heb ik niet geport. Mijn uitgangspunt bij het porten van TEX was de Unix TEX distributie. Ik werk dus met `web2c`, maar ik krijg dat programma niet meer helemaal correct aan het lopen. Op het moment doe ik het dus met de gegenereerde C-code van halverwege vorig jaar. Als ik METAFONT wil porten, moet ik ook een interface naar de WIMP schrijven etc. en op het moment zit ik niet zó ruim in de tijd. Vrijwilligers/belangstellenden zijn natuurlijk van harte uitgenodigd om mij te schrijven. Ze kunnen zo de sources etc. krijgen zoals ze op dit moment zijn.

Wil je METAFONT gebruiken, dan kun je de METAFONT (Pascal-) versie 1.0 van ArMaTuReS gebruiken. Deze heeft echter verschillende nadelen, nl. er is geen manier om het programma fatsoenlijk af te breken (de enige manier is METAFONT in een task window draaien, en `Kill` kiezen) en de nieuwere `ligtables` worden niet correct verwerkt.

4 Bugs

Allereerst: door problemen met `web2c` is het me nog niet gelukt een fatsoenlijke `triptex` te maken, dus de trip test heb ik niet gedraaid. (Ik vraag me dan ook af of ik de naam armTEX mag gebruiken voor mijn produkt, maar goed). De enige bug die ik heb kunnen vinden is het feit dat `\newlinechar` niet goed afgehandeld wordt, i.e., foutmeldingen van LATEX kunnen er i.p.v.

```
LaTeX error.
  See LaTeX manual for explanation
  Type H <return> for immediate help
! Float(s) lost.
```

zo uitzien:

```
LaTeX error.
  See LaTeX manual for explanation^^J
  Type H <return> for immediate help
```

Zoals duidelijk blijkt, wordt de `^^J` niet opgevat als `\newlinechar`.

Ik persoonlijk schuif alle bugs op mijn nogal wankele `web2c`. Als iemand een betere versie heeft, of wil maken, houd ik me aanbevolen.

Ik gebruik deze TEX-versie nu bijna een jaar en ik heb verder nog geen bugs kunnen vinden.

5 Hoe kan men aan armTEX komen?

Ik kan armTEX voor iedereen die erom vraagt op schijf zetten. Op het moment kost de hele distributie tussen de 8 en 30 schijven, afhankelijk van de hoeveelheid sources en extras. De programma's zijn binnenkort ook op een FTP-site te krijgen. Nadere mededelingen hierover volgen nog.

Nogmaals: ik zou het plezierig vinden als meerdere mensen met mij aan dit project zouden willen werken.

Het gebruik van MathTime in LaTeX.

Piet Tutelaers
Technische Universiteit Eindhoven
rcpt@urc.tue.nl

Onder invloed van de enorme hoeveelheid beschikbare PostScript type1 outline fonts heeft Knuth TeX3.0 in 1989 voorzien van het virtuele fontmechanisme zodat deze niet-TeX-fonts gebruikt kunnen worden. De uitbreidingen aan TeX waren minimaal[1]. De grote last kwam op de schouders te liggen van de ontwikkelaars van de previewers en dvi-drivers. Dat dit niet triviaal is, bewijst het feit dat er nog steeds geen goede PD dvi-driver bestaat voor HPLaserJets op UNIX die met virtuele fonts kan omgaan. Gelukkig bestaan er inmiddels previewers voor UNIX (xdvi), VAX-VMS (xdvi) en MSDOS (dviscr van emTeX) die ze wel begrijpen. En voor PostScript printers heeft Tom Rokicki veel tijd en energie gestoken in zijn dvips zodat veel van de voordelen van PostScript beschikbaar is voor TeX-gebruikers.

Bij dvips worden een aantal LaTeX stijlen meegeleverd waardoor het selekteren van PostScript fonts erg makkelijk is. Om in plaats van Computer Modern het PostScript font 'Times' te kiezen, hoef je alleen maar de stijloptie times te kiezen:

```
\documentstyle[times]{...}
```

Deze times optie levert echter geen PostScript uitvoer op voor formules. Eenvoudig omdat er geen goede PostScript fonts zijn die alle symbolen van cmmi, cmsy en cmex kunnen vervangen. Je krijgt dan ook PostScript uitvoer gemengd met Computer Modern zoals figuur 1 laat zien. Om te demonstreren wat het effect is van het vergroten en het verkleinen van fonts, heb ik met de `-E` optie van dvips een EPSF aangemaakt bestaande uit een stuk tekst met daarin enkele formules. Het origineel van deze tekst is weergegeven in het midden van figuur 1. Links en rechts ervan is respectievelijk het verkleinde en vergrootte deel hiervan te zien. Beoordeel zelf de kwaliteit van de formules. Met een fotozetter zou het effect nog duidelijker gedemonstreerd kunnen worden.

Nu heeft Michael Spivak PostScript fonts ontworpen die de computer moderne fonts cmmi, cmsy en cmex kunnen vervangen. Deze fonts zijn voor $135,- ($95,- voor educatieve instellingen) te koop bij de TeXplorators[2]. Dankzij deze schaalbare MathTime fonts ziet het resultaat er hiermee een stuk beter uit zoals figuur 2 laat zien.

In tabel 1 zijn de fonts die je in LaTeX kunt selekteren naast elkaar geplaatst. In de tweede kolom staan de computer moderne fonts die standaard worden geselekteerd, de derde kolom bevat de fonts die je krijgt met times en de laatste kolom bevat de mathtime keuzes. In de picture omgeving van LaTeX worden enkele niet-PostScript gekozen zoals lasy, circle en line. Dus om 100 procent PostScript uitvoer te maken kunt je beter de figuren in EPSF formaat opnemen.

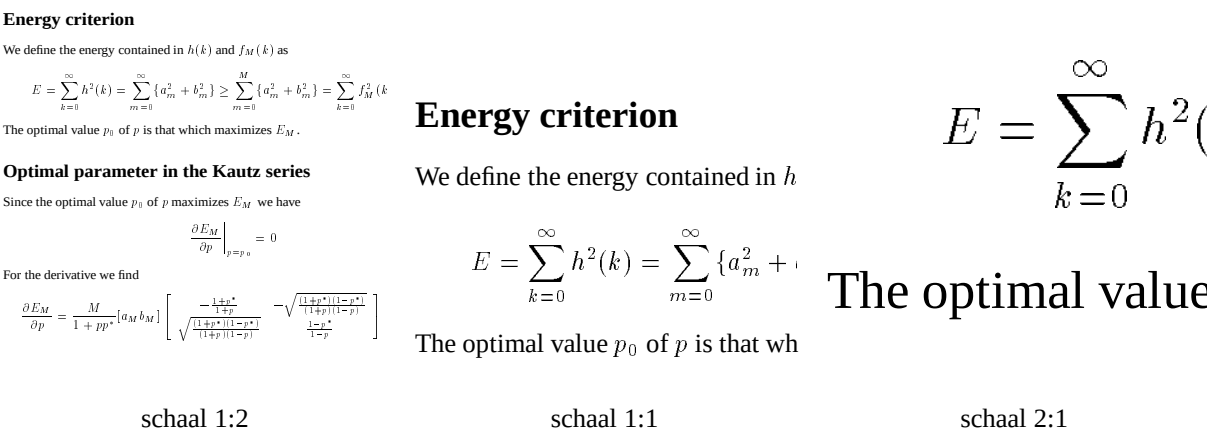
LaTeX	standaard	`times`	`mathtime`
`\rm`	cmr	ptmr	ptmr
`\it`	cmi	ptmi	ptmi
`\sl`	cmsl	ptmro	ptmro
`\bf`	cmbx	ptmb	ptmb
`\sc`	cmcsc	ptmrc	ptmrc
`\ss`	cmss	phvr	cmss
`\tt`	cmtt	pcrr	cmtt
`$ a $`	cmmi	cmmi	mtmi
`$\circ$`	cmsy	cmsy	mtsy
`$[]$`	cmex	cmex	mtex
`picture`	lasy	EPSF	EPSF

Figuur 1: *LaTeX fonts afhankelijk van gekozen stijl*

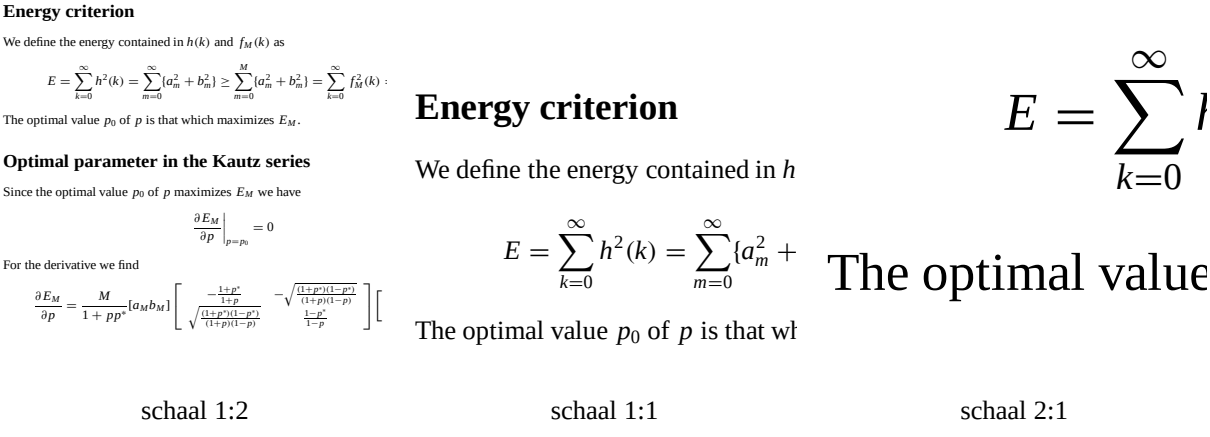
Het ptmr font, het font dat met `\rm` wordt geselekteerd bij de LaTeX stijl times of mathtime, is een virtueel font dat gebruikt maakt van Times-Roman en het Symbol font voor de griekse letters. Het ptmro font wordt gerealiseerd door Times-Roman te kantelen met PostScript commando's omdat er geen Times-Oblique font is. Voor small caps (ptmrc) worden de hoofdletters uit Times-Roman met een factor 0.8 verkleind bij gebrek aan een echt Times-SmallCaps font. Verder kiest times het Courier font (pcrr) als het teletype lettertype en Helvetica (phvr) als het schreefloze alternatief.

[1] Implementatie van 8-bits code, VPtoVF en VFtoVP

[2] The TeXplorators Corporation, 1572 West Gray, #377, Houston, TX 77019-4948 U.S.A.; FAX:(713) 523–6743



Figuur 1: *Een demonstratie van* `times`



Figuur 2: *Een demonstratie van* `mathtime`

Helaas biedt het MathTime pakket standaard geen ondersteuning voor LaTeX. Gelukkig heeft Don Hosek de plain TeX macros uit mtmac.tex omgezet naar de LaTeX stijl mathtime.sty zodat je met:

```
\documentstyle[mathtime]{...}
```

Times voor tekst en MathTime voor formules kunt selekteren. In mathtime.sty wordt gebruik gemaakt van het 'New Font Selektie Schema' (NFSS). Voor schreefloos en teletype worden (bewust?) de computer moderne fonts cmss en cmtt geselekteerd. Je kunt echter vrij gemakkelijk een eigen times.sty maken die gebruik maakt van mathtime.sty en deze twee fonts herdefinieert in de PostScript fonts Helvetica en Courier:

```
% times.sty
\input mathtime.sty

% Take Courier for \tt (teletype)
\new@fontshape{cour}{m}{n}{%
  <5>pcrr at 5pt%
  <6>pcrr at 6pt%
  <7>pcrr at 7pt%
  <8>pcrr at 8pt%
  <9>pcrr at 9pt%
  <10>pcrr at 10pt%
  <11>pcrr at 11pt%
  <12>pcrr at 12pt%
  <14>pcrr at 14pt%
  <17>pcrr at 17pt%
  <20>pcrr at 20pt%
  <25>pcrr at 25pt}{}
\subst@fontshape{cmtt}{m}{n}{cour}{m}{n}

% Take Helvetica for \ss (sans seriff)
\new@fontshape{helv}{m}{n}{%
    <5>phvr at 5pt%
    <6>phvr at 6pt%
    <7>phvr at 7pt%
    <8>phvr at 8pt%
    <9>phvr at 9pt%
    <10>phvr at 10pt%
```

```
    <11>phvr at 11pt%
    <12>phvr at 12pt%
    <14>phvr at 14pt%
    <17>phvr at 17pt%
    <20>phvr at 20pt%
    <25>phvr at 25pt}{}
\subst@fontshape{cmss}{m}{n}{helv}{m}{n}
```

Om een dvi-file die gebruik maakt van PostScript fonts te kunnen previewen heb je een viewer nodig die deze fonts kan verpunten ('renderen'). Adobe heeft hiervoor Display PostScript ontwikkeld. DPS biedt een library interface waarmee programma's kunnen worden gemaakt die fonts dynamisch kunt verpunten. Er zijn weinig previewers die hiervan gebruik maken. Een minder dynamische maar wel PD beschikbare oplossing biedt ps2pk. Met dit programma kun je type1 fonts omzetten naar PK-fonts. Deze PK-fonts kun je gebruiken in elke previewer en dvi-driver die met virtuele fonts kan omgaan. Op UNIX kun je het genereren van PK-fonts automatisch laten verlopen als je beschikt over de aangepaste versie van MakeTeXPK, ps2pk en de benodigde PostScript fonts. Op `ftp.urc.tue.nl` in `pub/tex/MathTime` bevindt zich een `README` file waarin uitgelegd wordt hoe je dat doet. Hier bevindt zich ook de benodigde versie van MakeTeXPK. Voor onze emTeX gebruikers op de TUE heb ik een versie gemaakt waarin alle benodigde type1 PK-fonts kant-en-klaar aanwezig zijn. Deze fonts heb ik gegenereerd met behulp van een shell-script en dvips op UNIX.

Hoewel MathTime geen perfekte oplossing biedt voor LaTeX, er ontbreekt bijvoorbeeld een font voor Italic-BoldMath, ben ik er toch zeer tevreden mee. Dankzij de flexibiliteit van schaalbare fonts kunnen we nu met LaTeX volwaardige PostScript genereren. Deze PostScript kun je door een fotozetter laten afdrukken. Of je kunt een A3-pagina opblazen tot A0 formaat zonder dat dit afbreuk doet aan de kwaliteit van de afdruk. Deze faciliteit wordt erg gewaardeerd door collegas die meedoen aan poster sessies op conferenties.

Aanvullende informatie van de leverancier

Van de MathTime leverancier (The TeXplorators Corporation) ontving de redactie de volgende aanvullende informatie:[3]

Site licenses are also available for large sites with many PostScript printers.

Price of *MathTime* is still $135 ($95 educational), including shipping.

Site license fees are:

- Up to 15 printers at one site: 3 times cost of individual license
- Up to 30 printers at one site: 6 times cost of individual license
- Unlimited number of printers at one site: 10 times cost of individual license

All licenses, including individual, can be upgraded at any time to another license simply by paying the difference in the fees.

Unlike programs, licensed for a single computer, most PostScript fonts are usually licensed for a single *printer* (this is partly for historical reasons, since in the pre-PostScript days all typesetters had their own proprietary fonts).

But the manual specifically recommends adding the *MathTime* fonts to the beginning of a `.ps` file that is being sent to a phototypesetting device, since they may have difficulties dealing with fonts not in `.pfb` format (on the other hand, if a typesetting service that deals with large amounts of mathematical material does have its own *MathTime* fonts, and downloads them, then this is both unnecessary and inefficient).

Consequently, the *MathTime* fonts are licensed for a single computer. They may actually be placed on more than one computer, provided that only one is used at a time. (For example, if you use one computer in your office and another at home, it's legitimate to keep the fonts on both, but not if the computer in your office is being used by your secretary, while you are using the one at home.)

In practice, many organizations have many computers producing TeX documents that are printed on a single PostScript printer, one that is essentially controlled, so far as downloading fonts is concerned, by a single computer. In that case, the other computers really need only the `.tfm` files for the *MathTime* fonts. In such situations it is specifically allowed that the `.tfm` files may be placed on the other computers.

As for future developments, there's nothing definite I can say yet. However, there are some possiblities that there will be improvements to the current fonts—if so, upgrades will be offerred for a nominal fee.

There's also the possiblity that we will have bold versions of the MTMI and MTSY fonts. Other possiblities for the future are Times versions of the AMS's msam and msbm fonts (except for their horrible Blackboard bold symbols, which are best taken from the Adobe Mathematical-Pi fonts), and a Script alphabet, both upper and lower case.

I should say that everything in the last paragraph is contingent on my eventually not losing any money on the development of the MathTime fonts. So far I'm still far from even, but developments that I can't comment on yet may change that picture.

[3] Michael Spivak, 30-mar-93.

The Future of TEX*

Philip Taylor

Royal Holloway and Bedford New College, University of London, Egham Hill, Egham, Surrey, UK
p.taylor@vax.rhbnc.ac.uk

July 1992

Abstract

TEX and the other members of Knuth's *Computers & Typesetting* family are arguably amongst the most successful examples of computer software in the world, having been ported to almost every conceivable operating system and attracting an allegiance that verges on the fanatical. Development work on this family has now ceased, and many members of the computer typesetting community are concerned that some action should be taken to ensure that the ideas and philosophy enshrined in TEX are not allowed simply to fade away. In this paper, we discuss some of the options available for perpetuating the TEX philosophy, and examine the strengths and weaknesses of the present TEX system. We conclude by postulating a development strategy for the future which will honour both the letter and the spirit of Knuth's wish that TEX, METAFONT and the Computer Modern typefaces remain his sole responsibility, and at the same time ensure that the philosophy and paradigms which are the strengths of TEX are not lost for ever by having artificial constraints placed on their evolution.

Keywords: TEX, extended TEX, NTS, New Typesetting System

"My work on developing TEX, METAFONT and Computer Modern has come to an end." [1] With these words, Professor Donald E. Knuth, creator of TEX, informed the world that the evolution of probably the most successful computer typesetting system yet developed had ceased, and that with the sole exception of essential bug fixes, no further changes would be made. TEX's version number will asymptotically approach π as bug fixes are made, and at the time of his death, it will be renamed 'TEX, Version π'; thereafter it will remain exactly as he last left it: a fitting and appropriate memorial to one of the most productive and inspired computer scientists (and mathematicians, and Bible scholars) that the world has ever known.

The future of TEX is therefore totally determined: why, then, is this paper entitled "*The Future of TEX*"? Because, primarily, TEX is already fifteen years old — four years as a child (TEX 78); eight years as an adult (TEX 82); and three years in maturity (TEX 3). Fifteen years is a long time in the lifespan of computer languages: Algol 68, for example, was certainly at or beyond its peak by 1982, and is today almost as rare as the Tasmanian wolf,[1] if not yet as dead as the Dodo:[2] a language must evolve, or die. (There are numerous natural languages which are almost certainly in terminal decline, despite the most strenuous efforts of a nucleus of active speakers to artificially prolong their lives: Cornish and Manx are surely dead; Gaelic must feature in any linguistic 'Red Book' of endangered languages; only Welsh, which alone among the British native minority tongues continues to evolve, shews any real resistance to morbidity and eventual death). If natural languages must evolve or die, how much more so must computer languages, whose evolution must keep pace with a technology which evolves at a rate so rapid that it is unmatched in the natural world even by irradiated fruit-flies.[3]

So, my underlying hypothesis is: TEX must evolve, or die. If we are to believe the evidence of our ears and eyes, the underlying TEX philosophy is already as anachronistic as the horse and cart: TEX represents the pinnacle of Neanderthal evolution, building on the genetic heritage of Runoff, Nroff, Troff, Ditroff and Scribe, whilst Cro-Magnon man, in the guise of Ventura Publisher, Aldus Pagemaker and Quark Xpress, is already sweeping over the face of the planet. The halcyon days are long since gone (or so it would seem)

*This article is reproduced by kind permission of the organisers of the EuroTEX '92 conference in Prague, Czechoslovakia, September 14–18, in the proceedings of which [4] it first appeared.

[1] *Thylacinus cynocephalus*

[2] *Raphus cucullatus*

[3] *Drosophila melanogaster*

when it was socially acceptable to: enter text; check it for spelling errors (by eye!); insert a series of formatting commands; pass the whole through an interpreter; identify the first error; correct the first error; pass the whole through the interpreter again; identify the second error; correct the second error; pass the whole through the interpreter for a third time; repeat for all subsequent errors. . .; pass the whole through the interpreter for the n^{th} time; then pass it through the interpreter again (to resolve forward- and cross-references); preview a facsimile of the final copy on the computer screen; notice a formatting error; and go right back to editing the file: our colleagues sit there clicking away on their mice[4] like demented death-watch beetles[5] and think us totally mad; and mad we surely must be, for we not only enjoy this mode of working, we seek to convert the demented mouse clickers into TEX users as well!

Why? What is it about TEX that is so totally addictive? Is it perhaps TEX's descriptive and character-oriented nature —the fact that, in direct opposition to current trends, TEX requires the user to think about what he or she wants to achieve, and then to express that thought as a series of words and symbols in a file, rather than as a series of ephemeral mouse movements on a screen? Is it, perhaps, its portability —the fact that implementations (almost entirely public domain) exist for every major operating system in the world? Is it the deterministic nature of TEX —the fact that a given sequence of TEX commands and text-to-be-typeset will always produce *exactly* the same results, regardless of the machine on which it is processed? Is it the 'boxes and glue' paradigm, which provides a simple but somewhat naïve model of black and white space on the printed page? The ease with which form and content can be separated? The implementation as a macro, rather than a procedural, language? (would a procedural TEX still be recognisably TEX?) Is it, perhaps, the incredible contortions through which one occasionally has to go to achieve a desired result? (Or the incredible elation when such contortions finally achieve their intended effect?) How many of these elements could be eliminated and still leave something that is recognisably TEX? I propose to return to these questions, and to attempt to answer some of them, later in this paper.

A related question: what is the potential lifespan of a TEX-based typesetting system, or for that matter, of any computer language? Of all the general purpose computer languages which have sprung into existence since the advent of compilers (which point in time really marks the beginning of all the computer languages that are in general use today), Cobol and Fortran are probably among the longest lived; but Fortran has evolved enormously since the days of Fortran 2 (which is as far back as my memory goes), whilst Cobol has evolved relatively little; Basic, too, is still with us, although the originators of Dartmouth Basic would find little to recognise in the 'Visual Basic' of Microsoft today. Algol 60 evolved via various routes into Algol 68, which for me represents the pinnacle of language design, but evolved no further, and is today reaching the end of its twilight years. Pascal, which owes much to the Algol family, gave birth to Modula, which itself became transmuted into Oberon; in a sense, this last example represents a failure of the evolutionary system, for in its heyday Pascal was almost universally adopted, giving birth to the UCSD 'P' system as well as making possible the unbelievably successful (and revolutionary) 'Turbo Pascal', whilst Modula, although lauded by computer scientists, remained of relatively limited acceptance and acceptability, and Oberon remained almost unknown without the walls of academia. Most recently, among the procedural languages at least, we come to 'C', and its bastard offspring 'C++'; these languages have an honourable history, tracing their roots back through 'B' (or so I am told —I have never encountered 'B' myself) to BCPL, the 'Basic Combined (or Cambridge, depending on one's background) Programming Language', itself derived from CPL which simply wasn't so basic! *En route*, data typing was acquired, and lost, and acquired again, and polymorphism was acquired with the advent of 'C++'. Other evolutionary lines are represented by Prolog, which epitomises the declarative family, and Lisp, which is the archetype of list processing languages (and which remains almost unchanged since its inception). Poplog, encompassing as it does representatives of all three families (Pop 11, based on Pop 2, Prolog and Lisp) is perhaps a unique synthesis. Finally one should not omit mention of that most modestly titled of all programming languages, APL: '*A* Programming Language'.

But this is not a history of programming languages: I cite the above examples only to place TEX within context, for although when teaching TEX to secretaries one does not necessarily stress the fact of its being a computer programming language *per se*, a computer programming language it most certainly is. Indeed, TEX is 'Turing complete', which is a computer scientist's jargon for saying that TEX could be used as a general purpose programming language since it has the necessary flexibility, although apart from the intellectual satisfaction there would be little point in so doing: TEX's *forte* is clearly computer typesetting, and only programmers or perverts could derive pleasure from coercing it into calculating cube roots or cosines!

So what is the common theme among all the languages cited above? Simply this: that almost every one of them has either given birth to a successor (which is not necessarily more successful: *cf.* Pascal → Modula → Oberon), or has simply fallen into disuse; Cobol and

[4] *Mus ordinatus microsoftiensis* or *Mus ordinatus applemacintoshii*

[5] *Xestobium rufovillosum*

Lisp alone, which occupy highly specialised niches, remain relatively unchanged, and of these only Cobol continues to play a significant rôle in mainstream computing (although Lisp remains the language of choice for many linguistic and related tasks).

It seems, then, that we have a choice: we can either allow natural selection to take its course, in which case TEX, having fulfilled its appointed rôle on this planet (which I assume is to teach us the merits of literate programming, whilst encouraging us to devote ever more time to the typesetting of beautiful papers, presumably at the expense of ever less time spent actually researching or writing them), will surely join XCHLF, JEAN & JOSS in the great bit-bin in the sky; or we can adopt a corporate responsibility for the future of TEX and intercede in the process of natural selection, taking steps to ensure that TEX evolves into a typesetting system which is so demonstrably superior to the miasma of mouse-based, menu-driven, manipulators of text and images which are currently snapping at its heels that no-one will be able to deny it its rightful place at the forefront of typesetting technology for the twenty-first century.

Let us consider the options which are available to us:

1. We can leave TEX exactly as it is: this is clearly a defensible position as it is exactly what Knuth himself intends to do; it would be extremely arrogant of us to suggest that we know better than Knuth in this respect.
2. We can enhance TEX by just enough that those who really understand its power, its limitations, and its inner workings agree that it no longer has demonstrable defects (i.e. there are some 'simple' typesetting tasks with which $\TeX_\pi$ could not deal correctly, but with which an enhanced TEX could).
3. We can enhance TEX by incorporating the combined wish-lists of its major practitioners, thereby seeking to make TEX all things to all men (and all women), whilst retaining its present 'look and feel'.
4. We can enhance TEX as in option 3 above, whilst taking the opportunity to re-consider, and perhaps substantially change, its present look and feel.
5. We can take the opportunity to do what I believe Knuth himself might do, were he to consider today the problems of typesetting for the first time: look at the very best of today's typesetting systems (clearly including TEX among these), and then design a *new* typesetting system, far more than just a synthesis of all that is best today, which addresses the needs and potential not only of today's technology, but that of the foreseeable future as well. We would need to find some way to incorporate that spark of genius which characterizes Knuth's work!

No doubt each of us will have his or her own ideas on the desirability or otherwise of each of these options; it is not my intention in this paper to attempt to persuade you that any one of them is clearly preferable; but I would be shirking my responsibilities were I not to caution that, in my opinion, option 3 appears to represent the worst of all possible worlds, representing as it does a clear case of 'creeping featurism' at its worst while not possessing any redeeming qualities of originality.

Option 1 is, as I have suggested above, clearly defensible, in that it is Knuth's own preferred position; despite my fears that TEX will succumb to the pressures of natural selection if it is adopted, it may be that TEX represents both the pinnacle and the end of an evolutionary line, and that future typesetting systems will be based on an entirely different philosophy (e.g. mouse-based).

Option 2 represents the most conservative evolutionary position and has, I believe, much to commend it, certainly in the short term: it would retain the present look and feel of TEX; and compatibility with current TEX programs, whilst not intrinsically guaranteed, could be ensured by careful design; at the very worst, one could envisage a command-line qualifier which would disable the extensions, leaving a true TEX 3 underneath. Although option 2 is in opposition to Knuth's expressed wishes, he has made it plain that he has no objection to such enhancements *provided that* the resulting system is not called TEX. I propose that we term the results of adopting option 2 'Extended TEX', both to indicate its nature, and, more importantly, to comply with the spirit as well as the letter of Knuth's wishes.

Option 3 is considerably less conservative, but does at least retain the present look and feel of TEX; it is completely open-ended in terms of the extensions made to TEX, and offers the opportunity to make sweeping enhancements (I hesitate to use the word 'improvements' for the reasons outlined above). Compatibility with current TEX programs need not prove problematic, provided that the design were adequately thought out, and again the possibility of a '`/noextensions`' qualifier provides a fallback position. The timescale for such an implementation would not be small if a new swarm of bugs is to be prevented, and it is not clear how future obsolescence is to be avoided: after all, if 'The Ultimate TEX' (as I will term it) includes all the proposed enhancements of TEX's major practitioners, what enhancements remain to be implemented in the future?

Option 4 represents the first attempt at a true re-design of TEX, allowing as it does the option to re-think TEX's look and feel, whilst continuing to incorporate many of its underlying algorithms. One could envisage, for example, an implementation of TEX in which text and markup were kept entirely separate, with a system of pointers from markup to text (and *vice versa*?). One advantage of such a scheme is that it would eliminate, at a stroke, the troublesome nature of the <space> character which currently complicates TEX; the escape character could become redundant, and the problems

of category codes possibly eliminated. Of course, this is just one of many such possibilities: once one abandons the look and feel of TEX, the whole world becomes one's typesetting oyster. One might term such a version of TEX 'Future TEX'.

Option 5 is without doubt the most radical: not only does it reject (at least, initially), TEX's look and feel, it challenges the entire received wisdom of TEX and asks instead the fundamental question: "How should computer typesetting be carried out?" In so doing, I believe it best represents Knuth's own thoughts prior to his creation of TEX 78, and, by extrapolation, the thoughts which he might have today, were he faced for the first time with the problems of persuading a phototypesetter to produce results worthy of the texts which it is required to set. I think it important to note that there is nothing in option 5 which automatically implies the rejection of the TEX philosophy and paradigms: it may well be that, after adequate introspection, we will decide that TEX does, in fact, continue to represent the state of the typesetting art, and that we can do no better than either to leave it exactly as it is, or perhaps to extend it to a greater or lesser extent whilst retaining its basic model of the typesetting universe of discourse; on the other hand, neither does it imply that we *will* reach these conclusions. I will call such a system 'A New Typesetting System' (to differentiate it from 'The New Typesetting System' which is the remit of NTS, *q.v.*).

The options outlined above are not necessarily mutually exclusive: we might decide, for example, to adopt option 2 as an interim measure, whilst seeking the resources necessary to allow the adoption of option 5 as the preferred long-term position (indeed, I have considerable sympathy with this approach myself). But no matter which of the options we adopt, we also need to develop a plan of campaign, both to decide which of the options is the most preferable (or perhaps to adopt an option which I have not considered) and then to coordinate the implementation of the selected option or options.

As many of you will be aware, a start has already been made to this end: at a meeting of DANTE (the German-speaking TEX Users' Group) earlier this year, Joachim Lamarsch announced the formation of a steering group, organised under the ægis of DANTE, to co-ordinate developments of TEX; this group, diplomatically called 'NTS' so as to avoid any suggestion that it is TEX itself whose future is being considered, is chaired by Rainer Schöpf; the members are listed in Appendix A. An e-mail discussion list has also been created (called NTS-L),[6] with an open membership;[7] all messages are automatically forwarded to members of the NTS team. At the time of writing this article, the group has not yet formally met: instead, we have been content to listen to the many positive suggestions which have been put via the medium of NTS-L. It is clear that there is no general consensus at the moment as to which of the five options outlined above is preferable; some argue for strict compatibility with existing TEX implementations, whilst others argue that we must grasp the nettle and take this opportunity to create a truly revolutionary typesetting system. Some, at least, are quite content to adopt the Knuthian position, and simply use TEX as it is: "TEX is perfect" was the subject of more than one submission to NTS-L. One of the more interesting facts to emerge from the discussion is the different ways in which TEX is perceived: some see it simply as a tool for mathematical typesetting; others want to be able to create the most complex graphics without ever leaving TEX's protective shell; many want to be able to typeset arbitrarily complex documents (not necessarily containing one line of mathematics), but are content to leave graphics, at least, without TEX's remit.

So far, this paper has been concerned primarily with generalities; but I propose now to look at some of the specific issues to which I have earlier merely alluded, and to offer some personal opinions on possible ways forward. I propose to start by attempting to answer the question which I believe lies at the very heart of our quest: "What is the essence of TEX?"

It seems to me that there are some aspects of TEX which are truly fundamental, and some which are merely peripheral: among the fundamental I include its descriptive and character-oriented nature, its portability, and its deterministic behaviour; I also include some elements which I have not so far discussed: its programmability (for example, the way in which loops can be implemented, even though they are not intrinsic to its design), its generality (the fact that it can be used to typeset text, mathematics, and even music), its device independence, and its sheer æsthetic excellence (the fact that, in reasonably skilled hands, it can produce results which are virtually indistinguishable from material set professionally using traditional techniques). Equally important, but from a different perspective, are the facts that it is totally documented in the ultimate exposition of literate programming (the *Computers & Typesetting* quintology), that it is virtually bug-free, that any bugs which do emerge from the woodwork are rapidly exterminated by its author, and finally that for higher-level problems (i.e. those which are at the programming/user-interface level rather than at the WEB level), there are literally thousands of skilled users to whom one can appeal for assistance. We should not forget, too, Knuth's altruism in making the entire source code[8] freely available with an absolute minimum of constraints. It is almost certainly true that this last fact, combined solely with

[6] `NTS-L@VM.URZ.Uni-Heidelberg.De`

[7] Send a message to `LISTSERV@VM.URZ.Uni-Heidelberg.De` with a single line body containing the text
`Subscribe Nts-L <given name> <SURNAME>`

[8] including source for the TEX and METAFONT books; this is frequently forgotten. . .

the sheer excellence of TEX, is responsible for TEX's widespread adoption over so much of the face of our planet today.

Among its more peripheral attributes I include its implementation as a macro, rather than as a procedural or declarative, language, and perhaps more contentiously, its fundamental paradigm of 'boxes and glue'. I hesitate to claim that boxes and glue are not fundamental to TEX, since in many senses they clearly are: yet it seems to me that if a descendant of TEX were to have detailed knowledge of the *shape* of every glyph (rather than its bounding box, as at present), and if it were perhaps to be capable of typesetting things on a grid, rather than floating in space and separated by differentially stretchable and shrinkable white space, but were to retain all of the other attributes asserted above to be truly fundamental, then most would recognise it as a true descendant of TEX, rather than some mutated chimera.

Without consciously thinking about it, I have, of course, characterized TEX by its strengths rather than its weaknesses.[9] But if we are to intervene in the processes of natural selection, then it is essential that we are as familiar with TEX's weaknesses as with its strengths: if it had no weaknesses, then our intervention would be unnecessary, and the whole question of the future of TEX would never have arisen. But whilst it is (relatively) easy to identify a subset of its characteristics which the majority of its practitioners (I hesitate to say 'all') would agree represent its fundamental strengths, identifying a similar subset of its characteristics which represent its fundamental weaknesses is far more contentious. None the less, identify such a subset we must.

Perhaps the safest starting point is to consider the tacit design criteria which Knuth must have had in mind when he first conceived of TEX, and which remain an integral part of its functionality today. TEX, remember, was born in 1978 —a time when computer memories were measured in kilobytes rather than megabytes, when laser printers were almost unknown, when the CPU power of even a University mainframe was probably less than that available on the desktops of each of its academics today, and when real-time preview was just a pipe dream.[10] Each and every one of these limitations must have played a part in TEX's design, even though Knuth may not have been consciously aware of the limitations at the time. (After all, we are only aware of the scarcity of laser printers in 1978 because of their ubiquity today; we aren't aware of the limiting effects of the scarcity of ion-beam hyperdrives because they haven't yet been invented...). But by careful reading of *The TEXbook* (and even more careful reading of `TEX.WEB`), we can start to become aware of some of the design constraints which were placed on Knuth (and hence on TEX) because of the limits of the then-current technology. For example, on page 110 one reads: "TEX uses a special method to find the optimum breakpoints for the lines in an entire paragraph, but it doesn't attempt to find the optimum breakpoints for the pages in an entire document. *The computer doesn't have enough high-speed memory capacity to remember the contents of several pages* [my stress], so TEX simply chooses each page break as best it can, by a process of 'local' rather than 'global' optimisation." I think we can reasonably deduce from this that if memory had been as cheap and as readily available in 1978 as it is today, TEX's page-breaking algorithm may have been very different. Other possible limitations may be inferred from the list of numeric constants which appear on page 336, where, for example, the limit of 16 families for maths fonts is stated (a source of considerable difficulties for the designers of the New Font Selection Scheme);[11] 16 category codes, too, although seemingly just enough, force the caret character (^) to serve triple duty, introducing not only 64-byte offset characters and hexadecimal character specifiers, but also serving as the superscript operator.

So, we may reasonably infer that the combined restrictions of limited high-speed memory, inadequate CPU power, and very limited preview and proof facilities, combined to place limitations on the original design of TEX; limitations the effect of which may still be felt today. It is perhaps unfortunate that in at least one of these areas, that of high-speed memory, there are still systems being sold today which have fundamental deficiencies in that area: I refer, of course, to the countless MS/DOS-based systems (without doubt the most popular computer system ever invented) which continue to carry within them the design constraints of the original 8088/8086 processors. Because of the ubiquity of such systems, there have been a fair number of submissions to the NTS list urging that any development of TEX bear the constraints of these systems in mind; despite the fact that I too am primarily an MS/DOS user, I have to say that I do not feel that the 64K-segment, 640K-overall limitations of MS/DOS should in any way influence the design of a new typesetting system. Whilst I feel little affinity for the GUI-based nature of Microsoft Windows, its elimination of the 640K-limit for native-mode programs is such a step forward that I am prepared to argue that any future typesetting system for MS/DOS-based systems should assume the existence of Windows (or OS/2), or otherwise avoid the 640K barrier by using techniques such as that adopted by Eberhard Mattes' *emTEX386*.[12] If we continue to observe the constraints imposed by primitive systems such as MS/DOS, what

[9] OK, I admit it: TEX *might* have weaknesses...

[10] Although on page 387 (page numbers all refer to *The TEXbook* unless otherwise stated), we find "Some implementations of TEX display the output as you are running".

[11] Frank Mittelbach and Rainer Schöpf

[12] *emTEX386* uses a so-called 'DOS extender'.

hope have we of creating a typesetting system for the future rather than for yesterday?

These might be termed the historical (or 'necessary') deficiencies of TEX: deficiencies over which Knuth essentially had no control. But in examining the deficiencies of TEX, we must also look to the needs of its users, and determine where TEX falls short of these, regardless of the reasons. The term 'users', in this context, is all-encompassing, applying equally to the totally naïve user of LATEX and to the format designers themselves (people such as Leslie Lamport, Michael Spivak, and Frank Mittelbach); for although it is possible for format designers to conceal certain deficiencies in TEX itself (e.g. the lack of a `\loop` primitive), the more fundamental deficiencies will affect both. (Although it is fair to say that a sure sign of the skill of a format designer is the ease with which he or she can conceal as many of the apparent deficiencies as possible). An excellent introduction to this subject is the article by Frank Mittelbach in *TUGboat*, 'E-TEX: Guidelines for future TEX' [2], and the subsequent article by Michael Vulis, 'Should TEX be extended?' [3]. Perhaps less accessible, and certainly more voluminous, are the combined submissions to NTS-L, which are archived at `TeX.Ac.Uk` as `Disk$TeX:[TeX-Archive.Nts]Nts-L.All` and at `Ftp.Th-Darmstadt.De` as `/pub/tex/documentation/nts-l/*`.

So, what are these so-called 'fundamental deficiencies'? No doubt each of us will have his or her own ideas, and the three references cited above will serve as an excellent starting point for those who have never considered the subject before. What follows is essentially a very personal view —one person's ideas of what he regards as being truly fundamental. It is not intended to be exhaustive, nor necessarily original: some of the ideas discussed will be found in the references given; but I hope and believe that it is truly representative of current thinking on the subject. Without more ado, let us proceed to actual instances.

1. *The lack of condition/exception handling:* It is not possible within TEX to trap errors; if an error occurs, it invariably results in a standard error message being issued, and if the severity exceeds that of 'warning'[13] (e.g. overfull or underfull boxes), user interaction is required. This makes it impossible for a format designer to ensure that all errors are handled by the format, and actually prevents the adoption of adequate defensive programming techniques. For example, it is not possible for the designer of a font-handling system to trap an attempt to load a font which does not exist on the target system.
2. *The inability to determine that an error has occurred:* The `\last...` family (`\lastbox`, `\lastkern`, `\lastpenalty`, `\lastskip`) are unable to differentiate between the absence of a matching entity on the current list and the presence of a zero-valued entity; since there is all the difference in the world between a penalty of zero and no penalty at all, vital information is lost.
3. *The hierarchical nature of line-breaking and page-breaking:* Once a paragraph has been broken into lines, it is virtually impossible to cause TEX to reconsider its decisions. Thus, when a paragraph spans two pages, the material at the top of the second page will have line breaks within it which are conditioned by the line breaks at the bottom of the previous page; this is indefensible, as the two occur in different visual contexts. Furthermore, it prevents top-of-page from being afforded special typographic treatment: for example, a figure may occur at the top of the second page, around which it is desired to flow text; if the paragraph has already been broken, no such flowing is possible (the issue of flowing text in general is discussed below). The asynchronous nature of page breaking also makes it almost impossible to make paragraph shape dependent on position: for example, a particular house style may require paragraphs which start at top of page to be unindented; this is non-trivial to achieve.
4. *The local nature of page breaking:* For anything which approximates to the format of a Western book, the verso-recto spread represents one obvious visual context. Thus one might wish to ensure, for example, that verso-recto pairs always have the same depth, even if that depth varies from spread to spread by a line or so. With TEX's present page breaking mechanism, allied to its treatment of insertions and marks, that requirement is quite difficult to achieve. Furthermore, by localising page breaking to the context of a single page, the risk of generating truly 'bad' pages is significantly increased, since there is no look-ahead in the algorithm which could allow the badness of subsequent pages to affect the page-breaking point on the current page.
5. *The analogue nature of 'glue':* TEX's fundamental paradigm, that of boxes and glue, provides an elegant, albeit simplistic, model of the printed page. Unfortunately, the flexible nature of glue, combined with the lack of any underlying grid specification, makes grid-oriented page layup impossible to achieve, at least in the general case. The present boxes and glue model could still be applicable in a grid-oriented version of TEX, but in addition there would need to be what might be termed 'baseline attractors': during the glue-setting phase, baselines would be drawn towards one of the two nearest attractors, which would still honour the constraints of `\lineskiplimit` (i.e. if the effect of drawing a baseline upwards were to bring two lines too close together, then the baseline would be drawn downwards instead).

[13] I use the VAX/VMS conventions of 'success', 'informational', 'warning', 'error' and 'severe error' as being reasonably intuitively meaningful here.

6. *The lack of any generalised ability to flow text:* TEX provides only very simple paragraph shaping tools at the moment, of which the most powerful is `\parshape`; but one could envisage a `\pageshape` primitive and even a `\spreadshape` primitive, which would allow the page or spread to be defined as a series of discrete areas into which text would be allowed to flow. There would need to be defined a mechanism (not necessarily within the primitives of the language, but certainly within a kernel format) which would allow floating objects to interact with these primitives, thereby providing much needed functionality which is already present in other (mouse-oriented) systems.
7. *An over-simplistic model of lines of text:* Once TEX has broken paragraphs into lines, it encapsulates each line in an `\hbox` the dimensions of which represent the overall bounding box for the line; when (as is usually the case) two such lines occur one above the other, the minimum separation between them is specified by `\lineskiplimit`. If any two such lines contain an anomalously deep character on the first line, and/or an anomalously tall character on the second, then the probability is quite great that those two lines will be forced apart, to honour the constraints of `\lineskiplimit`; however, the probability of the anomalously deep character coinciding with an ascender in the line below, or of the anomalously tall character coinciding with a descender in the line above, is typically rather small: if TEX were to adopt a 'skyline'[14] model of each line, rather than the simplistic bounding-box model as at present, then such line pairs would not be forced apart unless it was absolutely necessary for legibility that they so be. Note that this does not require TEX to have any knowledge of the characters' *shape*; the present bounding-box model for characters is still satisfactory, at least for the purposes of the present discussion.
8. *Only partial orthogonality in the treatment of distinct entities:* TEX provides a reasonably orthogonal treatment for many of its entities (for example, the `\new`... family of generators), but fails to extend this to cover all entities. Thus there is no mechanism for generating new instances of `\mark`s, for example. Similarly, whilst `\the` can be used to determine the current value of many entities, `\the \parshape` returns only the number of ordered pairs, and not their values (there is no way, so far as can be ascertained, of determining the current value of `\parshape`). It is possible to `\vsplit` a `\vbox` (or `\vtop`), but not to `*\hsplit` an `\hbox`. The decomposition of arbitrary lists is impossible, as only a subset of the necessary `\last`... or `\un`... operators is provided. The operatorless implicit multiplication of <number><dimen-or-skip register> (yielding <dimen>) is also a source of much confusion; it might be beneficial if the concept were generalised to <number><register> (yielding <register-type>). However, this raises many related questions concerning the arithmetic capabilities of TEX which are probably superficial to our present discussion. I would summarise the main point by suggesting that orthogonality could be much improved.
9. *Inadequate parameterisation:* TEX provides a very comprehensive set of parameters with which the typesetting process may be controlled, yet it still does not go far enough. For example, one has `\doublehyphendemerits` which provide a numeric measure of the undesirability of consecutive hyphens; it might reasonably be posited that if two consecutive hyphens are bad, three are worse, yet TEX provides no way of indicating the increased undesirability of three or more consecutive hyphens. Also concerned with hyphenation is `\brokenpenalty`, which places a numeric value on the undesirability of breaking a page at a hyphen; again it might be posited that the undesirability of such a break is increased on a recto page (or reduced on a verso page), yet only one penalty is provided. A simple, but potentially infinite, solution would be to increase the number of parameters; a more flexible solution might be to incorporate the concept of formula-valued parameters, where, for example, one might write something analogous to `\brokenpenalty = {\ifrecto |500| \else |200| \fi}`, with the implication of delayed evaluation.
10. *Inadequate awareness of æsthetics:* TEX is capable of producing results which æsthetically are the equal or better of any computer typesetting system available today, yet the results may still be poorer than that achieved by more traditional means. The reason for this lies in the increased detachment of the human 'operator', who now merely conveys information to the computer and sits back to await the results. When typesetting was accomplished by a human compositor, he or she was aware not only of the overall shape of the text which was being created, but of every subtle nuance which was perceivable by looking at the shapes and patterns created on the page. Thus, for example, rivers (more or less obvious patterns of white space within areas of text, where no such patterns are intended), repetition (the same word or phrase appearing in visually adjacent locations, typically on the immediately preceding or following line), and other æsthetic considerations leapt out at the traditional typesetter, whereas TEX is blissfully unaware of their very existence. Fairly complex pattern matching and even image processing enhancements might need to be added

[14] This most apposite and descriptive term was coined by Michael Barr.

to TEX before it was truly capable of setting work to the standards established by hot-metal compositors.

Clearly one could continue adding to this list almost indefinitely; every system, no matter how complex, is always capable of enhancement, and TEX is no exception to this rule. I have quite deliberately omitted any reference to areas such as rotated text and boxes, support for colour, or support for graphics, as I believe them to be inappropriate to the current discussion: they are truly *extensions* to TEX, rather than deficiencies which might beneficially be eliminated. But I believe I have established that there *are* areas in which TEX is capable of being improved, and would prefer to leave it at that.

This brings us therefore to the final theme: how should we proceed? The NTS-L approach is obviously helpful, in that it allows the entire (e-mail connected) TEX community to contribute to the discussion, but I see at least two problems:

1. Those who are not on e-mail[15] are essentially excluded from the discussion; I do not see any easy solution to this problem.
2. The views expressed are, in some cases, radically different, and I wonder whether we will ever converge on a universally acceptable decision.

The second is in many ways the more important issue (Knuth apart), for unless the decisions made are acceptable to a very large majority of the contributors, the group may split, with part electing to go one route and another part electing to adopt a different strategy. This could result in a proliferation of *Über-TEX*s, with a concomitant fragmentation of the user community. Natural selection would surely winnow out the real non-starters before too long, but I seriously worry about the effect of such a proliferation on the TEX community, and even on TEX itself: after all, if we can't agree amongst ourselves whether there should be a successor to TEX, and if so what functionality it should possess, the whole credibility of the TEX ethos will be called into question. I would not like this to happen.

Somehow, therefore, we have to find a generally acceptable solution. My intuitive feeling is that such a solution will either be conservative or radical, but nothing in between. (This may seem like a distinct hedging of bets, but I hope that my meaning is clear: I believe that a compromise solution, which tries to be all things to all people, is doomed to failure). I do truly believe that adopting both solutions (one conservative, one radical) may be the best way forward: as an initial step, we identify (as I have tried to do above) any true *deficiencies* of TEX —those that actually prevent it from accomplishing its stated aims —and rectify those, producing a system that is backwards compatible with present TEX implementations whilst being capable of achieving superior results. In parallel with this (which is intended to be a reasonably short term and straightforward project, requiring not too much in the way of resources), we start planning a truly radical New Typesetting System, with the same fundamental design desiderata as TEX (portability, freely available, fully documented, bug-free. . .), but designed for the technology of tomorrow[16] rather than that of today.

Considering first the conservative approach, we will need to identify what is feasible, as well as what is desirable. Clearly this will require advice from those who are truly familiar with `TEX.WEB`, as I see this approach purely as modifications to the WEB rather than as a re-write in any sense. Chris Thompson and Frank Mittelbach are obvious candidates here, and Frank is already a member of the NTS team; I would suggest that if we adopt this strategy, Chris be invited to participate as well. Once we have identified what is possible, we will need a reasonably accurate estimate of time-to-implement, and if this exceeds that which can be achieved with volunteer labour, we will need to seek funds to implement this solution. I would suggest that TUG be approached at this stage (obviously they will have been kept informed of the discussions), and asked if they are willing to fund the project. There seems no point in projecting beyond this stage in the present paper.

For the radical approach, familiarity with WEB is probably unnecessary, and indeed may be a disadvantage: if we are seeking a truly NEW Typesetting System, then detailed familiarity with current systems may tend to obfuscate the issue, and certainly may tend to constrain what should otherwise be free-ranging thoughts and ideas. We will need to consult with those outside the TEX world, and the advice of practising typographers[17] and (probably retired) compositors will almost certainly prove invaluable. But above all we will need people with vision, people who are unconstrained by the present limits of technology, and who are capable of letting their imagination and creativity run riot.

And what conclusions might such a group reach? Almost by definition, the prescience required to answer such rhetorical questions is denied to mere mortals; but I have my own vision of a typesetting system of the future, which I offer purely as an example of what a New Typesetting System might be. Firstly (and despite my quite ridiculous prejudices against windowing systems), I believe it will inherently require a multi-windowing environment, or will provide such an environment itself (that is, I require that it will make no assumptions about the underlying operating environment, but will instead make well-defined calls through a generic interface; if the host system supports a multi-windowing environment such as Microsoft Windows

[15] Knuth is not on e-mail. . .
[16] and beyond. . .
[17] Michael Twyman and Paul Stiff have indicated a keen desire to be involved in the project.

or the X Window System, the NTS will exploit this; if the host system does not provide such intrinsic support, then it will be the responsibility of the implementor to provide the multi-windowing facilities). I envisage that perhaps as many as eight concurrent displays might be required: linked graphic and textual I/O displays, through which the designer will be able to communicate the underlying graphic design in the medium of his or her choice (and observe in the other window the alternative representation of the design); an algorithmic (textual) display, through which the programmer will communicate how decisions are to be made; two source displays, one text, one graphic, through which the author will communicate the material to be typeset; and a preview display, through which an exact facsimile of the finished product may be observed at any desired level of detail. A further display will provide interaction (for example, the system might inform the user that some guidance is needed to place a particularly tricky figure), and the last will enable the user to watch the system making decisions, without cluttering up the main interactive window. Needless to say, I assume that the system will essentially operate in real time, such that changes to any of the input windows will result in an immediate change in the corresponding output windows. I assume, too, that the input windows will be able to slave other unrelated programs, so that the user will be able to use the text and graphics editors of his or her choice. Of course, not all windows will necessarily be required by all users: those using predefined designs will not need either the design-I/O or the algorithm-input windows, and will be unlikely to need the trace-output window; but the interaction window may still be needed, and of course the source-input windows unless the source, too, has been acquired from elsewhere. For just such reasons, the system will be capable of exporting any designs or documents created on it in plain text format for import by other systems.

And underneath all this? Perhaps no more than a highly refined version of the TEX processor; totally re-written, probably as a procedural language rather than a macro language (why procedural rather than, say, list processing or declarative? to ensure the maximum acceptability of the system: there are *still* more people in the world who feel comfortable with procedural languages than with any of the other major genres), and obviously embodying at least the same set of enhancements as the interim conservative design, together with support for colour, rotation, etc. The whole system will, of course, be a further brilliant exposition of literate programming; will be placed in the public domain; will be capable of generating DVI files as well as enhanced-DVI and POSTSCRIPT; and will be so free of bugs that its creators will be able to offer a reward, increasing in geometric progression, for each new bug found. . .

But we will need one final element, and I have deliberately left this point to the very end: we will need the advice of Don Knuth himself. Don has now distanced himself from the TEX project, and is concentrating on *The Art of Computer Programming* once again. This detachment is very understandable —TEX has, after all, taken an enormous chunk out of his working (and, I suspect, private) life —and I hope that we all respect his wish to be allowed to return once again to 'mainstream' computer science, mathematics, and Bible study. But I think it inconceivable that we can afford to ignore his advice; and if I were to have one wish, it would be this: that I would be permitted to meet him, for whatever time he felt he could spare, and discuss with him the entire NTS project. I would like to know, above all, what changes *he* would make to TEX, were he to be designing it today, rather than fifteen years ago; I would like to know if he agrees that the deficiencies listed above (and those that appear elsewhere) are genuine deficiencies in TEX, or are (as I sometimes fear) simply the result of an inadequate understanding of the true power and capabilities of TEX; and I would like to know how he feels about the idea of an 'Extended TEX' and of a New Typesetting System (I suspect he would be far more enthusiastic about the latter than the former). And I suppose, if I am honest, I would just like to say 'Thank you, Don', for the countless hours, days, weeks, months and probably years of pleasure which TEX has given me.

References

[1] Donald E. KNUTH: "The Future of TEX and METAFONT", in *TUGboat*, Vol. 11, No. 4, p. 489, November 1990.

[2] Frank MITTELBACH: "E-TEX: Guidelines for future TEX", in *TUGboat*, Vol. 11, No. 3, pp. 337–345, September 1990.

[3] Michael VULIS: "Should TEX be extended?", in *TUGboat*, Vol. 12, No. 3, pp. 442–447, September 1991.

[4] Zlatuška, Jiří(ed): *EuroTEX '92 Proceedings*, pp. 235–254, September 1992. Published by CSTUG, Czechoslovak TEX Users Group, ISBN 80-210-0480-0.

Appendix

Inaugural members of the NTS-L team

- Rainer Schöpf (Chairman)
- Peter Abbott
- Peter Breitenlohner
- Frank Mittelbach
- Joachim Schrod
- Norbert Schwarz
- Philip Taylor

E-TEX: Guidelines for Future TEX Extensions*

Frank Mittelbach

Electronic Data Systems (Deutschland) GmbH
Eisenstraße 56, D-6090 Rüsselsheim,
Federal Republic of Germany
mittelbach@mzdmza.zdv.uni-mainz.de

Abstract

With the announcement of TEX 3.0, Don Knuth acknowledged the need of the (ever growing) TEX community for an even better system. But at the same time, he made it clear, that he will not get involved in any further enhancements that would change the TEXbook.

TEX started out originally as a system designed to typeset its author's own publications. In the meantime it serves hundreds of thousands of users. Now it is time, after ten years' experience, to step back and consider whether or not TEX 3.0 is an adequate answer to the typesetting requirements of the nineties.

Output produced by TEX has higher standards than output generated automatically by most other typesetting systems. Therefore, in this paper we will focus on the quality standards set by typographers for hand-typeset documents and ask to what extent they are achieved by TEX. Limitations of TEX's algorithms are analyzed; and missing features as well as new concepts are outlined.

*Published in TUGboat, Volume 11 (1990), No 3 — 1990 Conference Proceedings.

E-TEX: Guidelines for Future TEX Extensions

Frank Mittelbach
Electronic Data Systems (Deutschland) GmbH
Eisenstraße 56, D-6090 Rüsselsheim, Federal Republic of Germany
Tel. +49 6142 803267
Bitnet: mittelbach@mzdmza.zdv.uni-mainz.de

Abstract

With the announcement of TEX 3.0, Don Knuth acknowledged the need of the (ever growing) TEX community for an even better system. But at the same time, he made it clear, that he will not get involved in any further enhancements that would change *The TEXbook*.

TEX started out originally as a system designed to typeset its author's own publications. In the meantime it serves hundreds of thousands of users. Now it is time, after ten years' experience, to step back and consider whether or not TEX 3.0 is an adequate answer to the typesetting requirements of the nineties.

Output produced by TEX has higher standards than output generated automatically by most other typesetting systems. Therefore, in this paper we will focus on the quality standards set by typographers for hand-typeset documents and ask to what extent they are achieved by TEX. Limitations of TEX's algorithms are analyzed; and missing features as well as new concepts are outlined.

1 Introduction

Last year at Stanford we celebrated the tenth birthday of the TEX project. Up to now, TEX has served thousands of users well and we expect it will continue to do so in the future. The longevity of TEX lies in

- the quality of its output
- its universal availability
- and its stability.

In the last few years, more and more users brought TEX from the universities into industry where it was challenged by new applications [33]. But time does not stand still, and what was at the top of its profession yesterday might prove to be obsolete tomorrow. TEX is still state of the art for the tasks it was designed to accomplish, but, with the growing understanding from several years' usage, we can now see where it will fail in high quality typesetting.

As a result of user pressure [27], Don Knuth announced a new version of TEX at Stanford, acknowledging the fact, that he did not foresee the need for 8-bit input [19]. At the same time, he made it clear, that he had decided to retire from this project and return to his long delayed topic "The Art of Computer Programming".

So TEX is finally frozen, and any further development will result in a different system no longer maintained by Knuth. The main purpose, therefore, of this paper is to give an overview of high quality typesetting requirements (covered and not covered by TEX 3.0) thereby, we hope, channeling future developments so that we do not end up with several incompatible "TEX-based systems", but rather with one system that will provide the same characteristics (i.e., quality, portability, and availability) as the current program.

TEX was designed as a low-level formatter, a stable kernel, of a typesetting system where extensions at both ends would be possible to take into account developments in printing technology (back end) and in user interfaces (front end) [14]. Thus, complaints about user unfriendliness of TEX are uncalled for, since such requirements can be handled by front ends either written in the TEX language itself like LATEX and, therefore, fully portable, or in an external language like ArborText's Publisher, or VAX Document, etc. These systems use TEX or a TEX-based system as the ultimate formatter but provide a user-friendly interface [32].

When we discuss missing features, we must distinguish carefully between things which can and should be handled by a front end system and things that are truly tasks for a formatter and cannot be

handled in TEX 3.0. In the following sections we analyze features required for high quality typesetting, discussing whether they can be handled by TEX primitives or by a suitable front end, or both. If they cannot be handled, we attempt to find ways to achieve the desired results. Finally, in section 12, we switch our attention to the concepts of the TEX language itself, outlining some ideas on how a language for a new system could describe the underlying concepts more clearly.

2 Line breaking

TEX's line breaking algorithm is clearly a central part of the TEX system. Instead of breaking a paragraph line by line, the algorithm regards paragraphs as a unit and searches for an 'optimal solution' based on the current values of several parameters. Consequently, a comparison of results produced by TEX and other systems will normally favour TEX's methods.

Such an approach, however, has its drawbacks, especially in situations requiring more than block style text of a fixed width. The final line breaks are determined at a time when information about the content of the current line has been lost (at least for the eyes of TEX, i.e., its own macro language), so that TEX provides no sort of post-processing of the final lines based on their content.

Furthermore, there is no way to influence the paragraph shape with regard to the current position on the page, since this information is not known *a priori*. See section 4 for further discussion of this topic.

The use of only four categories (tight, decent loose, very loose) to distinguish different glue–settings in adjacent lines seems somewhat inadequate The number of categories should be increased In addition, a more global approach (even beyond paragraph borders in certain circumstances), taking the overall variation of glue-setting into account might produce better results.

2.1 Line breaking parameters

While the algorithm provides a variety of parameters to influence layout, some important ones for quality typesetting are missing. There is no way to deal with vertical stripes produced by interword gaps falling into the same vertical position. A similar problem involves identical words one above the other, especially at the beginning of a new line. Both problems are distracting to the eyes of the reader and will destroy any effort to produce a beautifully broken paragraph. A good example is shown at the beginning of the third paragraph of section 4 where "...breaking algorithm ..." is repeated on two lines.

Another aspect of fine print is the assurance that the last line of a paragraph will not be too short. This is especially important in layouts which use paragraph indentation, where an undesired gap would be produced if the last line of one paragraph is shorter than the indentation of the next paragraph. Unknown to most TEX users, this can be prevented by a special setting of TEX's line breaking parameters as shown in example 1 in section 14. While other parts of this paper use this setting, this paragraph shows the undesired effect.

Hyphenation of consecutive lines is handled for up to two lines (`\doublehyphendemerits`), but there is no possibility of avoiding paragraphs like the current one and the next one, in certain circumstances. As one can easily observe, the number of hyphens in these paragraphs is artificially forced by setting some of TEX's line breaking parameters to unusual values. But in non-English languages (with longer word lengths on the average), such situations present real-life problems.

Another problem is the discrepancy between the first and later lines of a paragraph, produced by the implementation of the paragraph indentation. This is especially crucial in layouts with zero indentation, because space at the beginning of the first line (for example, from `\mathsurround`) will not vanish into the margin because of the implicit `\hbox` representing the indentation (even if not visibly present), while such space will be removed at the beginning of later lines. This will result in strange starting gaps.

3 Spacing

When block text is to be produced, it is necessary to change the interword or the intercharacter spacing, or both. Since variable intercharacter spacing is frowned upon by the experts (except in rare circumstances), a line breaking algorithm has to stretch or shrink the interword space starting from an optimal value given by the font designer until the final word positions are determined. Again, TEX has a well designed algorithm to take such stretchability into account. Additionally, each character has a so called `\spacefactor` assigned to it which will influence a following space, so that it is possible to enlarge or reduce the interword space after certain characters. As an example, compare the spacing after punctuation characters in this paragraph with other paragraphs.

There is no provision, however, for influencing the interword gaps in relation to the current characters on both word boundaries. If it is necessary to shrink a given line, not all gaps should shrink by the same amount. Instead, it is best to shrink more af-

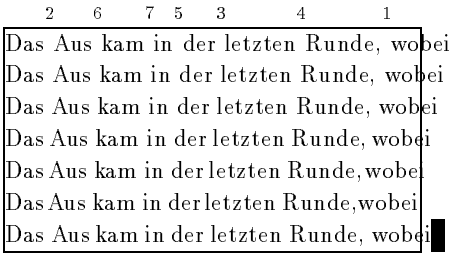


Figure 1: Interword spacing

The interword spaces are numbered in a way so that higher numbers denote spaces which should shrink less using the rules given by Siemoneit [28]. The last line shows the resulting overfull box which would be produced by standard TEX in this situation.

ter a comma, for example, than between 'then it' because of the different shape of the characters. There is no way to achieve such fine tuning in TEX except by manually adding `\hskip` in lines, which is intolerable. An example of this approach is shown in figure 1. Such a mechanism is clearly font dependent, and an implementation would, therefore, change both TEX and METAFONT, since the best place to store this information is in the `TFM` file. But even tables similar to `\sfcode` (fixed by the format or a macro) would be a big improvement, since most fonts in use tend to have similar shapes.

In TEX's concept for glue-setting an important distinction is made between stretchable and shrinkable glue: while the latter is only allowed to shrink to a fixed minimum (i.e., the natural width minus the shrink component), any given amount of stretchable glue is automatically allowed to stretch arbitrarily far.[1] The reason for this behavior is that it allows the line breaking algorithm to achieve 'emergency results' if no suitable line breaks are otherwise found. But this is undesirable in most circumstances, so that either the stretching should be bounded similarly to shrinking in all cases (resulting in some changes to the line and page breaking algorithms), or another class of glue should be added, for which the amount of stretching can be determined individually.

Don Knuth [18, pp. 394–395] gives an example of how to achieve hanging punctuation (together with special fonts, as he noted). Since this, too, is a sign of good quality typesetting, it is questionable whether such a scheme (that will make the ligature mechanism partly unusable, along with other side effects) is advisable or whether this should be a direct feature of a future program.[2]

4 Page breaking

A major problem with TEX is its page breaking algorithm. Page breaking is handled asynchronously by moving things at certain times from the list of recent contributions onto the "current page" until this list is filled with more items than will fit on the page in final form. The final page break is chosen by weighing badness (how full the page is if we break here) and penalties (how expensive it is to break here). Such penalties will be placed after some of the lines either by the line break algorithm or during macro expansion.

But good page layout usually requires taking pairs of facing pages into account as they will be seen by the reader. This is in itself not a real restriction, because one can view a double page as a huge case of two-column format, provided, of course, that both pages can be held simultaneously in memory. But, unfortunately, none of TEX's internal mechanisms can handle multi-column layout properly, so that such an approach has to avoid all internal features for page breaking like `\insert`, etc. Good examples that also show the limitations of TEX in this regard are the output routine of LATEX [22] and implementations of multi-column layout [4, 24], all bordering on the impossible.

But, even more important, the line breaking algorithm in conjunction with the page breaking algorithm pose unsolvable problems. When the final page break is chosen by TEX, all paragraphs which were once candidates for the current page are already divided up by the line breaking alogrithm, and this division cannot be undone for text carried over to the next page, since some of the necessary information (space at the line breaks, for example) is lost. This makes it impossible to change the page layout at a fixed place, e.g., at the top of a new page, to leave room for a small figure surrounded by text. Only in very restricted circumstances a solution can be found inside TEX [9], but documents of moderate complexity cannot be handled this way. A general solution to this problem can be included in the current TEX in an upward compatible manner. A prototype was designed at the University of Mainz shortly after the conference at Stanford [25].

It is an open question whether we should follow TEX's page breaking algorithm at all, since it was stopped short because of the space and time constraints of the computers available at the time of its development. In his PhD thesis [26], M. Plass considered several global optimization strategies, using a two pass system. His results open a wide field for future research work. Some of his ideas seem to be used in the Type & Set system [3].

1 Under normal circumstances, however, this is prevented by the badness function.

2 Starting with the next section the article uses hanging punctuation. The change in quality is clearly visible although improvement is still possible by making subtle ajustments to all characters (e.g., move the 'r' a tiny bit out, etc.) to reach a perfect alignment.

The main contribution of TEX82 to computer based typesetting was the step taken from a line-by-line paragraph breaking algorithm to a global optimizing algorithm.[3] The main goal for a future system should be to solve the similar, but more complex, problem of global page breaking.

5 Page Layout

For the tasks of page makeup, TEX provides the concept of output routines together with insertions and marks. The concepts of insertions and marks are tailored to the needs of a relatively simple page layout model involving only one column output, footnotes, and at the most simple figures once in a while.[4]

The mark mechanism provides some information about certain objects and their relative order on the current page, or more specifically, information about the first and last of these objects on the current page and about the last of these objects on any of the preceeding pages. Such information is necessary to construct certain kinds of running heads, e.g., one with the name of the current chapter or with information about the first and last word explained on the page, etc.

This is a global mechanism, however, so that only one class of objects can take advantage of the whole mechanism. If more than one class is implemented, some of the features of the mechanism are lost within one class.[5] As a consequence of this deficiency, one should extend the mark mechanism to a system of independent marks which can be allocated separately by a macro package.

The insertion mechanism seems to be derived from 'footnote applications', and later extended to allow for some simple kinds of floating insertions.[6] But placement of floating objects needs more than simply storing them in a huge box which is split at a certain point when the output routine is called. Floats are accompanied by captions and the like, which require differing treatment depending on their final placement on the page. Floats may vary in width even if they belong to the same class. On the other hand, deferred floats in one class may influence or even prohibit the placement of floats in other classes.

Some classes of insertions, like marginal notes, cannot be handled by the primitives at all. To provide such features LATEX, for example, defines its own memory management for floating objects. Naturally, such a mechanism is slow and space consuming. Additionally, the quality of page breaks is further reduced because it is difficult to maintain, for free, all the information provided by the insertion concept.

Another problem is the design decision that the page breaking mechanism is, at least in its crucial parts, available only in outer vertical mode; thus, for example, space for insertions is not taken into account when splitting a `\vbox`.

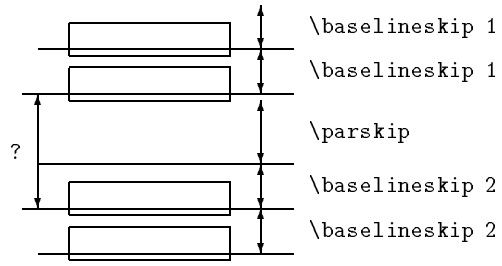


Figure 2: Baseline to baseline spacing

To implement a baseline to baseline dimension, for example between a paragraph and a heading (denoted by the question mark), the value for `\parskip` has to be determined in dependence of the `\baselineskip` of the second paragraph. Unfortunatly the value of `\baselineskip` used will be the one current at the end of the second paragraph while the `\parskip` has to be computed at its beginning.

For a designer TEX's model of interline glue determination is very unfamilar because it does not allow to specify baseline to baseline spacing in a page-spec without using lengthly and complicated internal computations (see figure 2 on the next page). This also means that it is nearly impossible to implement grid-oriented specs, i.e., where (nearly) all baselines fall into predetermined positions. This article uses a grid-oriented spec (which was partly handprepared) to show this aspect of high quality typeseting. Details are given in example 3.

The design of suitable primitives for this complex must go hand in hand with a new algorithm for page breaking, comprising probably the most drastic changes to the TEX system.

3 It should be noted that a similar algorithm was developed independently by J. Achugbue [2]. A comparison might lead to further enhancements.

4 The term 'one column output' means that all text is assembled using the same line width. Problems with variable line width are discussed in section 4. Of course, this already covers a wide range of possible multi-column layouts, e.g., the footnote handling in this article. But a similar range of interesting layouts is not definable in TEX's box-glue-penalty model.

5 The LATEX implementation provides an extended mark mechanism with two kinds of independent marks with the result that one always behaves like a `\firstmark` and the other like a `\botmark`. The information contained in the primitive `\topmark` is lost.

6 This is only a guess from studying [17].

6 Penalties — measurement for decisions

Line and page breaks in TEX are determined chiefly by weighing the "badness" of the resulting output[7] and the penalty for breaking at the current point. Such penalties are either inserted directly by the user (during macro expansion) or added later by certain TEX formatting routines.

The main problem posed by the implicit penalties is that they cannot be removed. If, for example, the line breaking algorithm decides to put a penalty after a line (e.g., from `\widowpenalty`), there is no way to prohibit a page break at this place via macro expansion except, of course, by setting the penalty in question to infinity. This is the result of TEX's algorithm that consecutive penalties p_1 and p_2 behave like $p_3 := \min(p_1, p_2)$.

Local changes to the offending penalty parameters are error prone and time consuming, since after each change the following page breaks might fall in different places. Additionally, future editions of the document are made more difficult because every correction of this sort might produce undesired results after a single change.

If we think in terms of the current TEX (i.e., assuming all major algorithms are unchanged), it would be better to adopt a different strategy in the case of consecutive penalties, either $p_3 := \max(p_1, p_2)$ or $p_3 := \frac{1}{2}(p_1 + p_2)$. For both functions, the boundary cases $p_i = \pm\infty$ would need special care. Breaking the chain of penalties could be performed as usual by grouping or `\kern0pt` or similar procedures as is already done with ligatures, etc.

As we mentioned, this is a solution within the framework of TEX82. If a totally different algorithm for page breaking is designed, the concepts of local penalties should be reconsidered, too, and probably be replaced by a different strategy.

7 Hyphenation

When typesetting text, especially in narrow columns, hyphenation is often inevitable in order to avoid unreadable, spaced out lines. But readability has many faces; one of the golden rules says [28] "Avoid more than two hyphenated lines in a row." As we mentioned in section 2, this cannot be specified in TEX (unless one disables even two consecutive hyphens).

Another problem is hyphenation of words at places which are allowed but which distort the meaning:

Stiefel-tern	Stief-eltern
Spargel-der	Spar-gelder[8]

One should probably always forbid such problematical hyphens by choosing appropriate patterns for Liang's algorithm [23]. But readability is also distorted by the hyphenation of very short syllables which give no or almost no information about the word hyphenated and, therefore, slow down the reading process considerably. With TEX 3.0 it is now possible to adjust the minimal number of letters to the left and right of a hyphen. This is necessary for many languages which often have long words and, for example, many two letter syllables like German. But there is no provision for provision for assigning weights to hyphenation points, i.e., it is a simple yes or no situation. One possible solution to this problem would be to add another class of demerits which could be applied via

$$\frac{\text{users value}}{\text{length of broken part}}$$

or a similar function. Of course, one probably has to distinguish between pre-break and post-break text (and/or length) in the formula.

Since the quality of a certain breakpoint also depends on the word (i.e., the meaning of the word-parts), we should consider whether such information could be provided by the hyphenation algorithm itself.

8 Box Rotation

TEX's concept of document representation is strictly horizontal and left to right oriented. Beside the problem of processing documents containing right-left or top-down oriented languages (which can be handled to some extent by special versions of TEX [21]), this also poses unnecessary restrictions in standard applications. Except by using `\special` (for PostScript devices) it is impossible to rotate certain parts of the document. While arbitary rotation is indeed next to impossible for most output devices, rotation by 90° can be handled in a simple manner by rotating the character cells. It should be easy to include some sort of `\rotate` primitive to TEX's language which would allow rotating hboxes and vboxes by multiples of 90 degrees.[9] This would allow inclusion for example, of landscape tables, etc., in a document, without the need to use real glue and scissors to add the page number or the running head.

9 Font Information

The ISO Draft Standard [1] contains hundreds of properties describing a font resource. While some of

7 This is in some sense a measure of the difference between the optimal and actual amount of white space on the line or page in question.

8 The meanings of the words are 'step-parents' and 'savings', but the first parts of the words in the left columns mean 'boot' and 'asparagus', respectively.

9 This would also require changes to the `dvi` language and thus changes in all driver programs.

them are accessible through TEX via `\fontdimen`, the majority are not. It seems advisable to add more of them to the set of TEX's parameters (for example, a recommended `\baselineskip`), in order to be able to make font family changes in documents more easily.

9.1 Virtual Fonts

Use of font families in TEX, which differ in character position, etc., from the defaults in the Computer Modern family, is difficult but can be achieved as proved in several projects [7, 35]. The proposed use of virtual fonts [20] may help to simplify matters in this regard. If it is possible to agree on standards for the position and the method of access for certain accented characters for the most common Latin alphabet languages, it should be possible to typeset multilingual documents (using the new features of TEX 3.0) without introducing unnecessary variants of standard fonts differing only in the availability of certain accented characters as 'real' letters.[10] A good survey of accented characters in Latin alphabet languages and a proposal for their access via ligatures is given by Haralambous [8].

9.2 Ligatures and Kerns

Unfortunately, ligatures as well as kerns differ from language to language. Take, for example, the 'fflʼ ligature which is not used in traditional German documents. On the other hand, such documents contain 'ch', 'ck' and 'ft' ligatures to obtain a better script. The following examples shows the difference:

Druckschrift	Druckschrift	(standard)
Druckschrift	Druckschrift	(German ligatures)

Since these special ligatures do not involve new letter shapes (at least not in most font families) it is possible to achieve the desired results simply with kerning. In the Computer Modern font family [15], both 'ch' and 'ck' are contained in kerning programs, but only for serif fonts. Other font families show similar deficiencies. Thus, for typesetting German documents, one either needs special physical fonts (or at least virtual fonts), or a way to manipulate ligature and kerning programs from within the TEX program. For reasons of portability, a controlled access to the ligature/kerning programs during font loading seems preferable.

10 Tables

Well-designed tables are difficult to typeset even for experienced hand composers. TEX's primitives `\halign` and `\valign` do a marvellous job in this respect, even in complex situations. There is one important subclass of tables, however, which cannot be handled at all, except with hand tailoring. It is not possible to specify combinations of horizontally and vertically spanned columns, e.g., an open curly brace spanning several rows in one column while the row structure is maintained on both sides.

Another feature often desired is the ability to specify tables spanning several pages. While this is difficult to achieve, since TEX's table primitives normally read the whole table before determining the column width, etc., it does not pose unsolvable problems with the advanced features of TEX 3.0.

11 Math

Mathematical typesetting is one of TEX's major domains where no other automatic typesetting system has been able to catch up. But even in this area several things could be improved.

The source code of $\mathcal{AMS}$-TEX [30] shows many interesting examples where Spivak circumvents limitations of TEX's formatting rules by introducing complex code to define functions that should perform standard tasks in mathematical typesetting. A detailed analysis of these problems (double accents, under accents, placement of equation numbers, etc.) would easily fill several pages; some comments can be found in Spivak's documentation [29].

While TEX's spacing rules for math are quite good, it seems at least questionable that many of them are hardwired into the program instead of being accessible through parameters. The table for spacing between different math-atoms is probably the most important example of this sort.

Another problematical feature of TEX's math typesetting routines is that sub-formulas are always boxed at natural width even if the top level mathlist is subject to stretching or shrinking. This might produce ugly results in certain circumstances. The concept of boxing sub-formulas has the additional disadvantage that such parts of a formula cannot be broken across lines. Therefore programming constructs like `\left`...`\right` which automatically determine the height of variably sized delimiters, cannot be used in complex displays.

12 TEX's language

The language of TEX is divided into two parts which are described as the mouth and the stomach of TEX [18]. This distinction is crucial in many applications since the output produced by TEX's gastronomical routines cannot be fed again into its mouth, i.e., its scanner. Actually, constructed boxes are post-processable to a limited extent (via `\lastbox`, `\unpenalty`, etc.) but arbitrary constructions cannot be handled this way because primitives for manipulating things like characters, rules, etc., are missing. In general, this distinction is the reason for

10 The use of the accent primitive of TEX is not recommended for standard accents of a language [18, p. 54] since it disables the hyphenation facility.

many obstacles in TEX-programming, which sometimes prevent any solution at all. A new system should remove this two-class society of internal commands.

Another severe problem of the language is its incompleteness regarding standard programming constructs (such as certain conditionals, an acceptable arithmetic parser, etc.), as well as special constructs suitable for typesetting. To determine, for example, the length of the last line in a paragraph (which is done automatically by TEX when typesetting displays), one has to use a complicated and lengthy computation, as shown in example 2, below. This example also shows one of the inconsistencies of TEX's language: `\prevgraf` has to be advanced using a scratch register because the direct use of `\advance` is forbidden. Many problems of this sort can be found by looking at appendix D of the *The TEXbook* [18, pp. 373–401] which is entitled "Dirty Tricks". Actually nine out of ten examples therein are used in the implementation of LATEX [22], which shows that these examples are far less exotic than the preface to this appendix suggests. As examples of missing programming constructs, conditionals, such as `\ifmathopen`, for determining math atoms should be mentioned.

Such problems explain the fact that general application software written in TEX (like LATEX) easily takes up more than a third of the available memory without typesetting even a single letter. For better and more stable front ends one needs a language where such tasks can be specified in a more elegant manner.

Some of TEX's restrictions in the language are due to the representation of dimensions in most TEX installations as 'real numbers', which are machine-dependent.[11] To make TEX nevertheless machine-independent, Knuth tried to prevent machine-dependent results generated in TEX's stomach from creeping into parts accessible to the scanner, or to influence internally any decisions about line or page breaks. As a result of this strategy, the use of the code in example 2 together with a finite `\parfillskip` (as in example 1) will nearly always produce the value `\maxdimen` instead of a decent one.[12] For the same reason, Knuth said in a conversation with the author at Stanford that he cannot permit the removal of arbitrary items in a constructed list since this would allow access to floating-point arithmetic. But there exists another way to achieve machine-independence, which would also eliminate the restrictions mentioned, namely to change from floating-point to fixed-point arithmetic[13] which can be done in a straightforward way, as Knuth himself has acknowledged [16, p. 46].

TEX is a macro-language with all the advantages and disadvantages. Anyone who ever wrote a relatively long application in TEX knows that debugging is extremely difficult. Transparent programming, as proposed by the author of TEX [10], is next to impossible: it is no problem to write three lines of TEX code that cannot be understood even by TEXperts without a second and third look. But it is much more difficult to write TEX code that performs a desired function, and is, at the same time, understandable to the average user. The examples given in section 14 are good test cases; they are all straightforward TEX-coding, but their precise meanings are difficult to understand without explanatory text.

Many problems arise from design decisions based on totally different semantic constructs, which have similar or identical syntactical structure. The most important examples are the curly braces and the dollar sign.[14] The curly braces are used both for delimiting arguments during macro expansion, as well as for the start and end of block structures that define the scope of certain declarations. In math mode they have the additional meaning of delimiting the scope of a sub-formula. Two consecutive dollar signs normally start or end a display formula, but in restricted horizontal mode they simply denote an empty math formula. Such concepts should be unraveled for the sake of clarity.

TEX's language is suitable for simple programming jobs. It is like the step taken from machine code (of the formatter) to assembly language. For complex programing tasks of general application software, especially from the viewpoint of logically tagged documents [5], a more powerful language with well-defined concepts for variable-bindings, procedures, etc, is preferable. While this aspect can be achieved in a front end programming language (which compiles into the TEX language) it is better to include it, for the sake of portability, in the TEX kernel. In the author's opinion, an ideal language should combine the advantages of procedural languages with the goodies of interpreter features.[15] As a side effect, such a language would be partly compilable.

13 Conclusion

The current TEX system is not powerful enough to meet all the challenges of high quality (hand) ty-

11 Actually, this only applies to internal dimensions representing stretching or shrinking of glue, computed kerns for accents, and some others.

12 The reason is given in module 1148 of the TEX program [16, p. 470].

13 Perhaps, using always the same floating-point algorithm (either available in the compiler library or simulated by the program) would be even better.

14 To be more exact, the three characters with `\catcode` one, two, and three.

15 Some sort of LISP-like system, but with primitives suitable for typesetting.

pesetting. The author shares Knuth's dream of a stable, low-level formatter which is able to produce documents of highest quality. But, unlike Knuth, he views the current TEX only as a very good prototype on the way to reach this goal.

As outlined in this paper, many important concepts of high quality typesetting are not supported by TEX 3.0. Further research is necessary to design a typesetting language which can handle these tasks properly.

The TEX user community needs an open mind for new developments that keep the 'TEX-System' the state of art in the field of computer typesetting. As Knuth is no longer involved in research on typography it is important for TUG to find an identity in *supporting* and *maintaining* 'the best typesetting program' and not only promoting the program that Knuth has given to the world. If we don't strike for even further quality our large community might fall back to insignificance.

One important step for TUG would be to initiate and (when advisable) support further research projects which will take up the challenges posed by the Stanford project.

14 Examples

Example 1 To avoid nearly empty lines at the end of a paragaph, the following code could be used:

```
\parfillskip \columnwidth
\advance \parfillskip -1.5\parindent
\advance \parfillskip 0pt minus \parfillskip
\advance \parfillskip 0pt minus  -1em
```

This setting was used throughout this article, for example, which led to some changes in section 2. With the standard setting (e.g., `0pt plus 1fil`), the first and third paragraph therein would have ended with the word part 'ods' (from 'meth-ods') and the word 'topic' respectively. Unfortunately, this solution is not perfect either, since it produces somewhat funny results with lines consisting of two very short words.

Example 2 The following code determines the length of the last line of the preceeding paragraph, using a feature of TEX built into mathematical displays. This code can be used to determine, for example, the amount of white space before an itemized list, or something similar. The example of code given is not really suitable for direct applications of this sort, since it simply displays the value found on the terminal. But it could easily be extended.

```
\def\getlastlinewidth{\ifhmode $$%
 \predisplaypenalty\@M \postdisplaypenalty\@M
 \abovedisplayskip-\baselineskip \belowdisplayskip\z@
 \abovedisplayshortskip\abovedisplayskip
 \belowdisplayshortskip\belowdisplayskip
 \showthe\predisplaysize
 $$\count@\prevgraf \advance\count@-\thr@@
 \prevgraf\count@ \else\typeout{*Not hmode*}\fi}
```

This example illustrates several important things. First, there is no elementary way to compute such important information. Second, it is one of the (not unusual) cases where information about the typesetting process can only be got by introducing undesired (since space-consuming) penalties, glues, and null-boxes in the output.

Example 3 To introduce a grid-oriented spec all flexible glue on the page has to be disposed off (execpt for `\skip\footins`) and the `\vsize` must be adjusted. Titles are set with `8pt + 4pt = \baselineskip` leading and we have to ensure that the above space is kept after a page break. Lists are set with `6pt + 6pt` so the inner lines are half way off. Page breaks insides lists would need special treatments, e.g., by increasing `\topskip` to keep the subgrid. Figures and examples in different type sizes are measured and necessary kerns added to keep the surrounding material in line. Again this approach only works if no page break intervence, which happens to be the case for this article. To use the badness calculation of TEX for determing page breaks a stretchable `\topskip` can be used. During the output routine this extra stretch must then be canceled again.

References

1. *Information Processing — Font Information Interchange, ISO/IEC JTC 1/SC 18/WG8 N1036*, February 1990.
2. Achugbue, James O. "On the line breaking problem in text formatting." *Proc. of the ACM SIGPLAN/SIGOA*, 2(1, 2), 1981.
3. Asher, Graham. "Type & Set: TEX as the engine of a Friendly Publishing System." Pages 91 – 100 in *TEX applications, uses, methods*, Malcolm Clark [6].
4. Benson, Gary, Debi Erpenbeck, and Jannet Holmes. "Inserts in a multiple-column format." Pages 727 – 742 in *1989 Conference Proceedings*, Christina Thiele [31].
5. Bryan, Martin. *SGML: an author's guide to the standard generalized markup language*. Addison-Wesley, Woking, England; Reading Massachusetts, second edition, 1988.
6. Clark, Malcolm, editor. *TEX applications, uses, methods*, Chichester,West Sussex, England, 1990. Ellis Horwood Limited. Exeter con-

ference July 1988.
7. Conrad, Arvin C. "Fine typesetting with TEX using native autologic fonts." Pages 521–528 in *1989 Conference Proceedings*, Christina Thiele [31].
8. Haralambous, Yannis. "TEX and latin alphabet languages." *TUGboat*, 10(3):342–345, November 1989.
9. Honig, Alan. "Line-Oriented Layout with TEX." Pages 159–184 in *TEX applications, uses, methods*, Malcolm Clark [6].
10. Knuth, Donald E. "Literate programming." *The Computer Journal*, 27:97–111, 1984. An expository introduction to `WEB` and its underlying philosophy.
11. Knuth, Donald E. *Computers & Typesetting*. Addison-Wesley, Reading, Massachusetts, 1986. Consists of [18, 16, 13, 12, 15].
12. Knuth, Donald E. METAFONT*: The Program*. Volume D of *Computers & Typesetting* [11], 1986.
13. Knuth, Donald E. *The* METAFONT*book*. Volume C of *Computers & Typesetting* [11], 1986.
14. Knuth, Donald E., September 1987. Talk given at Gutenberg Museum Mainz.
15. Knuth, Donald E. *Computer Modern Typefaces*. Volume E of *Computers & Typesetting* [11], July 1987. Reprint with corrections.
16. Knuth, Donald E. *TEX: The Program*. Volume B of *Computers & Typesetting* [11], May 1988. Reprint with corrections.
17. Knuth, Donald E. "The errors of TEX." Technical Report STAN-CS-88-1223, Stanford University, Department of Computer Science, Stanford, California 94305, September 1988.
18. Knuth, Donald E. *The TEXbook*. Volume A of *Computers & Typesetting* [11], May 1989. Eighth printing.
19. Knuth, Donald E. "The new versions of TEX and METAFONT." *TUGboat*, 10(3):325–328, November 1989.
20. Knuth, Donald E. "Virtual Fonts: More fun for Grand Wizards." *TUGboat*, 11(1):13–23, April 1990.
21. Knuth, Donald E. and Pierre MacKay. "Mixing right-to-left text with left-to-right text." *TUGboat*, 8(1):14–25, April 1987.
22. Lamport, Leslie. `latex.tex`, February 1990. LATEX source version 2.09.
23. Liang, Franklin Mark. *Word Hy-phen-a-tion by Com-put-er*. PhD thesis, Stanford University, Department of Computer Science, Stanford, CA 94305, August 1983. Report No. STAN-CS-83-977.
24. Mittelbach, Frank. "An environment for multicolumn output." *TUGboat*, 10(3):407–415, November 1989.
25. Mittelbach, Frank. Letter to Don Knuth, September 1989. Suggestions for the TEX 3.0 release. Published by R. Wonneberger in [34].
26. Plass, Michael Frederick. *Optimal Pagination Techniques for Automatic Typesetting Systems*. PhD thesis, Stanford University, Department of Computer Science, Stanford, CA 94305, June 1981. Report No. STAN-CS-81-970.
27. Rynning, Jan Michael. "Proposal to the TUG meeting at Stanford." *TEXline*, 10:10–13, May 1990. Reprint of the paper that triggered TEX 3.0.
28. Siemoneit, Manfred. *Typographisches Gestalten*. Polygraph Verlag, Frankfurt am Main, second edition, 1989.
29. Spivak, Michael. `amstex.doc`, 1990. Comments to [30].
30. Spivak, Michael. `amstex.tex`, 1990. $\mathcal{AMS}$-TEX source version 2.0 (without comments).
31. Thiele, Christina, editor. *1989 Conference Proceedings*, volume 10#4 of TUGboat. TEX Users Group, December 1989.
32. Wittbecker, Alan E. "TEX enslaved." Pages 603–606 in *1989 Conference Proceedings*, Christina Thiele [31].
33. Wonneberger, Reinhard. "TEX in an industrial environment." In Brüggemann-Klein, Anne, editor, *1989 EuroTEX Conference Proceedings*, 1990. To appear.
34. Wonneberger, Reinhard. "TEX yesterday, today, and tomorrow." *TEXhax*, 90(5), January 7 1990.
35. Youngen, R. E., W. B. Woolf, and D. C. Latterner. "Migration from computer modern fonts to times fonts." Pages 513–519 in *1989 Conference Proceedings*, Christina Thiele [31].

The LaTeX3 Project

Frank Mittelbach* **and Chris Rowley†**

26 February 1993
Version 1.00

Abstract

This is a brief sketch of the LaTeX3 Project: background, history, principles, aims and functionality. The new version of LaTeX is, like the current version, a freely available system for automated processing of structured documents, formatting them to the highest typographic standards by use of the TeX typesetting software.
Although its uses include a very large range of published documents, the importance of its unsurpassed ability to format mathematical formulas will not be forgotten in producing the new version.
It is being produced by an international group of volunteers under the technical direction of Frank Mittelbach.

1 Why a new version?

With TeX, Knuth designed a formatting system [7] that is able to produce a large range of documents typeset to extremely high quality standards. For various reasons, including its quality, portability, stability and availability, TeX spread very rapidly and can nowadays be best described as a world-wide de facto standard for high quality typesetting. Although it is most famous for its ability to typeset mathematics, it is being used for many other types of document, particularly those with multi-lingual requirements.

The TeX system is fully programmable. This allows the development of high-level user interfaces whose input is processed by TeX's interpreter to produce low-level typesetting instructions; these are input to TeX's typesetting engine which outputs the format of each page in a device-independent page-description language.

Many people have made use of this powerful feature of TeX and developed their own front-ends; the most commonly used such packages are $\mathcal{AMS}$-TeX and L$\mathcal{AMS}$-TeX by Michael Spivak [30, 29] and LaTeX by Leslie Lamport [8]. The development of $\mathcal{AMS}$-TeX was sponsored by the American Mathematical Society, a publishing house which now processes and typesets the majority of its output using TeX.

The principal aim of $\mathcal{AMS}$-TeX was to simplify the user interface to the sophisticated built-in math-typesetting capabilities of TeX. It therefore does not provide support for certain more general document processing requirements (such as symbolic cross-references, automatic numbering, etc.); it is therefore most appropriate for short articles which contain lots of formulas.

The LaTeX system, on the other hand, supports the needs of long documents such as textbooks and manuals. It was designed to separate content and form as much as possible by providing the user with a generic (i.e. logical rather than visual) markup interface; this is combined with style files in which the formatting is specified. Nevertheless, LaTeX also provides a complete set of direct formatting instructions; these allow the user to access the full power of TeX's typesetting expertise when preparing the final version of the document.

Recent years have shown that the concepts and approach of LaTeX have become widely accepted. Indeed, LaTeX has become the standard method of communicating and publishing documents in many academic disciplines. This has led to many publishers accepting LaTeX source for articles and books. The American Mathematical Society, for example, now provides a LaTeX style option [1] which makes the math-typesetting features of $\mathcal{AMS}$-TeX available to users of LaTeX. But the use of LaTeX in the publishing industry goes further: Elsevier Science Publishers, for example, are developing a system [24] which links SGML (for electronic storage of documents in databases) and LaTeX (for formatting these documents).

The use of LaTeX, together with SGML, is now also spreading into industrial environments, where the technical qualities of TeX together with the concepts of

*Eichenweg 29, W-6500 Mainz 1, Federal Republic of Germany, mittelbach@mzdmza.zdv.uni-mainz.de
†Open University, Walton Hall, Milton Keynes MK7 6AA, United Kingdom, c.a.rowley@open.ac.uk

LaTeX are considered a powerful combination of great potential importance to such areas as corporate documentation [33] and database publishing.

With the spreading use of SGML-compliant systems, such as the Grif editor, LaTeX again is a common choice as the formatter for high quality typeset output [9]. A typical SGML Document Type Definition (DTD) uses concepts similar to those of LaTeX. Therefore, as in the Euromath system, the formatting is often implemented by simply mapping document elements to LaTeX constructs rather than directly to 'raw TeX' [31]. This enables the sophisticated analytical and processing techniques built in to much of the LaTeX software to be exploited; and it avoids the need to program in TeX.

Such developments ensured that the uses of LaTeX have become widespread, both geographically and typographically. However, they have also pushed the system way beyond its original intended purpose. Moreover, as most people who have been involved in the production of style files will agree, they have demonstrated that the current version, whilst making the author's job easier, provides little assistance to developers of new applications. Thus there can be no doubts about the need for the continued development of LaTeX into a new, improved, front-end to TeX—one that will serve the typesetting needs of the nineties and beyond—so that is what we intend the LaTeX3 project to provide.

2 History

The original goals for the development of a new version of LaTeX are described in a paper [21] presented in 1989 at the 10th annual meeting of the TeX Users Group at Stanford—it was also at this time that Leslie Lamport expressed his full support for such a project. However, we have since discovered that those original goals did not touch many of the deficiencies of the current system.

New applications of LaTeX have highlighted many limitations of its interface (both for authors of documents and for designers of styles); and further research on such problems led us to the conclusion that one gains very little by just providing more and more specialized style files to solve this or that special problem. This is because many of these deficiencies and limitations have their source in LaTeX's internal concepts and design [21, 22, 23].

The most important of those original goals, and one that is still a core part of the LaTeX3 system and a central concern of the project team, is the provision of a good style design interface—one that allows easy implementation of various layouts. (Easy, of course, is relative: we mean 'as easy as possible, given the complexity of the task'.) In order to make LaTeX3 a fully flexible and extensible system, a major effort is needed in the near future in order to get this interface 'right'.

3 Aims

The principle aims guiding our work on the project's development are as follows.

- The LaTeX3 system will provide high quality typesetting for a wide variety of document types and typographic requirements.
- For authors, it will be easy to use since it will be highly automated, but controllable.
- For editors and designers, it will support the direct formatting commands which are essential to the fine-tuning of document layout.
- It will process complex structured documents and support a document syntax that allows automatic translation of documents conforming to commonly used SGML document type definitions into LaTeX documents—this syntax will therefore, for example, support the SGML concepts of 'attribute' (or 'named argument') and 'short reference', in such a way that these can be easily linked to the corresponding SGML features.
- LaTeX3 will be designed as an open system and, like the present version, it will be usable with any standard TeX system and will thus be available on a very wide range of platforms.
- Its highly modular design will provide a system that is flexible and extensible, with well-defined and fully documented interfaces.
- The code itself will also be thoroughly documented and the modular design will help to make the system easy to maintain and enhance.
- We shall also provide extensive catalogues containing many examples that are carefully designed to make the learning time for new users (including designers, editors and programmers) as short as possible.

4 Available now!

In some important areas, the extra facilities of LaTeX3 are already available to current LaTeX users.

- The New Font Selection Scheme (NFSS) is now in widespread use, providing very general and powerful tools for setting up and accessing all available fonts. A new version (NFSS2) has recently been released, which extends the flexibility of the system in the following areas of font management.
  - Scalable fonts, e.g., support for PostScript fonts.
  - Encodings, i.e., support for multilingual documents allowing change of font encoding (code page) within a document.
  - Math symbols, facilitating access to a wide range of symbols.
  - Math typesetting using any suitable family of fonts, eg Lucida or Adobe Times.
- With $\mathcal{AMS}$-LaTeX [1] the capabilities of LaTeX for mathematical typesetting have reached at least the standard of $\mathcal{AMS}$-TeX.

- Work by Frank Mittelbach and David Carlisle, together with valuable suggestions by several others, has made available more sophisticated tabular processing [15].
- Various forms of multiple-column formatting are also now available [14].
- Johannes Braams [2] has produced a new version of the 'Babel system' to support a wide range of languages.

5 New features

LaTeX3 will provide much new and enhanced functionality in addition to the above facilities; and its 'LaTeX3 programming language' will enable it to be further extended in a controlled way.

Here are further details of some of the many planned improvements.

- A robust author interface: providing interactive error recovery linked to an on-line help system.
- Languages: the typesetting will be customisable for use with different languages (as will the whole system, e.g. the error/help components). In particular, there will be support for mixed language documents and for multiple languages within one paragraph to be correctly hyphenated.
- Table formatting: a large number of extensions in this area will be available, including:
  - multi-page tables;
  - automated column width calculation;
  - a variety of designs for ruled tables;
- Float handling: the major problem in processing floats (e.g. tables and figures) is the precise specification of the designer's rules concerning how to position them. The new system will make it possible to implement a large range of such positioning algorithms, including the requirements of multi-column formatting, whilst also supporting explicit positioning commands.
- Many of the non-typesetting aspects of document processing will be automated, providing a flexible interface to cover a wide range of requirements and styles in the following areas:
  - citations and bibliographies;
  - cross-references;
  - indexing, etc.;
  - tables of contents, etc.;
  - multiple marks.
- The enhanced functionality in the area of mathematical typesetting will include:
  - alignments in displayed formulas;
  - alphabets, symbols, embellishments;
  - commutative (arrow) diagrams.

  These will extend many of the features which are already available in the current version of $\mathcal{AMS}$-LaTeX [1].

  The system will also support the elements defined in the 'standard SGML DTD for mathematical expressions' which is currently under development [32].
- The typesetting requirements of many other areas will also be addressed—some examples:
  - technical documentation (e.g. offset layout, change bars);
  - academic publishing in the humanities (e.g. critical text editions);
  - structural formulas in chemistry;
  - integration of graphical structures, including shading and colour;

6 Interfaces

As we said earlier, the design specification interface is probably the most significant single new feature of LaTeX3. It will have two clearly separated parts:

- the creation of the generic mark-up (e.g. environments) used by the author in creating the LaTeX form of the document;
- the specification of how (in SGML language) the document elements will be formatted.

This will simplify the production of different layouts for the same document type. It will also enable LaTeX documents to be created and modified by structured document editors, such as Arbortext Publisher and Grif. Indeed, these two parts are direct analogues of the following components of the Grif system: the 'generic structure definition (or S) language' and the 'presentation description (or P) language' [4].

However, in order to support the requirements of high quality typesetting, the LaTeX3 equivalent of the P language will need to be even richer than the Grif language. Despite the complexity of the task, this part of the 'designer interface' will help to make the following straightforward:

- specification of a wide variety of typographic design rules;
- linking of the elements in a document type to the desired formatting;
- specifying and modifying all of the parameters that influence the layout.

Another important interface will be the LaTeX3 programming language, used for producing enhancements and extensions: it will be an entirely new language based on data structures and operations suited to the kind of programming required by document processing applications and to the expression of visual components of the layout process. Built on this language there will be high-level generic functions that allow the straightforward expression of common layout components.

7 Resources

The majority of the work, including conceptualisation, modelling, prototyping, implementation and testing, is being undertaken by a dozen individuals under the technical direction of Frank Mittelbach. This work is, in

most cases including that of the technical director, done entirely in their spare time and so involves, as you can imagine, a lot of enthusiasm to keep the project alive. A large number of other individuals and organisations have contributed in one way or another to the effort, and we are confident that this number will continue to rise.

There are, nevertheless, many other tasks still to be done in support of the LaTeX3 project. These can be worked on concurrently with the development of the LaTeX3 kernel system. Furthermore, some of these tasks require special expertise not found among the core programming team. Initial research, analysis, and other work on these tasks by volunteers will, we are sure, greatly speed up the process of integrating a number of desirable features into LaTeX3.

For this reason a 'volunteer task list' has been set up: this describes briefly the individual tasks, which require a wide variety of expertise and time involvement. It will be updated at regular intervals and a copy is available, via anonymous `ftp` from the following sites: `niord.shsu.edu`, directory `[fileserv.vol-task]`; `ftp.uni-stuttgart.de`, directory `soft/tex/vol-task`. For access via mail server, send mail to `fileserv@shsu.bitnet`, with no subject line and in the body write: `sendme vol-task`, or to `mail-server@rus.uni-stuttgart.de`, with no subject line and in the body write: `send soft/tex/vol-task/vol-task.tex`. If you are unable to retrieve a copy via electronic networks, please contact Chris Rowley.

To provide a means of communication with a large number of LaTeX users an electronic mailing list has been installed at Heidelberg. To subscribe to this list send a mail message to `listserv@vm.urz.uni-heidelberg.de`, with one line as the body of the message (substituting your own names):

```
subscribe LaTeX-L
       Your_first_name Your-surname
```

One of the major, and growing, problems is how to bring people from all over the world together to discuss the open questions and find new solutions. It is important that these meetings involve people from outside the project since we very much need the views and experience of typesetters, designers, publishers, etc. to help eliminate the flaws in the system and to find new and better solutions.

We have so far held two 'open workshops', in London, UK and Boston, USA; these were hugely successful and showed that further workshops of this kind are essential if we are to provide LaTeX3 with a good designer interface.

It is now clear that our ability to maintain the current progress will depend on adequate financial support being available for the following purposes:

- enhancement of computer equipment and software for the core development team;
- purchase of books on typography and other related subjects;
- essential expenses (travel, accommodation, etc.) for meetings of the project's core development team; and also for meetings with others, for example: testers of early versions; publishers; typographic designers; suppliers of related software, etc.

References and Bibliography

This bibliography contains some items which are not specifically referenced in this article: these contain further information about the LaTeX3 project. It also contains entries concerning BibTeX, which will be reimplemented and enhanced by Oren Patashnik for use with LaTeX3.

[1] American Mathematical Society, Providence, Rode Island. *AMS-LaTeX Version 1.1 User's Guide*, December 1990.

[2] Johannes Braams An update on the `babel` system *TUGboat*, 14, to appear 1993.

[3] Malcolm Clark. What is the LaTeX3 project? *TeX and TUG NEWS*, 1(1):4, February 1992.

[4] Grif S.A, St Quentin en Yvelines. *Grif Languages: Grif 2.2*, 1992.

[5] Mary Guenther, editor. *TeX 90 Conference Proceedings*, March 1991. Published as TUGboat 12#1.

[6] Alan Hoenig. LaTeX3, TUG and You. *TeX and TUG NEWS*, 2(1):2–4, July 1992.

[7] Donald E. Knuth. Computers & Typesetting. Addison-Wesley, Reading, Massachusetts, 1986.

[8] Leslie Lamport. *LaTeX: A Document Preparation System*. Addison-Wesley, Reading, Massachusetts, 1986.

[9] Helmut Lenzing. The euromath project. *Euromath Bulletin*, 1(1):13–20, 1992.

[10] Frank Mittelbach. LaTeX 2.10. In Lincoln K. Durst, editor, *1990 Conference Proceedings*, page 444, September 1990. Published as TUGboat 11#3.

[11] Frank Mittelbach. LaTeX2.09 → LaTeX3. *TeXline*, (14):15–18, February 1992.

[12] Frank Mittelbach. LaTeX3. *Die TeXnische Komödie*, 4(2):15–22, August 1992.

[13] Frank Mittelbach. LaTeX3 project. *TeX Gebruikers Group*, 92(1):87–90, May 1992.

[14] Frank Mittelbach. The `multicol` package. Distribution of style options that allow multi-column layout with LaTeX, May 1992.

[15] Frank Mittelbach and David Carlisle. The `array` package. Distribution of style options that extend LaTeX's array and tabular facilities, May 1992.

[16] Frank Mittelbach and Chris Rowley. The LaTeX3 project fund. *Die TeXnische Komödie*, 3(4):13–15, December 1991.

[17] Frank Mittelbach and Chris Rowley. LaTeX209 → LaTeX3. *TUGboat*, 13(1):96–101, April 1992.

[18] Frank Mittelbach and Chris Rowley. The LaTeX3 project fund. *TeX and TUG NEWS*, 1(1):5–6, February 1992.

[19] Frank Mittelbach and Chris A. Rowley. The pursuit of quality—how can automated typesetting achieve the highest standards of craft typography? In C. Vanoirbeek and G. Coray, editors, *Electronic Publishing '92*, pages 260–273, Cambridge, April 1992. Cambridge University Press.

[20] Frank Mittelbach, Chris Rowley and Michael Downes. Volunteer work for the LaTeX3 project. LaTeX3 project paper, September 1992.

[21] Frank Mittelbach and Rainer Schöpf. With LaTeX into the nineties. In Christina Thiele, editor, *1989 Conference Proceedings*, volume 10#4 of TUGboat, pages 681–690. TeX Users Group, December 1989.

[22] Frank Mittelbach and Rainer Schöpf. LaTeX dans les années 90. *Cahiers GUTenberg*, (6):2–14, July 1990.

[23] Frank Mittelbach and Rainer Schöpf. Towards LaTeX 3.0. In Guenther [5], pages 74–79. Published as TUGboat 12#1.

[24] N.A.F.M. Poppelier. SGML and TeX in scientific publishing. In Guenther [5], pages 105–109. Published as TUGboat 12#1.

[25] David Rhead. Could LaTeX do more for chemists? *TeXline*, (12):2–4, December 1990. Suggestions for LaTeX3.

[26] David Rhead. Towards BibTeX style-files that implement principal standards. *TeXline*, (10):2–8, May 1990.

[27] David Rhead. How might LaTeX3 deal with citations and reference lists? *TeXline*, (13):13–20, September 1991. Suggestions for LaTeX3.

[28] Chris Rowley. LaTeX209 → LaTeX3: an update. In Mimi Burbank, editor, *1992 Annual Meeting Proceedings*, pages 390–391, October 1992. Published as TUGboat 13#3.

[29] Michael D. Spivak. *$\mathcal{AMS}$-TeX The Synthesis*. The TeXplorators Corporation, Houston, 1989.

[30] M. D. Spivak, Ph.D. *The Joy of TeX, A Gourmet Guide to Typesetting with the $\mathcal{AMS}$-TeX macro package*. American Mathematical Society, Providence, Rode Island, second edition, 1990.

[31] Eric van Herwijnen. *Practical SGML*. Wolters Kluwer Academic Publishers, Dordrecht, 1990.

[32] Björn von Sydow. The design of the euromath system. *Euromath Bulletin*, 1(1):39–48, 1992.

[33] Reinhard Wonneberger. Structured document processing: the LaTeX approach. *Computer Physics Communications*, (61):177–189, 1990.

[34] Reinhard Wonneberger and Frank Mittelbach. BibTeX reconsidered. In Guenther [5], pages 111–124. Published as TUGboat 12#1.

PostScript en LaTeX, de komplimentariteit in praktijk

Michel Goossens

CERN, CN Division
CH121 Geneva 23, Switzerland
goossens@cernvm.cern.ch

Abstract

In dit artikel toon ik aan hoe PostScript en LaTeX een hoge graad van samenhorigheid bezitten, die het mogelijk maakt om de voordelen van beide systemen te kombineren om dokumenten elektronisch te publiceren. Allereerst vertel ik hoe, samen met de dvi-vertaler `dvips` en het stijlbestand `epsfig`, het invoegen van PostScript materiaal in een (LA)TeX bestand heel eenvoudig wordt. Samen met de stijl `rotating` kan men bijna elk gewenst globaal grafisch effekt verkrijgen zonder per-se een PostScript guru te zijn. In het tweede gedeelte van het artikel geef ik een kort overzicht van enkele op PostScript gebaseerde stijlbestanden, die bepaalde nuttige visuele effekten genereren, zoals grijze raampjes, kleurentypografie en het overdrukken van tekst. In het laatste gedeelte toon ik hoe eenvoudig het is om PostScript fonts te gebruiken in LaTeX met het nieuwe fontselektie systeem (NFSS) van Frank Mittelbach.

Een woord vooaf

Dit artikel is een verkorte uit het Engels vertaalde versie van een hoofdstuk uit *het boek 'A LaTeX Companion'*, dat ik samen met Frank Mittelbach (de leider van het LaTeX3 project) en Alexander Samarin (ISO, Internationale Standaards Organisatie, Genève), heb geschreven, en dat in september door Addison-Wesley wordt uitgegeven. *Dit boek zal in ongeveer 400 bladzijden een overzicht geven van al wat een gevorderd LaTeX gebruiker nodig heeft om haar/zijn LaTeX dokument in de gewenste struktuur en vorm te gieten.* Kort samengevat ziet de inhoudstabel er als volgt uit: na een kort overzicht van wat LaTeX (wel en niet) is behandelen we de strukturele basisbegrippen (hoofdstukken en paragrafen, lijsten, verbatim omgevingen, verschillende vormen van noten, meerdere kolommen, de kop- en voetteksten, drijvend materiaal). Een volledig hoofdstuk is gewijd aan tabellen en aan het nieuwe fontselektie systeem (NFSS) waarvan weldra de tweede, erg uitgebreide versie, uitkomt. Vervolgens bespreken we de mogelijkheden op het gebied van wiskundig tijpzetten met $\mathcal{AMS}$LaTeX en behandelen het probleem van grafisch materiaal, eerst op een drukker-onafhankelijke manier (`picture` omgeving en de `epic` en `eepic` stijlen) en dan met PostScript (zoals in dit artikel). De volgende hoofdstukken bevatten een gedetailleerde bespreking van hoe men een zakenregister (index) genereert met *MakeIndex* en hoe men literatuurlijsten en bibliografische gegevensbanken creëert en beheert met BibTeX. Het laatste hoofstuk vertelt hoe men zijn LaTeX stijlen kan dokumenteren en verspreiden met de stijl `doc` en docstrip. De eerste appendix beschrijft waar men terecht kan op het internet of via elektronische post om LaTeX stijlbestanden en informatie te verkrijgen, terwijl de tweede appendix een gedetailleerd overzicht geeft van LaTeX's gevorderde begrippen (het definiëren van opdrachten en omgevingen, het gebruik van tellers en lengtes, de konstruktie en het nut van raampjes en tenslotte hoe men met LaTeX arithmetiek bedrijft).

De helft van de opbrengst van dit boek gaat direkt naar de financiering van het LaTeX3 projekt, zodat U, door het kopen van dit boek, zich niet alleen een handig referentiewerk aanschaft, maar ook ervoor zorgt dat LaTeX in de toekomst nóg beter wordt.

1 Het samenbrengen van tekst en PostScript grafisch materiaal

In dit artikel beschrijf ik enkele LaTeX stijlen die, samen met Tom Rokicki's `dvips` [1] (en waarschijnlijk ook andere dvi-vertaalprogramma's) het mogelijk maken PostScript's uitgebreid opdrachtenpalet [2, 3, 4, 5, 6, 7] te gebruiken om bepaalde grafische effekten te verkrijgen. Het enige waar U voor moet opletten indien U gebruik wilt maken van deze functies, is dat hun effekt slechts zichtbaar wordt indien U ook beschikt over een PostScript drukker of visualizator, een programma dat PostScript kan omzetten in een beeld op het scherm van een werkstation of een persoonlijke computer (PC of Mac). Een voorbeeld van zo'n programma in het publieke domein is ghostview (dat deel uitmaakt van het GNU software projekt).

Niettegenstaande het feit dat het mogelijk is om via de TeX opdracht `\special` willekeurige teksten door te

spelen aan de dvi-vertaalprogramma's (in ons geval elementaire PostScript opdrachten), is het in het algemeen aan te raden meer generische konstrukties te gebruiken. De eerste stijl, die ik bespreek, heet `epsfig` en maakt het U eenvoudig om een PostScript beeld in te voeren in een LaTeX bestand. Zoals de naam het aanduidt gaat het hier over een 'Encapsulated PostScript' (EPS) [8] bestand d.w.z. een bestand dat, o.a. informatie bevat over zijn dimensies, meer in het bijzonder een regel bevat met de struktuur:

```
%%BoundingBox: llx lly urx ury
```

met `llx` en `lly` de x- en y-dimensies (in PostScript punten, waar 1 PostScript punt $= 0,35278$mm) van de linker beneden hoek en `urx` en `ury` de x- en y-dimensies van de rechter bovenhoek van een (virtuele) rechthoek, die het gehele beeld omsluit. Aan de hand van deze dimensies kan de opdracht `epsfig` de plaats berekenen, die opengelaten moet worden om het beeld te kunnen afdrukken. Kleine beeldjes, zoals '  ' en '⌀' kunnen om het even overal in het document worden gebruikt.
De volledige syntax van de opdracht is:

```
\epsfig{file=fn,%
        height=ht,width=wd,%
        clip=,rotate=angle,silent,%
        bbllx=blx,bblly=bly,%
        bburx=brx,bbury=bry}
```

`file` De naam van het Encapsulated PostScript bestand (je kan ook `figure=` gebruiken).

`height` De gewenste hoogte van het beeld op het gedrukt blad (in één van de TeX eenheden). Indien deze parameter en de volgende (`width`) niet gespecificeerd zijn dan zal het beeld zijn 'natuurlijke' grootte hebben, d.w.z. de dimensies gespecificeerd op de `BoundingBox` lijn in het Encapsulated PostScript bestand zelf. Waneer men de breedte (`width=`) specificeert en geen hoogte (`height=`), dan wordt deze laatste automatisch aangepast zodanig dat de verhouding tussen beide dimensies onveranderd blijft.

`width` De gewenste breedte van het beeld op het gedrukt blad (in één van de TeX eenheden). Indien deze parameter en de vorige (`height`) niet gespecificeerd zijn dan zal het beeld zijn 'natuurlijke' grootte hebben, d.w.z. de dimensies gespecificeerd op de `BoundingBox` lijn in het Encapsulated PostScript bestand zelf. Waneer men de hoogte (`height=`) specificeert en geen breedte (`width=`), dan wordt deze laatste automatisch aangepast zodanig dat de verhouding tussen beide dimensies onveranderd blijft.

`clip` Wanneer deze optie wordt gekozen, dan worden de delen van het beeld die buiten de 'BoundingBox' vallen, weggeknipt. Opgelet dat `"clip="` geen waarde neemt, maar dat het '='-teken aanwezig moet zijn.

`rotate` De draaihoek (in graden, de positieve richting gaat tegen de richting van de wijzers van een uurwerk in) voor het gehele beeld.

`silent` Met deze optie werkt `\epsfig` 'stilletjes', d.w.z. zonder informatie op het beeldscherm (of in de log) te schrijven.

`bbllx` De **x**-coördinaat van de linker benedenhoek van de omschreven rechthoek (BoundingBox).

`bblly` De **y**-coördinaat van de linker benedenhoek van de omschreven rechthoek (BoundingBox).

`bburx` De **x**-coördinaat van de rechter bovenhoek van de omschreven rechthoek (BoundingBox).

`bbury` De **y**-coördinaat van de rechter bovenhoek van de omschreven rechthoek (BoundingBox).

Wanneer de BoundingBox parameters **vóór** de naam van de figuur staan (`figure=` or `file=`), dan negeert `\epsfig` de parameters in het Encapsulated PostScript bestand (`\epsfig` zal dan zelfs het bewuste bestand niet trachten te lezen), en gebruikt de dimensies gespecifieerd in de opdracht zelf. Dat kan interessant zijn om een bepaald gedeelte van het beeld weg te knippen (clipping), of om PostScript bestanden te gebruiken die de BoundingBox informatie niet bevatten of waar deze laatste verkeerd is. U moet wel oppassen dat u geen spaties laat in de argumenten van `\epsfig`, want dan werkt die opdracht niet; als U verplicht bent een `\epsfig` opdracht over verschillende regels uit te schrijven, dan moet U elke lijn afsluiten met een %-teken (zoals ik ook deed in de beschrijving van de opdracht hierboven).

1.1 Simpele figuren

De eenvoudigste situatie is wanneer U tevreden bent met de 'natuurlijke' dimensies van het beeld (zoals gespecificeerd in het bestand zelf op de `BoundingBox` lijn). Dan heeft U geen schaalfaktor nodig en `\epsfig` berekent de plaats die het plaatje inneemt eenvoudig uitgaande van deze dimensies. Een eenvoudige manier om de natuurlijke dimensies van een beeld te kennen is het te drukken op een PostScript printer. De linker benedenhoek van het beeld wordt gepositioneerd daar waar de `\epsfig` opdracht wordt gegeven. Indien de opgegeven breedte (`width`) of hoogte (`height`) niet overeenstemmen met de respektieve natuurlijke dimensies, dan word het beeld met de nodige schaalfaktor vergroot of verkleind.

Figuur 1 op pagina 104 is een prent waarvoor we een hoogte van 5cm wensen. De `\epsfig` opdracht werd in een `\mbox` gestopt om de figuur te centreren met behulp van een `center` omgeving.

1.2 Proeffiguren

Sommige PostScript figuren hebben een lange tijd nodig om gedrukt te worden. Voor zulke figuren is er een 'proef' (draft) optie, die niet het PostScript beeld zelf afdrukt, maar ze vervangt door een rechthoek met dezelfde dimensies als het uiteindelijke beeld in de tekst zelf, tezamen met de naam van de figuur (zie figuur 2 op pagina 104). De opdracht `\psdraft` schakelt om naar proefmode, zodat alle volgende `epsfig` opdrachten 'draft' rechthoekjes, in plaats van de oorspronkelijke figuren, zullen drukken. De opdracht `\psfull`, keert terug naar de normale mode. Een ander voordeel van de draft-mode is dat er geen TeX `\special` opdrachten in het dvi-bestand worden geschreven, zodat het test dokument bekeken kan worden met een willekeurig visualizatie-programma, dat niet noodzakelijk PostScript behoeft te verstaan.

1.3 Schikking van beeldmateriaal

We kunnen nu de mogelijkheden van de `\epsfig` opdracht en van LaTeX kombineren om ons beeldmateriaal optimaal op het blad te schikken. De figuren 3 tot 5 op pagina 119 tonen schikkingen, verkregen door gebruik te maken van `minipage` omgevingen.

2 Het roteren van tekst- en beeldmateriaal

Het komt vaak voor dat we een figuur of tabel willen draaien, enerzijds om de presentatie van de informatie te verbeteren (bijvoorbeeld een tabel wijder dan de kolombreedte), anderzijds om bepaalde visuele effekten te bereiken. Een handige manier om deze draaifunktionaliteit te verkrijgen is met de stijl `rotating`, ontwikkeld door Sebastian Rahtz en Leonor Barroca [9], die enkele nieuwe LaTeX omgevingen definiëert die ik hierna één voor één onder de loep wil nemen.

2.1 Het roteren van tekst

De `rotate` omgeving draait het omsloten materiaal over het opgegeven aantal graden (positief is met de wijzers mee, zoals in PostScript). Deze omgeving neemt echter aan dat het materiaal geen plaats inneemt, zodat U deze omgeving kunt gebruiken om speciale effekten te verkrijgen. Plaats het TeX kommentaar teken % aan het einde van de opdracht `\begin{rotate}{52}%` — U moet de opdracht altijd volgen met tekst of met %, om te voorkomen that ongewenste spaties in de tekst opduiken.

```
Begin van de zin \begin{rotate}{-52}%
   NTG
\end{rotate} Einde van de zin
```

Begin van de zin NTG Einde van de zin

Indien U wenst dat LaTeX de nodige ruimte openlaat om het materiaal te plaatsen, dan gebruikt U de omgeving `turn`:

```
Begin van de zin \begin{turn}{52}%
   NTG
\end{turn} Einde van de zin
```

Begin van de zin NTG Einde van de zin

De omgeving '`sideways`' is een speciaal geval, van de `turn` omgeving, met een draaihoek van -90:

```
Begin van de zin \begin{sideways}%
   NTG
\end{sideways} Einde van de zin
```

Begin van de zin NTG Einde van de zin

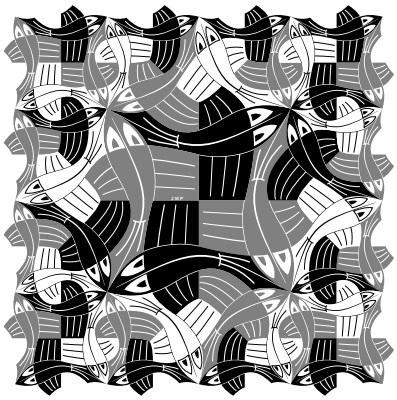
Wanneer men met hele paragrafen tekst werkt, dan vindt men dat TeX ramen ('boxes') in het algemeen zowel een hoogte als een diepte hebben. In dit geval vinden de rotaties plaats rond het uiterste linker punt waar het raam de basislijn raakt. Dit kan soms tot eigenaardige en onverwachte resultaten leiden, zoals figuur 6 op pagina 120 toont. Voor elk beeldje tonen de lijntjes links en rechts de positie van de basislijn. U kunt de positionering van de ramen min of meer kontroleren door het specificeren van de (optionele) positie parameter die beschikbaar zijn bij bepaalde LaTeX opdrachten.

```
\newcommand{\B}{A b C d E f G h I j K
                l M n O p Q}
Begin   \begin{turn}{45}
          \parbox[t]{16mm}{\B}
        \end{turn}
Midden  \begin{turn}{45}
          \parbox[b]{16mm}{\B}
        \end{turn}
Einde
```

Begin Midden Einde

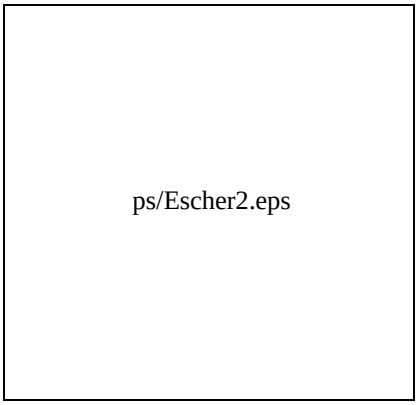
2.2 Het roteren van tabellen

Materiaal in tabellen (LaTeX's `tabular` en `array` omgevingen) kunnen op een analoge manier gedraaid worden. De voorbeelden in figuur 7 op pagina 120 tonen hoe 'onzichtbare regels' gebruikt worden om de hoogte of de breedte van de kolommen te kontroleren.



```
\begin{center}
 \mbox{\epsfig{file=ps/Escher2.eps,height=50mm}}
\end{center}
\caption[]{Een eenvoudig gecentreerde figuur}
\label{fig:simple}
```

Figuur 1: *Een eenvoudig gecentreerde figuur*




```
% ga naar test (draft) mode
 \psdraft

 \begin{center}
  \mbox{\epsfig{file=ps/Escher2.eps,height=50mm}}
 \end{center}
 \caption[]{Een figuur in test (draft) mode}
 \label{fig:draft}
% ga terug naar de normale mode
 \psfull
```

Figuur 2: *Een figuur in test (draft) mode*

Rotaties kunnen gekombineerd worden, als volgt:

```
\begin{sideways}
\begin{tabular}{l@{\qquad}r}
\em Zin \rule{0pt}{3cm}
           & \begin{rotate}{-90}%
Frequentie\end{rotate}  \\[1mm]
 \hline
 Goeie dag & 33\\
 Tot ziens & 34\\
 \hline
\end{tabular}
\end{sideways}
```

Zin	Frequentie
Goeie dag	33
Tot ziens	34

Een ingewikkelder (en wellicht meer voorkomend) voorbeeld is tabel 1 op pagina 105, die een `sideways` omgeving in een `sidewaystable` omgeving toont. Deze laatste omgeving draait niet enkel de tabel zelf, maar ook zijn onderschrift. De `sidewaystable` omgeving — en zijn analoog voor figuren, de `sidewaysfigure` omgeving — nemen echter een volledig pagina in, en men moet dus wel een beetje uitkijken om zeker te zijn dat men de pagina wel degelijk op een aanvaardbare manier kan vullen.

	gebruikersprijzen		produktieprijzen		werkloosheid %	
	3 maanden	1 jaar	3 maanden	1 jaar	laatste	1 jaar
Australië	+0,4	+0,8	+4,6	+2,1	11,4	10,5
België	+1,9	+2,2	−3,0	−0,7	8,4	7,7
Canada	+1,0	+1,7	+3,3	+2,9	11,8	10,3
Duitsland	+3,2	+3,7	−0,8	+0,5	7,1	6,3
Frankrijk	+1,4	+2,1	−1,9	−0,9	10,4	9,9
Groot Brittanië	+1,9	+3,0	+1,4	+3,3	10,3	8,9
Italië	+4,2	+4,8	−2,1	+1,0	10,6	10,3
Japan	+0,4	+1,1	−2,6	−1,3	2,2	2,0
Nederland	+5,1	+2,9	−2,1	+0,1	4,5	4,4
Spanje	+6,6	+5,1	+0,2	+1,2	15,3	15,4
Verenigde Staten	+3,4	+3,0	+1,4	+1,2	7,2	6,9
Zweden	+5,7	+1,3	−3,7	−1,3	5,4	3,2
Zwitserland	+2,4	+3,3	−0,6	−0,1	3,9	1,7

```
renewcommand{\arraystretch}{1.2}
\begin{sideways}
\begin{tabular}{@{}>{\bf}l@{\qquad}%
   >{$}c<{$}@{\quad}>{$}r<{$}@{\qquad}%
   >{$}c<{$}@{\quad}>{$}r<{$}@{\qquad}%
   >{$}r<{$}@{\quad}>{$}r<{$}}
 & \multicolumn{2}{c}{\textbf{gebruikerspr"yzen}}
 & \multicolumn{2}{c}{\textbf{produktiepr"yzen}}
 & \multicolumn{2}{c}{\textbf{werkloosheid \%}}\\
 & \multicolumn{1}{c}{\textbf{3 maanden}}
 & \multicolumn{1}{c}{\textbf{1 jaar}}
 & \multicolumn{1}{c}{\textbf{3 maanden}}
 & \multicolumn{1}{c}{\textbf{1 jaar}}
 & \multicolumn{1}{c}{\textbf{laatste}}
 & \multicolumn{1}{c}{\textbf{1 jaar}}\\\hline
Australi"e &+0,4&+0,8&+4,6&+2,1&11,4&10,5 \\\hline
Belgi"e    &+1.9&+2,2&-3,0&-0,7&8,4& 7,7 \\\hline
Canada     &+1,0&+1,7&+3,3&+2,9&11,8&10,3 \\\hline
Duitsland  &+3,2&+3,7&-0,8&+0,5&7,1& 6,3 \\\hline
Frankr"yk  &+1,4&+2,1&-1,9&-0,9&10,4& 9,9 \\\hline
Groot Brittani"e
           &+1,9&+3,0&+1,4&+3,3&10,3& 8,9 \\\hline
 Itali"e   &+4,2&+4,8&-2,1&+1,0&10,6&10,3 \\\hline
Japan      &+0,4&+1,1&-2,6&-1,3&2,2&2,0 \\\hline
Nederland  &+5,1&+2,9&-2,1&+0,1&4,5&4,4 \\\hline
Spanje     &+6,6&+5,1&+0,2&+1,2&15,3&15,4 \\\hline
Verenigde Staten
           &+3,4&+3,0&+1,4&+1,2&7,2&6,9 \\\hline
 Zweden    &+5,7&+1,3&-3,7&-1,3&5,4&3,2 \\\hline
Zwitserland&+2,4&+3,3&-0,6&-0,1&3,9&1,7 \\\hline
\end{tabular}
\end{sideways}
```

Tabel 1: *Hoe men een tabel kan draaien*

2.3 Het roteren van figuren

Met de `rotating` stijl kunt U niet enkel tekst maar ook PostScript beeldmateriaal draaien. Een voorbeeld ziet U in figuur 8 op pagina 121, waar een Encapsulated PostScript beeld in het dokument wordt ingevoerd met de `\epsfig` opdracht, en daarna wordt gedraaid naar keuze. Het is interessant om voor elk geval naar de positie van de basislijn te kijken.

3 Het kleuren van ramen

Om de nadruk te leggen op een bepaald gedeelte van de tekst is het dikwijls nuttig het bedoelde materiaal te omlijnen of het met een grijze achtergrond te kleuren.

De stijl `psboxit` (geschreven door J. Maillot) bouwt een PostScript raam boven op een TEX raam, gebruik makend van de parameters berekend voor deze laatste. Om de mogelijkheden van deze stijl te benutten moeten de PostScript opdrachten geïnitialiseerd worden, door de opdracht `\PScommands` te geven aan het begin van het dokument.

De basisopdracht is `\psboxit`, gedefiniëerd as volgt:

`\psboxit{`*PS opdrachten*`}{`*TEX materiaal*`}`

Deze opdracht onderwerpt het materiaal gespecificeerd in het tweede argument '*TEX materiaal*' aan het effekt van de PostScript opdrachten in het eerste argument '*PS opdrachten*'. Het stijlbestand definiëert verscheidene PostScript proceduren, zoals `cartouche` en `rectcartouche`. Deze laten toe zekere visuele effekten te verkrijgen, zoals getoond in het voorbeeld hieronder:

```
--\psboxit{5 cartouche}{NTG}--
--\psboxit{rectcartouche}{\spbox{DANTE}}--
--\psboxit{box .7 setgray fill}%
          {\spbox{TUG}}--
```

–NTG– – DANTE – – TUG –

De bijkomende opdracht `\spbox` werkt zoals `\fbox`, d.w.z. men verkrijgt een raam rond het ingeschreven materiaal met een afstand `\fboxsep` toegevoegd in elke richting. Het enige verschil is dat `\spbox` geen lijnen afbeeldt rond het raam. Het belangrijkste doel van `\fboxsep` is raam te bouwen met identieke afmetingen als `\fbox`, maar met een gekleurde achtergrond. In het algemeen kunt U natuurlijk zelf opdrachten definiëren, zoals:

```
\newcommand{\grijsraam}[1]%
{\psboxit{box .7 setgray fill}{\fbox{#1}}}
```

Indien U grotere ramen wilt kleuren, dan gebruikt U het best de `boxitpara` omgeving:

```
\begin{boxitpara}{box 0.7 setgray fill}
Amsterdam is een grote stad.
Antwerpen heeft een mooie kathedraal.
Groningen ligt in het noorden.
Leuven ligt in het zuiden.
\end{boxitpara}
```

Amsterdam is een grote stad. Antwerpen heeft een mooie kathedraal. Groningen ligt in het noorden. Leuven ligt in het zuiden.

3.1 Werken met kleuren

Echte kleuren, in tegenstelling met verschillende grijsniveau's, zijn natuurlijk enkel te verkrijgen wanneer men over een kleurendukker beschikt. `dvips` maakt het mogelijk om PostScript's kleurenopdrachten te gebruiken, en stelt hiervoor de stijl `dpscolor` en `dpsblack` ter beschikking. Deze laatste stijl neutraliseert alle PostScript kleurenopdrachten van `dpscolor`, zodat men een bestand dat deze opdrachten bevat, zonder problemen kan afdrukken op een zwart-wit printer. `dpscolor` biedt een pallet van 68 kleuren (de komplete lijst vindt U in het stijlbestand zelf).

De kleur van de achtergrond voor de lopende en volgende bladzijden wordt bepaald door de opdracht `\background`, met als parameter de naam van de kleur. Indien men een groene achtergrond wenst dan zegt men:

```
\background{Green}
```

Wanneer men dan terug wilt naar de gewone (default) witte achtergrond, dan geeft men:

```
\background{White}
```

`dpscolor` heeft twee soorten kleuropdrachten; de eerste heeft één argument, om korte teksten te kleuren, bijvoorbeeld:

```
\Blue{De volgende tekst wordt in 't
      blauw gezet.}
```

De volgende tekst wordt in 't blauw gezet.

terwijl de tweede de kleur voor de tekst herdefiniëert, en deze definitie onveranderd laat tot aan de volgende kleuropdracht.

```
\textGreen De volgende tekst is in
  't groen. En dat bl"yft ie tot we
  de kleur herdefini"eren.
\textBlack De tekst wordt terug zwart
```

De volgende tekst is in 't groen. En dat blijft ie tot we de kleur herdefiniëren. De tekst wordt terug zwart

Bijkomende kleuren kunnen gedefiniëerd worden in funktie van het CMYK (Cyan, Magenta, Yellow, blacK) kleurenmodel, waarin men een kleur specifiëert aan de hand van de intensiteit (een getal tussen 0.0 en 1.0) van elk van haar vier kleurenkomponenten.

```
\Color{.2 .3 .4 .5}{De kleur van deze
       tekst werd gespecifi"eerd  als
       een CMYK kwadruplet}
```

De kleur van deze tekst werd gespecifiëerd als een CMYK kwadruplet

References

[1] Tomas Rokicki. *DVIPS: A TeX Driver.* `dvips` Distribution, 1993.

[2] Adobe Systems Incorporated. *PostScript Language Reference Manual*. Addison-Wesley, Reading, Second edition, 1990.

[3] Adobe Systems Incorporated. *PostScript Language Tutorial and Cookbook.* Addison-Wesley, Reading, 1985.

[4] Glenn C. Reid. *PostScript Language Program Design*. Addison-Wesley, Reading, 1988.

[5] Glenn C. Reid. *Thinking in PostScript*. Addison-Wesley, Reading, 1990.

[6] Stephen E. Roth, editor. *Real World PostScript*. Addison-Wesley, Reading, 1988.

[7] Ross Smith. *Learning PostScript—A Visual Approach*. Peachpit Press, 1085 Keith Avenue, Berkeley, CA 94708, 1990.

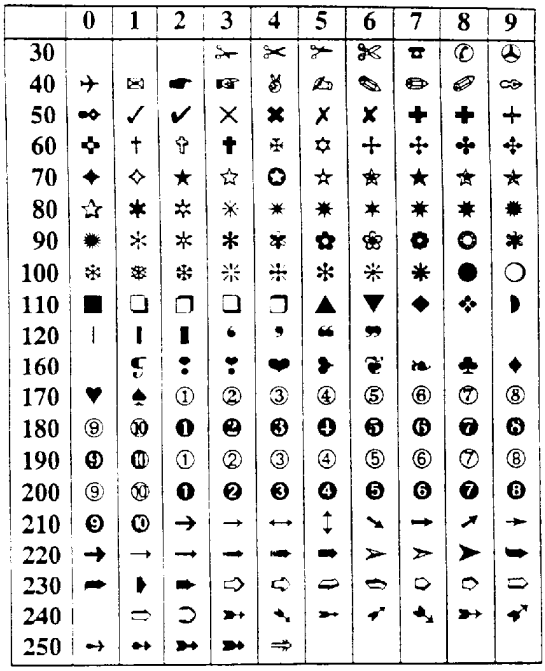
[8] Peter Vollenmeider. *Encapsulated PostScript: Applications for the MacIntosh and PC*. Prentice-Hall and Carl Hanser, 1990.

[9] Sebastian Rahtz and Leonor Barroca. A style option for rotated objects in LaTeX. *TUGBoat*, 13(2):156–180, July 1992.

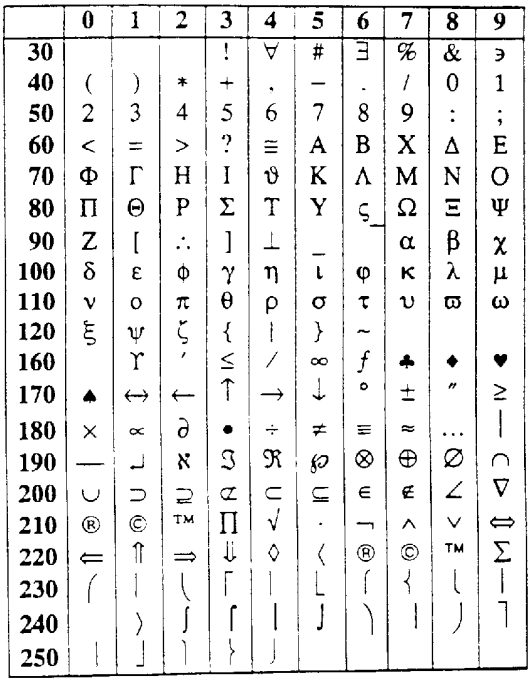
[10] Karl Berry. Filenames for fonts. *TUGBoat*, 11(4):517–520, November 1990.

Familie	Serie	Vorm	Externe namen
Families met schreef (serif)			
Times	m	n, it	`Times-Roman(ptmr), Times-Italic(ptmri)`
	b	n, it	`Times-Bold(ptmb), Times-BoldItalic(ptmbi)`
Palatino	m	n, it	`Palatino-Roman(pplr), Palatino-Italic(pplri)`
	b	n, it	`Palatino-Bold(pplb), Palatino-BoldItalic(pplbi)`
NewCenturySchoolbook	m	n, it	`NewCenturySchlbk-Roman(pncr),` `NewCenturySchlbk-Italic(pncri)`
	b	n, it	`NewCenturySchlbk-Bold(pncb),` `NewCenturySchlbk-BoldItalic(pncbi)`
Bookman	m	n, it	`Bookman-Light(pbkl), Bookman-LightItalic(pbkli)`
	b	n, it	`Bookman-Demi(pbkd), Bookman-DemiItalic(pbkdi)`
Schreefloze (sans serif) families			
Helvetica	m	n, it	`Helvetica(phvr), Helvetica-Oblique(phvro)`
	b	n, it	`Helvetica-Bold(phvb),` `Helvetica-BoldOblique(phvbo)`
	c	n, it	`Helvetica-Narrow(phvrrn),` `Helvetica-Narrow-Oblique(phvron)`
	bc	n, it	`Helvetica-Narrow-Bold(phvbrn),` `Helvetica-Narrow-BoldOblique(phvbon)`
AvantGarde	m	n, it	`AvantGarde-Book(pagk),` `AvantGarde-BookOblique(pagko)`
	b	n, it	`AvantGarde-Demi(pagd),` `AvantGarde-DemiOblique(pagdo)`
Schrijfmachine font			
Courier	m	n, it	`Courier(pcrr), CourierOblique(pcrro)`
	b	n, it	`Courier-Bold(pcrb), Courier-BoldOblique(pcrbo)`
Versier en fantasie fonts			
Symbol	m	n	`Symbol(psyr)`
ZapfChancery	m	n	`ZapfChancery-MediumItalic(pzcmi)`
ZapfDingbats	m	n	`ZapfDingbats(pzdr)`

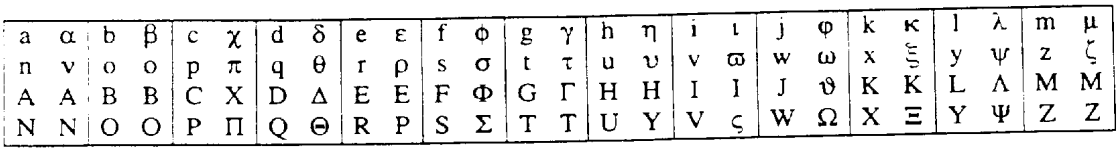
Tabel 2: *Klassificatie van de 35 basis PostScript fonts (tussen haakjes de naam volgens K. Berry)*



Tabel 3: *Het ZapfDingbats PostScript font*



Tabel 4: *Het Symbol PostScript font*



Tabel 5: *De Griekse karakters in het Symbol PostScript font*

References

[1] Tomas Rokicki. *DVIPS: A TeX Driver.* `dvips` Distribution, 1993.

[2] Adobe Systems Incorporated. *PostScript Language Reference Manual*. Addison-Wesley, Reading, Second edition, 1990.

[3] Adobe Systems Incorporated. *PostScript Language Tutorial and Cookbook.* Addison-Wesley, Reading, 1985.

[4] Glenn C. Reid. *PostScript Language Program Design*. Addison-Wesley, Reading, 1988.

[5] Glenn C. Reid. *Thinking in PostScript*. Addison-Wesley, Reading, 1990.

[6] Stephen E. Roth, editor. *Real World PostScript*. Addison-Wesley, Reading, 1988.

[7] Ross Smith. *Learning PostScript—A Visual Approach*. Peachpit Press, 1085 Keith Avenue, Berkeley, CA 94708, 1990.

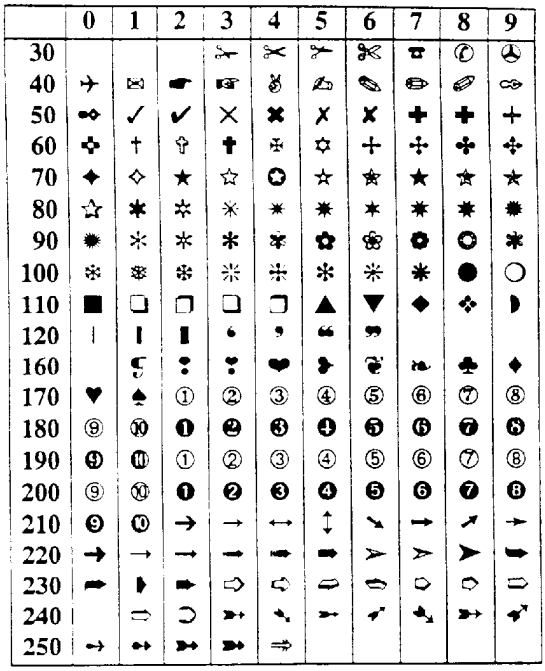
[8] Peter Vollenmeider. *Encapsulated PostScript: Applications for the MacIntosh and PC*. Prentice-Hall and Carl Hanser, 1990.

[9] Sebastian Rahtz and Leonor Barroca. A style option for rotated objects in LaTeX. *TUGBoat*, 13(2):156–180, July 1992.

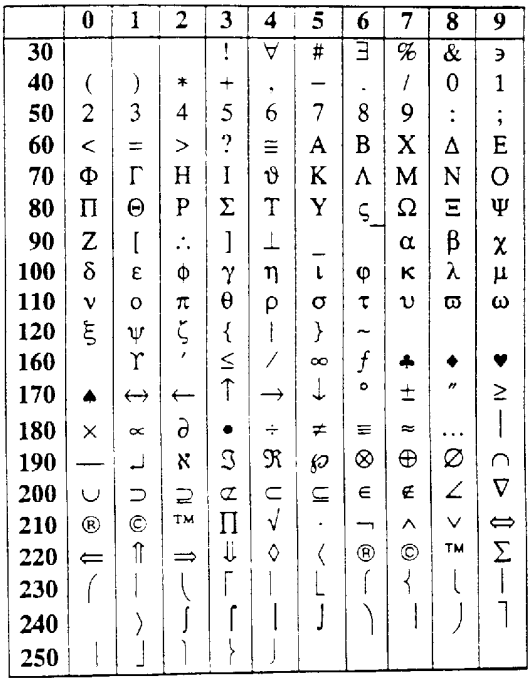
[10] Karl Berry. Filenames for fonts. *TUGBoat*, 11(4):517–520, November 1990.

Familie	Serie	Vorm	Externe namen
Families met schreef (serif)			
Times	m	n, it	`Times-Roman(ptmr), Times-Italic(ptmri)`
	b	n, it	`Times-Bold(ptmb), Times-BoldItalic(ptmbi)`
Palatino	m	n, it	`Palatino-Roman(pplr), Palatino-Italic(pplri)`
	b	n, it	`Palatino-Bold(pplb), Palatino-BoldItalic(pplbi)`
NewCenturySchoolbook	m	n, it	`NewCenturySchlbk-Roman(pncr),` `NewCenturySchlbk-Italic(pncri)`
	b	n, it	`NewCenturySchlbk-Bold(pncb),` `NewCenturySchlbk-BoldItalic(pncbi)`
Bookman	m	n, it	`Bookman-Light(pbkl), Bookman-LightItalic(pbkli)`
	b	n, it	`Bookman-Demi(pbkd), Bookman-DemiItalic(pbkdi)`
Schreefloze (sans serif) families			
Helvetica	m	n, it	`Helvetica(phvr), Helvetica-Oblique(phvro)`
	b	n, it	`Helvetica-Bold(phvb),` `Helvetica-BoldOblique(phvbo)`
	c	n, it	`Helvetica-Narrow(phvrrn),` `Helvetica-Narrow-Oblique(phvron)`
	bc	n, it	`Helvetica-Narrow-Bold(phvbrn),` `Helvetica-Narrow-BoldOblique(phvbon)`
AvantGarde	m	n, it	`AvantGarde-Book(pagk),` `AvantGarde-BookOblique(pagko)`
	b	n, it	`AvantGarde-Demi(pagd),` `AvantGarde-DemiOblique(pagdo)`
Schrijfmachine font			
Courier	m	n, it	`Courier(pcrr), CourierOblique(pcrro)`
	b	n, it	`Courier-Bold(pcrb), Courier-BoldOblique(pcrbo)`
Versier en fantasie fonts			
Symbol	m	n	`Symbol(psyr)`
ZapfChancery	m	n	`ZapfChancery-MediumItalic(pzcmi)`
ZapfDingbats	m	n	`ZapfDingbats(pzdr)`

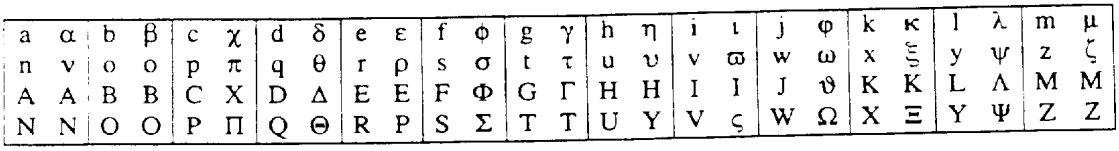
Tabel 2: *Klassificatie van de 35 basis PostScript fonts (tussen haakjes de naam volgens K. Berry)*



Tabel 3: *Het ZapfDingbats PostScript font*

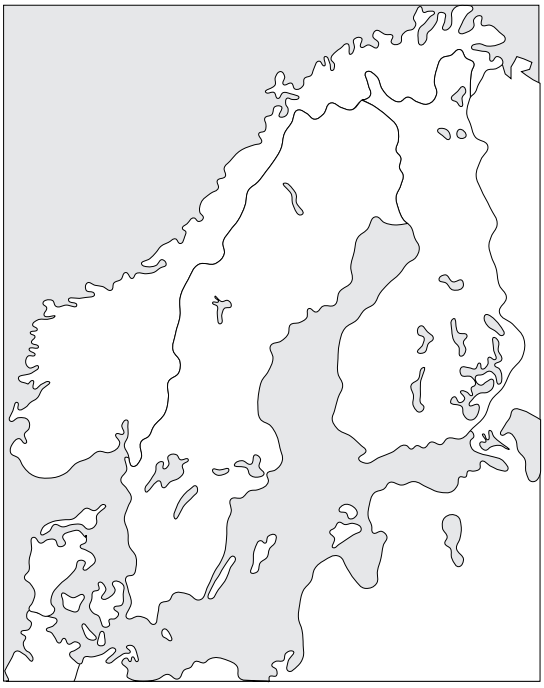


Tabel 4: *Het Symbol PostScript font*



Tabel 5: *De Griekse karakters in het Symbol PostScript font*

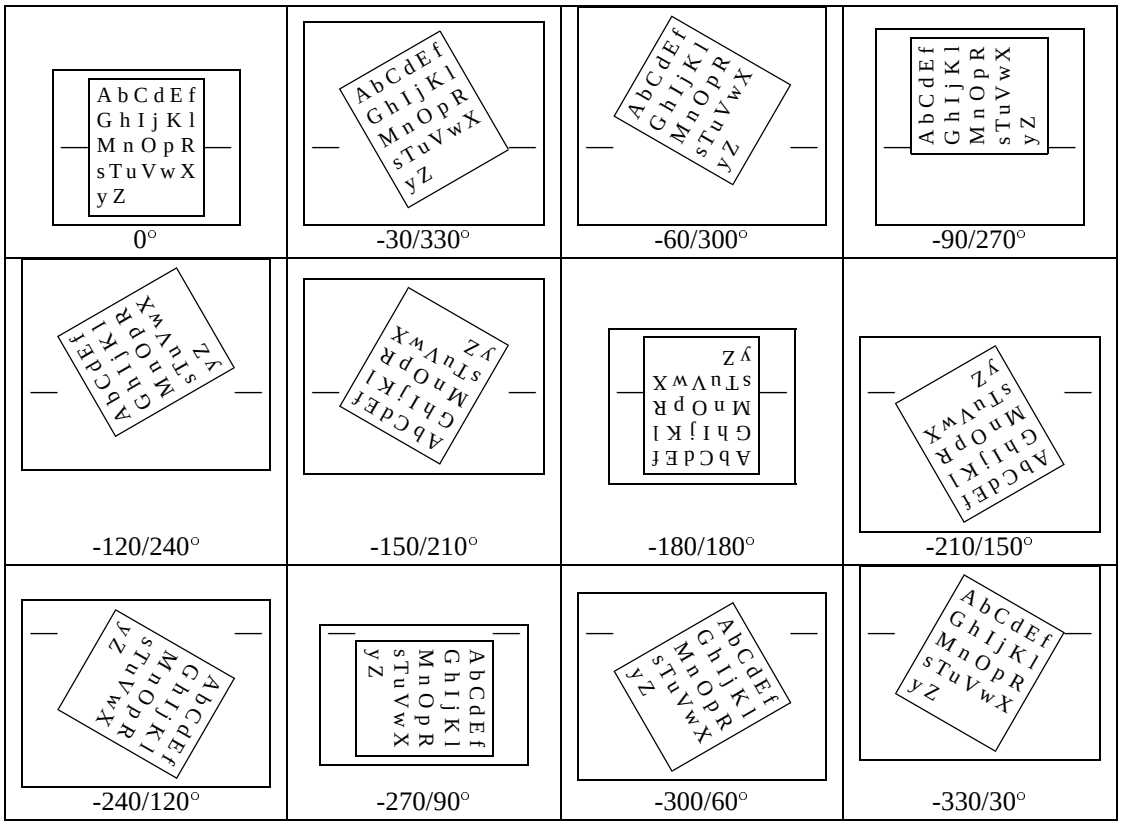
```
\begin{figure*}[p]
  \begin{minipage}[b]{.495\linewidth}
     \centering\mbox{\epsfig{figure=ps/Scandinavia.eps,width=.9\textwidth}}
     \caption{Noord Europa}   \label{fig:draai1}
  \end{minipage}\hfill
  \begin{minipage}[b]{.495\linewidth}
      \centering\mbox{\epsfig{figure=ps/Africa.eps,width=.9\textwidth}}
      \caption{De kaart van Afrika} \label{fig:draai2}
  \end{minipage}
  \centering\mbox{\epsfig{figure=ps/USA.eps,width=.70\textwidth}}
 \caption{De Verenigde Staten van Amerika}     \label{fig:draai3}
\end{figure}
```



Figuur 3: *Noord Europa*

Figuur 4: *De kaart van Afrika*

Figuur 5: *De Verenigde Staten van Amerika*



Figuur 6: *Het draaien van paragrafen*

Kolom 1	Kolom 2	Kolom 3
A	B	C
D	E	F
G	H	I

```
\begin{tabular}{rrr}\\[3mm]
\begin{rotate}{-45}Kolom 1\end{rotate}&
\begin{rotate}{-45}Kolom 2\end{rotate}&
\begin{rotate}{-45}Kolom 3\end{rotate}\\
\hline A& B& C\\ D& E& F\\ G& H& I\\ \hline
\end{tabular}
```

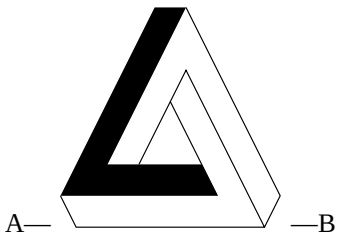
Kolom 1	Kolom 2	Kolom 3
A	B	C
D	E	F
G	H	I

```
\begin{tabular}{rrr}
\begin{turn}{-45}Kolom 1\end{turn}&
\begin{turn}{-45}Kolom 2\end{turn}&
\begin{turn}{-45}Kolom 3\end{turn}\\
\hline A& B& C\\ D& E& F\\ G& H& I\\\hline
\end{tabular}
```

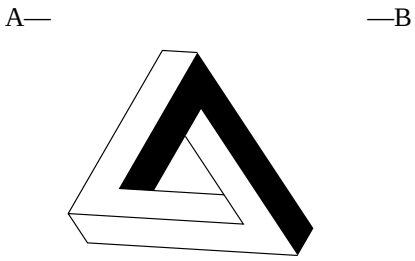
Kolom 1	Kolom 2	Kolom 3
A	B	C
D	E	F
G	H	I

```
\begin{tabular}{rrr}\\[5mm]
\begin{rotate}{-45}Kolom 1\end{rotate}
\rule{.5cm}{0pt}&
\begin{rotate}{-45}Kolom 2\end{rotate}
\rule{.5cm}{0pt}&
\begin{rotate}{-45}Kolom 3\end{rotate}
\rule{.5cm}{0pt}\\
\hline A& B& C\\ D& E& F\\ G& H& I\\\hline
\end{tabular}
```

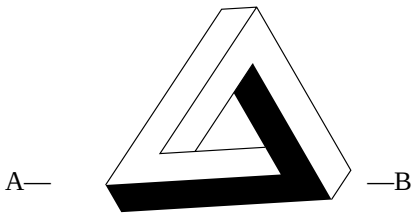
Figuur 7: *Het draaien van informatie in tabellen*



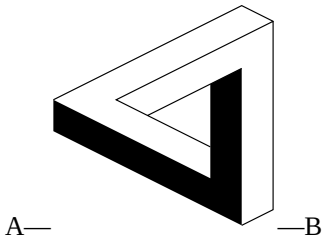

```
A---
 \epsfig{figure=ps/Escher3.eps,width=30mm}
---B
```




```
A---\begin{turn}{-240}
\epsfig{figure=ps/Escher3.eps,width=30mm}
\end{turn}---B
```




```
A---\begin{turn}{240}
\epsfig{figure=ps/Escher3.eps,width=30mm}
\end{turn}---B
```




```
A---\begin{sideways}
\epsfig{figure=ps/Escher3.eps,width=30mm}
\end{sideways}---B
```

Figuur 8: *Verschillende manieren om een beeld te draaien*

Virtual Fonts: Great Fun, Not for Wizards Only

Yannis Haralambous

yannis@gat.citilille.fr

1 Introduction

This paper deals with *virtual fonts*. I would like to present some examples of their astonishing possibilities, taken from everyday typesetting (or almost). First of all, what is a virtual font?

If we look at it from TEX's point of view, then we will not see any difference: for TEX, a virtual font is just like any other font. We have to admit that TEX is in a sad position: it does all this beautiful typesetting, and never sees any result, since for it a font consists just of... boxes. TEX knows only about TFM files. All the beauty of typefaces is unknown to it: the box of a Garamond 'a' looks exactly like the one for the same letter, or any other letter, in Courier[1].

So TEX will never see what it has produced; this pleasure belongs entirely to the DVI driver[2]. The driver will search for a font matching the name specified in the DVI file, pick the right character, and either display it on the screen, or print it on the page. But what kind of information is the driver looking for? Before the arrival of virtual fonts there were three main kinds of fonts: PXL fonts (these are obsolete now), PK fonts (which are still the standard "bitmap" fonts for platform-independent TEX systems) and PostScript fonts. The latter are scalable fonts, in other words, they are defined by mathematical equations just like METAFONT fonts, with the main difference that the work usually done by METAFONT, namely the rasterization of outlines, is done inside the printer, by an engine called "PostScript interpreter".

Both PK and PostScript fonts have good and bad sides. The main problem with PK fonts is that they tend to occupy a lot of space on the storage media (usually a hard disk). And the occupied space depends on the resolution of the printer: for 2540 dpi machines, PK files can get quite big. The good sides are manifold: first of all a PK file is what we get out of a METAFONT run, so usually it looks good; and METAFONT makes fonts fit to our printer: we have complete control on every pixel of our output, no matter how small this pixel can be. And of course, PK fonts can be used on every printer.

PostScript fonts cost money, at least the most commonly used ones. They work only with PostScript printers (unless we have an utility, called ATM, but even then there is no guarantee). They are send to the printer in form of curves and lines; the printer then fills these with pixels. These fonts do not have the same encoding as TEX fonts. A PostScript mechanism allows a straightforward re-encoding, usually done by the driver. Nevertheless in some situations re-encoding is not enough: if a character is not present in a font it may be available in some auxiliary font. For example, Adobe places small caps in an auxiliary font, called "Expert" font; this font does not contain uppercase letters. Making a new font, incorporating both uppercase and small cap characters can take a lot of time, and time is money. Not to mention the fact that such a font will not be a standard Adobe font and hence will have to accompany the document at all stages of production (at the risk of being lost, corrupted, substituted and so on...).

So in both cases (PK and PostScript) there is at least one big problem: either storage space, or cost.

We will show solutions to these problems, based on virtual fonts. The concept of *virtual font* is very simple: things that can appear inside a DVI file are gathered together and presented to TEX as a single character. TEX will ask for one character; the driver will replace this "virtual" character by something else: some other character (re-encoding), or a cluster of characters (for example a character plus an accent), or even a whole page (including PostScript graphic by the means of `\special` commands). We can define a font where each character is virtually mapped to a PostScript figure[3]; we can then "hyphenate" or "kern" or "perform ligatures" with figures (think of the Maya script...)

So here is already a possible construction for a virtual font: take a piece of DVI code (for example out of a DVI file) and consider this to be one character of our virtual font. This technique is used in Eberhard Mattes' QDTeXVPL. The advantage of this method: we can gather every possible TEX instruction in one character; the disadvantage: we can do no more than that. Examples: if we want a quick approximation of an 'a with acute accent', ask TEX to typeset á, by the instruction `\'a`, and borrow that code from the DVI

[1] The author axiomatically considers Adobe Times-Helvetica-Courier to be the trinity of the most annoying fonts.

[2] And to us, humans...

[3] Purists may object this example, since PostScript is by no means TEXware. The author can be excused by the fact that D. E. Knuth in his original paper on virtual fonts, gives an example containing real PostScript code.

file into our virtual font; if, for any reason, we want a 'dotless j' (for example to obtain ˆ) and this character is not provided in the PostScript font, we cannot have it virtually either: there is no DVI command for erasing parts of characters[4].

The clean (but not always quick) way to produce virtual fonts, is out of VPL (Virtual Property List) files; these are PL files with some additional commands[5]

To compile PL files into TFM ones, and vice versa, we use the utilities PLtoTF and TFtoPL. The situation is similar for virtual fonts: we have tools VPtoVF and VFtoVP which convert a VPL file into a VF and a TFM file.

Let's start giving examples of useful or temptative virtual fonts.

2 A Dutch font

There are two specific features of Dutch typesetting, which can become automatic by changing the property list of DC fonts:

- the 'ij' ligature, in names like Eijkhout, Nijhof, Huijqen and in thousands of Dutch words;
- the fact that letter 'ë' becomes 'e' when the word is hyphenated just before this letter: conciërge will be hyphenated as conci-erge.

The automatization of the 'ij' ligature can be done very easily. We just need to add the lines

```
(LABEL C i)
(LIG C j H BC)
(STOP)
```

to the PL file (and the equivalent lines for the 'û' ligature (dont forget to set \uccode"eb="cb).

The problem of the disappearing dieresis is solved by a begin-of-word ligature: when the word is hyphenated, the letter 'ë' suddenly is at the beginning of the next line, and hence at the left boundary of the remainder of the word. If we set the rule *'ë' at the beginning of a word shall become 'e'*, then our problem is solved[6]. This can be written in the PL file as follows:

```
(LABEL BOUNDARYCHAR)
(/LIG H EB C e)
(/LIG H CB C E)
(STOP)
```

These additions will make any DC font behave 'the Dutch way'. Of course we will have to give these fonts new names, for example NLDC. But, since we haven't changed a single pixel of the original DC fonts, it would be very convenient if we could use PK files of DC fonts for our new fonts as well.

For this, we will convert our PL file into a VPL file: let's 'dutchify' font dcr10, by converting dcr10.pl into nldcr10.vpl:

Two new entries are necessary in the preamble:

```
(VTITLE Dutch DC font)
(MAPFONT D 0 (FONTNAME dcr10))
```

The FONTNAME field is the weak point of virtual fonts: the font name which appears in this field necessary depends on the operating system: for example if we want to use a DC Sans-Serif Bold Italic 10 points font, and write dcssbxti10 into that field, then the day when a MS-DOS[7] user wants to use our (compiled) virtual font, he/she will have a rather unpleasant surprise.

The MAPFONT command points to other fonts, it is followed by the internal number of each font. Since we take all of our characters from font dcr10, this is the only font we will define. It takes the internal number 0, which is the default font number. The Dutch font is ready.

Before we go any further, I would like to slightly change the subject and remind the reader that a **properly written file has a properly written header**, and by this I mean a header written using Nelson Beebes filehdr and Robert Solovays checksum specifications. Here is an example of such a header for the nldcr10.vpl file:

```
(COMMENT ********************************
 @Font-Property-List-file{
   author     = "Yannis Haralambous",
   version    = "alpha",
   date       = "28 March 1993",
   time       = "15:50:42 MET",
   filename   = "nldcr10.vpl",
   address    = "187, rue Nationale
                 59800 Lille
                 France",
   FAX        = "(33) 20.40.28.64",
   checksum   = "63003 1955 6452 40079",
   email      = "yannis@gat.citilille.fr",
   codetable  = "ISO/ASCII",
   supported  = "yes",
   docstring  = "Experimental virtual
                 property list file for
                 Dutch, based upon the
```

[4] The only solution would be to write raw PostScript code and put a white mask in front of the dot of 'j'; not a very clean solution, though.

[5] And one important conceptual difference: they contain the (system dependent) names of TFM files of the different fonts from which characters are taken. This makes VF fonts themselves system-dependent, and not only *their names* as in the case of "real" fonts!

[6] At least in most cases; this solution is not 100% clean: for reasons bound to the line-breaking algorithm of TEX, if for example we introduce a hyphen inside the word ('conci-ërge') and it happens to be hyphenated just after that hyphen, the dieresis will remain. . .

[7] MS-DOS is a computer operating system which appeared on planet Earth at the end of the twentieth century, and assumed that filenames would never require more than 8 letters of the Latin alphabet.

```
                dcr10 font."
}***********************************)
```

Nelson has developed a set of Emacs-LISP macros to create and update such headers automatically, inside GNU-Emacs. The checksum is calculated and verified by Robert Solovays `checksum` utility.

3 A font for Welsh

The Welsh language, spoken by some 250,000 people, is notorious for its long words (Don Knuth mentions Llanfairpwllgwyngyllgogerychwyrndrobwllllantysiliogogogoch in the TEXbook) and for its multitude of accents: it needs the five standard vowels (‘a’, ‘e’, ‘i’, ‘o’, ‘u’), and letters ‘y’ and ‘w’, *all* with acute, grave, dieresis and circumflex accent.

Note that all of these (both the letters and the accents) are present in the Cork encoding, what we need is their combination: to be able to hyphenate words (of the length of Dons example) with accented letters, these have to be included in the font as separate characters. We can always obtain the letter ‘ŵ’ by using the primitive `\accent`, but this will block the hyphenation process, which is of vital importance for this language.

So we’ll have to define a variant of the Cork font, featuring in addition to the already existing ‘a’, ‘e’, ‘i’, ‘o’, ‘u’ with all four accents and ‘ý’, ‘ÿ’ also the letters ‘ỳ’, ‘ŷ’, ‘ẃ’, ‘ẁ’, ‘ŵ’, ‘ẅ’. Let’s give the variant of `dcr10`, the name `cydcr10` (`CY` is the ISO two-letter code for Welsh[8]).

The problem is slightly different than for the Dutch virtual font: once again we will remain in the same font, but this time we will have to combine characters: to get ‘ŵ’ we will combine a ‘w’ and a circumflex accent. Because of the width of the ‘w’ letter, the two signs must be superposed with a certain offset. How do we calculate this offset for every possible DC font?

The proper way to do it, would be by using METAFONT. We can ask the latter to write the exact offset between the letter ‘w’ and the circumflex accent in the `.log` file and then use that information. The advantage of this method is that the accent is always well positioned. The disadvantage is that this method is rather slow: for every font needed, one must run METAFONT and then edit the `VPL` file.

A different method is used by Eberhard Mattes’ `QDTeXVPL` (Quick and Dirty TEX to `VPL`). This method leaves TEX the task of placing the accent, using the `\accent` mechanism, and then collects the information. The disadvantage of this method is that the accent will always be placed in the middle and at the same relative height. But of course, this can be corrected manually by editing each `VPL` file.

Here is what to do: suppose we are placing letters ‘ỳ’, ‘ŷ’, ‘ẃ’, ‘ẁ’, ‘ŵ’, ‘ẅ’ at positions `0xA0`, `0xA7` – `0xA9`, `0xB3`, `0xB5` (and ‘Ỳ’, ‘Ŷ’, ‘Ẃ’, ‘Ẁ’, ‘Ŵ’, ‘Ẅ’ at positions `0x80`, `0x87` – `0x89`, `0x93`, `0x95`). We will prepare the following short TEX file:

```
\input qdtexvpl
\font\f=dcr10
\texvpl{^^80}{\f\‘Y}
\texvpl{^^87}{\f\^Y}
\texvpl{^^88}{\f\’W}
\texvpl{^^89}{\f\‘W}
\texvpl{^^93}{\f\^W}
\texvpl{^^95}{\f\"W}
\texvpl{^^a0}{\f\‘y}
\texvpl{^^a7}{\f\^y}
\texvpl{^^a8}{\f\’w}
\texvpl{^^a9}{\f\‘w}
\texvpl{^^b3}{\f\^w}
\texvpl{^^b5}{\f\"w}
\end
```

where we have one `\texvpl` command for each character: the first argument is its position and the second its description using plain TEX macros.

Let’s call this file `welsh-wannabe.tex`. Next, we will run TEX on it and obtain `welsh-wannabe.dvi`. Then we will run `QDTeXVPL` with the following command line:

```
QDTeXVPL -d10.0 welsh-wannabe.dvi \
                    welsh-wannabe.vpl
```

Here is part of the resulting `VPL` file:

```
(CHARACTER O 200
    (CHARWD R 0.749817)
    (CHARHT R 0.939255)
    (CHARDP R 0.000000)
    (MAP
        (PUSH)
        (MOVERIGHT R 0.124969)
        (MOVEDOWN R -0.244980)
        (SETCHAR O 0)
        (POP)
        (SETCHAR C Y)))
```

We see the description of character `'200`, namely ‘Ỳ’. The dimensions `CHARWD`, `CHARHT` and `CHARDP` are calculated out of the superposition of the boxes of the grave accent and of the letter ‘Y’ (Mattes puts both the grave accent and the ‘Y’ in a box and calculates the dimensions of this box).

The `MAP` field describes the character at position `O 200`. First we memorize the current position by a `PUSH` command. Then, we move 0.124969 em to the right, and 0.244980 em up. Next, we select font 16 and set character `0x00`, namely the grave accent. By a `POP` command we return to the position of the last `PUSH` (the usual `DVI` fifo stack) and typeset the letter ‘Y’ as if nothing happened.

[8] This is by no means the announcement of a Welsh font encoding standard. A subgroup of the Technical Working Group on Multiple Language Coordination will take care of that: the new font encoding will be part of the Welsh TLP (TEX Language Package).

Now we have to merge this file with the PL file of the dcr10 font and call the resulting file cydcr10.vpl. Don't forget the VTITLE and MAPFONT fields, similar to those of the Dutch font. (Don't forget Nelsons file header!) By making some cut-and-paste[9] operations we replace the previous character metrics by the new descriptions.

Are we finished? Certainly not. A font is more than just a bunch of characters. We must also take a look at the relationships between these characters, namely ligatures and kernings. We have to check that the previous characters were not involved in some ligatures (that's easy with DC fonts which have only a dozen ligatures, but less trivial with Arabic fonts which have about 5,000 of them).

And then we will have to generate kerning pairs for the new characters: it is logical that in most cases 'ŵ' should be kerned exactly like 'w', and so on. To automate this procedure, the author has written an utility called AdjKerns, which generates kerning pairs for a font out of existing pairs and a set of rules (for example 'œ' should be kerned like 'o' on the left and like 'e' on the right).

Here is how to proceed: first we will write a small file with the rules (this file depends only on the encoding), let's call it welsh-krn-rules.kcf (kcf for kerning configuration file). Its contents will be:

```
KERN H 80 LIKE C Y
KERN H 87 LIKE C Y
KERN H 88 LIKE C W
KERN H 89 LIKE C W
KERN H 93 LIKE C W
KERN H 95 LIKE C W
KERN H A0 LIKE C y
KERN H A7 LIKE C y
KERN H A8 LIKE C w
KERN H A9 LIKE C w
KERN H B3 LIKE C w
KERN H B5 LIKE C w
```

By executing the program with the following command line:

```
AdjKerns -c welsh-krn-rules.kcf \
                        cydcr10.vpl
```

it will read all kerning pairs and ligatures, disregard those of the previous characters, generate new pairs according to the rules in welsh-krn-rules.kcf and overwrite the file cydcr10.vpl. The latter will be the VPL of our newborn Welsh font.

4 PostScript "Expert" Fonts

When Don Knuth wrote the Computer Modern fonts, he found it very natural to include the 'ff', 'ffi' and 'ffl' ligatures in the standard font encoding, and to make a separate font for small caps. Unfortunately the makers of PostScript fonts have decided otherwise: (1) 'ff', 'ffi' and 'ffl' were left outside, (2) small caps are sometimes available, but in an auxiliary font *which does not include uppercase letters!*

Virtual fonts allow us TEX-users to use PostScript fonts without renouncing to our habits and use both the 'f-ligatures' and small caps with the same ease as we use DC fonts.

Let us resume the situation, by taking a concrete example: the beautiful Centaur font (drawn by Bruce Rogers in 1928), commercialized by Adobe Systems:

- the plain PostScript font CentaurMT (MT for MonoType) contains the usual set of characters (uppercase and lowercase), including 'fi' and 'fl' ligatures;
- the 'expert' PostScript font CentaurExpertMT, contains the 'ff', 'ffi' and 'ffl' ligatures, small caps A B C D E F G H I J K L M N O P Q R S T U V W X Y Z, as well as olstyle digits 0 1 2 3 4 5 6 7 8 9. It does not contain uppercase letters.

And here is what we want:

- a plain font centaur with uppercase, lowercase letters and the five 'f-ligatures': 'ff', fi', 'fl, 'ffi' and 'ffl';
- a small caps font centaursc with uppercase letters, small capitals, and no 'f-ligatures'. Eventually we could put the oldstyle digits in this font.

In this way it will be straightforward to write « une fille flamande affligée mais affreusement raffinée » or „DIE GROSSEN DREI B SIND BACH, BEETHOVEN, BRAHMS".

The reader has already guessed the solution: these fonts will be virtual, and will each one take characters from both PostScript fonts. The TFM files of the original PostScript fonts will have names starting by an 'r' (raw): rcentaur and rcentaurexp. Both centaur and centaursc will use characters from fonts rcentaur and rcentaurexp.

There is an utility to do this job automatically: Vulcano, written in 1992 by the author. Vulcano will read both AFM files and according to a certain configuration file will use the information to write a VPL file for a plain or a small caps font.

The configuration files are called dc.vcf and dc-sc.vcf (VCF for 'Vulcano Configuration File'). Here is the command line (splitted in two lines) for making the plain font:

```
Vulcano -c dc.vcf -p rcentaur.pl \
 -v centaur.vpl  -a CentaExpMT.afm \
        rcentaurexp.pl CentaMT.afm
```

The -c option is used to indicate the configuration file, -p gives the name of the PL file of the raw PostScript font, -v the name of the VPL file we wish to have, -a is used to add supplementary AFM files (we can use -a up to 255 times: theoretically we can take every

[9] Emacsians say "kill-and-yank"...

character from a different PostScript font. . .), the two arguments of `-a` are the AFM file name and the name of the resulting raw PL file.

To obtain the small caps font we just have to modify the command line a bit:

```
Vulcano -c dc-sc.vcf -p rcentaur.pl \
 -v centaursc.vpl -a CentaExpMT.afm \
        rcentaurexp.pl CentaMT.afm
```

The reader will notice that the arguments of `-p` and `-v` haven't changed: that is natural since the information Vulcano gathers out of the AFM files is exactly the same, and hence the resulting raw PL files will be the same for both runs.

It is very easy to make new VCF files. Let's take a look to their syntax. In PostScript, every glyph has a name; the 'encoding' actually affects character positions to these names. The first part of a VCF file consists of a list of PostScript names for glyphs appearing in the font, followed by the various positions they occupy in the specific encoding:

```
START Encoding DC (YH 26mar93)
grave -> 0
acute -> 1
circumflex -> 2
tilde -> 3
...
udieresis -> 252
yacute -> 253
thorn -> 254
germandbls -> 255
STOP
```

The second part of a VCF file, concerns the eventual boundary character, left boundary ligatures and ordinary ligatures:

```
START Ligatures 14
21 45 =: 22
27 105 =: 30
27 108 =: 31
33 96 =: 189
39 39 =: 17
44 44 =: 18
45 45 =: 21
60 60 =: 19
62 62 =: 20
63 96 =: 190
96 96 =: 16
102 102 =: 27
102 105 =: 28
102 108 =: 29
STOP
END
```

The use of glyph names in this second part would lead to inconsistencies, here is an example: the glyph `hyphen` is used in two cases in the DC font, at position `45` for the word separator, and at position `127` as the line-break-hyphen. As we can see in the list, a ligature is provided for character `45` but *not* for character `127`; this distinction would not be possible on the PostScript glyph name level.

And now let's take a look at the VPL files created by Vulcano; this time we have two fonts to combine:

```
(MAPFONT D 0 (FONTNAME rcentaur))
(MAPFONT D 1 (FONTNAME rcentaurexp))
```

Here is the code for characters 'ff' and 'fi':

```
(CHARACTER D 27 (MAP (SELECTFONT D 1)(SETCHAR D 86))(COMMENT PS: ff)
   (CHARWD R 552)
   (CHARHT R 673)
   (CHARDP R 1)
   (COMMENT
      (LIG i -> ffi)
      (LIG l -> ffl)
      )
   )
(CHARACTER D 28 (MAP (SETCHAR D 174))(COMMENT PS: fi)
   (CHARWD R 479)
   (CHARHT R 674)
   (CHARDP R 8)
   )
```

In the first case we switch to font 1 (`rcentaurexp`) and select character `86`, in the second case we remain in font 0 (the default font `rcentaur`) and take character `174`.

5 How to Get Rid of Virtual Fonts

If you are sending your file to a correspondant who does not have the same virtual fonts as you (for example if you are using a Dutch or Welsh virtual font and want to switch back to the original DC fonts), then there is an utility for 'de-virtualizing' your `DVI` file.

This utility, called `DVIcopy` and written by Peter Breitenlohner, will replace each virtual character by its real extension. It is a 'must' if you plan to have your own personalized virtual fonts.

6 Availability

All mentionned tools are in the public domain. They are written in ANSI C and can be found on various servers: `QDTeXVPL` can be found at Stuttgart (IP `129.69.1.12`) in directory

`/soft/tex/fonts/utilities/qdtexvpl`

while `Vulcano` and `AdjKerns` are kept in Paris (IP `129.199.104.3`) in `/pub/tex/yannis`. Nelson Beebes `filehdr` package as well as Robert Solovays `checksum` can be found in Salt Lake City (IP `128.110.198.2`), in `/pub/tex/bib` and `/pub/tex/pub`.

`DVIcopy` is written in Pascal `WEB` (a C version of it can be obtained through Tom Rokickis `WEB2C` translator). It can be fetched from Stuttgart, directory `/soft/tex/systems/pc/utilities`.

7 Conclusion

The examples in this paper should not be considered as effective TEXware, but merely as hints on areas open to further development. Many people have written tools to create virtual fonts: Jiří Zlatuška (`ACCENTS`), Alan Jeffrey (tools in `AWK`) and others.

The author hopes that his effort in convincing the reader that virtual fonts are *not* for wizards only, has succeeded. So, have fun!

The Birth of a Virtual Font
The AdjKerns Utility

Yannis Haralambous

yannis@gat.citilille.fr

1 Introduction

Making a baby can be a difficult process: you need at least two people (with the adequate tools) you need some favorable circumstances, and you need motivation.

What about virtual fonts? well, we have motivation enough: several languages are not covered by DC fonts, in particular Welsh, Esperanto, Maltese, many African languages (the glyphs for the latter are included in Joerg Knappens FC fonts) etc. These languages need virtual fonts to be typeset. The circumstances are favorable enough. Up to now, though, we were missing one tool.

Here is the problem: you want to substitute a character in the DC font by some special character you need in your target language. OK. How to construct the virtual representation for this character? well, for that we have Eberhard Mattes' `QDTeXVPL`. It will give us the description of all characters we need. Now suppose you replace an 'A' with some accent, by an 'E' with some other accent. What about all the kerning pairs involving that 'A' you took away? And what all those new kerning pairs you need for the 'E'?

Do you want to replace the old kerning pairs by new ones, by hand?, well, you can try; but if you ever attempt to take a coffee break, then you will never be able to recognize again what you have changed, and what remains to be changed. Not to mention the nasty habit of PL files of having kerning pairs applying to more than one characters. This optimizes parsing speed, but surely makes it more difficult to make manual changes.

The utility I'm presenting does all this automatically. It will read a `[V]PL` file and will

- remove all kerning pairs involving removed characters;
- introduce kerning pairs for new characters;
- add or remove ligatures.

To illustrate the use of this program, I provide a real-world example, namely a virtual font for Esperanto. You will follow the whole process of giving birth to this font, starting by its mother: the common `dcr10` font.

2 How it works

`AdjKerns` has been written in ANSI C[1]. It should run everywhere; it has been tested on a Macintosh (MPW 3.2 and Think C 5.0).

The command line should look like this:

```
AdjKerns [-c foo.kcf] \
    [-o <something>]  <input file>
```

The input file is a `PL` or a `VPL` file[2]. Out of this file `AdjKerns` will read only the `LIGTABLE`, the rest will remain untouched.

The rules on how to modify the `LIGTABLE` are read from the configuration file `foo.kcf`. This file is optional; if you don't include any rules, then `AdjKerns` will just rewrite your `[V]PL` file, by ignoring any optimization attempts of the `LIGTABLE`. The file will be bigger, but the result will be the same; it will be easier to modify, though.

The `-o` option specifies the output file. If no such is specified, then the input file is overwritten (you don't need that file anyway...).

Let's see now the syntax of the `KCF` file. The following lines are allowed:

```
KERN <char> LIKE <char>
KERN <char> LEFT LIKE <char>
KERN <char> RIGHT LIKE <char>
KERN <char> LEFT LIKE <char>
                 RIGHT LIKE <char>
UNKERN <char>
ADD <lig> <char> <char> -> <char>
ADD <lig> BOUNDARYCHAR <char> ->
                          <char>
REMOVE <lig> <char> <char> -> <char>
REMOVE <lig> BOUNDARYCHAR <char> ->
                           <char>
% comments, like in TeX
```

where `<char>` means (like in a `[V]PL` file, either `C` followed by a character, or `D` followed by a decimal number, or `O` followed by an octal number, or `H` followed by a hexadecimal number. `<lig>` means one

[1] Be aware! its author is a lousy C programmer, so *don't* trust the program as you trust TEX. Keep a critical eye for possible bugs, and report them!

[2] If you look closer you'll see that there is no big difference; the latter has a few more possibilities, otherwise the syntax is the same.

of the following: `LIG`, `/LIG`, `LIG/`, `/LIG/`, `/LIG>`, `LIG/>`, `/LIG/>`, `/LIG/>>`.

Everytime you kern some letter, all previous kerning pairs and ligatures are erased. By the `UNKERN` command you can erase these, without inserting a new character. Commands are executed sequentially, and there is no consistence test; so you can ask something and then the opposite: nobody will complain.

3 An Example: Esperanto

Esperanto needs the following characters which are not provided in the DC font: ĉ, ŝ, ˆ, ĝ, ĥ, ŭ (and the corresponding uppercase letters). Dirk Everdeen which is an Esperanto guru (and has prepared a β version of Esperanto hyphenation patterns) asked me to use the following input mechanism: these letters should be accessed by ligatures `cx`, `sx`, `jx`, `gx`, `hx`, `ux`, because the letter 'x' is not used in Esperanto.

Since the "phantom" letter 'x' comes *after* the real letter, there is no problem with kerning (like in the case of the input ligature `ts` of the `wncyr` fonts). So this method could eventually be adopted as a standard transliteration of Esperanto. But this is another issue[3].

Our problem now is to create a virtual font with these letters, the right kerning pairs and the right ligatures. I have chosen positions which would harm the least amount of languages: only Turkish and Slovakian are excluded when using this virtual font.

How to call it? well, of course `eocr10` since `eo` is the two-letter code for Esperanto.

3.1 First step: character description

How do we describe the accented characters? Let's use Eberhard Mattes' `QDTeXVPL`. For this we need a configuration file, which will specify the positions of new characters, and their descriptions in the form of TEX macros. Here is this file (let's call it `esperanto-wannabe.tex`):

```
\font\f=dcr10
\input qdtexvpl
\texvpl{^^ad}{\f\^s}
\texvpl{^^8d}{\f\^S}
\texvpl{^^a0}{\f\^c}
\texvpl{^^80}{\f\^C}
\texvpl{^^a8}{\f\^h}
\texvpl{^^88}{\f\^H}
\texvpl{^^a9}{\f\^\j}
\texvpl{^^89}{\f\^J}
\texvpl{^^a7}{\f\^g}
\texvpl{^^87}{\f\^G}
\texvpl{^^b8}{\f\u u}
\texvpl{^^98}{\f\u U}
\bye
```

The good thing is that we can quickly describe the characters using TEX macros, like `\^`, `\u`, `\j`. The bad thing is that this is all we can do. Fine tuning must be done by hand.

You run this file through TEX and get file `espernato-wannabe.dvi`. Then you run `QDTeXVPL` with the following command line:

```
QDTeXVPL -d10.0 esperanto-wannabe.dvi \
                esperanto-wannabe.vpl
```

Here is an extract of what we get:

```
(MAPFONT D 16
    (FONTNAME dcr10)
    (FONTCHECKSUM O 30523766474)
    (FONTDSIZE R 10.000000))
(CHARACTER O 255
    (CHARWD R 0.394347)
    (CHARHT R 0.694275)
    (CHARDP R 0.000000)
    (MAP
        (MOVERIGHT R -0.052765)
        (SELECTFONT D 16)
        (SETCHAR O 2)
        (MOVERIGHT R -0.447113)
        (SETCHAR C s)))
(CHARACTER O 215
    (CHARWD R 0.555420)
    (CHARHT R 0.939255)
    (CHARDP R 0.000000)
    (MAP
        (PUSH)
        (MOVERIGHT R 0.027771)
        (MOVEDOWN R -0.244980)
        (SELECTFONT D 16)
        (SETCHAR O 2)
        (POP)
        (SETCHAR C S)))
...
```

Now we have to replace these characters inside `dcr10.pl` and name the new file `eocr10.vpl` (hopefully someday somebody will find the time to write some utility to do this automatically).

3.2 Second step: adjusting kerning pairs and ligatures

As you saw, we have modified character descriptions, but neither kerning pairs, nor ligatures. This will be done by `AdjKerns`. Here is the necessary configuration file `eo.kcf`:

```
% Kerning and ligature configuration
% for Esperanto virtual fonts, based
% on DC Semi-official (not yet
% approved by TWGMLC)
KERN H AD LIKE C s
KERN H 8D LIKE C S
KERN H A0 LIKE C c
KERN H 80 LIKE C C
KERN H A8 LIKE C h
KERN H 88 LIKE C H
KERN H A9 LIKE C j
KERN H 89 LIKE C J
KERN H A7 LIKE C g
```

[3] To be solved by the Esperanto subgroup of the Technical Working Group on Multiple Language Coordination.

```
KERN H 87 LIKE C G
KERN H B8 LIKE C u
KERN H 98 LIKE C U
ADD LIG C s C x -> H AD
ADD LIG C S C x -> H 8D
ADD LIG C S C X -> H 8D
ADD LIG C c C x -> H A0
ADD LIG C C C x -> H 80
ADD LIG C C C X -> H 80
ADD LIG C h C x -> H A8
ADD LIG C H C x -> H 88
ADD LIG C H C X -> H 88
ADD LIG C j C x -> H A9
ADD LIG C J C x -> H 89
ADD LIG C J C X -> H 89
ADD LIG C g C x -> H A7
ADD LIG C G C x -> H 87
ADD LIG C G C X -> H 87
ADD LIG C u C x -> H B8
ADD LIG C U C x -> H 98
ADD LIG C U C X -> H 98
```

Note that we have included ligatures both for ‘Cx’ and ‘CX’ etc.: the former will be used in text, the latter in titles. Now we run `AdjKerns`, with the following command line:

```
AdjKerns -c eo.kcf eocr10.vpl
```

After some lines:

```
************************************************************

     This is program AdjKerns (= Adjust Kerning Pairs)

                    Version beta-1
          written for you by a lousy C programmer
                (Yannis Haralambous, 1993)

        It shall help you make efficient virtual fonts
        This software belongs to the public domain

************************************************************
Loading input file preamble...
Reading kern and ligature data...
Reading lines from configuration file and executing them...
.............................
Finished reading the configuration file
Writing down new lig/kern data...
Writing input file remaining data...
```

we get the expected result (our file `eocr10.vpl` is overwritten.

3.3 Final step: using the font

Once we have the `VPL` file, we are done. Run `VFtoVP` and use the font. Here is a small text file to test the font (file `eotest.tex`):

```
\magnification=1200
\font\myfont=eocr10
\myfont
Berto staras antaux la vendejo. Sxi estas rigardanta la
montran fenestregon. Si vidas multajn obiektojn. En la
montra fenestrego trovigxas skatoloj kai faruno. Sub
klosxo kusxas fromagxo, kiu havas multajn truojn. Ankaux
botelojn sxi estas vidanta. La boteloj, kiujn sxi estas
rigardanta, estas egale grandaj. En la mezo de la
fenestrego pendas afisxo, sur kiu estas skribita la
frazo: ,,Cxi tie oni parolas Esperanton‘‘. La pordo
de la vendejo estas malfermita. En la vendejo en angulo
oni vidas barelon kun haringoj. Haringoj estas fisxoj.
La fromagxa klosxo estas farita el vitro. La posedanto
de la vendejo nun estas vendanta fumajxojn kaj cindrujon
al sinjoro. Berto demandas la vendistinon, cxu cxokolado
kaj dolcxajxoj estas haveblaj. La vendistino neas tion.
Ankaux teo, kafo kaj rizo ne plu estas haveblaj. Vino
estas trinkajxo. Berto volas acxeti unue fromagxon kaj
due haringojn. La acxetita fromagxo kostas unu marko. Nun
```

```
sxi estas portanta la acxetitan hejmen.
\end
```

taken from a Esperanto reader my mother had at school (nice brown and thin „Nachkriegspapier“).

4 Go forth, etc. etc.

I am tempted to say “Go forth and make masterpieces of virtual fonts”, but I have a small request: if you wish to make fonts for languages (Welsh, Maltese, etc.) please contact the Technical Working Group on Multiple Language Coordination first; it might very well be that somebody is already preparing this language; and if not, you may be that person. But in any case, we can share our experiences to make *consistent* and *compatible* TEX Language Packages.

5 Availability

You will find all the mentionned files, as well as the sources (and Macintosh-executables) of `AdjKerns` on `spi.ens.fr`, directories

`/pub/tex/yannis/adjkerns` and
`/pub/tex/yannis/adjkerns/examples`.

`QDTeXVPL` can be found in Stuttgart, directory

`/soft/tex/fonts/utilities/qdtexvpl`.

A TEX Language Package for Esperanto is prepared by the TWGMLC (Technical Working Group on Multiple Language Coordination).

When TEX and Metafont Work Together*

Alan Hoenig

Department of Mathematics, John Jay College
Mail: 17 Bay Avenue
Huntington, NY 11743 USA
(516) 385-0736
ajhjj@cunyvm

Abstract

When TEX and Metafont communicate to each other, they can do more together than they can alone. This presentation concentrates on two illustrations of this principal, and urges readers to come up with more.

When TEX becomes sensitive to information passed to it from Metafont, it is possible to prepare diagrams and figures using Metafont and then to have TEX prepare labels which can be precisely positioned within the figure. When communication goes the other way, TEX and Metafont can prepare special purpose fonts which (among other things) can be set along curved baselines. Illustrations of both techniques are presented.

Keywords: TEX, Metafont, labelling figures, figures with labels, curvilinear typesetting, special effects typesetting, typesetting special effects

1 Introduction

When we force TEX and METAFONT to talk to each other, they can do different things and more interesting things than when they work in isolation, and I will give two major examples of this. For the sake of completeness, though, I will try to cover these examples in depth, even though I am more interested in the overview than the details. Plus, I plan to allow myself the luxury of several major digressions whenever I feel like it.

2 Setting the Stage: Labelled Diagrams; Non-standard Typesetting

If you need graphics, especially mathematical figures, for a document, I strongly recommend Metafont. Parabolas and normal curves look like they are supposed to look, and all elements of the picture are placed where they are supposed to be.

But there is a great problem. Figures often need labels or tags to be positioned precisely with respect to the figure, and this Metafont does not give us. I've seen people insert graphics prepared by other programs in their TEX documents that come with text in some typeface completely different from the Computer Modern that's usually used in the main text. This is an unacceptable deviation from Knuth's vision, in my opinion, and my first example will be a discussion of ways to use TEX to label figures.

A second issue concerns the setting of type along non-horizontal baselines. I freely admit that this is not the kind of thing TEX was designed for, nor is it the kind of thing that TEX users should normally care about, unless they need to set type around a circular university or institutional seal. Nevertheless, it's fun (that's definitely part of Knuth's vision!), and it fits right in with the context of this presentation. It will also be possible to apply a variety of other special effects to the type for something a bit out of the ordinary.

3 A Basic Idea

TEX and Metafont have strengths in quite different areas. For example, Metafont is quite good at arithmetic, while TEX is not. TEX, on the other hand, is decent at reading and writing files, while Metafont is *seriously* deficient in that area.

The idea behind having TEX and Metafont work together is to have each do the things each is good at, and communicate these results to the other.

The best way to transfer information from TEX to Metafont is to ask TEX to write this information to a file using statements like figure 1. [1] TEX is good at this, and we can do this so the information in the file conforms to standard Metafont syntax. Then, Metafont simply

* An earlier version has appeared in EuroTEX '92 conference proceedings.

[1] Figures containing bits of text delimited by horizontal rules correspond to the slides that accompanied the oral presentation of this paper.

needs to `input` this file to have access to the data. Fortunately, Metafont does possess the ability to read text files.

```
\newwrite\out=file.mf
  ⋮
\write\out{z1=(w, h); %Text for Meta-
font!}
```

Figure 1: *TEX talks to* Metafont.

Going the other way, from Metafont to TEX, is somewhat problematical. One way I have used is to have Metafont create a dummy font and encode numerical information as the numeric value of a kern between `char0` (say) and `char1`.

Let's make this more precise. Suppose the result of a Metafont calculation is -6.18 pt. Then the Metafont statement

```
ligtable 0:  1 kern -6.18pt#;
```

(don't forget the sharp symbol!) records this value in the `tfm` file for this dummy font as the kern between `char0` and `char1`. Here's how TEX can retrieve this value.

```
\font\foo=dummy \newdimen\result
\setbox0=\hbox{\foo\char0 \char1}
          \result=\wd0
\setbox0=\hbox{\foo\char0{}\char1}
          \advance\result by-wd0
```

Now, `\result` holds the value `-6.18 pt`. In practice, additional hackery can strip the '`pt`' away leaving a pure numeric value. By the way, TEX boxes can have negative widths, so it *is* possible to store negative quantities this way.

MF:

```
ligtable 0:  1 kern -6.18pt#;
```

TEX:

```
\font\foo=dummy \newdimen\result

\setbox0=\hbox{\foo\char0 \char1}
          \result=\wd0
\setbox0=\hbox{\foo\char0{}\char1}
          \advance\result by-wd0
```

Figure 2: Metafont *talks to TEX*.

In the old Metafont, we were restricted to 256 kern pairs per font, and it's easy to use these up in this kind of scheme. In the new Metafont, each font may have up to 32k kern pairs. It is hardly likely that we will run out now, so I recommend that you upgrade your TEX and Metafont if you haven't done so already.

4 Labelling Figures

What would labelling a figure involve? At the time Metafont creates the figure, it should take note of the positions of each label. Then, it should pass that information to TEX.

1. Create the figure.
2. Identify the anchor points.
3. Store the coordinates of the labels.
4. Have TEX typeset the figures in the usual way.
   ```
   \font\fig=figfont
    ...
   {\fig\char0}
    ...
   ```

Figure 3: *How to label figures using TEX and* Metafont.

More precisely, here's the sequence we must follow.

1. Create the figure with Metafont as a character in a special figure font.
2. Identify the points in the figure which will be the anchors for the labels. In Metafont we use notation z_1 or `z1` to refer to key points used for drawing; we might use the notation lab_1 or `lab1` for these anchor points. The tag "`lab`" refers to "labels."
3. Metafont will store the coordinates of the lab_i as kerns between `char0` and successive characters of the figure font. The components of each of the lab_i measures the horizontal and vertical offsets from the reference point (origin) of the figure.
4. TEX typesets the figures in the usual way:
   ```
   \font\fig=figfont
   ...
   {\fig\char0}
   ```
   A set of TEX macros allows placing the labels using the trick mentioned above. One might say `\point{This is a label}` and so on.

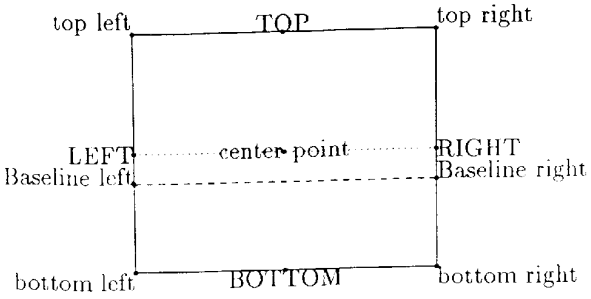


Figure 4: *References for all pointing commands*

These macros are enclosed in two sets of macros, the so-called `labtex` macros (one for TEX, and one for Metafont). `labtex.tex` imagines that each label is enclosed in a rectangular box. Figure 4 shows the eleven different pointing commands for positioning the label box with respect to the label point in the Metafont diagram.

```
\up3pt    \down     \right      \left
\extradx  \extrady  \everylabel

\tlpoint  \ltpoint
\tpoint
\rtpoint  \trpoint
\rpoint
\Brpoint  \rBpoint
\brpoint  \rbpoint
\bpoint
\blpoint  \lbpoint
\Blpoint  \lBpoint
\lpoint
\cpoint
```

Figure 5: `labtex` *pointing commands and "tinkering" commands.*

This figure also shows how tight the labels are in the absence of some tinkering. We can tinker either by hand or semi-automatically. To adjust a label placement by hand, we can use the commands `\up`, `\down`, `\right`, or `\left` to move the label by a certain amount. For example, instead of saying `\cpoint{A label.}`, we can say

```
\cpoint{\down3pt\right2pt A label.}
```

We can also adjust all labels automatically by means of three parameters which work in the background. `\extradx` and `\extrady` place extra space to the right/left or above/below the label text. A token `\everylabel` (akin to `\everypar`, `\everyhbox`, `\everymath`, and so on) allows some global adjustments to the labels themselves. For example, if you set

```
\everylabel=\ninepoint
```

you would get smaller labels automatically.

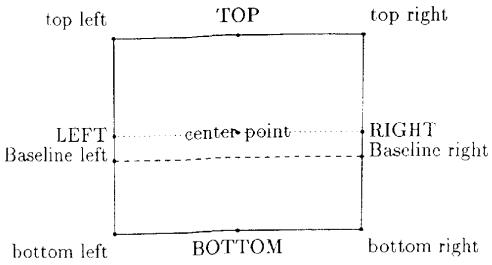


Figure 6: *Automatic tinkering with label placement.*

Figure 6 show how these labels look using these automatic enhancements. Both `\extradx` and `\extrady` have been set to 3 pt. By the way, although this particular figure could have been produced with TEX (after all, it consists only of horizontal and vertical line segments, which TEX can draw), it wasn't. In particular, observe that Metafont can also draw dotted and dashed lines.

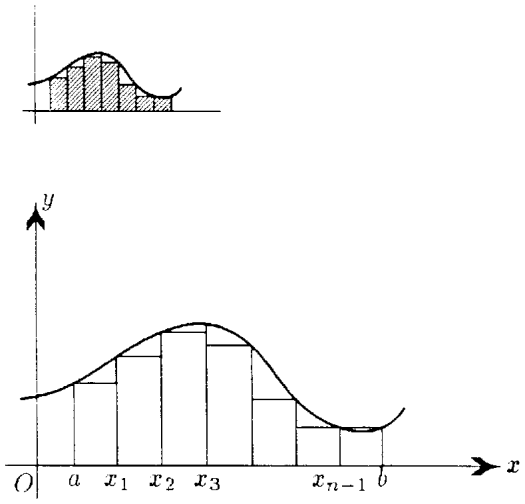


Figure 7: *Developing the theory of integrals.*

One of my hidden agendas in this project is to remind people of the power and utility of Metafont as a figure and diagram generator. With Metafont, you can do all the things you expect in a figure—and more, now that it is possible to anchor labels in the diagram.

Less well known is Metafont's ability to add shading to a drawing. Figure 7 displays a figure that was copied from the book I learned calculus from. (Published by Addison-Wesley, Knuth's publisher, it seems to use Monotype Modern 8a, the inspiration for Computer Modern, as its text face.) There are two versions of this figure for a reason. The larger version has labels but no shading—standard Metafont cannot handle shading for such large pictures. The smaller version was the largest I could generate with the shading, but alas, there was no room for labels in this case.

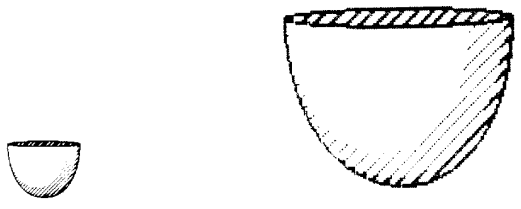


Figure 8: *Special effects with* Metafont.

Figure 8 shows one way that Metafont can draw a three-dimensional bowl. (If it doesn't look sufficiently 3-D, blame the author and not Metafont.) It's fun generating shading, but you quickly run up against the memory capacity of normal Metafont, as I did in figure 7. If the user community vocalizes a need for this kind of Metafont work, I hope the implementer community will respond with versions of "Really Big Metafont."

4.1 Digression: Applying Shading to Regions in Metafont

I'd like to briefly digress to indicate the method I used to shade Metafont regions. The crucial idea behind this was suggested to me by Yannis Haralambous.

Suppose we want to apply a striped vertical pattern to a simple rectangle, as schematically illustrated in figure 9. It's helpful to remember that regions created by Metafont are patterns of pixels. In this figure, the absence of a pixel is indicated by a period and the presence of a single pixel by a 1. Metafont, though, can doubly, triply, or multiply write on a pixel so that pixel values may possess other whole number values. Pixels may also be multiply erased, and possess negative whole number values as well. These observations are crucial in what follows.

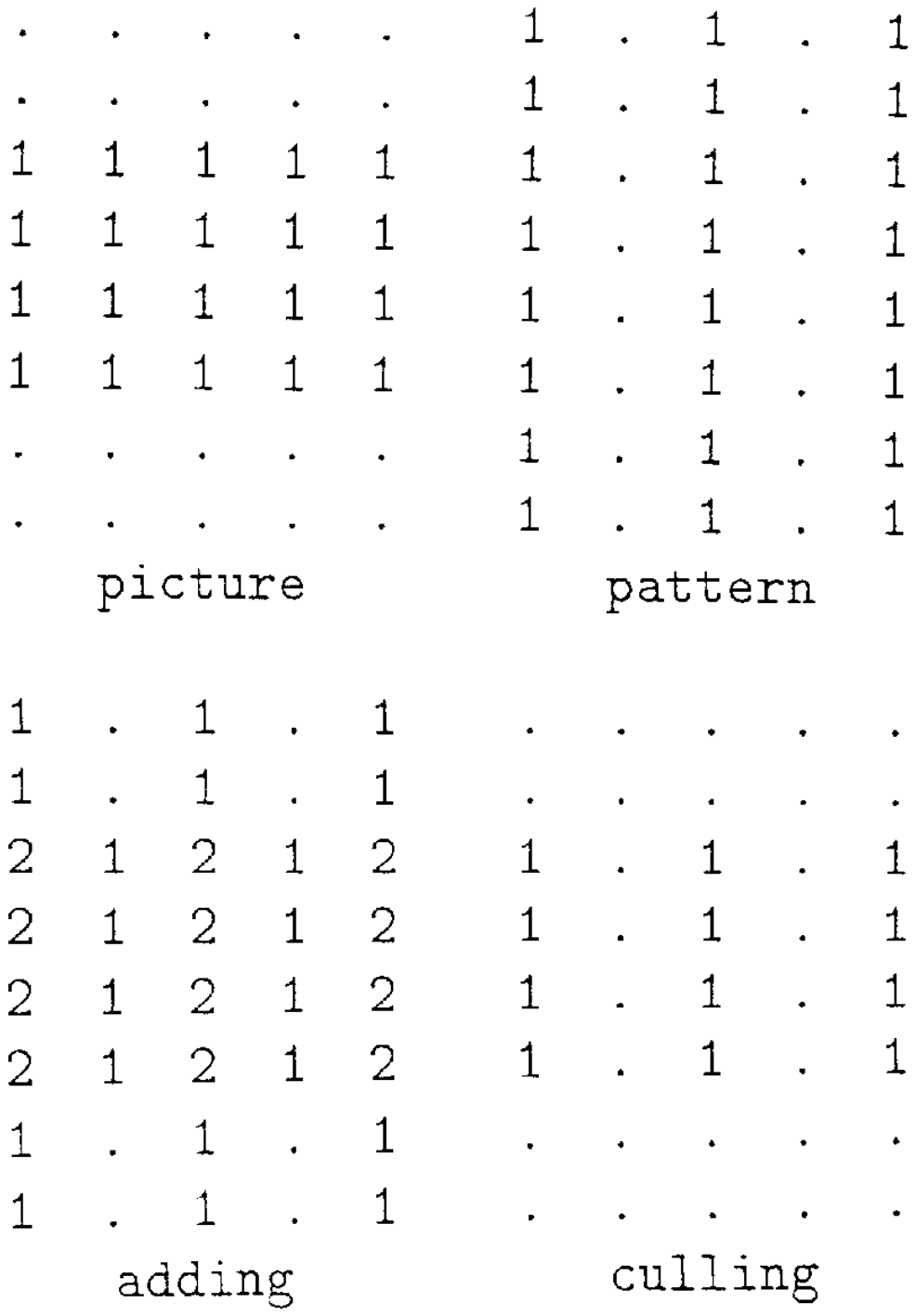


Figure 9: *Applying a pattern to a region in* Metafont.

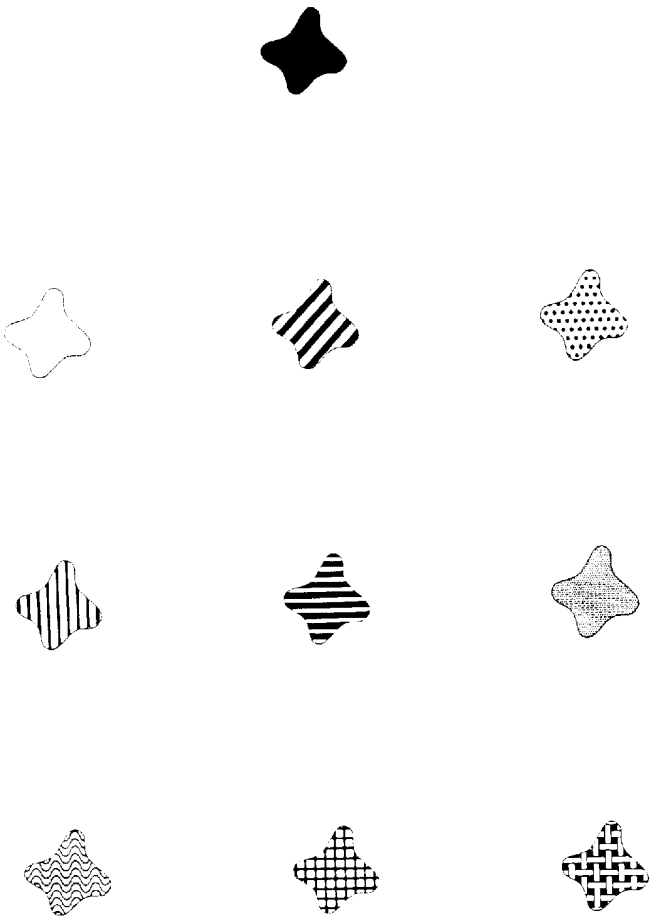


Figure 10: *A region and some background patterns.*

What happens when we add the pattern to the region? To add two pictures together, simply add the corresponding pixel values. Consequently, figure 9 shows the new picture. We see that some pixels now have values of 2, and *it is precisely these pixels that we wish to preserve in the final picture*, as these pixels trace out the striped pattern only in the region of the original shape (in this case, a rectangle). But now our work is done, for we can instruct Metafont to strip away all pixels except those with a value of two. Once this culling is done, we are left with the striped rectangle that we were after, as shown finally in figure 9.

A similar set of operations allows us to outline any region, but as this is treated in *The* Metafont*book*, I will say no more about it here.

There is nothing special about stripes and rectangles; this reasoning applies to arbitrary shapes and arbitrary patterns. I can't resist showing off some patterns. In figure 10, we see a general amorphous shape on which I have drawn an outline and patterns using these operations. Of course, we can use letterforms of a font on which to superimpose these patterns, and you see an example of "Computer Modern Candystripe" in figure 11, which is based on `cminch` (or `cm-half-inch`, to be precise). Figure 14 shows some more fonts. The author of the Computer Modern fonts has provided so many hooks into the programs generating the fonts that very little modification of the existing Metafont programs was needed to generate these fonts.

4.2 Other Work with Labelled Figures

Returning now to my main discussion, I'd like to complete the discussion by referring to other work done in the field of labelling figures.

First is the `diagramf` package developed by Alan Jeffrey. He uses a variant method which is superior in many respects to the one I have just outlined.

Metafont is able to produce only three kinds of files—`gf` pixel files, `tfm` font metric files, and the ubiquitous `log` files. I used the `tfm` to communicate with TEX, but he uses the `log` file in the following manner. The Metafont `message` command displays its argument on screen, but also records it in the log file. If we create a macro that automatically appends a flag to any message, then a simple filter can process the `log` file, collecting all this special `message` data. In a Unix environment, `grep` does this straightforwardly, but even on other platforms, it's simple enough to create a utility to do that for you.

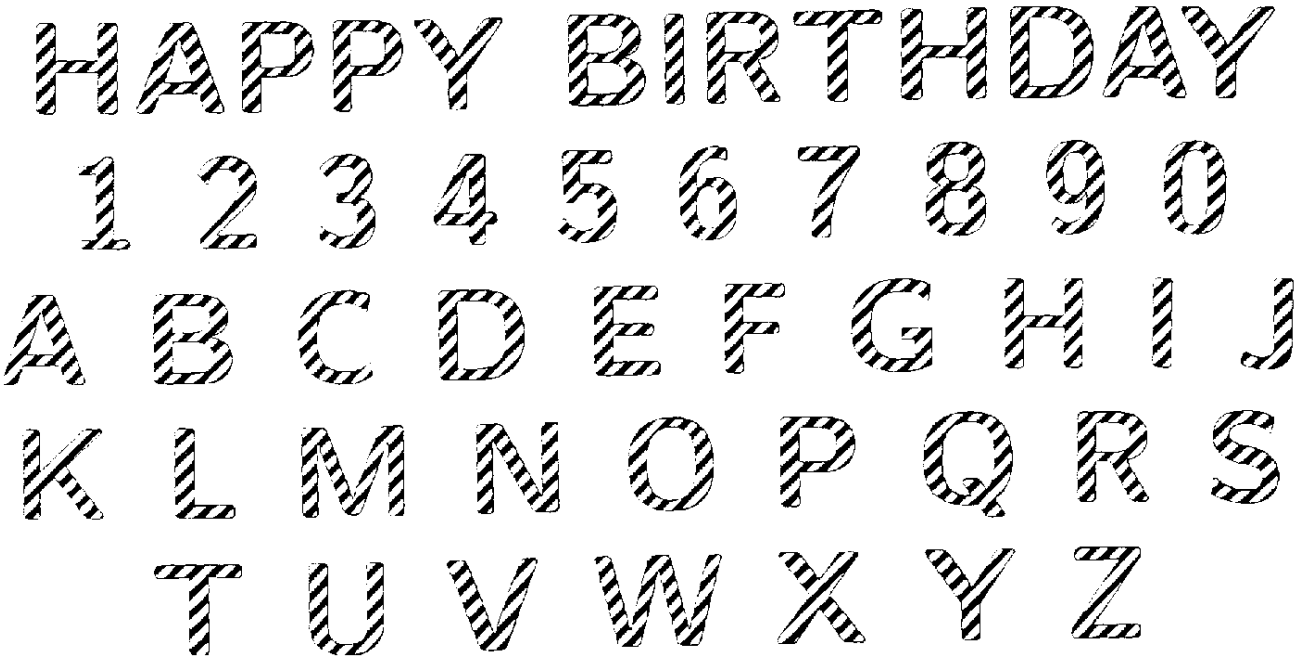


Figure 11: *Computer Modern Candy Stripe.*

Although this dependence on `grep` is outside the TEX-Metafont loop, its advantage is that you can pass more than numeric data to TEX. In the method I outlined previously, it was our job to keep straight which label text belongs to which label position. With `diagramf`, you combine the text with the position data in the Metafont file, so no further record keeping is necessary.

Using Diagramf

1. MF: create figure.
2. grep: strip log file.
3. TEX: run file to measure label text.
4. Return to beginning to resolve forward references.

Figure 12: *The steps in using* `diagramf`.

There is a further advantage to this method. After TEX typesets the label text, it can pass metric information back to Metafont. As a result, Metafont can revise the diagram to conform to the text. Figure 13 shows one result of this in practice.

There are three other packages users should be aware of. Both allow the placement of text at positions inside previously prepared PostScript figures. These diagrams can be produced by other applications like Adobe Illustrator, MatLab, Fontographer, and so on. Craig Barratt of Stanford has produced the `PSFrag` style for use with LATEX, while Jean Orloff of CERN the `PSBox` style, for use with either LATEX or plain TEX.

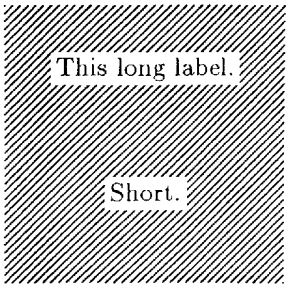


Figure 13: *Cutouts fit around their labels.*

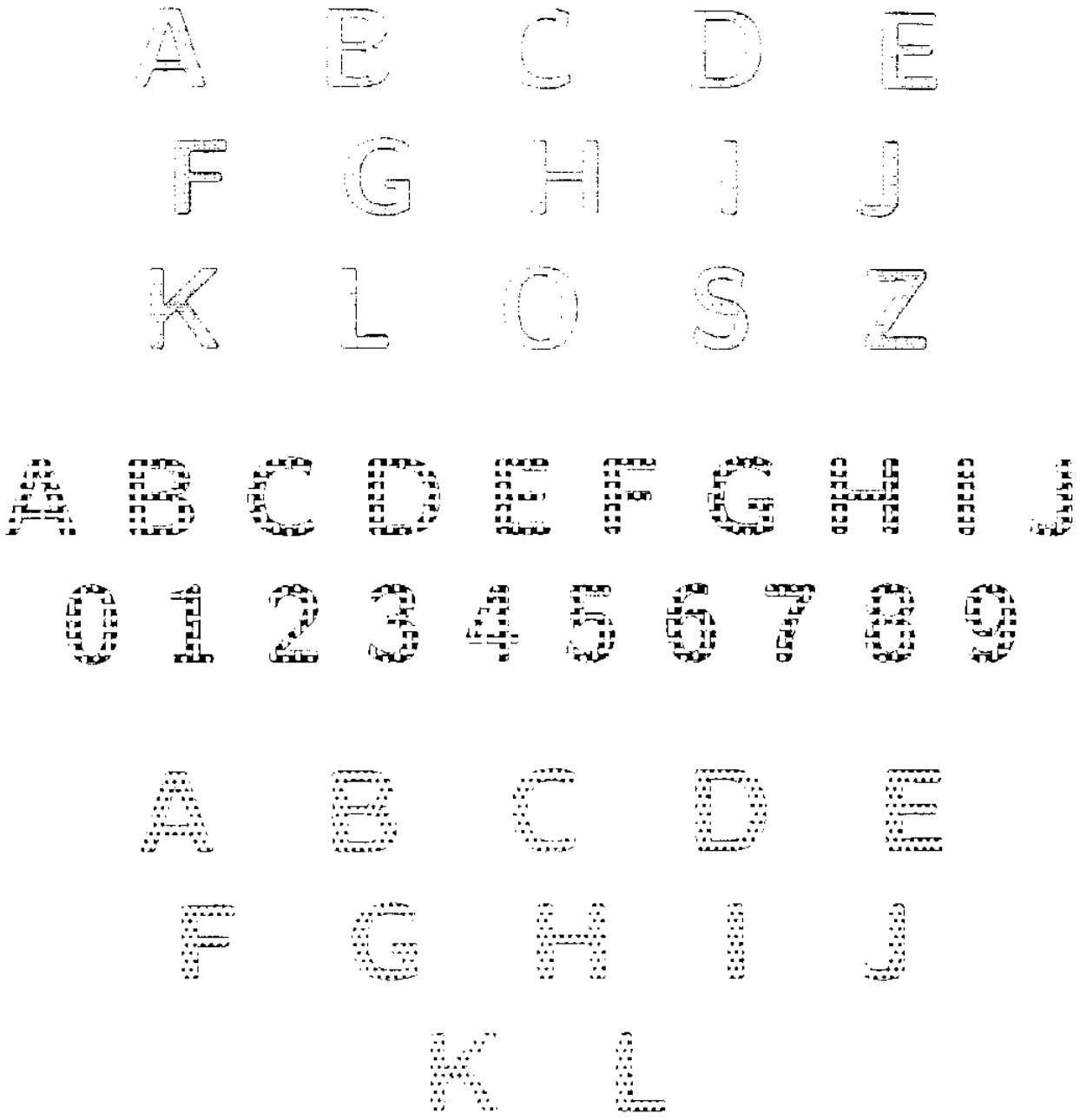


Figure 14: *More fun fonts.*

Figure 14: *More fun fonts.*

Just recently, I became aware of another such package that I'd like to mention. This is the `mfpic` macro package by Thomas Leatherum. It's not a terribly extensive package—the user manual TEXs out to four pages of `\magstephalf` text—but its product cycle is the inverse to that of `labtex` and `diagramf`. That is, you start with TEX code, and `mfpic` generates the appropriate Metafont files. I am excited about `mfpic` for a simple reason. Since all the `mfpic` commands are *TEX* commands, this package is therefore a perfect introduction to Metafont to those users who are too timid to tackle Metafont itself. Actually, since the pictures are created with `mfpic`'s TEX commands, you fit the labels into the picture using usual `\kern` commands rather than the special sleight of hand that I or Alan Jeffrey used. (Or so I gather from a very brief perusal of this package.)

Labeling Packages

- `labtex`: ask the author
- `diagramf`: archives such as Aston (tex.ac.uk) and SHSU (niord.shsu.edu)
- `PSFrag`: Aston
- `PSBox`: Aston
- `mfpic`: SHSU

Figure 15: *Labeling packages and how to get them.*

The author of `mfpic` makes no claim as to the completeness of his package. It is modeled on the famous PiC-TEX macros which attempt to the same thing, but entirely within TEX. Tom notes that `mfpic` is not complete, as not all of PiC-TEX's capabilities are a part of `mfpic`. He would like to see further enhancements made to `mfpic`, but he has no plans himself to do this.

A final approach to the problem of combining text with figures is a simple one: write your own implementation of a Metafont-like language to do this. That's the approach John Hobby took with his MetaPost system.

The macro packages I just mentioned appear to be freely available for use and can be downloaded from many good archives. I obtained `diagramf` from the `niord` server (anonymous ftp from `niord.shsu.edu` in directory `diagramf`) and the other two from the Aston archives (anonymous ftp from `tex.ac.uk`). `mfpic` is also available from `niord`.

5 TEX to Metafont

It's much easier to get TEX to talk to Metafont. TEX has many more file handling commands at its disposal, and it's perfectly possible to get TEX to create a file whose contents conform to proper Metafont syntax. Later, Metafont can read the file.

What purpose is served by having TEX talk to Metafont in this fashion? TEX can record metric information about sequences of letters it will typeset. Then, Metafont can use this information to create a character based upon the positions/measurements/states of characters which came before.

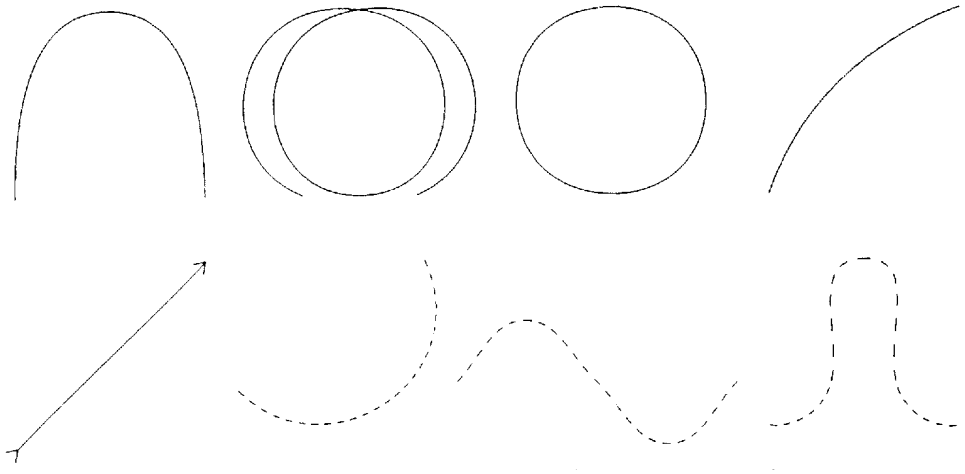


Figure 16: *Convex paths and non-convex paths (drawn with dashed lines).*

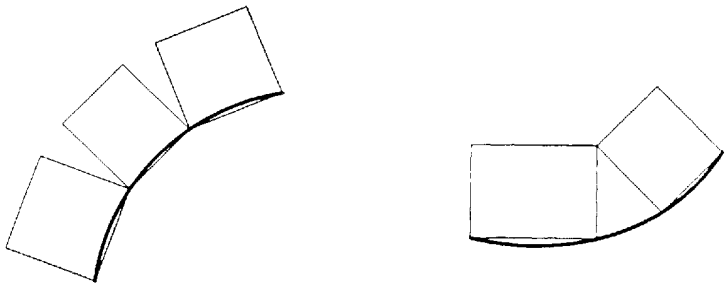


Figure 17: *Why we need a convex path.*

As a first application, suppose we want to typeset along a convex path. Informally speaking, a *convex path* is one which "sheds water." More precisely, if we start driving along it from the left to the right, we never turn our steering wheel to the left. See figure 16.

Figure 17 provides insight as to why we need a convex path. We imagine that the type sits on chords connecting to points of the curve. If the curve is convex, then the problem of placing adjacent types is a problem of the geometry of the curve. We simply butt the adjacent types at their bases, as in the left of figure 17. (The rectangles in this figure represent the bounding boxes of each character.) If the curve is not convex, the problem of placing adjacent type requires us to make sure that

the types are far enough apart so they intersect only at their tops. How far apart shall they be? This is a much more difficult problem in general, although I shall say something about a special case below.

5.1 A Three-Pass Method

A three-pass method is necessary to create the special purpose font for curvilinear typesetting. Note that the end product is a special purpose font limited for *one use* only—the typesetting of one particular message around one particular path. This is quite different from the normal way we use Metafont. Normally, of course, a Metafont font is appropriate for an infinite number of messages. Fortunately, the processes involved—TEX and Metafont—do their jobs so swiftly that the making of special purpose fonts is not at all onerous.

Measure.
Make.
Mount.

Figure 18: *Measure—make—mount.*

Three passes are needed to create the special purpose font. First we *measure* the type with `measure.tex`. Then we *make* the type with macros contained in `make.mf` (plus others). Finally, we *mount* the letters on the page with `mount.tex`.

Let's examine the steps one by one.

5.2 Step One: Measuring the Message

TEX examines the text of your curved message in the first pass. The macros in this file, `measure.tex` examines each individual character in your message but don't typeset it. Using the techniques of exercise 11.5 (page 67) of *The TEXbook*, `measure` takes note of each character in the message and its width. It prepares two files for later use by Metafont. The file `measure.wid` contains `wid`th information about the individual characters in the message, while `measure.cha` contains information about the letters and `cha`racters themselves. Both these files are created using standard Metafont syntax. Shortly, Metafont will read these files.

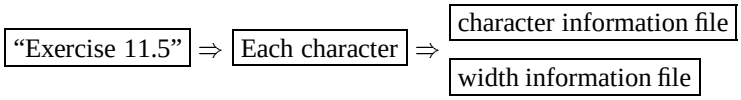


Figure 19: *Measuring the text.*

5.3 Step Two: Making the Message

One underlying assumption is that we are using the standard complement of Computer Modern fonts for curved typesetting. In preparation for step two, it is necessary to make a few changes to the standard font files. (Actually, such a strong assumption is not necessary. It is sufficient to have Metafont descriptions of the fonts, and with the appearance of utilities to convert outline font descriptions to Metafont descriptions, this is not a burden at all.)

The most important involves the actual program files. Typically, these have names like `romanu.mf` (containing the programs for Roman, uppercase letters) or `punct.mf` (with the programs for punctuation). In `romanu`, the program structure is something like

```
cmchar "The letter A";
beginchar("A", ...
 ...
endchar;
```

where the ellipses indicate the presence of material that is irrelevant to this discussion. We want to embed these definitions into Metafont macros. After all, there may be more than one 'A' in a curved text, and we need to be able to create as many A's as necessary. A series of tedious but trivial modifications transform the programs into subroutines. The idea is to relate a letter `n` with a subroutine `n_` which is a function of one argument—the angle of rotation that is necessary to rotate the letter and keep it on the curved baseline. The code fragment above might become

```
def A_(expr n, rotation_angle)=
 currenttransform:=identity rotated
                              rotation_angle;
 def t_=transformed currenttransform enddef;
 cmchar "The letter A";
 beginchar(n, ...
  ...
 endchar;
enddef;
```

and so on for the remaining character programs. We have to change the name of this file, and adjust the remaining files to call this file rather than the standard `romanu.mf` file.

- Fiddling with the macros.
- Measure special position.
- Measure angle of rotation.
- Pass the offset information to TEX.

Figure 20: *Making the special letters.*

Here are some other things we need do in the `make` step of the cycle. I use Metafont to draw the path along which the typesetting will go, and this path becomes the first character in our new font, `char0`.

Now for each character in the message, Metafont determines the position of this letter on the path, and records this information for subsequent retrieval by TEX.

First, suppose the point z_0 marks our current position on the curve. Then we use Metafont's `solve` macro to find the point z_1 such that the length of the chord $z_1 - z_0$ is the same as the width of the current character. (Plain Metafont sets the value of `solve`'s tolerance to `0.1`; we need to decrease to `tolerance=0.001` or even smaller). It's easy for Metafont to determine the angle of the chord, and thus the angle of rotation for the letter. (As you see, we are approximating the path by a series of chords which inscribe the path such that each face of the polygonal approximate path will be the exact width of each character in the message.)

TEX will eventually need two pieces of information about each letter in order to typeset it properly—the x- and y-offsets of that letter from the previous letter. We pass this information to TEX using kerning pairs (as we did at the outset of this discussion to pass coordinate pairs to TEX in order to label a Metafont diagram).

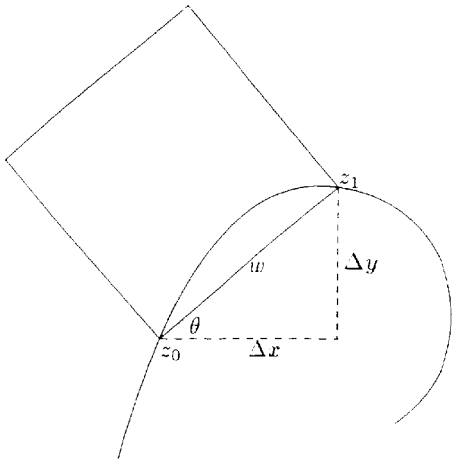


Figure 21: *Needed information.*

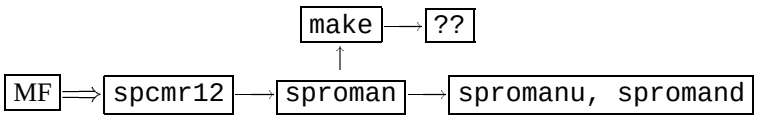
The Metafont process uses so many more files than TEX does that it's worth examining the alterations to the normal Metafont production cycle in a slight more detail. Normally, to produce a font of `cmr12` (for example), Metafont first reads a parameter file (`cmr12.mf`), which calls the driver file (`roman.mf`), which finally calls the program files (`romanu.mf, romanl.mf, romand.mf,` etc.).

MF ⟹ `cmr12` ⟶ `roman.mf` ⟶ `romanu.mf, romand.mf, ...`

I mirror this process, except I rename the files by adding the prefix 'sp' (for special purpose)

MF ⟹ `spcmr12` ⟶ `sproman.mf` ⟶ `spromanu.mf, spromand.mf`

and add two new files to the process. The first is `make.mf`, which contains the actual macros for analyzing and storing the curved baseline. The driver file `sproman` calls this file. A second file contains the actual definition of the path together with any special alterations to the macros to achieve whatever special effects you are after. You call this file whatever you want, but don't forget to `input` it in the `make` file. (In the absence of such a file, `make` defines defaults for all such special purpose things.)



The '??' in this figure represents the file you need to prepare and which will contain the special definitions pertaining to your path.

As a practical matter, all fonts are named `spcmr12` in this scheme. As with any meta-font, you have to convert the generic pixel files into packed pixel `pk` files, and place it and the corresponding font metric file in the places in your system where TEX expects to find them. I have gotten into the habit of transferring these files into these distant directories with new names indicative of whatever they illustrate.

```
\font\rofont=spiral
  ⋮
\curvetype{ABCDEFG ...} % text for curved message
```

Figure 22: *Mounting—placing—the curved type on the page.*

5.4 Step Three: Mounting the Message

Finally, it's TEX's turn again. The file `mount.tex` contains all the positioning macros, as well as the ma-

chinery for extracting the position information from the kerns. All you have to do is tell TEX the name of the special purpose font containing your message, for example,

```
\font\rofont=spiral
```

5.5 Examples

In figure 23 you see some text typeset along a wiggly, self-intersecting curve. Unfortunately, it doesn't seem to be possible to have TEX automatically make space so it doesn't overprint at the point of intersection.

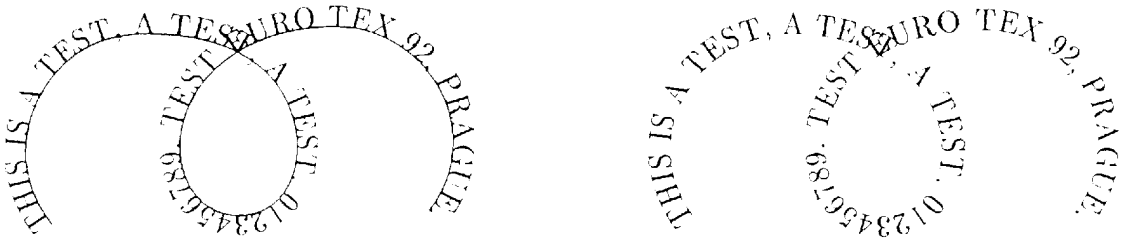


Figure 23: *Type along a wiggly curve.*

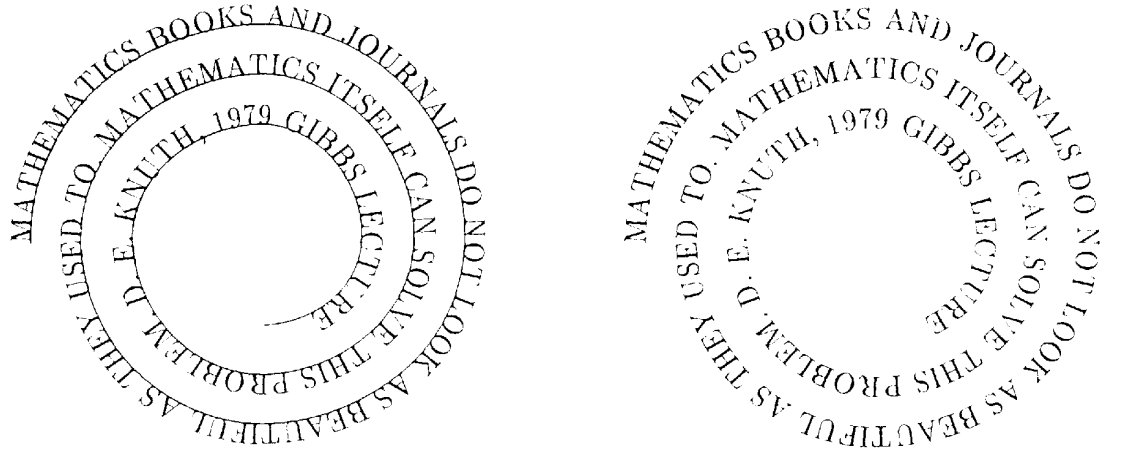


Figure 24: *Type along a spiral.*

In figure 24, TEX typesets along a spiral. It is harder than you think to draw a spiral in Metafont, so it is nice to be able to print the curved type twice—with and without the underlying path, so we can convince ourselves that TEX and Metafont have remained true to the curve. For some reason, certain device drivers choked when trying to print this bit of text.

Figure 25 was inspired by a promotional piece developed by Blue Sky Research, purveyors of TEXtures for the Macintosh. The radii of the ellipse are in golden ratio $\frac{1+\sqrt{5}}{2}$, a particularly pleasing ratio. It proved to be straightforward to achieve the font change.

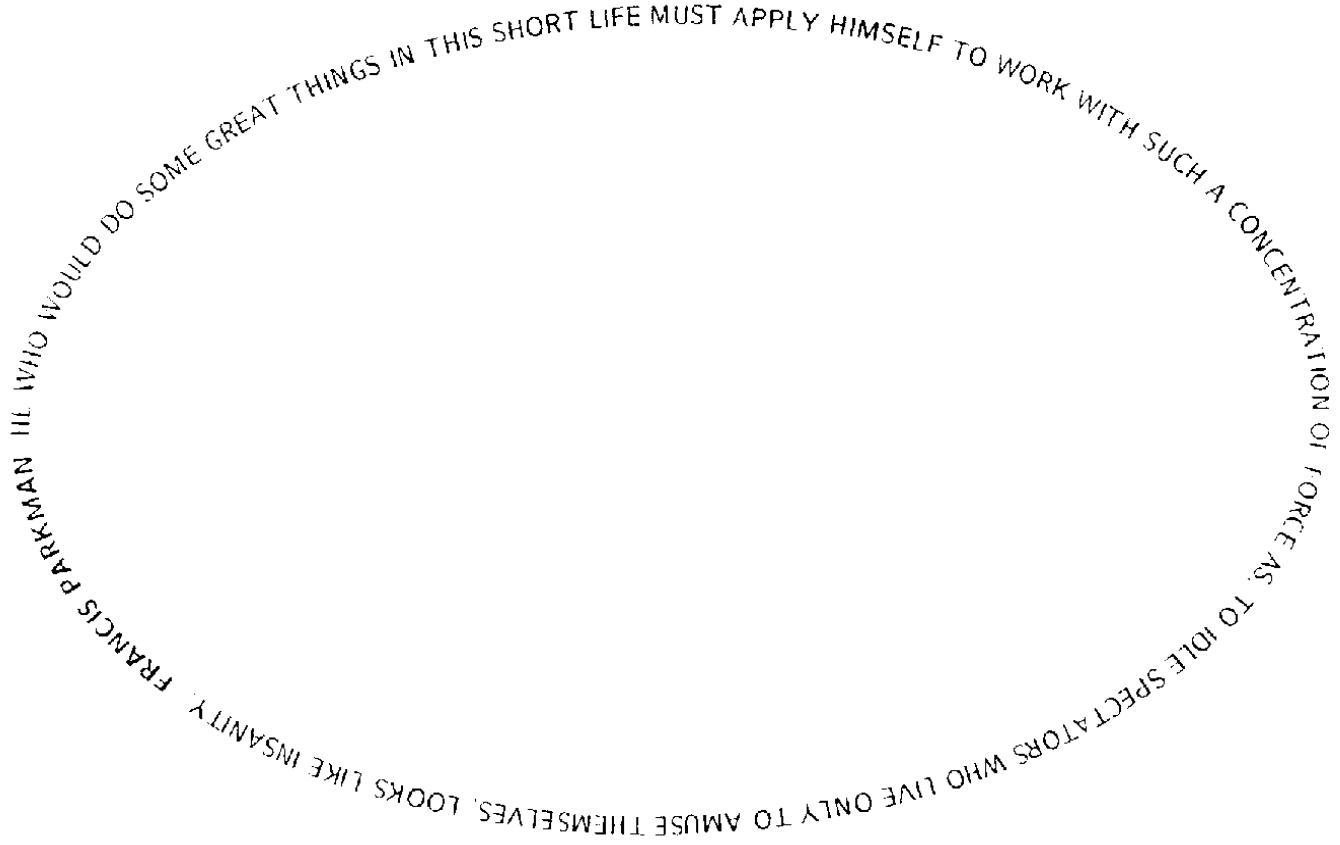


Figure 25: *Typesetting along a golden ellipse.*

Metafont rotates the type by modifying an underlying transformation called

`currenttransform`

which is a hook that Knuth provided for the purpose of doing last minute things to letters. It is helpful of thinking of a transformation as something that changes the shape, position, or orientation of a curve in certain allowable ways. You won't be surprised to learn that we can shift a curve up or down in position, and that we can rotate it, but we may also do a few other things such as

- uniformly magnify a curve;
- magnify a curve in the x- or y-directions separately;
- skew a figure; and
- perform any number of these all at once.

Transformations

- uniformly magnify a curve;
- magnify a curve in the x- or y-directions separately;
- skew a figure; and
- perform any number of these all at once.

Figure 26: *Legal transformations in* Metafont.

As long as we are rotating letters, perhaps we can apply other of these transformations to the letters of our text. For example, in figure 27 we have requested that Metafont shrink each letter at the same time it applies the rotation.

In the next figure, figure 28, you see what happens when we apply a transformation that does *not* include a rotation. Each letter has been skewed up or down so it conforms to the profile of the 'sunrise' curve. Actually, the base of each letter is straight, and coincides with an inscribed chord of the curve. (Transformations which apply curvature like that are generally not allowed in Metafont.) As long as the individual characters are so much smaller than the dimensions of the curve, viewers don't seem to mind this innocent deception.

5.6 Non-Linear Transformations in Metafont

It may be that Metafont is capable of making some limited non-linear transformations after all. I'd like to digress yet again to discuss this possibility.

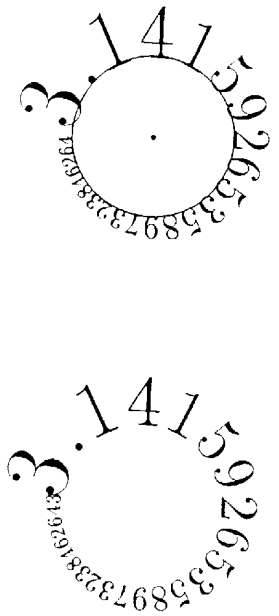


Figure 27: *A new representation for* π.

A Metafont transformation is a collection of six constants which Metafont uses in specified ways to make the transformation. In figure 30, which I'll show presently, I needed to specify a transformation whose six constants were not constant but depended instead on the horizontal distance. For any given column of pixels, that meant that the transform was constant for all the pixels in that column, but would change a bit when we proceeded to the next column.

Figure 28: *Along a* Metafont *sunrise.*

I stored the original, untransformed object as a Metafont picture and I made the transformations on a pixel-by-pixel level. The transformed image is stored in yet another picture variable.

1. Update transformation for new column.
2. Pixel by pixel, find out where 'ink' starts and stops.
3. Find the transforms for 'start' and 'stop' values.
4. Continue on that column.
5. Next column.

Figure 29: *Performing non-linear transformations.*

1. Prepare for examining a new column by updating the transformation for that column.
2. Start examining the column from the bottom up. Look for the first pixel in that column that is black. Keep examining pixels until the pixels are clear. At this point, we have a pair of numbers, the first of which records the point at which the pixels are turned on, and the second the point at which they are turned off.
3. Find the transforms of those values. In the image picture, turn on the pixels for that column between these two values.
4. Continue analyzing that column.
5. Continue on to the next columns.

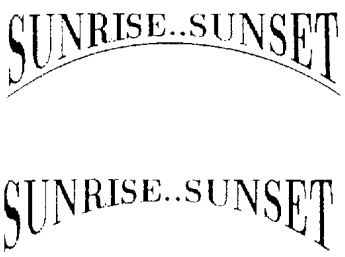


Figure 30: *Nonlinear* Metafont *transformations.*

Figure 30 shows the result of one non-linear transformation. Metafont is not happy at working at the pixel level; it takes about five minutes to generate these images on a 386 PC. (Normally, Metafont could create comparable text in a matter of seconds.)

If you look carefully, you'll see that the boundaries of the transformed letters are more jagged than you expect. This is work in progress, and I hope that this embarrassing state of affairs can be easily corrected.

5.7 An Application for Curved Typesetting

The only real application I can think of for curvilinear typesetting is for setting text around the circumference of a circular university or institutional seal. Despite the simplicity of the path—a mere circle, compared to some of the non-standard curves we have looked at—there are a number of interesting problems that need to be solved in connection with this application.

1. Need to center text.
2. Need to alter kern values.
3. Need to set bottom part along a concave path!

Figure 31: *Problems for seals.*

- The text needs to be *centered* with respect to the circumference. That is, it must be properly positioned along the seal.
- Unlike "standard" curvilinear typesetting, we explicitly *do* need to include the kerns between characters. The text of the seal might not look right otherwise. Furthermore, we may need to be able to alter some of the kerns between characters (because

after all, type may look different along a circle than along a straight line), and we need a simple way to make this alteration.

- Text along a seal almost always comes in two parts. The top part along the top of the seal is typeset along a convex path, the circle itself. But there is a bottom portion, and this is set along the inside of a larger circle. This text must be set along a non-convex path, and so we must address and deal with the issue of non-convex typesetting to successfully typeset the seal.

The centering is relatively easily done, since we can pretend to set the type at 9:00, and measure the angle of the ray connecting the final letter with the circle's origin. Half this angle is the angular offset we use to recalibrate the rotation angles of the letters, which we do before creating the characters.

We handle kern information by storing the kern information in a third file in the initial, measuring process. In the second step, Metafont reads this information in and uses it to modify the width information of the characters. Of course the kern file is just an Ascii file, and we can edit any of the values if some fine tuning of the kerns is needed.

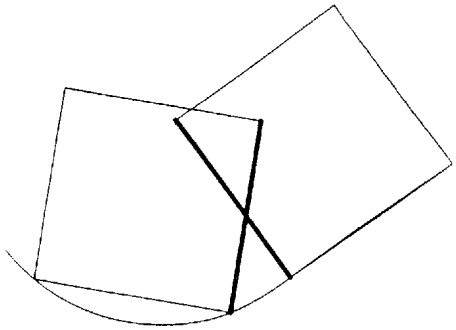


Figure 32: *Where to put neighboring letters on inscribed text.*

Finally, we handle the typesetting of the bottom portion of the text of the seal as another application of Metafont's handy `solve` macro. Progress along any Metafont path can be measured using a so-called time parameter t. In any pair of characters, we imagine drawing a line along the right edge of the left member of the pair, and another along the left edge of the right character. `solve` can find the proper value of t to use when placing the right-hand character so the two lines intersect at the tops of the types.

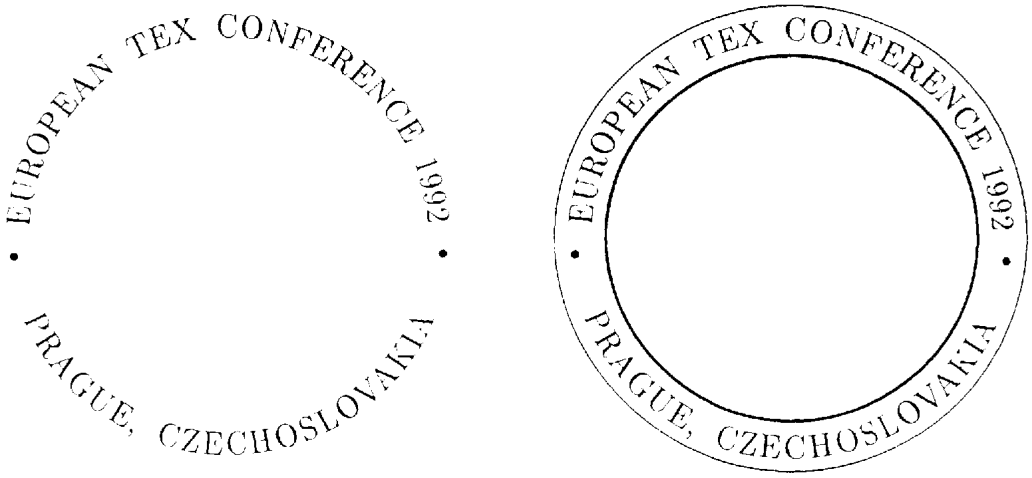


Figure 33: *A possible letterhead logo?*

Figure 33 shows an example of this "logo" typesetting which might be appropriate for a conference.

Generation of letterforms by mathematical means was first tried in the fifteenth century; it became popular in the sixteenth and seventeenth centuries; and it was abandoned (for good reasons) during the eighteenth century. Perhaps the twen- tieth century will turn out to be the right time for this idea to make a comeback, now that mathematics has advanced and computers are able to do the calculations. Modern printing equipment based on raster lines—in which metal "type" has been replaced by purely combinatorial patterns of zeroes and ones that specify the desired position of ink in a discrete way—makes mathematics and computer science increasingly relevant to printing. We now have the ability to give a completely precise definition of letter shapes that will produce essentially equivalent results on all raster-based machines. Moreover, the shapes can be defined in terms of variable parameters, making it possible for designers to perform valuable experiments that were previously unthinkable. (This is drawn from the preface to *The Metafont-book* of Donald E. Knuth.)

Figure 34: *Embedding an irregular figure in text.*

6 What Else?

There are three additional applications of TEX-Metafont communication that I can think of. Are there any more? I feel sure there are plenty.

Near the centre of the State of New-York lies an extensive district of country, whose surface is a succession of hills and dales, or, to speak with greater deference to geographical definitions, of mountains and valleys. It is among these hills that the Delaware takes its rise; and flowing from the limpid lakes and thousand springs of this regions, the numerous sources of the Susquehanna meander through the valleys, until, uniting their streams, they form one of the proudest rivers of the United States. The mountains are generally arable to the tops, although instances are not wanting, where the sides are jutted with rocks, that aid greatly in giving to the country that romantic and picturesque character which it so eminently possesses. The vales are narrow, rich, and cultivated; with a stream uniformly winding through each. Beautiful and thriving villages are found interspersed along the margins of the small lakes, or situated at those points of the streams which are favorable to manufacturing; and neat and comfortable farms, with every indication of wealth about them, are scattered profusely through the vales, and event to the mountain tops. Roads diverge in every direction, from the even and graceful bottoms of the valleys, to the most rugged and intricate passes of the hills. Academies, and minor edifices of learning, meet the eye of the stranger, at every few miles, as he winds his way through this uneven territory; and places for the worship of God, abound with that frequency which flows from unfettered liberty of conscience. (From *The Pioneers* of James Fenimore Cooper.)

Figure 35: *Embedding the State of New-York.*

The first, about which I will say very little, is an application of data display using TEX. I hope to talk about it at a conference like this in a year or two.

The second was suggested to me in conversation with Yannis Haralambous. I might like to embed some irregularly shaped object within text, as in figures 34 and 35. I pass to Metafont the coordinates of the outline of the object. Metafont can measure the points at which horizontal baselines intersect the path, and obligingly pass that information back to TEX. TEX can use that information in a `\parshape` command to create the properly shaped paragraph.

Yannis and I have begun talking about yet a third application of these principles, and that is toward typesetting in Korean. The bad news is that there are over 2000 "letters" in the Korean "alphabet," far too many for any single font. The good news is that these letters—actually syllables—are composed from less than 30 separate components. Each component represents a sound in the Korean language. Each component can be coded as a Metafont macro. We are optimistic that the source file can pass to Metafont the order in which the sounds make up the syllable. Syllables appear to be built up from the phonemic components in straightforward ways.

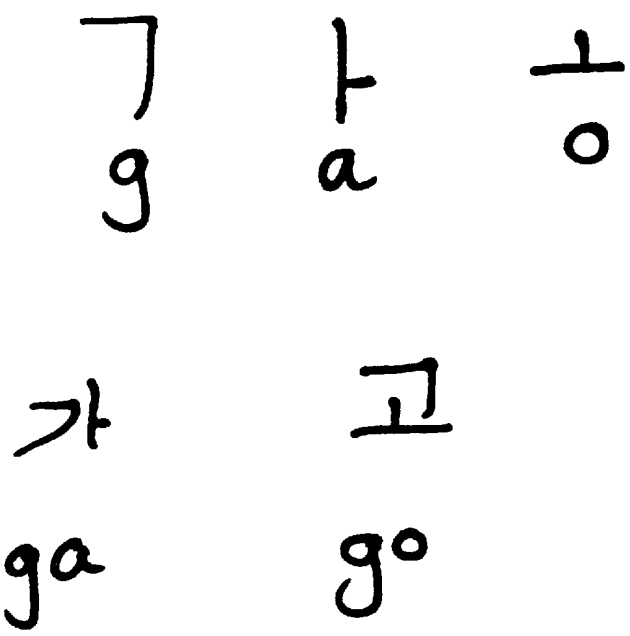


Figure 36: *Creating Korean type.*

Then Metafont can create a special purpose font for the text. There is room for 256 characters in a font, which is roughly equivalent to a page of text in a typical book.

We expect, therefore, that it will take somewhere on the order of 200 special fonts to typeset a typical novel.

7 Conclusion

I have discussed a couple of ways in which when TEX and Metafont work together, the results are more than either could do singly. *That* is the point to stress, rather than the applications themselves. I hope listeners will take this lesson to heart and develop their own TEX-Metafont applications.

Getallen

David van Leeuwen

david@rulgm0.leidenuniv.nl

Inleiding

Een groot priemgetal is miljardtweehonderdvierendertigmiljoenvijfhonderdzevenenzestigduizendnegenhonderdnegenenveertig, en dat is op zich niet zo bijzonder, zij het dat ik het als `\getal 1234567949` heb ingetikt. Getallen uitspellen in het Nederlands is mogelijk, dankzij de 'logische' opzet. Natuurlijk zijn getallen tot veertien uitzondering (zoals in veel talen getallen onder de twintig), kennelijk gebruiken we die getallen vaak, en beneden de honderd hebben we die vreemde omkering van tiental en eenheid, telefoonnummers moeten we dan ook *nooit* groeperen in twee cijfers, want dat kan tot verwarring leiden. Maar voor grotere getallen groeperen we netjes in drie cijfers (die we ook eigenlijk in de notatie zouden moeten schrijven, bij voorkeur met *thinspaces*, als in 1 234 567 949), en tellen we duizendtallen, miljoenen, miljarden, etcetera.

De code

Laten we nu eens naar de code krijgen die ik in bovenstaand voorbeeldje heb gebruikt. Eerst definiëren we wat tellertjes en een slik-in macro, die er voor zorgt dat ik `\getal` kan gebruiken als plain TEX's `\number`:

```
\newcount\i\newcount\j\newif\ifhyphen              % hulp tellertjes
\def\getal{\hyphenfalse\afterassignment\nul\i=}    % slik in macro
\def\nul{\ifnum\i=0 nul\else\getall\fi}            % nul lijkt bijzonder
```

Het te kraken getal staat nu in `\i` (dit is nu even geen ı!) Het probleem is verlegd naar `\getall`, en het vervelende getal 0 is al behandeld. Als we weten dat `\i` een 1000-tal bevat, kunnen we dit 1000-tal uitschrijven, met 'duizend' erachter. Het probleem is dan verlegd tot het uitschrijven van een getal onder de duizend. Natuurlijk geldt dit ook voor 1 000 000-tallen, en zelfs voor 100-tallen. Daarom introduceren we de macro `\num` ⟨tal⟩ ⟨schrijfwijze⟩, die een ⟨tal⟩-tal afsplitst van het (hulp)getal `\j` (= `\i`), dat uitschrijft, daar ⟨schrijfwijze⟩ achteraanzet, en de rest als probleem `\i` achterlaat:

```
\def\num#1 #2 {\divide\j by #1 \ifnum\j>1 {\i=\j\getall}\fi
  #2\multiply\j by #1 \advance\i by -\j\getall}
```

Hier zie je een voorbeeld van het gebruik van locale registers met TEX, binnen de haken {...} moeten we een deelprobleem oplossen, en veranderen tijdelijk de waarde van `\i`. Een ander probleem bij het uitschrijven van getallen is dat we van die lange woorden krijgen, we moeten ze dan zeer waarschijnlijk afbreken. Voor de veiligheid zetten we `\-` tussen elke lettergreep, maar dit mag niet als eerste, we hebben een vlag nodig die ons vertelt of we de eerste lettergreep al gezet hebben. De macro `\?` zet mogelijk een afbreukstreepje

```
\def\?{\ifhyphen\-\else\global\hyphentrue\fi}          % mag afbreukstreepje?
```

Dan nu de macro `\getall`. Bij aankomst staat het uit te schrijven getal in het register `\i`. We beginnen met de kleinste getallen:

```
\def\getall{\ifcase\i\or \?een\or \?twee\or \?drie\or \?vier\or \?vijf\or
  \?zes\or \?ze\-ven\or \?acht\or \?ne\-gen\or \?tien\or \?elf\or \?twaalf\or
  \?der\-tien\or \?veer\-tien\else \groterdanveertien \fi}
```

Wel, dat was eenvoudig maar saai. Vervolgens moeten we getallen beneden de honderd behandelen. De tigheid is voldoende onregelmatig om hem helemaal uit te schrijven. 'Omkeren' gebeurt met een `\edef`. Tenslotte moeten eenheid en tigheid aan elkaar geplakt worden, met uitzonderingen voor twee, drie en tien. (Neem aan dat " een goede trema-macro voorstelt.)

```
\def\groterdanveertien{\j=\i          % hulp register
 \ifnum\i<100 \divide\j by 10         % \j is tiental
  \edef\tig{\ifcase\j\or tien\or twin\-tig\or
    der\-tig\or veer\-tig\or vijf\-tig\or zes\-tig\or ze\-ven\-tig\or
    tach\-tig\or ne\-gen\-tig\fi}     % en \tig is het uitgeschreven tiental
  \multiply\j by 10 \advance\i by-\j\getall  % \i is de rest, de eenheid, <10
  \ifnum\j=10 \else\ifcase\i\or\-en\or"en\or"en\else\-en\fi\fi\?\tig % 2,3->"
 \else\groterdanhonderd\fi}
```

Wel, als we dit allemaal kunnen, kunnen we de macro `\num` gebruiken om de duizendtallen en zo uit te schrijven:

```
\def\groterdanhonderd{\ifnum\i<1000 \num100 \?hon\-derd   % 100<=\i<1000
 \else\ifnum\i<1100 \num1000 \?dui\-zend % 1000<=\i<1100, ‘duizend...’
  \else\ifnum\i<2000 \num100 \?hon\-derd % 1100<=\i<2000, ‘...tienhonderd...’
   \else\ifnum\i<1000000 \num1000 \?dui\-zend             % 1000<=\i<1000000
    \else\ifnum\i<1000000000 \num1000000 \?mil\-joen
     \else\num1000000000 \?mil\-jard
 \fi\fi\fi\fi\fi}
```

Zo kunnen we op een eenvoudige wijze grote getallen opschrijven, die we bijvoorbeeld hebben uitgerekend. Dit voorkomt fouten bij overtikken.

Een voorbeeld

Fibonacci getallen f_n zijn te vinden door $f_n = f_{n-1} + f_{n-2}$, en de eerste paar zijn in het Nederlands uitgeschreven achtereenvolgens:

0: nul; 1: een; 1: een; 2: twee; 3: drie; 5: vijf; 8: acht; 13: dertien; 21: eenentwintig; 34: vierendertig; 55: vijfenvijftig; 89: negenentachtig; 144: honderdvierenveertig; 233: tweehonderddrieëndertig; 377: driehonderdzevenenzeventig; 610: zeshonderdtien; 987: negenhonderdzevenentachtig; 1597: vijftienhonderdzevenennegentig; 2584: tweeduizendvijfhonderdvierentachtig; 4181: vierduizendhonderdeenentachtig; 6765: zesduizendzevenhonderdvijfenzestig; 10946: tienduizendnegenhonderdzesenveertig; 17711: zeventienduizendzevenhonderdelf; 28657: achtentwintigduizendzeshonderdzevenenvijftig; 46368: zesenveertigduizenddriehonderdachtenzestig; 75025: vijfenzeventigduizendvijfentwintig; 121393: honderdeenentwintigduizenddriehonderddrieënnegentig; 196418: honderdzesennegentigduizendvierhonderdachttien; 317811: driehonderdzeventienduizendachthonderdelf; 514229: vijfhonderdveertienduizendtweehonderdnegenentwintig; 832040: achthonderdtweeëndertigduizendveertig; 1346269: miljoendriehonderdzesenveertigduizendtweehonderdnegenenzestig; 2178309: tweemiljoenhonderdachtenzeventigduizenddriehonderdnegen; 3524578: driemiljoenvijfhonderdvierentwintigduizendvijfhonderdachtenzeventig; 5702887: vijfmiljoenzevenhonderdtweeduizendachthonderdzevenentachtig; 9227465: negenmiljoentweehonderdzevenentwintigduizendvierhonderdvijfenzestig; 14930352: veertienmiljoennegenhonderddertigduizenddriehonderdtweeënvijftig; 24157817: vierentwintigmiljoenhonderdzevenenvijftigduizendachthonderdzeventien; 39088169: negenendertigmiljoenachtentachtigduizendhonderdnegenenzestig; 63245986: drieënzestigmiljoentweehonderdvijfenveertigduizendnegenhonderdzesentachtig; 102334155: honderdtweemiljoendriehonderdvierendertigduizendhonderdvijfenvijftig; 165580141: honderdvijfenzestigmiljoenvijfhonderdtachtigduizendhonderdeenenveertig; 267914296: tweehonderdzevenenzestigmiljoennegenhonderdveertienduizendtweehonderdzesennegentig; 433494437: vierhonderddrieëndertigmiljoenvierhonderdvierennegentigduizendvierhonderdzevenendertig; enzovoorts.

International quotations

Johannes Braams

j.l.braams@research.ptt.nl

Introduction

At the conference in Portland last year, Christina Thiele asked me if I was willing to write an article for ttn about quotation marks. She said that she suspected that there would be many people out there who would want to know how to produce quotation marks in a little piece of, say, french, text that they include in their document that is otherwise written in english. All this is of course covered in the babel language specific style files, but for those who don't want to use babel but do need the occasional quotation mark I wrote this article.

Let me start with a disclaimer: I am not familiar with all the typographic conventions that are in use in the various countries all over the world. Also I can only give you macros for those languages for which a language specific babel file exists. So, there may be more conventions. If you know of them: please inform me, so that I can enlarge my collection.

Yet another disclaimer: If you have access to fonts with the EC (or Cork) encoding, you do not need most of the hackery in this article. These fonts include the characters that are needed so they need not be constructed.

The first part of this article[1] deals with the macros needed to typeset the quotation marks needed for various languages. First I'll introduce some internal hackery, then the macros that really produce quotation marks. I also give some shorthands to make the macros easy to use. Finally I will give some examples of the use of the macros. In the examples I will show the source text as well as the typeset result.

Some help macros

Because the macros in this part of the code are not intended for use in a document we'd like to 'hide' them from the user. Usually this is done by changing the category code of an 'other' character to letter. More often than not the '@' is chosen for this purpose. Sometimes the underscore is used for this purpose as well. Here I too will use the '_'. Of course the category code needs to be reset later so we store the current value in `\uscatcode`.

```
\chardef\uscatcode=\catcode‘\_
\catcode‘\_=11\relax
```

In some languages we need to lower quotation marks to the baseline. For this purpose we use the macro `\set_low_box`. It has one argument and uses `\box0` to gives us its result. This macro comes from `german.tex`.

It first typesets a comma in box register 2 and its argument in box register 1. Then it computes the distance that the argument has to be lowered to reach the baseline. Finally it lowers the contents of box register 0 — using box register 0 again for the result — and adjusts the values for the height and depth of the box.

```
\def\set_low_box#1{\setbox2\hbox{,}%
  \setbox0\hbox{#1}%
  \dimen0\ht0 \advance\dimen0 -\ht2%
  \setbox0\hbox{\lower\dimen0 \box0}%
  \ht0\ht2 \dp0\dp2}
```

Using macros for quotation marks sometimes disturbes the spacefactor. Therefore we also need a macro to preserve it. This macro also stems from `german.tex`. It first checks if it is executed in horizontal mode, if that is the case the current spacefactor is stored in the macro `\_SF`. Outside horizontal mode the macro `\_SF` is empty. Then `\save_sf_q` typesets its argument and resets the spacefactor.

```
\def\save_sf_q#1{{\ifhmode
  \edef\_SF{\spacefactor\the\spacefactor}
  \else
  \let\_SF\empty \fi \leavevmode #1\_SF}}
```

Producing the quotation marks

In languages such as Dutch, German and Czech, the opening quotes are traditionally typeset at the baseline.

```
\def\loq{\protect\_loq}
\def\_loq{\save_sf_q{\set_low_box{‘’}%
\box0\kern-.04em}}
```

In Germany also the closing quotes are different from what is provided by TEX. They use something that looks like the english opening quotes as closing quotes. Obviously, if one didn't do anything about it, the spacing would be wrong. Therefore we need yet another macro, `\icq`.

```
\def\icq{\protect\_icq}
\def\_icq{\save_sf_q{%
            \kern-.07em‘‘\kern.07em}}
```

[1] When the source of this article is run through LATEX it produces a file called `quoting.tex` which contains all the code that is described and used in the article.

In french typography, a very different kind of quotes is used, the so called ‘guillemets’. To realise these guillemets, various macros have been floating around the net. According to the `french.sty` package by Bernard Gaulle, one of the oldest definitions is:

```
\def\oog{\protect\_oog}
\def\_oog{\leavevmode\ifdim\lastskip>0pt%
          \unskip\penalty-9\hskip0.35em %
                          minus 0.35em\fi
          \raise .27ex\hbox{%
              $\scriptscriptstyle\ll$}%
          $\,$\nobreak\ignorespaces}
\def\ocg{\protect\_ocg}
\def\_ocg{\leavevmode\ifdim\lastskip>0pt%
          \unskip\penalty10000\fi
          \nobreak$\,$\leavevmode
          \raise .27ex\hbox{%
          $\scriptscriptstyle\gg$}}
```

But, this does not *really* give the result that the french would like. Therefore, if you have access to the LaTeX-symbol fonts it is better to use the following definition for the guillemets:

```
\chardef\lg='050
\chardef\rg='051
\def\og{\protect\_og}
\def\_og{\hbox{\ly\lg\kern-0.2em\lg%
                       \kern+0.2em}}
\def\cg{\protect\_cg}
\def\_cg{\hbox{\ly\kern+0.2em\rg%
                    \kern-0.2em\rg}}
```

In the code above the control sequence `\ly` is used. It is an internal macro from the old font selection schem in LaTeX. When you use the nfss you will have to define it:

```
\ifx\undefined\selectfont
\else
  \def\ly{\fontfamily{lasy}\fontseries{m}
          \fontshape{n}\selectfont}
\fi
```

Easy usage

In the previous section a couple of macros have been defined to make it possible to use various quotes. But it would be nice if one did’t have to do so much typing each time they are used. To provide easy acces, it is common use in language specific files to introduce active characters. For the macros presented here, we could introduce three active characters, the `"`, `<` and the `>`.

```
\def\dq{"}\catcode‘\"=\active
\def\lt{<}\catcode‘\<=\active
\def\gt{>}\catcode‘\>=\active
```

As you may have noticed, I saved a copy of the non active version of each character in a control sequence. These are needed later on, when the active character has inspected its argument and decides that it needs to insert the non-active version of itself.

We use the active `"` to acces the low opening quotes and the german closing quotes, the other two are used to produce the guillemets. Here is the definition of the active characters:

```
\def"#1{\ifx#1‘\loq{}\else
        \ifx#1’\icq{}\else\dq#1
        \fi\fi}
\def<#1{\ifx#1<\ifmmode\lt\lt\else%
                    \og{}\fi\else\lt#1\fi}
\def>#1{\ifx#1>\ifmmode\gt\gt\else%
                    \cg{}\fi\else\gt#1\fi}
```

But, be carefull when introducing new active characters. You have to make sure that they get deactivated at the right moment. Therefor we need to add them to macros such as `\dospecials` and — in case you use LaTeX — `\@sanitize`. A safe way of doing this was found by Bernd Raichle. It involves using an extra macro `\add_special`.

```
\chardef\atcatcode=\catcode‘\@
\catcode‘\@=11\relax
\def\add_special#1{\begingroup
    \def\do{\noexpand\do\noexpand}
    \def\@makeother{%
        \noexpand\@makeother\noexpand}
    \edef\x{\endgroup
      \def\noexpand\dospecials{%
                    \dospecials\do#1}
      \expandafter\ifx\csname%
              @sanitize\endcsname\relax
                  \else
        \def\noexpand\@sanitize{%
               \@sanitize\@makeother#1}
      \fi}
    \x}
\catcode‘\@=\atcatcode\relax
```

Once that macro is defined we use it to tell TeX to treat our active characters with caution.

```
\add_special\"
\add_special\<
\add_special\>
```

Wrapping up

These macros were defined while the category code of the ‘_’ was changed. It must not be forgotten to ‘undo’ that change:

```
\catcode‘\_=\uscatcode\relax
```

Examples

A quotation in Dutch might look like this:

```
Hij zei: "‘Ga je mee?’’.
```
Hij zei: „Ga je mee?”.

But in French (using the definition that uses the LaTeX symbol font) it would be:

```
Il disait: << Tu va? >>.
```
Il disait: « Tu va? ».

Whereas in Italian it might look like:

```
Lui dice: >>Andiamo?<<.
```
Lui dice: »Andiamo? «.

To show the difference between the two definitions given for the guillemets, this is wat the ‘old’ version looks like:

```
Il disait: \oog\ Tu va? \ocg.
```
Il disait: ≪ Tu va?≫.

Typesetting number sequences

FIFO and some more . . .

Kees van der Laan

Hunzeweg 57
9893 PB Garnwerd, The Netherlands
cgl@rug.nl

Abstract

Typesetting sequences of numerical values, represented via symbolic names which get their values on the fly, is dealt with. The sorting of the sequence is done by a linear sorting algorithm, of complexity $O(n^2)$. Three or more consecutive numbers are typeset as a range.
The objective was to encode typesetting sequences of numbers as simple, concise, general, compatible, modular, orthogonal, and . . . , as possible in TEX.[1]

Keywords: Typesetting sequences, citation lists, lists of references, linear sorting, FIFO, plain TEX, macro writing, education.

1 Introduction

To think about typesetting sequences of numerical values looks a bit peculiar. I was pulled in this direction when thinking about the 'range notation' problem, which was posed at the tex-nl@hearn discussion list.[2] The idea is that for example the numbers 1, 2, 3 should appear in print as 1–3. This is a trifle when the numbers are known a priori. When symbolic names are used, which get their numerical values on the fly, one has to resort to macros. The macros must take care of ordering the sequence and proper typesetting the numbers.

For that purpose the macro `\typseq`—mnemonics: typeset sequence—was written.

There are various choices possible with respect to the TEX implementation of symbolic names for numbers. The most direct form is `\chardef\`$\langle symbolic\ name\rangle = \langle number\rangle$.[3]

The problem was solved in the Polya, 1957, way. First the kernel problem of typesetting an ordered sequence of numbers in range notation was considered. Collateral the independent problem of ordering a sequence had to be solved.[4] And finally the merging of both aspects and the TEX encoding.

The intended scope of readers consists of those who not only favor the use of LATEX and TEX, but also like to understand what is going on, and otherwise strive after keeping the encoding simple and concise.

Notations

`\ea`, `\nx`, and `\ag`, are used as shorthand for `\expandafter`, `\noexpand`, respectively `\aftergroup`.

2 Example of use

If we have:[5]

```
\chardef\dekker=5, \chardef\forsythe=3,
\chardef\reinsch=4, \chardef\knuth=11,
```

then

```
\typseq{\dekker,\knuth,\forsythe,\reinsch}
```

yields [3–5, 11].

3 Stepping stone

Suppose we have a non-descending sequence of numbers and we like to typeset these in range notation. For example 1, 2, 5, 7, 8, 9, 10 as [1, 2, 5, 7–10].

My solution is the invocation

```
\cpr{1,2,5,7,8,9,10},
```

backboned by[6]

[1] Not blurred by safeguarding goodies or limited by a particular application.

[2] Not dealt with is the typesetting of large amount of data via tables or graphs, with or without statistical methods, or the typesetting of encoded data like van Wijngaarden's method to typeset millions of prime numbers on an A4 or two.

[3] This is restricted by 256. If needed one can use `\def\`$\langle symbolic\ name\rangle\{\ldots\}$, or `\mathchardef`. . . .

[4] For sorting general sequences in TEX, via $O(n \log n)$ algorithms, see van der Laan, 1993.

[5] That these names stand for incomparable outstanding (numerical) mathematicians is a mere coincidence.

[6] Declarations are omitted. For FIFO see van der Laan, 1992b, or the listing of the file typseq.tex included below.

```
\def\cpr#1{{\def\process{\processc}\bs
 \fifo#1,\ofif,\prtfl\es}}
\def\processc#1{%
           \init{#1}\def\processc##1%
 {\ifnum##1=\lst\else\ifnum##1=\slst
  \lst=\slst\advance\slst1{}\else
  \prtfl\sepn\init{##1}\fi\fi}}
\def\init#1{\frst=#1\lst=#1\slst=#1{}%
   \advance\slst1 }
\def\prtfl{\the\frst\ifnum\frst<\lst
 \advance\frst1{}\ifnum\frst=\lst\sepn
 \else\nobreak--\nobreak\fi\the\lst\fi}
\def\bs{[}\def\es{]}\def\sepn{, }
```

Explanation

`\cpr` This is an independent macro for just typesetting a sequence in range notation.

`\processc` The encoding makes use of the FIFO paradigm[7] to process each element. In order to perform the appropriate action we must look ahead, or postpone the typesetting, both to an unknown depth. It seems natural to postpone *as long as needed* the typesetting, with the first and last elements of the range so far—not necessarily different—stored in the counter variables `\frst`, respectively `\lst`.
The mechanism of redefinition is used to account for initialization. The first definition of `\processc` invokes `\init`—to give the counter variables `\frst` and `\lst` the value of the current list element—followed by a *redefinition* of `\processc`.[8] In the latter redefinition an element is skipped when it equals[9] the previous one. If it equals the successor of `\lst` then `\lst` gets the value of its successor. Otherwise 'the range' is typeset followed by the value of the separator, `\sepn`. The typesetting is handled by the macro `\prtfl`. The degenerate case, `\frst`=`\lst`, yields a single number. After having typeset the range we are in a pseudo initial state. `\frst` (and `\lst`) must get the value of the list element at hand. This is done again via the invocation of `\init`.

At the end of the list we must typeset appropriately the values of `\frst` and `\lst`. This is done via the invocation of `\prtfl`.

If we look at sorting as a CISO—Collective-In-Smallest-Out—process, then it is rather straightforward to combine sorting with `\processc`, because `\processc` handles the range typesetting independent of how the successive arguments are obtained.[10]

4 The macro \typseq

Purpose. The purpose of the macro `\typseq` is to typeset automatically a sequence in ascending order in range notation.

Input. The argument of `\typseq` is the sequence of *symbolic* names—representing the numerical values—separated by commas.

Result. The values of the sequence items are typeset in ascending order in range notation, separated by the value of `\sepn`, and delimited by the values of `\bs`, respectively `\es`.

Design. My encoding of automatic typesetting sequences of numbers comes down to

- Store the input sequence as a list with active list separators,
- Sort the list via appropriate definition of the active list separator,
- Typeset the list appropriately.

The file typseq.tex

```
%Shorthands
\let\ag=\aftergroup
\let\ea=\expandafter\let\nx=\noexpand
%Counters
\newcount\frst%First value of range
\newcount\lst %Last value of range
\newcount\slst%Successor \lst
%Newif-s
\newif\ifnoe%   Mnemonics: if not empty
%Parameters: separators
\def\sepn{, }%  Number separator
%Parameters: brackets
\def\bs{[}\def\es{]}
%FIFO with comma as separator
\def\fifo#1,{\ifx\ofif#1\ofif\fi%
 \process{#1}\fifo}\def\ofif#1\fifo{\fi}
%Store, sort and typeset
\def\typseq#1{{\strseqaslst{#1}\bs\srt\prtfl%
 \es}}% Local scope because of redef-s.
%Store sequence as list
\def\strseqaslst#1{\let\process=\processs
\xdef\list{\fifo#1,\ofif,}}
%ProcessS stores consecutive elements
%(preceded and) separated by \ls as a list.
\def\processs#1{\nx\ls\nx#1}
%Mod from Syntactic Sugar, MAPS92.1, p135
\def\srt{% Assumed is \list contains
%        symbolic names separated by \ls.
 \loop\ifx\empty\list\noefalse%
      \else\noetrue\fi%
%Test for NOEmpty list.
 \ifnoe \first\list%  \min=first element
```

[7] For FIFO and especially (variant) TEX encodings of the principle, see van der Laan, 1992b.

[8] This mechanism is general and elegant for coping with begin situations, where the first action is the only one different from the rest.

[9] For an ordered sequence 'less than' could be used. 'Equals' allows for non-ordered sequences too: just numbers which form a range are compressed.

[10] This prompts another approach for solving the problem: maintaining a priority queue. My case rests.

```
      \list%  Find minimum and store \min
      \processc\min%  Typset\min
%Delete minima from \list.
  {\def\ls####1{\ifx####1\min\else%
   \nx\ls\nx####1\fi}\xdef\list{\list}}%
 \repeat}%   end \srt
%List Separator.
\def\ls#1{\ifnum#1<\min\let\min=#1{}\fi}
%Pop up first element of list #1
\def\first#1{\def\lop\ls##1##2\pol{%
   \let\min=##1{}}\ea\lop#1\pol}

%Compressing sequences; it is assumed
%that elements are separated by commas.
\def\cpr#1{{\def\process{\processc}\bs%
 \fifo#1,\ofif,\prtfl\es}}
%Typeset element or keep track of range
\def\processc#1{\init{#1}\def\processc##1%
 {\ifnum\lst=##1{}\else\ifnum\slst=##1%
  \lst=\slst\advance\slst1{}\else%
  \prtfl\sepn\init{##1}\fi\fi}}
\def\init#1{\frst=#1\lst=#1\slst=#1{}%
   \advance\slst1{}}
%Print range: \frst-\lst (or \lst).
\def\prtfl{\the\frst\ifnum\frst<\lst{}%
 \advance\frst1{}\ifnum\frst=\lst\sepn%
 \else\nobreak--\nobreak\fi\the\lst\fi}
\endinput            %dec 92; cgl@rug.nl
%Test/example program. \tracingmacros=1
%Typeset sequence: \chardef\a=1
\chardef\b=27\chardef\c=134  %all <256!
\typseq{\c,\a,\b}.
\bye
```

Explanation

`\typseq` This composition macro invokes the macros for storing (`\strseqaslst`),[11] sorting (`\srt`), and typesetting (`\processc`).

`\strseqaslst` The arguments of the macro are the sequence elements separated by commas. First the data are stored in a list, via the use of the FIFO paradigm.[12] `\ls` is used as active list separator.

`\srt` We loop through the `\lists` until the `\list` is exhausted. In each step the minimum value is determined, typeset, and deleted from the list.[13]

- Finding the minimum.
  To initiate the process the first element is considered to be the minimum. Then the `\list` is invoked with the active list separator the function to find better minima.
- Delete minimum.
  The deletion of the minima from the list is TEX specific. We first redefine locally the list separator, `\ls`, with the function to delete the element if it equals the minimum value.[14] Then `\list` is `\xdef`-ed with `\list` as replacement text! This expands `\list` and provides in `\list` the non-minima, again separated by `\ls`-s.[15]
  Concise and elegant isn't it?
- Typesetting.
  We must account for the range notation—three or more consecutive numbers are represented as $\langle first\ number\rangle$–$\langle last\ number\rangle$—and other conventions like the separation symbol, and enclosing the total within square brackets. The latter have been parameterized into `\sepn`, `\bs`, respectively `\es`.

5 Variation

My earlier variant of `\storeseqaslst` made use of `\ag`, as follows

```
\def\strseqaslst#1{{\ag\def\ag\list\ag{%
 \let\process\processs\fifo#1,\ofif,}}}
%with
\def\processs#1{\ag\ls\ag#1}
```

When an application provides just numbers—for example page numbers after index items—the above `\strseqaslst` can be adapted to store these numbers in `\ls\1\ls\2\ls` ... `\ls\`$\langle n\rangle$, via

```
\def\strseqaslst#1{{\ag\def\ag\list\ag{%
 \let\process\processs\global\n0
 \fifo#1,\ofif,}}}
%with
\def\processs#1{\global\advance\n1
 \ea\xdef\csname\the\n\endcsname{#1}%
 \ag\ls\ea\ag\csname\the\n\endcsname}
```

The encoding of `\strseqaslst` can be made robust with respect to redundant spaces. I refrained from this at the moment.

`\strseqaslst` is superfluous when the convention is adopted to separate and precede the arguments of `\typseq` by `\ls`.

For other representations of the numbers, for example in superscript, one can redefine `\prtfl`. Penalties can be inserted in the replacement text of `\sepn`, to inhibit line breaks. The question is: How much?

6 Looking back

The data structure `\list` is peculiar. Not only obeys it to the queue access method, that is from left to right via the execution of `\list` with active list separator `\ls`, but individual elements can be accessed via their

[11] Mnemonics: store-sequence-as-list.

[12] This is in the spirit of the generation of n-stars, TEXbook Appendix D.1 p.374. No `\aftergroup` is necessary in this particular case, however.

[13] This has been published earlier and perhaps a bit hidden, in Syntactic Sugar, van der Laan, 1992a.

[14] Note that multiple occurrences of the minimum is accounted for.

[15] The four #-s in the definition of `\ls`—redefinition within `\srt`, line 36–37—look peculiar. We must account for the hidden definition of the loop `\body`, which makes that the definition of `\ls` is nested two levels deep!

names too. During the invocation of the example the data structure is modified as follows[16]

```
sequence :\d,\k,\f,\r
list     :\ls\d\ls\k\ls\f\ls\r ,
```

and gradually into `\list` equals `\empty` while sorting. Neat![17]

7 Epilogue

Typesetting of sequences on the fly occur when dealing with citation lists, or a list of references to figures and the like. LaTeX's `\cite` doesn't automatically order sequences or typeset ranges.

The range representation was posed as a problem at the tex-nl@hearn discussion list. A (LaTeX) style was offered in reply.[18] This style was composed by Ronald Kappert out of the work of Arseneau and Green. Kappert reported that Arseneau has improved upon the style. The improved style is available on file server niord.shsu.edu. According to Kappert it is in general use, especially by those who have long lists of citations. I myself don't use long citation lists. I am happy with the system of 'name followed by year.' As author, and also as reader, I know by heart the name and year of most of the works to refer to. This is structurally simple. It also makes the usual multi-pass processing of a list of references superfluous, because there is no information needed which will be created later. It is already there. However, in practice authors are restricted by journal conventions.

The hardest thing was not to introduce bells-and-whistles and to keep it as straight as possible. I refrained from introducing robustness with respect to arguments with unnecessary spaces.

Ronald Kappert is kindly acknowledged for his remarks and suggestions in proofing the article.

8 TeXniques used

- Creating a list of dynamical length via FIFO (and as alternative with `\aftergroup`).
- Sorting a list via repeated execution of `\xdef\list{ \list}`, and appropriate use of the active list separator.
- Handling recursion (loop) initialization, such that some action on first traversal is different from the action via the same name, on later traversals.

References

[1] Jeffreys, A (1990): Lists in TeX's mouth. *TUGboat* 11, no. (2), 237–244.

[2] Kappert, R (1992): scite.sty. (From the file server; it is compiled of work from D. Arseneau and I. Green: overcite.sty, drftcite.sty, cite.sty. Actually, I. Green made the style citesort.sty, which has been assimilated by D. Arseneau in his styles.)

[3] Knuth, D.E (1984): *The TeXbook*, Addison-Wesley.

[4] Laan, C.G. van der (1992a): Syntactic Sugar. MAPS92.2, 130–136. (Submitted TUG '93.)

[5] Laan, C.G. van der (1992b): FIFO & LIFO sing the BLUes. MAPS92.2, 139–144. (Submitted TUGboat.)

[6] Laan, C.G. van der (1992c): Tower of Hanoi, revisited. *TUGboat* 13, no. (1), 91–94. Also in MAPS92.1, 125–127.

[7] Laan, C.G. van der (1993): Sorting in BLUe, MAPS93.1. (Submitted TUG '93.)

[8] Lamport, L (1986): LaTeX, user's guide & reference manual. Addison-Wesley.

[9] Polya, G (1957): How to solve it. Anchor.

[16] Symbolic names abbreviated to their first letter.

[17] Instead of emptying the `\list` I could have maintained a property list, for example a permutation array. Another approach. My case rests.

[18] My code consists of the modules: storing, sorting, and typesetting in range notation (each $\approx$ 10 lines), next to the composition (1 line).

Sorting in BLUe

Kees van der Laan

Hunzeweg 57,
9893PB Garnwerd, The Netherlands
cgl@rug.nl

Abstract

Macros for number and lexicographic sorting are supplied. Data can originate from the copy, from file, or generated automatically. Lexicographic sorting allows words with ligatures and diacritical marks. Applications treated are: sorting with respect to report generation with TEX as a database tool, sorting and compressing `index.tex`, Knuth's index reminders file, and sorting control sequences separately.
It is illustrated by various examples that a set can be sorted within TEX once the ordering of the set is defined and encoded in a comparison macro, in compliance with the parameter macro `\cmp`.

Keywords: Sorting, index preparation, database handling, multiple sorting keys, macro writing, education.

Introduction

Sorting is a fundamental process. With respect to TEX, sorting was needed by Amy Hendrickson for sorting address labels [15], by Alan Jeffreys [16] and by Lincoln Durst [9] for sorting index items, to name but a few. Donald Arseneau, Ian Green, Ronald Kappert [19], and myself [25], have used sorting within TEX for citation lists. For aspects with respect to index generation see [8] and [31]. Available is Makeindex [6], [27], to cooperate with LATEX, and Salomon's plain TEX version of it [33].

All the sorting with respect to index items are external, outside of TEX.[1] This is practical, but sorting within plain is possible.[2] An advantage of TEX is that it allows for abstraction with respect to the kind of data.

Normally, number sorting and lexicographic sorting are done by different routines. This is necessary because the exchange and comparison are generally tied up with the data type. Within TEX the exchange is independent of the type, and the relational operator can be used as parameter by the sorting macro. Furthermore, secondary (and more) keys can be accounted for. The latter facility is not always available in the external sorters.

The efficiency of a sorting process depends upon the character of the data. A nearly sorted list, or a small number of items, can be handled effectively by a linear sorting routine. A non-increasing sorted list can better be walked through in reverse order than sorted. In general sorters of complexity $O(n \log n)$ are efficient for random data. Quick sort comes in handy when only part of an array has to be sorted.

For a discussion of the wide area of sorting and searching, see [20], and for programming templates, see [37].[3] For the Dutch speaking community there is the nice introduction [2].

The challenge is to encode $O(n \log n)$ sorting algorithms in TEX in a simple but flexible way.[4] Issues to address are

- a data structure must be chosen
- macros to fill the data structure
- abstracting from the sorting algorithm—heap sort, quick sort, ...
- parameterizing the comparison and exchange operations,
- abstracting in lexicographic sorting from the ASCII[5] ordering, and the
- handling of ligatures and diacritical marks.

In the first section the printing of sequences is treated. The storing of the data is considered in the second section. The sorting is elaborated on in the sections 3 and 4: sorting of numbers, respectively lexicographic sorting

[1] However, citations lists are sorted within TEX.

[2] If not for the encoding challenge.

[3] Any sorting macro should implement the algorithm with the comparison and exchange operator as parameters.

[4] Compatibility of number and lexicographic sorting has been strived after, where the particular sorting variant can be realized by appropriate `\let`-equals of the parameters.

[5] ASCII is the abbreviation of American Standard Code for Information Interchange. An ASCII table—associating each character with a number—is provided in the TEXbook, p. 367.

in the presence of the Dutch ij-ligature and diacritical marks.[6] In the fifth section the applications: sorting address labels, sorting and compressing Knuth's index reminders file, and sorting of control sequences separately, are dealt with. In the appendices I supplied the listings of the files: `heap.tex`, `quick.tex`, `sort.tex` and my testdriver `sort.tst`.

There are so many details in sorting and the TEX encoding of it, that I hope that the remainder is not too concise for those who are really interested in the details of the TEX encoding. On the other hand, I hope it won't contain too much for those who just like to get an idea of the possibilities of TEX with respect to sorting.[7]

Approach. The three processes: initialization, sorting and typesetting, are separately and independently designed.

For filling the data structure I considered it handy to have a few macros which store from

- copy (`\seq`...`\qes`),
- a file (`\storefrom`), or
- a process, which (randomly) generate elements (`\storerandomn`, `\storerandomw`).

For sorting I provided

- the Ben Lee User level macros (`\sortn`, `\sortaw`, `\sortw`), and
- the blue collar macros (`\heapsort`, `\quicksort`).

For typesetting the data structure I used the macros `\prtn`, respectively `\prtw`.[8]

Files. The file `sort.tex` contains the macros for storing (`\seq`...`\qes`, `\storefrom`, and `\storerandomn`, `\storerandomw`), for sorting (`\sortn`, `\sortaw`, `\sortw`), and for typesetting (`\prtn`, `\prtw`, and `\prtind`). Apart from these, the file contains the common definitions of the `\heapsort` and `\quicksort` macros, as well as variants for the parameter macros.

The files `heap.tex` and `quick.tex` contain the `\heapsort`, respectively `\quicksort`, macro along with specific auxiliaries.

My testdriver is the file `sort.tst`.

Definitions and notations. A sequence is defined as a row of numbers, respectively words, separated by spaces. The structure `\csname`$\langle k\rangle$`\endcsname`, is associated with an array with index $k = 1, 2, \ldots, n$. To denote in the documentation a value pointed by the number $\langle k\rangle$, I made use of `\val`$\{\langle k\rangle\}$, with `\def\val#1{\csname#1\endcsname}`. Macro names take suffix -n, -w, when specific for number, respectively word data. For example `\sortn` stands for sort numbers, `\prtw` stands for print words. I have typeset the in-line results of the examples in bold face.

For transferring values to a macro, I generally refrained from the (optional) parameter mechanism, as it is used in nowadays high-level programming languages. Instead I used Knuth's parameter TEXnique, which comes down to providing definitions and using these by invokations, eventually after a `\let`-equal.

I have used the shorthand notation `\ea`, `\nx`, and `\ag` for `\expandafter`, `\noexpand`, respectively `\aftergroup`. `\k` is used as counter to loop through the values $1, 2, \ldots, n$, the index domain. `\n` contains the maximum number of sequence elements, n. `\ifcontinue` is used for controlling loops. The array and the counter `\status` had to be maintained globally, because of the nesting of loops.

1 Typesetting elements

After sorting the typesetting must be done. In general this is dependent upon the application and will demand Hi-TEXnique.[9] For simplicity and in order to concentrate on the sorting aspects I typeset the sequence element after element, via `\prtn`, or `\prtw`.

Example (Typesetting a number sequence)[10]

```
\def\1{314}\def\2{1}\def\3{27}\n3 \prtn
```

yields: **314, 1, 27**.

Example (Typesetting a word sequence)

```
\def\1{ik}\def\2{j\ij}\def\3{h\ij}\n3 \prtw
```

yields: **ik jij hij**.

1.1 TEX encoding

Design choice. The elements are typeset in the default font. The separator is parameterized into `\sep`. Number sequences are typeset in range notation.

Input. The array `\`$\langle k\rangle$, $k = 1, 2, \ldots, n$, and the counter `\n` with value $\langle n\rangle$,[11] and optionally a value $\langle kzero\rangle, \geq 0$, in `\kzero`.

[6] Adaptable to other ligatures and accents.

[7] Ben Lee User, BLU for short, can always page through the provided headings and grasp 'what it is all about' from the included examples.

[8] The file `sort.tex` contains also `\prtind`, to typeset `index.tex`.

[9] Think for example of Knuth's typesetting of the index of the TEXbook, p. 261–263. It is in the chapter on OTR-s (Output Routines) with aura '... the following material will take you all the way to the rank of Grandmaster, i.e., a person who can design output routines.'

[10] In the examples `\def`-s are used to define a one digit as control symbol. `\csname...\endcsname` must be used for two or more digits.

[11] The defaults for the parameter `\sep`—`\sepn`, respectively `\sepw`—are provided in the file `sort.tex`.

Result. The array $\langle kzero+1\rangle : \langle n\rangle$ is typeset. `\kzero` is default 0.

The macros

```
\def\prts{{\k\kzero%print \1,...\n
 \def\sep{\let\sep\sepw}%
 \loop\ifnum\k<\n\advance\k1
    \sep\csname\the\k\endcsname
 \repeat}}\let\prtw\prts
%
\def\prtn{{\k\kzero%Print ranges
 \loop\ifnum\k<\n\advance\k1
  \ea\prc\csname\the\k\endcsname
 \repeat\prtfl}}
%
\def\prc#1{\init{#1}\def\prc##1{%
 \ifnum##1=\lst\else\ifnum##1=\slst
    \lst\slst\advance\slst1 \else
       \prtfl\sepn\init{##}\fi\fi}}
%
\def\prtfl{\the\frst\ifnum\frst<\lst
 \advance\frst1 \ifnum\frst=\lst\sepn
   \else\nobreak--\nobreak\fi\the\lst}
%
\def\init#1{\frst#1\lst#1\slst#1\advance
 \slst1{}}
```

Explanation. Abstraction of the lower index into `\kzero`, default 0, makes it possible to typeset parts of the array. The elements are separated by the separator given in, `\sepn`, respectively `\sepw`. The encoding is TEX specific. Each first time the loop is traversed the invokation of `\sep` redefines itself with the actual separator. On subsequent traversals the provided separator is typeset.

The replacement text of `\prtn` and `\prtw` is a group, and therefore the loop's `\body` cannot redefine the `\body` of an outer loop.

In order to account for number ranges `\prtn` uses `\prc`, a simplified version of `\processc`, borrowed from [25].

2 Storing a sequence

As data structure the following TEX-specific encoding[12] is used.

$$\texttt{\csname}\langle k\rangle\texttt{\endcsname},\quad k=1,2,...,n.$$

Writing to, respectively reading from, the k^{th} element goes via[13]

$$\texttt{\ea\def\csname}\langle k\rangle\texttt{\endcsname}\{\langle k^{th}elem.\rangle\},$$

and

$$\texttt{\csname}\langle k\rangle\texttt{\endcsname}.$$

When a counter `\k`, which takes the values $1,2,\ldots,n$, is used, then TEX requires `\the\k` for the index number $\langle k\rangle$.

To get the hang of it. The reader must be aware of the differences between

- the index number, $\langle k\rangle$
- the counter variable `\k`, with the value $\langle k\rangle$ as index number
- the control sequences $\backslash\langle k\rangle, k=1,2,\ldots,n$, with as replacement texts the items to be sorted.

When we have `\def\3{4} \def\4{5} \def\5{6}` then
`\3` yields **4**,
`\csname\3\endcsname` yields **5**, and
`\csname\csname\3\endcsname\endcsname` yields **6**.

Similarly, when we have
`\k3 \def\3{name} \def\name{action}` then
`\the\k` yields **3**,
`\csname\the\k\endcsname` yields **name**, and
`\csname\csname\the\k\endcsname\endcsname` yields **action**.[14] To exercise shorthand notation the last can be denoted by `\val{\val{\the\k}}`.

Another `\csname...` will execute `\action`, which can be whatever you provided as replacement text.

2.1 From copy

Elements available in the copy of an author are stored via

$$\texttt{\seq}\langle sequence\rangle\texttt{\qes}.$$[15]

Example (Storing numbers from copy)
`\seq1 314 27\qes` stores the elements. For verification `\prtn` yields: **1, 314, 27**.

Example (Storing words from copy)
`\seq ik j{\ij} h\ij\qes` stores the elements. For verification `\prtw` yields: **ik jij hij**.

TEX encoding

Design choice. The sequence is stored in an array via the FIFO TEXnique [23]. The process is independent of the type. Numbers or words (text) can be stored by the same macro.

Input. Data from the user copy preceded by `\seq` and followed by the separator `\qes`. The elements must be separated by a ␣, which is not gobbled by TEX's mouth. (In practice this means that words ending with a control sequence—`\i`, `\j`, or for Dutch `\ij`—must have braces around that control sequence.)

[12] Functionally equivalent to an array. Amy Hendrickson [15] used arrays in TEX although she did not call them as such. Adrew Greene [13], while playing around in TEX's mind, associated already the array concept with `\csname...`.

[13] Actually, I used `\gdef`-s, `\xdef`-s, and `\the\k`.

[14] Confusing, but powerful!

[15] Mnemonics: sequence. This abstracts from all the `\def`-s, casu quo `\csname...\endcsname`-s, as provided in the examples.

Result. The array $\langle k \rangle,\ k = \langle kzero + 1 \rangle, 2, \ldots, n,$ with the sequence elements as values. The counter `\n` will contain the value $\langle n \rangle$. $\langle kzero \rangle$ is a bias, with default value 0.

The macros

```
\def\seq#1\qes{%
          \k\kzero\fifow#1 \wofif{} }
%
%and auxiliaries
\def\fifow#1 {\ifx\wofif#1\n\k\wofif\fi
 \processw{#1}\fifow}
               \def\wofif#1\fifow{\fi}
%
\def\processw#1{\advance\k1
 \ea\gdef\csname\the\k\endcsname{#1}}
```

Explanation. The idea is that the elements from the copy enclosed by `\seq` and `\qes`—and appended in the macro `\seq` by `␣\wofif{}␣`[16]—are processed as arguments of the macro `\fifow`. This macro has a `␣` as endseparator. When `\wofif` is encountered the number of elements is stored in `\n` and the recursion is terminated by the invokation of `\wofif`. The latter macro gobbles all the tokens—in this case `\fi \processw{#1}`—up to and including the next invokation of `\fifow`. Its replacement text inserts a new `\fi`, to correct the disturbed `\if...\fi` balance.

The macro `\processw` maintains the (index) counter and actually stores each element, globally.

2.2 From a file

In applications the words (and other information like page numbers[17] for index preparation) are gathered into a file for later, usually external, processing.

Example (Storing from file)
If the file `index.tex` contains the records

```
word !3 314
word !1 27
tag !1 1
word !1 1
```

then

```
\storefrom{index.tex}
```

stores the elements from the file into the array.

For verification the array is printed by

```
\begin{quote}
\let\sepw\\\prtw\unskip.
\end{quote}
```

with result[18]

word !3 314
word !1 27
tag !1 1
word !1 1

TEX encoding

Specification. Records from a user specified file are to be read into the array. On termination the counter `\n` contains the number of stored elements.

Input. The file with the elements given per line. `\kzero` is default 0.

Result. The array $\langle k \rangle,\ k = \langle kzero \rangle + 1, 2, \ldots, n,$ with the elements as values. `\n` contains the upper bound of the array, $\langle n \rangle$.

The macro

```
\def\storefrom#1{%#1 is file name
 \openin\rec#1 \k\kzero \continuetrue
 \loop\ifeof\rec\continuefalse\fi
 \ifcontinue\advance\k1 \read\rec to\xyz
  \ea\global\ea\let\csname\the\k\endcsname\xyz
 \repeat\advance\k-1\n\k\closein\rec}
```

Explanation. The `\newread\rec` has been specified in the file `sort.tex`. TEX appends a `\par` to the opened file, therefore I had to decrement the counter `\k` by 1 at the end. After `\rec#1` a `␣` is mandatory; an empty group is not recognized as terminator. Because of the lack of an `\ifnoteof` and of the way `\loop` has been encoded—TEXbook, p. 219, an `\else` cannot be used in the body of the loop as part of the termination—I used the `\newif\ifcontinue` for controlling the loop. The bias $\langle kzero \rangle$ is handy for merging index files.

2.3 From a generator

Although the automatic generation of data is only used in the tests, it seemed worthwhile for me to include these macros too, as an example of how data can be created and stored.

Numbers

A random number generator—the macro `\rnd`—has been encoded in TEX by Reid [30]. I added `\storerandomn` to store the specified number of random numbers in the array.

Example (Storing random generated numbers)

```
\rndnum5 \storerandomn5\prtn
```

yields:[19] **1, 88, 62, 27, 1.**

[16] The empty group is needed because spaces after control sequences are gobbled. Beware!

[17] Known by the OTR—Output Routine—only. For writing the index reminders to the file `index.tex` see the TEXbook, p. 424, the macro `\writeit` and auxiliaries. A simplified encoding will be provided in Manmac BLUes, see elsewhere in this MAPS.

[18] Note that I had to add an `\unskip`. `\\` is LATEX's newline.

[19] More clearly, I could have provided `\rndnum=5␣` and `\storerandomn{5}`, to emphasize the different syntactical roles of the number 5.

The macros. The encoding of my macro is straightforward, once I decided to use Reid's random generator macro, \rnd [30].

```
\def\storerandomn#1{%#1 number
                    %of r-numbers
 \n#1\k0{\loop\ifnum\k<\n\advance\k1 %
                               \rnd\ea
  \xdef\csname\the\k\endcsname{%
                   \the\rndval}%
 \repeat}}
%
\def\rnd{\global\multiply\rndnum371
 \global\advance\rndnum1
 \ifnum\rndnum>99999
  \rndtmp\rndnum \divide\rndtmp100000
  \multiply\rndtmp100000
  \global\advance\rndnum-\rndtmp
 \fi\global\rndval\rndnum
 \global\divide\rndval1000 }
```

Words

Reid [30] introduced his macros for generating random paragraphs. I added \storerandomw to store the specified number of random words in the array.

Example (Storing random generated words)

```
\rndnum5 \storerandomw5\prtw
```

yields: **ajqjjhfn fyi uednas ahw zr.**

The macros

```
\def\storerandomw#1{%#1 number of words
 \n#1\nw\n{\loop\ifnum0<\nw
  {\ag\defarr\ag{\randomword}}%
                          \advance\nw-1
 \repeat}}%end s-r-w.
%
\def\defarr{\ea\gdef%
               \csname\the\nw\endcsname}
%
\def\randomword{\rnd\nc\rndval
 \divide\nc15\advance\nc2
 \loop\ifnum0<\nc\randomchar%
                           \advance\nc-1
 \repeat}%end r-word
%
%Random character is modified
\def\randomchar{\rnd
 \multiply\rndval29\divide\rndval100
 \ifnum\rndval=26\rndval0 \fi
 \ifnum\rndval>26\rndval4 \fi
%Mod cgl: I \ag-ed the letter
 \ea\ag\ifcase\rndval
  a\or b\or c\or d\or e\or f\or g\or h\or
  i\or j\or k\or l\or m\or n\or o\or p\or
  q\or r\or s\or t\or u\or v\or w\or x\or
  y\or z\fi}
```

Explanation. Although the same approach as for storing random generated numbers has been followed, I had to modify the code due to the need for intermediate storing of each random generated letter. A random word consists essentially of random numbers, mapped onto letters. The numbers are generated in an inner loop, via Reid's macro \randomchar. Because of the nesting of loops I had to group the inner loop. Realizing this, prompted a TEX specific way for storing the letters generated in the inner loop. The letters are placed after the enclosing group via \aftergroup. When the group is ended the word is stored as replacement text of $\backslash\langle nw\rangle$. (The tokens for the definition are \ag-ed before the inner loop; the closing brace is already after the innerloop.)

3 Sorting of numbers

Example

\seq314 1 27\qes\sortn yields: **1, 27, 314**.

3.1 Design choices

The backbone of my 'sorting in an array' is the data structure

$$\backslash\text{csname}\langle k\rangle\backslash\text{endcsname}\{\langle k^{th}elm.\rangle\},\quad k=1,2,\ldots,n,$$

with k the role of array index and n the number of items to be sorted.

The encoding is parameterized by \cmp, the comparison macro, which differs for numbers, strings, and in general when more sorting keys have to be dealt with.[20] The result of the comparison is stored globally in the counter \status.

3.2 TEX encoding

Input. The elements are assumed to be stored in the array $\backslash\langle k\rangle$, $k=1,2,\ldots n$. The counter \n must contain the value $\langle n\rangle$.

Result. The sorted array \1, \2, ... $\backslash\langle n\rangle$, with $\backslash\text{val}1 \le \backslash\text{val}2 \le \ldots \le \backslash\text{val}\langle n\rangle$.

The macros

```
\def\sortn{\let\cmp\cmpn\sort\prtn}
%
\def\cmpn#1#2{%#1, #2 must expand into numbers
%Result: \status= 0, 1, 2 if
%        \val{#1} =, >, < \val{#2}.
 \ifnum#1=#2\global\status0 \else
  \ifnum#1>#2\global\status1 \else
             \global\status2 \fi\fi}
%
\def\sort{\heapsort}.
```

Explanation. The above shows the structure of each of the Ben Lee User sorting macros.

Sorting: \sortn. A (pointer) \def\sortn{...} is introduced which has as replacement text the setting of the parameter \cmp, and the invokations of the actual sorting macro and the macro for typsetting the sorted sequence.

[20] For an example see the sorting of Knuth's index reminders in section 5.

Comparison operation: `\cmpn`. The result of the comparison is stored globally in the counter `\status`. The values $0, 1, 2$ denote $=, >, <$, respectively.

Exchange operation: `\xch`. The values can be exchanged via[21]

```
\def\xch#1#2{%#1, #2 counter variables
 \edef\aux{\csname\the#1\endcsname}\ea
 \xdef\csname\the#1\endcsname{\csname
   \the#2\endcsname}\ea
 \xdef\csname\the#2\endcsname{\aux}}.
```

3.3 Some testing

Apart from the examples as given above, `\sortn` has been tested on sequences of random numbers. Some idea of the efficiency was obtained and no reasonable restrictions with respect to the number of items to be sorted, other than the installation limitations, were encountered. For this purpose use has been made of Reid's random number generator in TEX, `\rnd` [30].

Timings. On my 8086 MS-DOS PC `\sortn` (without time needed to create the array, but with the time needed to write the sorted array to the dvi-file) had the (near-linear) performance

No	≈ Time
15	13 seconds
50	1 minute
200	5 minutes.

The University's VAX8650[22] needed ≈ 1.75 minutes for sorting 500 numbers.[23] The measurements were done with `\heapsort` as sorting macro.

3.4 Variation

For short sequences algorithms of complexity $O(n^2)$ are generally used.[24]

```
%O(N*N) sorting.
\def\sort{\bubblesort}
%
\def\bubblesort{%Data in \1, \2,...\<n>.
{\loop\ifnum1<\n{\k\n
  \loop\ifnum1<\k\advance\k-1 \cmp\k\n
   \ifnum1=\status\xch\k\n\fi
  \repeat}\advance\n-1
 \repeat}}%end \bubblesort
```

4 Lexicographic sorting

Given the blue collar workers `\heapsort`, respectively `\quicksort`, we have to encode the comparison macro in compliance with the parameter macro `\cmp`. But, ... lexicographic sorting is more complex than number sorting. We lack a general comparison operator for strings,[25] and we have to account for ligatures and diacritical marks.

In creating a comparison macro for words, flexibility must be built in with respect to the ordering of the alphabet, and the handling of ligatures and diacritical marks.

Example (Sorting ASCII words)

```
\seq a b aa ab bc bb aaa\qes\sortw
```
yields: **a aa aaa ab b bb bc**.

Example (Sorting words with ij-ligature)

```
\seq{\ij}st{\ij}d {\ij} {\ij}s in tik
 t\ij\qes\sortw
```
yields: **in tik tij ij ijs ijstijd**.

Example (Sorting accented words)

```
\seq b\'e b\‘e \'a\'a ge\"urm geur aa a
ge{\ij}kt be ge\"\i nd gar\c con\qes
\sortw
```
yields: **a aa áá be bé bè garçon geïnd geur geürm geijkt**.

Reculer pour mieux sauter. Because of the complexity and the many details involved I recede with simplified cases as stepping stones. I'll first guide you through the encoding of the comparison macro for

- one-(ASCII)letter-words, and
- ASCII strings, of undetermined length,

after which we will come back to the main track of the encoding of the general comparison macro.

One-(ASCII)letter-words. The issue is to encode the comparison macro, in compliance with the parameter macro `\cmp`. Let us call this macro `\cmpolw`.[26] Its task is to compare one-letter words and store the result of each comparison globally in the counter `\status`. As arguments we have `\def`-s with one letter as replacement text.

```
\def\cmpolw#1#2{%#1, #2 are def-s
%Result: \status= 0, 1, 2 if
%        \val{#1} =, >, < \val{#2}.
 \ea\chardef\ea\cone\ea‘#1{}%
 \ea\chardef\ea\ctwo\ea‘#2{}%
 \global\status0 \lge\cone\ctwo}
%
\def\lge#1#2{%#1, #2 are letter values
```

[21] For a better and more general macro, see section 4 about lexicographic sorting. Here the definitions are completely expanded, which is not necessary and therefore inefficient.

[22] Just to give the reader an idea because VMS is a time sharing system.

[23] As expected with 0–99 as printed result. Neat!

[24] A nice example of encoding nested loops. Should be part of courseware about macro writing in TEX.

[25] It is not part of the language, nor provided in plain. Victor Eijkhout [10] supplied one. The (limited) predecessor of my comparison macro has appeared in [23]. Those macros don't abstract from the ASCII ordering or allow for accented words and ligatures.

[26] Mnemonics: compare one letter words.

```
%Result: \status= 0, 1, 2 if #1 =, >, < #2. \let\cmp\cmpaw\sort\prtw.
 \ifnum#1>#2\global\status1 \else
   \ifnum#1<#2\global\status2 \fi\fi}
%
\seq z y A B a b d e m n o p z z u v c g
q h j I i l k n t u r s f Y\qes
\let\cmp=\cmpolw\sort\prtw
```

The above yields: **A B I Y a b c d e f g h i j k l m n n o p q r s t u u v y z z z**.

Explanation `\cmpolw`**..** In order to circumvent the abundant use of `\expandafter`-s, I needed a two-level approach: at the first level the letters are 'dereferenced,' and the numerical value of each replacement text is provided as argument to the second level macro, `\lge`.[27]

ASCII words. The next level of complexity is to allow for strings, of undetermined length and composed of ASCII letters. Again the issue is to encode the comparison macro, in compliance with `\cmp`. Let us call the macro `\cmpaw`[28]. Its task is to compare ASCII words and store the result of each comparison globally in the counter `\status`.

The problem is how to compare strings letter by letter. Empty strings are equal. This provides a natural initialization for the `\status` counter. As arguments we have `\def`-s with words of undetermined length as replacement text.

```
\def\cmpaw#1#2{%#1, #2 are def-s
%Result: \status= 0, 1, 2 if
%        \val{#1} =, >, < \val{#2}.
 {\let\nxt\nxtaw\cmpc#1#2}}
%
\def\cmpc#1#2{%#1, #2 are def-s
%Result: \status= 0, 1, 2 if
%        \val{#1} =, >, < \val{#2}.
\global\status0 \continuetrue
{\loop\ifx#1\empty\continuefalse\fi
      \ifx#2\empty\continuefalse\fi
 \ifcontinue\nxt#1\nxtt \nxt#2\nxtu
     \lge\nxtt\nxtu
     \ifnum0<\status\continuefalse\fi
 \repeat}\ifnum0=\status
 \ifx#1\empty\ifx#2\empty\else
                 \global\status2 \fi
 \else\ifx#2\empty\global\status1 \fi
 \fi\fi}
%
\def\nxtaw#1#2{\def\pop##1##2\pop{\gdef
 #1{##2}\chardef#2'##1}\ea\pop#1\pop}
%
\seq a b aa ab bc bb aaa\qes
```

The above yields: **a aa aaa ab b bb bc**.

Explanation

Comparison: `\cmpaw`**..** The macro is parameterized over the macro `\nxt`. The main part of `\cmpaw` has been encoded as `\cmpc`. (That part is also used in the general case.)

We have to compare the words letter by letter. The letter comparison is done by the already available macro `\lge`. The `\lge` invokation occurs within a loop, which terminates when either of the strings has become empty. I added to stop when the words considered so far are unequal. At the end the status counter is corrected if the words considered are equal and one of the #-s is not empty: into 1, if #1 is not empty, and into 2, if #2 is not empty.

Head and tail: `\nxt`**..** The parameter macro `\nxt` has the function to yield from the replacement text of its first argument the ASCII value of the first letter and deliver this value as replacement text of the second argument.[29] The actual macro `\nxtaw` pops up the first letter and delivers its ASCII value—a `\chardef`—as replacement text of the second argument. Note that the first parameter is globally redefined for the emptiness-test after the loop.

4.1 Design choices

Sorting of words with ligatures and accents is done via the ordering defined in a so-called ordering table.[30] This entails that the comparison of letters has to be generalized such that accents are recognized too. For the (accented)letter-to-number conversion the ` is replaced by a table look-up.

Comparison operation. A special situation arises with diacritical marks. Within the context of sorting the commands for diacritical marks have been redefined with the function to provide for the control symbol together with the accompanying letter an appropriate value from the ordering table.

Note that when the ASCII ordering is sufficient, no ordering table is needed. For that case `\cmp` can be `\let`-equal to `\cmpaw`.

Ordering table. In Dutch the ij is peculiar. It is mostly used as a ligature[31] and in that role its lexicographic position is between `x` and `z`.[32] This character is not accounted for in the ASCII table, therefore we

[27] Mnemonics: letter greater or equal. A nice application of the use of `\ea`, `\chardef`, and the conversion of a character into a number: `'`. Note that the values of the upper case and lower case letters differ (by 32) in ASCII.

[28] Mnemonics: compare ASCII words.

[29] Splitting up into 'head and tail,' is treated in the TEXbook, Appendix D.2, p. 378, the macro `\lop`. There use has been made of token variables instead of `\def`-s.

[30] In the ordering table numbers are associated to letters, ligatures and accented letters.

[31] Also as two characters. For example in bi-jection (hyphen for emphasis) and the like.

[32] I was quite surprised to find out that my Dutch dictionary does *not* sort on the ij-ligature?!?

need an ordering table. In some other languages similar situations exist, so the idea of abstraction from the ASCII ordering is useful,[33] if not for the handling of diacritical marks. For the input I adopted the convention to supply `\ij`. The ordering table is implemented via a list of `\chardef`-s.[34] Numbers are not assigned consecutively in the ordering table, leaving room for accented letters.

I have provided the same values for upper and lower case letters. The successor values are reserved for the accents: acute, grave, umlaut, and hat. I also accounted for the cedille.

4.2 TEX Encoding

Purpose. To sort words with (Dutch) accents and ij-ligature.

Input. The elements are assumed to be stored in the array $\backslash\langle k\rangle$, $k = 1, 2, \ldots n$. The counter `\n` must contain the value $\langle n\rangle$.

The default settings are done in the file `sort.tex`. The macro `\accdef` contains the modified accent definitions, and the macro `\accstr` the string of accent control sequences.

Result. The sorted array `\1`, `\2`, $\ldots$ $\backslash\langle n\rangle$, with `\val1` $\leq$ `\val2` $\leq \ldots \leq$ `\val`$\langle n\rangle$.

The macros

```
%Modifications/addenda to number sorting
%Sorting and typesetting.
\def\sortw{{\accdef\let\cmp\cmpw\sort}%
                                    \prtw}
%
%Compare words.
\def\cmpw#1#2{%#1, #2 are def-s.
%Result: \status= 0, 1, 2 if
%        \val{#1} =, >, < \val{#2}.
 \let\nxt\nxtw\cmpc#1#2}
%
%Yield value of next (accented) letter.
\def\nxtw#1#2{\def\pop##1##2\pop{%
 \gdef#1{##2}\def\head{##1}}%tail and head
 \ea\pop#1\pop
 \ea\loc\head\accstr%\head in accentcs?
 \iffound\let\acs\head
  \ea\pop#1\pop%next tail and head
  \ea\let\ea#2\csname ot\acs\head\endcsname
 \else\ea\let\ea#2\csname ot\head\endcsname
 \fi}
%
\def\accstr{\'\`\"\^\c}
%
\def\accdef{%
 \def\'##1{##1g}\def\`##1{##1a}
                                %acute grave
 \def\"##1{##1t}\def\c##1{##1c}
                                %trema cedille
 \def\^##1{##1h}%hat
 \def\i{i}\def\j{j}}%dotless i, j

\def\loc#1#2{\def\locate##1#1##2\end
 {\ifx\empty##2\empty\foundfalse
 \else\foundtrue\fi}\ea\locate#2.#1\end}
%
%Ordering table
\chardef\ota32 \chardef\otA32
 \chardef\otaa33 \chardef\otag33
 \chardef\otat34 \chardef\otah35
%et cetera, see Appendix C
```

Explanation

Sorting: `\sortw`. A (pointer) `\def\sortw{...}` is introduced, which has as replacement text the insertion of the accent definitions via the invokation of `\accdef`, the setting of the parameter `\cmp`, and the invokations of the actual sorting macro and the macro for typesetting. A group is used in order to keep the temporary redefinitions of the accents local. `\accdef` yields the modified definitions for the accents and special letters like `\i`, and `\j`. The purpose of these definitions is to get the right value from the ordering table.

Comparison operation: `\cmpw`. The parameter macro `\nxt` is `\let`-equal to `\nxtw`. The comparison is done by the common `\cmpc`,[35] which stores globally the result of the comparison in the counter `\status`.

Head and tail: `\nxtw`. This macro peels a token[36] from the first argument, selects the associated value form the ordering table and delivers the latter value in the second argument, as a `\chardef`. The complication is that when we have an accent we have to consider the next token too, and select the associated numerical value for the combination.[37]

The remainder of the word, the tail, is delivered globally in the first argument, for the afterloop emptiness-test. The local macro `\pop` yields the head and the tail of a word. `\loc` determines whether the token in `\head` is an accent control symbol.[38]

Exchange operation: `\xch`. No expansion of the accents must take place, therefore the already stored data are copied via the `\let`-equal TEXnique.

```
\def\xch#1#2{%#1, #2 counter variables
```

[33] As communicated by Wlodek Bzyl, with respect to Czech.

[34] Remember that a `\chardef` name can be used as a number.

[35] For an explanation see the subsection about ASCII words.

[36] Or debraced group.

[37] A neat use of `\ea`, `\let`, and `\csname...`, with as result a `\chardef`!

[38] This is a generalization of the search for $\langle char\rangle \in \langle string\rangle$ [23]. On second thoughts, I consider this a neat generalization. The temporary redefinitions are parameterized into `\def\accdef...` .

```
\ea\let\ea\auxone\csname\the#1\endcsname
\ea\let\ea\auxtwo\csname\the#2\endcsname
\ea\global\ea\let\csname\the#2\endcsname
\auxone
\ea\global\ea\let\csname\the#1\endcsname
\auxtwo}.
```

To verify your understanding, what is the result[39] of

```
\m3\n4\def\3{first}\def\4{second}
\xch\m\n
\the\m, \the\n; \3, \4.
```

4.3 Some testing

Apart from the examples as given above, lexicographic sorting has been tested on sequences of random words. For this purpose use has been made of Reid's [30] work for generating random paragraphs in TEX.

Timings. On my 8086 MS-DOS PC the (word) sorting, without the time needed to create the array but with the time needed to write to the dvi-file, had the performance

No	≈ Time
15	30 seconds
50	3 minutes.

The University's VAX8650[40] needed ≈ 5 minutes to sort 500 random words. The measurements were done with `\heapsort` as sorting macro, with each word of random length.

5 Applications

5.1 Sorting address labels

Amy Hendrickson [15] used sorting of address labels to illustrate various macro writing TEXniques. However, she used external sorting routines. Here I will do the sorting within TEX, and enrich her approach further by separating the mark-up phase from the data base query and the report generating phases. Because this paper concentrates on sorting aspects, let us assume that each address is supplied as a definition, with the definitions of the name and address components as replacement text. Furthermore, it is handy to create a list of all the addresses: the names of the address definitions separated by `\as`, the address separator.[41] For the imaginative toy adresses of the three composers: Schönberg, Webern, Strawinsky, the structures look like as follows.

```
\def\schonberga{\def\initial{A}
 \def\sname{Arnold}\def\cname{Sch\"onberg}
 \def\street{Kaisersallee}\def\no{10}
 \def\county{}\def\pc{9716HM}
 \def\phone{050-773984}\def\email{as@tuw.au}
 \def\city{Vienna}\def\country{AU}}
%
\def\strawinskyi{\def\initial{I}
 \def\sname{Igor}\def\cname{Strawinsky}
 \def\street{Longwood Ave}\def\no{57}
 \def\county{MA}\def\pc{02146}
 \def\phone{617-31427}
 \def\email{igor@ai.mit.edu}
 \def\city{Boston}\def\country{USA}}
%
\def\weberna{\def\initial{A}
 \def\sname{Anton}\def\cname{Webern}
 \def\street{Amstel}\def\no{143}
 \def\county{Noord-Holland}\def\pc{9893PB}
 \def\phone{020-225143}\def\email{aw@uva.nl}
 \def\city{Amsterdam}\def\country{NL}}
%
%and the list
\def\addresslist{\as\strawinskyi
 \as\weberna\as\schonberga}
```

For the typesetting I made use of the following simple address label format[42]

```
\def\tsa{%The current address info is set
 \par\initials \cname \par
 \no\ \street\ \city\par
 \pc\ \county\ \country\par}
%
\def\initials{\ea\fifo\initial\ofif}
\def\fifo#1{\ifx\ofif#1\ofif\fi#1. \fifo}
\def\ofif#1\fifo{\fi}
```

Example (Database query: selection of addresses per country)
Suppose we want to select (and just `\tsa` them for simplicity[43]) the inhabitants from Holland from our list. This goes as follows.

```
\def\search{NL}
\def\as#1{#1\ifx\country\search\tsa\fi}
\addresslist
```

The above yields the result
A. Webern
143 Amstel Amsterdam
9893PB Noord-Holland NL

[39] Answer: 3, 4; second, first.

[40] Just to get the flavor of it because VMS is a time sharing system.

[41] By this set-up we can do a lot more than just sorting address labels. What about mailmerge? What about 'TEX as a database report generator?' Jurriens [18] coined the term, although most of the work there was done via UNIX scripts.

[42] The encoding of printing special address labels has been worked out by for example Damrau & Wester [7]. It is left as an exercise to the reader to modify `\tsa` such that address labels are typeset in an m-by-n grid, each label of size h-by-w with parameters m, n (counters), and h, w (dimensions).

In this example `\initials` is not used. It has been added to allow for multiple initials of which all letters must end with a period.

[43] We could also create a new address list for that country and apply another query, or just sort.

Example (Sorting address labels)
Amy's example can be done completely within TEX, as follows.

```
%Prepare sorting
\def\as#1{\advance\k1 \ea\xdef\csname
 \the\k\endcsname{\ea\gobble\string#1}}
%
\def\gobble#1{}
%
\k0{}\addresslist%Create array to be sorted
\n\k\def\prtw{}%Suppress default \prtw
\sortw%Sort the list
%Typeset addresses, alphabetically ordered
\k0
\loop\ifnum\k<\n\advance\k1
 \csname\csname\the\k\endcsname\endcsname
 \vskip1ex\tsa
\repeat
```

The above yields the results

A. Schönberg
10 Kaisersallee Vienna
9716HM AU

I. Strawinsky
57 Longwood Ave Boston
02146 MA USA

A. Webern
143 Amstel Amsterdam
9893PB Noord-Holland NL

Remarks. The automatic mark-up of address data supplied in a TEX independent way, is not the subject of this paper. The given set-up allows to add, in any order, the address information to the database, under the restriction that definitions with the same names must be used for the address components.[44] The list must be modified too.

As can be seen from the above, and also in Amy's free format, it is not easy to keep the file ordered while extending the database. Therefore sorting is needed, such that the database can be extended in an arbitrary way. Database TEXniques have it that modifications to the data are independent from the report generating, thanks to the sorting tools.

5.2 Sorting Knuth's index reminders

An index reminder, as introduced by Knuth, consists of index material to be further processed for typesetting an index. In the TEXbook, p. 424, Knuth gives the syntax of an index reminder

$$\langle word\rangle\sqcup!\langle digit\rangle\sqcup\langle page\ number\rangle.$$

The reminders, one per line, are written to a file because only the OTR knows the page numbers. Knuth considered this file, `index.tex`,

> '. . . a good first approximation to an index.'

He also mentions the work of Winograd and Paxton [36][45] for automatic preparation of an index. Here we will provide a second approximation to an index: the index reminders are sorted and compressed. The sorting is done on the three keys

primary key: $\langle word\rangle$
secundary key: $\langle digit\rangle$, and
tertiary key: $\langle page\ number\rangle$.

The compressing comes down to reducing the index reminders with the same $\langle word\rangle$ $\langle digit\rangle$ part to one, with instead of one page number all the relevant page numbers in non-decreasing order.

We assume that the index reminders are already stored in the array.[46] Similarly, I didn't bother about writing the sorted and reduced array to a file. It is up to the index preparator what to do with the array and how to typeset it. Furthermore, it is not complete, because of subentries, subsubentries, or 'see. . . ,' and 'see also. . . ,' which are not considered here.[47]

Example (Sorting on primary, secundary and tertiary keys)

```
\def\1{z !3 1}\def\2{a !1 2}\def\3{a !1 3}
\def\4{a !1 1}\def\5{ab !1 1}\def\6{b !0 1}
\def\7{aa !1 1}\def\8{a !2 2}\def\9{aa !1 2}
\n9\k0\kk0
\let\cmp\cmpir\sort\let\sepw\\\null
\hfil\vtop{\hsize2cm\noindent
 after sorting\\[.5ex]\prtw}
\hfil\vtop{\hsize2.5cm\noindent
 after reduction\\[.5ex]\redrng\prtw}
\hfil\vtop{\hsize2cm\noindent
 typeset in\\index:\\[.5ex]\prtind.}\hfil
```

The above yields[48]

after sorting:	after reduction:	typeset in index:
a !1 1	a !1 1–3	a 1–3
a !1 2	a !2 2	\a 2
a !1 3	aa !1 1, 2	aa 1, 2
a !2 2	ab !1 1	ab 1
aa !1 1	b !0 1	b 1
aa !1 2	z !3 1	⟨z⟩ 1.
ab !1 1		
b !0 1		
z !3 1		

[44] Of course one can change the chosen names.

[45] Later Lamport provided makeindex and Salomon a plain version of it, to name but two persons who contributed to the development. The Winograd Paxton Lisp program is also available in Pascal.

[46] The process for storing the contents of `index.tex` in the array `\1`, `\2`,. . . ,$\backslash\langle n\rangle$, has been described in Storing from a file, section 2, and will not be repeated here.

[47] An approach to handle sub(sub)entries is to allow for composite primary keys, for example separated by `\se`, respectively `\sse`. In `\decom, and \typind` we have to account for the various possibilities. I will come back to the issue of typesetting Indexes within TEX, another time.

[48] The unsorted input can be read from the verbatim listing.

Design

Given the sorting macros we just have to encode the special comparison macro in compliance with `\cmpw`: compare two 'values' specified by `\def`-s. Let us call this macro `\cmpir`.[49] Each value is composed of

- a word (action: word comparison),
- a digit (action: number comparison), and
- a page number (action: (page) number comparison).

The macros read as follows.

```
\def\cmpir#1#2{%#1, #2 defs
%Result: \status= 0, 1, 2 if
%        \val{#1} =, >, < \val{#2}
 \ea\ea\ea\decom\ea#1\ea;#2.}
%
\def\decom#1 !#2 #3;#4 !#5 #6.{%
 \def\one{#1}\def\four{#4}\cmpaw\one\four
 \ifnum0=\status%Compare second key
   \ifnum#2<#5\global\status2 \else
     \ifnum#2>#5\global\status1 \else
       %Compare third key
       \ifnum#3<#6\global\status2
       \else\ifnum#3>#6\global\status1 \fi
       \fi
     \fi
   \fi
 \fi}
```

Explanation. I needed a two-level approach. The values are decomposed into their components by providing them as arguments to `\decom`.[50] The macro picks up the components

- the primary keys, the $\langle word \rangle$,
- the secundary keys, the $\langle digit \rangle$, and
- the tertiary keys, the $\langle page\ number \rangle$.

It compares the two primary keys, and if necessary successively the two secondary and the two tertiary keys. The word comparison is done via the already available macro `\cmpaw`.

To let this work with `\sort`, we have to `\let`-equal the `\cmp` parameter to `\cmpir`.

Reducing duplicate word-digit entries

The idea is that the same index entries, except for their page numbers, are compressed into one, thereby reducing the number of elements in the array. Instead of one page number all the relevant page numbers are supplied in non-descending order in the remaining reminder, in range notation. The macro is called `\redrng`[51] and is given below

```
\def\redrng{%Reduction of \1,...,\n, with
%page numbers in range representation
 {\k1\kk0
 \ea\let\ea\record\csname\the\k\endcsname
 \ea\splitwn\record.\let\refer\word
 \let\nrs\empty\prcrng\num
 \loop\ifnum\k<\n\advance\k1
  \ea\let\ea\record\csname\the\k\endcsname
  \ea\splitwn\record.%
  \ifx\refer\word%extend \nrs with number
    \prcrng\num
  \else%write record to \kk
   \advance\kk1 \strnrs \ea\xdef
   \csname\the\kk\endcsname{\refer{} \nrs}
   \let\nrs\empty\init\num\prcrng\num
    \let\refer\word
  \fi
 \repeat\ifnum1<\n\advance\kk1 \strnrs\ea
  \xdef\csname\the\kk\endcsname{\word{}
   \nrs}\global\n\kk\fi}}
%auxiliaries
\def\splitwn#1 !#2 #3.{\def\word{#1 !#2}%
 \def\num{#3}}
%
\def\prcrng#1{\init{#1}\def\prcrng##1{%
 \ifnum##1=\lst\else\ifnum##1=\slst
  \lst\slst\advance\slst1 \else
  \strnrs\init{##1}\fi\fi}}
%
\def\strnrs{\dif\lst\advance\dif-\frst
 \edef\nrs{\ifx\nrs\empty\else\nrs\sepn\fi
  \the\frst\ifnum0<\dif
   \ifnum1=\dif\sepn\the\lst
   \else\nobreak--\nobreak\the\lst
   \fi
  \fi}}
```

Explanation. The encoding is complicated because while looping over the index reminders either the reminder in total or just the page number has to handled. The handling of the page numbers is done with modified versions of `\prc`, `\prtfl`, called respectively `\prcrng` and `\strnrs`.[52] I encoded to keep track of the numbers in the macro `\nrs`, in the case of duplicate word-$digit$-entries. Another approach is while typesetting the array element to process the page numbers via `\prc` [25].

Typesetting index entries

Knuth has adopted the following conventions for coding index entries.

Mark up	Typeset in copy*	In `index.tex`
`^{...}`	...	`... !0` $\langle page\ no \rangle$
`^^{...}`	'silent'	`... !0` $\langle page\ no \rangle$
`^|...|`	|...|	`... !1` $\langle page\ no \rangle$
`^|\...|`	|\...|	`... !2` $\langle page\ no \rangle$
`^|<...>|`	$\langle ... \rangle$	`... !3` $\langle page\ no \rangle$

* `|...|` denotes manmac's, TUGboat's,... verbatim.

[49] Mnemonics: compare index reminders

[50] Mnemonics: decompose. In each comparison the def-s are 'dereferenced,' that is their replacement texts are passed over. This is a standard TEXnique: a triad of `\ea`-s, and the hop-over-s to the second argument.

[51] Mnemonics: reduce (in range notation). The macro `\red`, which does not yield the page numbers in range notation is supplied in the file `sort.tex` too.

[52] Mnemonics: processc with ranges, respectively store numbers.

The typesetting as such can be done via the following macro.

```
\def\typind#1{%#1 a def
 \ea\splittot#1.%
 \ifcase\digit\word\or
        {\tt\word}\or
  {\tt\char92\word}\or
  $\langle\hbox{\word}\rangle$\fi{}
 \pagenrs}
%
\def\splittot#1 !#2 #3.{\def\word{#1}%
 \chardef\digit#2{}\def\pagenrs{#3}}
%
\def\prtind{{\def\\{\hfil\break}\k\kzero
 \def\sep{\let\sep\sepw}%
 \loop\ifnum\k<\n\advance\k1 \sep
  \ea\typind\csname\the\k\endcsname
 \repeat}}
```

The typesetting of the index à la TEXbook Appendix I has been dealt with in the Grandmaster chapter of the TEXbook, p. 261–263.

5.3 More than one index

Erik Frambach posed the following question on the tex-nl@hearn discussion list

> How to prepare automatically two index files: one for commands and one for the rest?

A solution to this problem is to create the information in two files, one for the control sequences and the other for the rest. This works independently of the used tool. Another solution is splitting the `index.tex` file, depending upon the $\langle digit \rangle$ code.[53] Knuth associated control sequences with code 2, when writing the index entry to `index.tex`, TEXbook p. 423.

Example (Separate sorting of control sequences)

```
\def\1{wd !2 7}\def\2{wrd !1 1}
\def\3{wd !2 2}\def\4{a !1 1}
\def\5{wd !2 5}\def\6{wd !2 3}
\def\7{z !3 7}\def\8{wrd !1 5}
\def\9{wd !2 1} \n9
\let\sepw\\\null
\hfil\vtop{\hsize=2cm\noindent
 data:\\[.5ex]\prtw}
\hfil\vtop{\hsize=2.2cm\sortcs\noindent
 after splitting:\\[.5ex] {\n\pk\prtw}
  \\[.5ex]\kzero\pk\prtw}
\hfil\vtop{\hsize2.5cm\let\cmp\cmpir
 {\low1\up\pk\quicksort}
 {\low\pkone\up\n\quicksort}\noindent
 after sorting\\both parts,\\
 compressing,\\and typesetting:\\[.5ex]
 \redrng\n3 \prtind\\[.5ex]\typind\4.}
```

yields[54]

data:	after splitting:	after sorting both parts, compressing, and typesetting:
wd !2 7	wrd !1 5	a 1
wrd !1 1	wrd !1 1	wrd 1, 5
wd !2 2	z !3 7	⟨z⟩ 7
a !1 1	a !1 1	\wd 1–3, 5, 7.
wd !2 5	wd !2 5	
wd !2 3	wd !2 3	
z !3 7	wd !2 2	
wrd !1 5	wd !2 7	
wd !2 1	wd !2 1	

Encoding

```
\def\getdig#1 !#2 #3.{\def\dig{#2}}
%
\def\sortcs{\global\k0\global\pk\n
\global\pkone\pk\global\advance\pkone1
%Invariant: 1:k non-cs-s,
%             and pk+1:n cs-s
\loop\global\advance\k1
\ifnum\k<\pkone
 \ea\ea\ea\getdig\csname\the\k\endcsname.%
 \if2\dig{\continuetrue%        <--
cs<=>2!
  \loop
   \ifnum\k=\pk\global\pkone\pk
    \global\advance\pk-1 \continuefalse
   \else\ea\ea\ea\getdig\csname\the\pk
    \endcsname.%
    \if2\dig\global\pkone\pk
     \global\advance\pk-1
     \ifnum\k=\pk\continuefalse\fi
    \else\xch\k\pk\global\pkone\pk
                  \global\advance\pk-1
     \continuefalse
    \fi
   \fi
  \ifcontinue
  \repeat}%
 \fi
\repeat}%Result\1:\pk non-cs, \pkone:\n cs
```

Explanation. Suppose that the file `index.tex` is stored in the array. Loop through the array and compare the $\langle digit \rangle$ with 2. In case of a control sequence swap this index entry with an appropriate entry at the end.

The invariant of the loop is: $\backslash 1 : \backslash\langle k \rangle$[55] contains no control sequences, and $\backslash\langle pk+1 \rangle : \backslash\langle n \rangle$ contains control sequences.

As result the array is partitioned with the control sequences at the end of the array, that is the replacement texts of $\backslash\langle pk+1 \rangle : \backslash\langle n \rangle$.

[53] Conversely, merging two separate index files is easy, and can be done via `\storefrom{`$\langle 1^{st}$-$file \rangle$`} \kzero\n \storefrom{`$\langle 2^{nd}$-$file \rangle$`}`.

[54] The sorting of the control sequences can be done via a slightly more efficient `\cmpir`, because of the same $\langle digit \rangle$.

[55] Not pk, but k!

6 Epilogue

A glossary or an index is usually processed outside of TEX, that is via other tools. 'How to encode in TEX,' was explored via the classic example of sorting. No robustness was strived after. The encodings have been kept as simple and flexible as possible.[56] As a consequence no attention has been paid to safeguarding goodies like the prevention of name confusions with those already in use by an author.

Silent redefinitions do occur when not alert. Beware!

Looking back. Much of the work has been done in the spirit of

> Abstraction is our only mental tool
> to master complexity E.W. Dijkstra
>
> A professional starts
> where an amateur ends G.E. Forsythe

7 TEXniques used

The printing of a sequence parameterized by the separator.
FIFO and the active list separator to store a sequence in an array.
Parameter separators to select parts of an argument.
Peeling off characters one by one from a string.
Expanding the parameters before the invokation of the macro (Use of triads of `\ea`-s).
Using the ASCII values of characters for comparison.
Transforming numbers into characters.
Generating random elements for testing.
`\ag` to store random generated letters as words.
Setting up an address database.
Selecting from a database via queries implemented via the active list separator.
Maintaining a heap structure.
Initialization of loops and recursion: on first traversal some actions are different from the rest.
Ending recursion via gobbling up the tokens including the invokation for the next level.
Parameterizing sorting with respect to the comparison operation.
`\chardef`-s to parameterize the ordering table of the alphabet.
Sorting (accented) words.
Sorting on keys, with composite values.
Compressing index reminders.
Nested `\csname`.
Not only exchanging expansion order but also processing order, via `\ag`.

Hard things. One can rhetorically question whether the macros have been coded in a near optimal way?[57] I'm convinced that the basic approach

> to parameterize as much as possible

is a good thing. I also believe that the modular approach to encode small pieces, with clear functional tasks, is the way to build something of a reasonable size, and to keep it readable and maintainable. Literate programming avant la lettre?

Often I needed the `\xdef` functionality, but partially expanded. As a typical example the following.
Instead of

```
\xdef\<name>{\csname\the\k\endcsname}
```

I had to use (also with `\global`)

```
\ea\let\ea\<name>\csname\the\k\endcsname
```

when I incorporated the handling of accents. Also for the non-accent case the latter is better, because it leaves the contents of $\backslash\langle name\rangle$ untouched. It is not clear to me whether the use of token variables instead, would have been better.

Exchanging the order of expansion is abundantly used in the TEXbook. In generating random words I needed to delay the storing. In that particular case—reasonable size of the wordlength—I could fruitfully made use of `\ag`. From this I learned that the use of `\ag` comes in when 'stomach' processes have to be exchanged on the fly.

A TEXfall is that `\global`, so easily used with counters and definitions, does not extend to `\newif`-s. In first instance I tried to keep track of the status of the comparison of two strings by Booleans, at an inner level. Because, I could not use them globally otherwise than adapting the `\newif` macro, I have used a counter—`\status`—instead.

Another TEXfall is that the `\body` of a loop is silently redefined when nesting loops without scope braces. This occurs for example when in a loop a macro is invoked which contains an (unbraced) inner loop in its replacement text. This is different from the TEXfall where the first (inner) `\repeat` is mistaken for the outer one. Difficulties with nesting of loops, especially to keep quantities local, have been alluded to earlier by Pittman [29].

In debugging I traced every comparison and exchange via `\immediate\write16...`-s.

In articles like this it is difficult to circumvent unwanted spaces when in horizontal mode. My solution is to do the sorting in vertical mode and when done typeset in horizontal mode. I have taken notice of Eijkhout's suggestions [11].

[56] But, alas, full of details.

[57] Indeed, because of the many ways one can encode in TEX, it is very hard, if not impossible, to decide which code is best. Perhaps we have to get used to it that programming is like life. Polymorph! Knuth experienced similar things as can be distilled from 'Always remember, however, that there's usually a simpler and better way to do something than the first way that pops into your head.' The TEXbook, p. 373. Apart from the set-up, much has been given an afterthought or two.

On the other hand in the encoding of `\seq` I had to insert an empty group after `\wofif` in order to retain the separator ␣. This insertion of the empty group was also necessary in `\redrng` when rewriting the array: not `\` but `{}` ! I also had to compensate for Southall's 'buses and weirdness'-TEX effect, about which he lectured so vividly at the 1990 SGML-TEX meeting at Groningen.

Conclusion

I believe that my macros can be of use for preparing indexes completely within TEX. In the discussion about the NTS (New Typsetting System), in [28] and [35], it is argued to think in pre- and post-processing outside of TEX. Sorting index items is neither a pre- nor a post-process. Generally it is done in between. A file with index reminders is written while TEXing the compuscript. Then external sorting and the like is done outside of TEX, and finally the typesetting of the index is done again by TEX. Now all can be done within TEX, despite the good and abundant external sorters available.

At the danger of being accused of misusing TEX as

> '. . . another American screw driver'[58]

for situations not envisioned in the design, I found that encoding a non-trivial example in TEX illustrates the power of TEX's language. But, . . . I also sadly endured TEX's negative side

> Encoding in TEX is error-prone! if not for being so unusual.

This despite of its author being the initiator of literate programming.[59]

> A discipline of TEX encoding? Absolutively!

Acknowledgements

Ronald Kappert is kindly acknowledged for his suggestions and remarks while proofing.

References

[1] Alexander, J.C (1986): Tib, a reference setting package. *TUGboat* 8, no. (2), 102.

[2] Amstel, J.J van, J Bomhoff, G.J Schoenmakers (1978): Inleiding tot het programmeren 1. Academic Service.

[3] Arseneau, D (1992): overcite.sty, drftcite.sty, cite.sty. (from the file server)

[4] Bentley, J (1986): Programming Pearls. Addison-Wesley.

[5] Bechtolsheim, S von (1989): `\csname` and `\string`. *TUGboat* 10, no. (2), 203–206. (Apart from the basics, SvB discusses the convenient (read non-double) loading of macro files, and the cross-referencing. Common to both applications is the use of `\csname` and `\string`. No nesting of `\csname`, and no mentioning of associating `\csname...` with arrays.)

[6] Chen, P, M Harrison (1987): Automatic index preparation. CSB-TR 87/347. UCB. (A nice survey of issues relevant to preparing indexes automatically. Existing indexing tools in use in various systems are discussed. The paper emerged from the experience gained in writing the makeindex C program.)

[7] Damrau, J, M Wester (1991): Form letters with 3-across labels capability. TUG '91. *TUGboat* 12, no. (4), 510–516.

[8] Durst, L (1989): Bibliographic citations, or variations on the old shell game. *TUGboat* 10, no. (3), 390–394.

[9] Durst, L (1991): Some tools for making indexes. *TUGboat* 12, no. (2), 248–252.

[10] Eijkhout, V (1992): TEX by Topic. Addison-Wesley.

[11] Eijkhout, V (1993): The bag of tricks. *TUGboat* 13, no. (4), 494–495.

[12] Green, I (1992): citesort.sty. (from the file server)

[13] Greene, A.M (1989): TEXreation—Playing Games with TEX's mind. *TUGboat* 10, no. (4), 691–705.

[14] Hendrickson, A (1989): MacroTEX.

[15] Hendrickson, A (1990): Getting TEXnical: Insights into TEX macro writing techniques. *TUGboat* 11, no. (3), 359–370.

[16] Jeffreys, A (1990): Lists in TEX's mouth. *TUGboat* 11, no. (2), 237–244.

[17] Jensen, K, N Wirth (1975): PASCAL user manual and report. Springer-Verlag.

[18] Jurriens, T.A (1992): TEX as database. MAPS92.2, 100–101.

[19] Kappert, R (1992): scite.sty. (A compilation of the earlier versions of Arseneau and Green. From the file server.)

[20] Knuth, D.E (1973): The Art of Computer Programming 3. Sorting and searching. Addison-Wesley.

[21] Knuth, D.E (1984): *The TEXbook*, Addison-Wesley.

[22] Laan, C.G van der (1992a): Syntactic Sugar. MAPS92.2, 130–136. (Submitted to TUG '93.)

[23] Laan, C.G van der (1992b): FIFO & LIFO sing the BLUes. MAPS92.2, 139–144. (To appear TUGboat 14.1. An earlier version has appeared in the EuroTEX '92 proceedings.)

[24] Laan, C.G van der (1992c): Tower of Hanoi, revisited. *TUGboat* 13, no. (1), 91–94. Also: MAPS92.1, 125–127.

[25] Laan, C.G van der (1993): Typesetting number sequences. MAPS93.1. (4 pages. Submitted TUGboat.)

[26] Lamport, L (1986): LATEX, user's guide & reference manual. Addison-Wesley.

[58] To paraphrase Perlis.

[59] With the purpose to program like writing literature. Not only to be processed by computers, but also to be read by humans, with pleasure!

[27] Lamport, L (1987): Makeindex, an index processor for LaTeX. (A clear user's guide for using makeindex—a C program—together with LaTeX.)
[28] Palais, R (1992): Moving a fixed-point. *TUGboat* 13, no. (4), 425–432.
[29] Pittman, J.E (1988): Loopy.TeX. *TUGboat* 9, no. (3), 289–291.
[30] Reid, T.J (1987): Floating figures at the right — and— Some random text for testing. *TUGboat* 8, no. (3), 315–320.
[31] Salomon, D (1989): Macros for indexing and table-of-contents preparation. *TUGboat* 10, no. (3), 394–400.
[32] Salomon, D (1992a): NTG's Advanced TeX course: Insights & Hindsights. MAPS 92 Special. 252p.; revised $\approx 500p$.
[33] Salomon, D (1992b): Index preparation for TeX related documents. MAPS92.2, 111–114. (The adaptation of makeindex for plain is discussed.)
[34] Spivak, M.D (1989): L$\mathcal{AMS}$-TeX. TeXplorators.
[35] Taylor, P (1992): The future of TeX. Proceedings EuroTeX '92. 235–254. (Reprinted in *TUGboat* 13, no. (4), 433–442.)
[36] Winograd, T, B Paxton (1980): An indexing facility for TeX. *TUGboat* 1, no. (x), A1–A12. (The work uses TeX version 1.x and a Lisp program. It does not contain such a clear user guide as for makeindex [27]. It provides numbers in range notation and allows for subentries, and cross-references like see ... and see also .... The program can merge files. The program has been converted into Pascal some years later. The latter version is available on file servers.)
[37] Wirth, N (1976): Algorithms + Data Structures = Programs. Prentice-Hall.
[38] Youngen, R.E (1992): TeX-based production at AMS. MAPS92.2, 63–68.

Appendix A: Heap sort

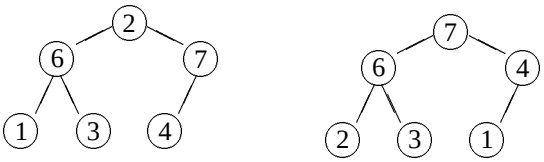
The process consists of two main steps, [2], [20]

- creation of a heap
- sorting the heap

with a sift operation to be used in both.

In comparison with my earlier release of the code in MAPS92.2, I adapted the notation with respect to sorting in *non-decreasing* order.[60]

What is a heap? A sequence $a_1, a_2, \ldots, a_n$, is a heap if $a_k \geq a_{2k} \wedge a_k \geq a_{2k+1}, k = 1, 2, \ldots, n \div 2$, and because a_{n+1} is undefined, the notation is simplified by defining $a_k > a_{n+1}, k = 1, 2, \ldots, n$.
A tree and one of its heap representations of $2, 6, 7, 1, 3, 4$ read



The algorithm. In PASCAL-like notation the algoritm, for sorting the array a[1:n], reads

%heap creation
$l := n$ **div** $2 + 1;$
while $l \neq 1$ **do**
$l := l - 1; sift(a, l, n)$ **od**
%sorting
$r := n;$
while $r \neq 1$ **do**
$(a[1], a[r]) := (a[r], a[1])\%exchange$
$r := r - 1; sift(a, 1, r)$ **od**
%sift #1 through #2
$j := \#1$
while $2j \geq \#2 \wedge (a[j] < a[2j] \vee a[j] < a[2j + 1])$ **do**
$mi := 2j +$ **if** $a[2j] > a[2j + 1]$ **then** 0 **else** 1 **fi**
$exchange(a[j], a[mi])\ j := mi$ **od**

Encoding

Purpose. Sorting values given in an array.

Input. The values are stored in the control sequences `\1`, ..., `\`$\langle n \rangle$. The counter `\n` must contain the value $\langle n \rangle$. The parameter for comparison, `\cmp`, must be `\let`-equal to `\cmpn`, for numerical comparison, to `\cmpw`, for word comparison, to `\cmpaw`, for word comparison obeying the ASCII ordering, or to a comparison macro of your own. (The latter macro variants, and in general the common definitions for `\heapsort`, and `\quicksort`, are supplied in the file `sort.tex`.)

Output. The sorted array `\1`, `\2`, ... `\`$\langle n \rangle$, with `\val1` $\leq$ `\val2` $\leq \ldots \leq$ `\val`$\langle n \rangle$.

Source

```
%heapsort.tex                          Jan, 93
\newcount\n\newcount\lc\newcount\r
\newcount\ic\newcount\uone
\newcount\jc\newcount\jj\newcount\jjone
\newif\ifgoon
%Non-descending sorting
\def\heapsort{%data in \1 to \n
\r\n\heap\ic1
{\loop\ifnum1<\r\xch\ic\r
   \advance\r-1 \sift\ic\r
\repeat}}
%
\def\heap{%Transform \1..\n into heap
 \lc\n\divide\lc2{}\advance\lc1
 {\loop\ifnum1<\lc\advance\lc-1
```

[60] It is true that the reverse of the comparison operation would do, but it seemed more consistent to me to adapt the notation of the heap concept with the smallest elements at the bottom.

```
  \sift\lc\n\repeat}}
%
\def\sift#1#2{%#1, #2 counter variables
 \jj#1\uone#2\advance\uone1 \goontrue
 {\loop\jc\jj \advance\jj\jj
  \ifnum\jj<\uone
   \jjone\jj \advance\jjone1
   \ifnum\jj<#2 \cmpval\jj\jjone
         \ifnum2=\status\jj\jjone\fi\fi
   \cmpval\jc\jj\ifnum2>\status%
                         \goonfalse\fi
 \else\goonfalse\fi
\ifgoon\xch\jc\jj\repeat}}
%
\def\cmpval#1#2{%#1, #2 counter variables
%Result: \status= 0, 1, 2 if
%values pointed by
%              #1 =, >, < #2
 \ea\let\ea\aone\csname\the#1\endcsname
 \ea\let\ea\atwo\csname\the#2\endcsname
 \cmp\aone\atwo}
\endinput                       %cgl@rug.nl
```

Explanation

\heapsort. The values given in \1,...\⟨n⟩, are sorted in non-descending order.

\heap. The values given in \1,...,\⟨n⟩, are rearranged into a heap.

\sift. The first element denoted by the first (counter) argument has disturbed the heap. Sift rearranges the part of the array denoted by its two arguments, such that the heap property holds again.

\cmpval. The values denoted by the counter values, supplied as arguments, are compared.

Examples (Numbers, words)

```
\def\1{314}\def\2{1}\def\3{27}\n3
\let\cmp\cmpn\heapsort
\begin{quote}\prtn,\end{quote}
%
\def\1{ab}\def\2{c}\def\3{aa}\n3
\let\cmp\cmpaw\heapsort
\begin{quote}\prtw,\end{quote}
and
\def\1{j\ij}\def\2{ge\"urm}\def\3{gar\c con}
\def\4{\'el\'eve}\n4
\let\cmp\cmpw {\accdef\heapsort}
\begin{quote}\prtw\end{quote}
```

yields

1, 27, 314,
aa ab c,

and

élève garçon geürm jij.

Appendix B: Quick sort

The quick sort algorithm has been discussed in many places, for example [20]. Here the following code due to Bentley [4], p. 112, has been transliterated.

```
procedure QSort(L,U)
if L<U then Swap(X[l], X[RandInt(L,U)])
   T:=X[L] M:=L
   for I:=L+1 to U do
       if X[I]<T M:=M+1
          Swap(X[M], X[I]) fi
   od Swap(X[L], X[M])
   QSort(L, M-1) QSort(M+1, U)
fi
```

Encoding

Purpose. Sorting of the values given in the array \⟨low⟩, ..., \⟨up⟩.

Input. The values are stored in \⟨low⟩, ..., \⟨up⟩, with $1 \leq low \leq up \leq n$. The parameter for comparison, \cmp, must be \let-equal to \cmpn, for number comparison, to \cmpw, for word comparison, to \cmpaw, for word comparison obeying the ASCII ordering, or to a comparison macro of your own. (The latter macros, and in general the common definitions for \heapsort, and \quicksort, are supplied in the file sort.tex.)

Output. The sorted array \⟨low⟩, ... \⟨up⟩, with \val⟨low⟩ ≤ ... ≤ \val⟨up⟩.

Source

```
%quick.tex                          Jan 93
\newcount\low\newcount\up\newcount\m
\def\quicksort{%Values given in
%\low,...,\up are sorted, non-descending.
%Parameters: \cmp, comparison.
 \ifnum\low<\up\else\brk\fi
%\refval, a reference value selected at random.
 \m\up\advance\m-\low%Size-1 of ar-
ray part
 \ifnum10<\m\rnd\multiply\m\rndval
   \divide\m99 \advance\m\low \xch\low\m
 \fi
 \ea\let\ea\refval\csname\the\low\endcsname
 \m\low\k\low\let\refvalcop\refval
 {\loop\ifnum\k<\up\advance\k1
   \ea\let\ea\oneqs\csname\the\k\endcsname
   \cmp\refval\oneqs\ifnum1=\status
      \global\advance\m1 \xch\m\k\fi
   \let\refval\refvalcop
  \repeat}\xch\low\m
 {\up\m\advance\up-1 \quicksort}%
 {\low\m\advance\low1 \quicksort}\krb}
%
\def\brk#1\krb{\fi}\def\krb{\relax}
\endinput                           %cgl@rug.nl
```

Explanation. At each level the array is partitioned into two parts. After partitioning the left part contains values less than the reference value and the right part contains values greater than or equal to the reference value. Each part is again partitioned via a recursive call of the macro. The array is sorted when all parts are partitioned.

[61] If the array is big enough. I chose rather arbitrarily 10 as threshold.

In the TEX encoding the reference value as estimate for the mean value is determined via a random selection of one of the elements.[61] Reid's [30] `\rnd` has been used. The random number is mapped into the range [low : up], via the linear transformation `\low` + (`\up` − `\low`) ∗ `\rndval`/99.[62]

The termination of the recursion is encoded in a TEX peculiar way. First, I encoded the infinite loop. Then I inserted the condition for termination with the `\fi` on the same line, and not enclosing the main part of the macro. On termination the invokation `\brk` gobbles up all the tokens at that level up to its separator `\krb`, and inserts its replacement text: a new `\fi`, to compensate for the gobbled `\fi`.

Examples (Numbers, words)

```
\def\1{314}\def\2{1}\def\3{27}\n3
\low1\up\n\let\cmp\cmpn
\quicksort
\begin{quote}\prtn,\end{quote}
%
\def\1{ab}\def\2{c}\def\3{aa}
\def\4{\ij}\def\5{ik}\def\6{z}\def\7{a}\n7
\low1\up\n\let\cmp\cmpw
\quicksort
\begin{quote}\prtw,\end{quote}
and
\def\1{j\ij}\def\2{ge\"urm}\def\3{gar\c con}
\def\4{\'el\'eve}\n4
\low1\up\n\let\cmp\cmpw
{\accdef\quicksort}
\begin{quote}\prtw.\end{quote}
```

yields

1, 27, 314,
a aa ab c ik ij z,

and

élève garçon geürm jij.

Appendix C: The file sort.tex

This file contains the common definitions of `\heapsort` and `\quicksort`, the macros for storing, the macros for sorting, the macros for typesetting, some variants for the parameter macros, and the ordering table.

```
%sort.tex                          Jan 93
%Shorthands
\let\ag=\aftergroup
\let\ea=\expandafter\let\nx=\noexpand
%Counters
\newcount\n\newcount\k\newcount\kk\n=0
\newcount\kzero%Start value in prt k-loops
\newcount\pk\newcount\pkone%Used in sortcs
\newcount\frst%First value of range
\newcount\lst %Last value of range
\newcount\slst%Successor \lst
\newcount\dif %Difference \lst-\frst
\newcount\nw  %Number of words
\newcount\nc  %Number of characters/comp
\newcount\numex %Number of exchanges
\newcount\rndval%Random number
\newcount\rndnum%Seed random generator
\newcount\rndtmp%Temporary value
\newcount\status%Status comparison
%Newif-s
\newif\ifcontinue%controls loops
\newif\iffound%locating accent cs
\newif\ifproof\prooftrue
%
%Storing: from copy
\def\seq#1\qes{\k\kzero\fifow#1 \wofif{} }
%Auxiliaries: FIFO
\def\fifow#1 {\ifx\wofif#1\n\k\wofif\fi
 \processw{#1}\fifow}
\def\wofif#1\fifow{\fi}
\def\processw#1{\advance\k1 \ea
 \gdef\csname\the\k\endcsname{#1}}
%
%Storing: from file
\newread\rec
\def\storefrom#1{%#1 is file name
 \openin\rec#1 \k\kzero \continuetrue
 \loop\ifeof\rec\continuefalse\fi
 \ifcontinue\advance\k1 \read\rec to\xyz
  \ea\let\csname\the\k\endcsname\xyz
 \repeat\advance\k-1\n\k\closein\rec}
%
%Storing: random numbers
\def\storerandomn#1{%#1 number of numbers
 \n#1\k0
 \loop\ifnum\k<\n\advance\k1 \rnd\ea
  \xdef\csname\the\k\endcsname{\the\rndval}
 \repeat}
%
%With, due to Reid, 1987
\def\rnd{\global\multiply\rndnum371
 \global\advance\rndnum1
 \ifnum\rndnum>99999
  \rndtmp\rndnum \divide\rndtmp100000
  \multiply\rndtmp100000
  \global\advance\rndnum-\rndtmp
 \fi\global\rndval\rndnum
 \global\divide\rndval1000 }
%
%Storing: random words
\def\storerandomw#1{%#1 number of words
 \n#1\nw\n\def\defarr{\ea\gdef
  \csname\the\nw\endcsname}
 {\loop\ifnum0<\nw{\ag\defarr\ag{%
   \randomword}}\advance\nw-1
 \repeat}}%end s-r-w.
%
\def\randomword{\rnd \nc\rndval
 \divide\nc15 \advance\nc2
 \loop\ifnum0<\nc\randomchar
   \advance\nc-1
 \repeat}%end r-word
%
%Random character is modified
\def\randomchar{\rnd
 \multiply\rndval29 \divide\rndval100
 \ifnum26=\rndval\rndval0 \fi
 \ifnum26<\rndval\rndval4 \fi
%Mod cgl: I \ag-ed the letter
\ea\ag\ifcase\rndval
 a\or b\or c\or d\or e\or f\or g\or h\or
 i\or j\or k\or l\or m\or n\or o\or p\or
 q\or r\or s\or t\or u\or v\or w\or x\or
 y\or z\fi}%end r-char
%
%Typeset
%Parameters: Separators
\def\sepn{, }%Number separator
```

[62] Note that the number is guaranteed within the range.

```
\def\sepw{ } %Word separator
\let\sep\sepw
%
\def\prc#1{\init{#1}\def\prc##1{%
 \ifnum\lst=##1{}\else\ifnum\slst=##1{}%
  \lst\slst\advance\slst1{}\else
  \prtfl\sepn\init{##1}\fi\fi}}
%
\def\init#1{\frst#1\lst\frst \slst\frst
   \advance\slst1 }
%
%Print range: \frst-\lst (or \lst).
\def\prtfl{\the\frst\ifnum\frst<\lst
 \advance\frst1 \ifnum\frst=\lst\sepn
 \else\nobreak--\nobreak\fi\the\lst\fi}
%
%Printing sequences
\def\prts{{\k\kzero%print \1,...\n
 \def\sep{\let\sep\sepw}%
 \loop\ifnum\k<\n\advance\k1
  \sep\csname\the\k\endcsname
 \repeat}}%end \prts
%
\let\prtw\prts
%
\def\prtn{{\k\kzero%Print number sequence
 \loop\ifnum\k<\n\advance\k1
  \ea\prc\csname\the\k\endcsname
 \repeat\prtfl}}%end \prtn
%
\def\typind#1{%#1 a def
 \ea\splittot#1.%
 \ifcase\digit\word\or
         {\tt\word}\or
  {\tt\char92\word}\or
  $\langle\hbox{\word}\rangle$\fi{}
  \pagenrs}
%
\def\splittot#1 !#2 #3.{\def\word{#1}%
  \chardef\digit#2{}\def\pagenrs{#3}}
%
\def\prtind{{\def\\{\hfil\break}\k\kzero
 \def\sep{\let\sep\sepw}%
 \loop\ifnum\k<\n\advance\k1
  \sep\ea\typind\csname\the\k\endcsname
 \repeat}}
%
%Sorting in O(nlog n)
\def\sortn{\let\cmp\cmpn\sort\prtn}
%
\def\sortaw{\let\cmp\cmpaw\sort\prtw}
%
\def\sortw{\let\cmp\cmpw{\accdef\sort}\prtw}
%
\def\sort{\heapsort}
%
%Paramaters: ij and accent string
\def\accstr{\‘\’\"\^\c}
%
\def\accdef{\def\i{i}\def\j{j}%
 \def\’##1{##1a}\def\‘##1{##1g}%
 \def\"##1{##1t}\def\^##1{##1h}%
 \def\c##1{##1c}}
%
\def\ij{ij}
%
%Sorting parameters: exchange macro
\def\xch#1#2{%#1, #2 counter variables
 \ea\let\ea\auxone\csname\the#1\endcsname
 \ea\let\ea\auxtwo\csname\the#2\endcsname
 \ea\global\ea\let\csname\the#2\endcsname
 \auxone
 \ea\global\ea\let\csname\the#1\endcsname
 \auxtwo}
%
%Sorting parameters: number comparison
\def\cmpn#1#2{%#1, #2 are def-s
%Result: \status= 0, 1, 2, if
%        \val{#1} =, >, < \val{#2}
 \ifnum#1=#2\global\status0 \else
   \ifnum#1>#2\global\status1 \else
               \global\status2 \fi\fi}
%
%Parameters: comparison of words
\def\cmpw#1#2{%#1, #2 are def-s
%Result: \status= 0, 1, 2, if
%        \val{#1} =, >, < \val{#2}
 \let\nxt\nxtw\cmpc#1#2}
%
\def\cmpaw#1#2{%#1, #2 are defs with as
%replacement text the words.
%Result: \status= 0, 1, 2, if
%        \val{#1} =, >, < \val{#2}
 \let\nxt\nxtaw\cmpc#1#2}
%
\def\cmpc#1#2{%#1, #2 are def-s
%Result: \status= 0, 1, 2, if
%        \val{#1} =, >, < \val{#2}
\ifproof\global\advance\nc1
        \let\aa#1\let\bb#2\fi
 \global\status0 \continuetrue
 {\loop\ifx\empty#1\continuefalse\fi
       \ifx\empty#2\continuefalse\fi
  \ifcontinue\nxt#1\nxtt\nxt#2\nxtu
             \lge\nxtt\nxtu
  \repeat}\ifnum0=\status
 \ifx\empty#1\ifx\empty#2\else
                  \global\status2 \fi
 \else\ifx\empty#2\global\status1 \fi
 \fi\fi
 \ifproof\immediate\write16{\aa
  \ifnum0=\status=\else
   \ifnum1=\status>\else
                  <\fi\fi\bb.}
 \fi%end ifproof
}
%
\def\lge#1#2{%#1 and #2 letter values
%Result: \status= 0, 1, 2, if
%             #1 =, >, <  #2.
%and \continuefalse if #1=/#2.
 \ifnum#1=#2{}\else\continuefalse
  \ifnum#1<#2\global\status2 \else
             \global\status1 \fi
 \fi}
%
\def\nxtw#1#2{\def\pop##1##2\pop{%
 \gdef#1{##2}\def\head{##1}}%head and tail
 \ea\pop#1\pop%split in head and tail
 \ea\loc\head\accstr%\head is an accent cs?
 \iffound\let\acs\head
  \ea\pop#1\pop%next head and tail
  \ea\let\ea#2\csname ot\acs\head\endcsname
 \else\ea\let\ea#2\csname ot\head\endcsname
 \fi}
%
\def\loc#1#2{\def\locate##1#1##2\end
 {\ifx\empty##2\empty\foundfalse
 \else\foundtrue\fi}\ea\locate#2.#1\end}
%
%Parameters:  for ASCII words
\def\nxtaw#1#2{%Result: value of first
%letter of string supplied in #1 is delivered
%in #2. (To be used as a number (\chardef)).
%#1, #2 are control sequences.
 \def\pop##1##2\pop{\gdef#1{##2}%
  \chardef#2‘##1{}}\ea\pop#1\pop}
%
```

```
\def\cmpir#1#2{%#1, #2 defs
%Result: \status= 0, 1, 2 if
%        \val{#1} =, >, < \val{#2}
 \ea\ea\ea\decom\ea#1\ea;#2.}
%
\def\decom#1 !#2 #3;#4 !#5 #6.{%
 \def\one{#1}\def\four{#4}\cmpaw\one\four
 \ifnum0=\status%Compare secondary keys
   \ifnum#2<#5{}\global\status2 \else
     \ifnum#2>#5{}\global\status1 \else
                %Compare tertiary keys
       \ifnum#3<#6{}\global\status2 \else
         \ifnum#3>#6{}\global\status1 \fi
       \fi
     \fi
   \fi
 \fi}
%
\def\red{%Reduction of \1,...,\n
 \k0\kk0\let\refer\empty
 \loop\ifnum\k<\n\advance\k1
  \ea\let\ea\record\csname\the\k\endcsname
  \ea\splitwn\record.%
  \ifx\refer\word%extend with number
    \ea\xdef\csname\the\kk\endcsname{%
       \csname\the\kk\endcsname, \num}%
  \else%write record to \kk
   \advance\kk1\let\refer\word\ea\global
   \ea\let\csname\the\kk\endcsname\record
  \fi
 \repeat\n\kk}
%
\def\redrng{%Reduction of \1,...,\n, with
%range representation of page numbers
 {\k1\kk0
 \ea\let\ea\record\csname\the\k\endcsname
 \ea\splitwn\record.\let\refer\word
 \let\nrs\empty\prcrng\num
 \loop\ifnum\k<\n\advance\k1
  \ea\let\ea\record\csname\the\k\endcsname
  \ea\splitwn\record.%
  \ifx\refer\word%extend \nrs with number
    \prcrng\num
  \else%write record to \kk
    \advance\kk1 \strnrs
    \ea\xdef\csname\the\kk\endcsname{\refer{}
      \nrs}\let\nrs\empty\init\num\prcrng\num
    \let\refer\word
  \fi
 \repeat\ifnum1<\n
  \advance\kk1 \strnrs
  \ea\xdef\csname\the\kk\endcsname{\word{}
    \nrs}
 \global\n\kk\fi}}
%
\def\prcrng#1{\init{#1}\def\prcrng##1{%
 \ifnum##1=\lst\else\ifnum##1=\slst
  \lst\slst\advance\slst1 \else
  \strnrs\init{##1}\fi\fi}}
%
\def\strnrs{\dif\lst\advance\dif-\frst
 \edef\nrs{\ifx\nrs\empty\else\nrs\sepn\fi
  \the\frst\ifnum0<\dif
   \ifnum1=\dif\sepn\the\lst
   \else\nobreak--\nobreak\the\lst
   \fi
  \fi}}
%
\def\splitwn#1 !#2 #3.{\def\word{#1 !#2}%
 \def\num{#3}}
%
\def\getdig#1 !#2 #3.{\def\dig{#2}}
%
\def\sortcs{\global\k0\global\pk\n
\global\pkone\pk\global\advance\pkone1
%Invariant: 1:k non-cs; pk+1:n control seq-s
\loop\global\advance\k1
\ifnum\k<\pkone
 \ea\ea\ea\getdig\csname\the\k\endcsname.%
 \if2\dig{\continuetrue
  \loop
   \ifnum\k=\pk\continuefalse
   \else\ea\ea\ea\getdig\csname\the\pk
                          \endcsname.%
    \if2\dig\else\xch\k\pk\continuefalse\fi
   \fi\global\pkone\pk\global\advance\pk-1
  \ifcontinue
  \repeat}%
 \fi
\repeat}%Result\1:\pk non-cs, \pkone:\n cs
%
%Parameters: Ordering table
\chardef\ota32 \chardef\otA32
 \chardef\otaa33 \chardef\otag33
 \chardef\otat34 \chardef\otah35
\chardef\otb39 \chardef\otB39
\chardef\otc46 \chardef\otC46
 \chardef\otcc47 \chardef\otcc47
\chardef\otd53 \chardef\otD53
\chardef\ote60 \chardef\otE60
 \chardef\otea61 \chardef\oteg62
 \chardef\otet63 \chardef\oteh64
\chardef\otf67 \chardef\otF67
\chardef\otg74 \chardef\otG74
\chardef\oth81 \chardef\otH81
\chardef\oti88 \chardef\otI88
 \chardef\otit91 \chardef\otih92
\chardef\otj95 \chardef\otJ95
 \chardef\otjt98
\chardef\otk102 \chardef\otK102
\chardef\otl109 \chardef\otL109
\chardef\otm116 \chardef\otM116
\chardef\otn123 \chardef\otN123
\chardef\oto130 \chardef\otO130
 \chardef\otoa131 \chardef\otog132
 \chardef\otot133 \chardef\otoh134
\chardef\otp137 \chardef\otP137
\chardef\otq143 \chardef\otQ143
\chardef\otr150 \chardef\otR150
\chardef\ots157 \chardef\otS157
\chardef\ott164 \chardef\otT164
\chardef\otu171 \chardef\otU171
 \chardef\otut174 \chardef\otuh175
\chardef\otv178 \chardef\otV178
\chardef\otw185 \chardef\otW185
\chardef\otx192 \chardef\otX192
\chardef\otij199 \chardef\otIJ199
\chardef\oty200 \chardef\otY200
\chardef\otz206 \chardef\otZ206
\endinput                  %cgl@rug.nl
```

Appendix D: The file sort.tst

Writing macros is one thing and testing another. I find testing software as difficult as writing a variant from scratch. For convenience I have provided my (plain) testdriver below.

The test path—Which sorting worker? Tracing on/off? How many random data?—is determined in a dialogue with TEX. Rudimentary, but useful.

```
%sort.tst                          Jan 93
%Separately needed is index.tex, as data.
\input sort.tex
\input heap.tex
```

```
\input quick.tex
\immediate\write16{Heap sort as sorter? (y/n):}
\read16 to\yesorno
\if y\yesorno Heap sort.
   \def\sort{\heapsort}
\else Quick sort.
   \def\sort{\low1\up\n\quicksort}
\fi (\number\day/\number\month/\number\year)
\immediate\write16{Proofing/Tracing? (y/n):}
\read16 to\yesorno
\if y\yesorno\prooftrue
 \nopagenumbers\tracingmacros2
\else\prooffalse
\fi

\smallskip Numbers.\par
\seq314 1 27\qes
Input: \prtn.\par
Result:\sortn.
\immediate\write16{Result: \1, \2, \3,
           ...\csname\the\n\endcsname.}

\smallskip Words (ASCII).\par
\seq a b aa ab bc bb aaa\qes
Input: \prtw.\par
Result: \sortaw.
\immediate\write16{Result: \1, \2, \3,
\4, \5, \6,...\csname\the\n\endcsname.}

\smallskip Words.\par
\seq a b aa ab bc bb aaa\qes
Input: \prtw.\par
Result: \sortw.
\immediate\write16{Result: \1, \2, \3,
\4, \5, \6 ...\csname\the\n\endcsname.}

\smallskip Accented words.\par
\def\1{z}\def\2{c}\def\3{\'a\'a}\def\4{\"ab}
\def\5{ge\"urm}\def\6{ge\"{\i}nd}
\def\7{gar\c{c}on}\def\8{a}\def\9{ge{\ij}kt}
\n=9
Input: \prtw.\par
Result: \sortw.
\immediate\write16{Result: \1, \2, \3,
           ...\csname\the\n\endcsname.}

%Test and timing: random generated elements
\smallskip Sort numbers.\par
\immediate\write16{Give seed for r-generator:}
\read16 to\seed
\immediate\write16{Give maximum of num-
bers to be
   generated:}
\read16 to\total \n\total
Seed=\seed. \rndnum\seed
\storerandomn\n \par
Input: \prtn.\par
Result: \sortn.
\immediate\write16{Result: \1, \2, \3,
           ...\csname\the\n\endcsname.}

\smallskip Sort words. \par
\immediate\write16{Give seed for r-generator:}
\read16 to\seed
\immediate\write16{Give maximum of words to be
   generated:}
\read16 to\total \n\total
Seed=\seed. \rndnum\seed
\storerandomw\n \par
Input: \prtw.\par
Result: \sortw.
\immediate\write16{Result: \1, \2, \3,
           ...\csname\the\n\endcsname.}

\smallskip Sort index reminders.\par
\storefrom{index.tex}
{\def\\{\hfil\break}\let\sepw\\
\let\cmp\cmpir\k0\kk0 \null
\hfil\vtop{\hsize2.25cm\noindent
  Data:\sepw\prtw}
\hfil\vtop{\hsize2.5cm\sort\noindent
  After sorting:\sepw\prtw}
\hfil\vtop{\hsize3.5cm\redrng\noindent
  After reduction:\sepw\prtw}
\hfil\vtop{\hsize3cm\noindent
  Typeset:\sepw\prtind.}
\immediate\write16{Index rem: \1, \2, \3,
           ...\csname\the\n\endcsname.}

\smallskip Frambach's example.\par
\def\1{wd !2 7}\def\2{wrd !1 1}
\def\3{wd !2 2}\def\4{a !1 1}
\def\5{wd !2 5}\def\6{wd !2 3}
\def\7{z !3 7}\def\8{wrd !1 5}
\def\9{wd !2 1} \n9
\let\sepw\\\null
\hfil\vtop{\hsize2cm\noindent
 Data:\sepw\prtw}
\hfil\vtop{\hsize2.5cm\sortcs\noindent
 After splitting:\sepw{\n\pk\prtw}
  \sepw\kzero\pk\prtw}
\hfil\vtop{\hsize3cm\let\cmp\cmpir
 {\low1\up\pk\quicksort}
 {\low\pkone\up\n\quicksort}\noindent
 After sorting\sepw both parts,\sepw
 and compressing:\sepw\redrng\n4 \prtw}
\hfil\vtop{\hsize3cm\noindent\n4
  Typeset:\sepw\prtind.}
}
\immediate\write16{EF's exam: \1, \2, \3,
           ...\csname\the\n\endcsname.}
\bye                          cgl@rug.nl
```

Appendix E: Contents

Manmac BLUes

or how to typeset a book via TEX

Kees van der Laan

Hunzeweg 57,
9893 PB Garnwerd, The Netherlands
cgl@rug.nl

Abstract

The manmac macros are enumerated. A user's guide is provided, and the encodings are explained. As enhancements the writing of index reminders to the file `index.tex` is elaborated upon, and how to incorporate AMS fonts and non-CM fonts is referred to.
In the appendixes I provided the source of manmac and my personalized report template. With respect to the latter, I played with the idea of formatting the MAPS specials series in this way.

Keywords: Computer-assisted typography, manmac, style/format, customizing, index preparation, plain TEX, fonts, macro writing, education.

Introduction

This is the first article in a series, where basic, general, and public domain macro collections are discussed. In progress are AMS BLUes, and TUGboat BLUes.

Manmac is the collection of macros, used by Don Knuth, to format the TEXbook.[1] They are discussed as an example format in the TEXbook, Appendix E, p. 412–425. The encodings comprise some 10 pages.

Manmac is a gem. Everybody should *know* about it. It contains handy macros, and it demonstrates various macro writing TEXniques. The macros are not alluded to in the index of the TEXbook, alas, so it is hard to find out about their existence.

Why? For quite a while I have looked for a simple collection of macros to typeset my day-to-day notes.

I started with LATEX and even typeset a book with it.[2] I also exercised LATEX with respect to adaptation,[3] but experienced LATEX as too complex and inconsistent, and too time-consuming to customize. For a Ben Lee User[4] type LATEX is there and ready *for use*, for sure.

$\mathcal{AMS}$-TEX has its virtues, if not for the excellent user's guides, and extensive handy math fonts. Its compatibility with LATEX, via $\mathcal{AMS}$-LATEX, is another strong point. But, ... it is too much biased towards Spivak's modifications of plain's math mark-up.

Spivak's L$\mathcal{AMS}$-TEX is beautiful but complex. Another aspect is—like LATEX—that it is a superseding package. Not a toolbox to cooperate with whatever is already in use. Moreover, the math mark-up is biased towards $\mathcal{AMS}$-TEX. The encodings of commutative diagrams and tables are very well developed. Also his scheme for general symbolic cross-referencing is very good.

I like Amy Hendrickson's MacroTEX-toolbox approach very much, but, ... alas she has not finished it, and it is not in the public domain.

`TUGboat.sty` and `LTUGboat.sty` are appealing. A wealth of experience gained in high-quality computer-assisted typesetting has been embodied. But, ... it is not basic, and not designed from one logical viewpoint as root. It has plain and LATEX user interfaces. From the early days of TUG it has grown out as a format upon plain alone. Only quite recently it has been mirrored into a LATEX style.[5]

Doob's macros for typesetting his 'Gentle introduction ...' were considered, certainly, but they were too limited, and not what I was looking for.

[1] Also included in the file is the handful of extra macros for typesetting the Metafont book.

[2] 'Publiceren met LATEX.' CWI Syllabus 19 (Dutch).

[3] `trspar.sty`, for making transparencies via LATEX. This is different from SliTEX because the same structuring control sequences as those provided by the common LATEX styles are used (the same input), and because I encoded to distill intelligent headers and footers from the arguments of the commands, automatically.

[4] BLU for short, alias Beginning LATEX User.

[5] Personally, I use these styles heavily. Especially in relation with writing in TEX about (LA)TEX.

> To return to the roots and from there understand all that has been built upon plain, that is my aim, that serves a lifetime.

And so after wandering around for quite a while, I returned to the roots. To manmac, Knuth's macros for formatting the TEXbook. Perhaps, I appreciated his macros more after having studied the epigons.

Mimi Jett and Daniel Olson, EPS,[6] arrived earlier at the same conclusion: their[7] modifying manmac workshops are a great idea! But, . . . should a user start from there nowadays?

My thesis is that

- a publisher should start from what AMS provides—they do their complete production by ($\mathcal{AMS}$)-(LA)TEX, some 100,000 pages per year
- a self-publishing author[8] can better start from the `tugboat.sty`-les
- a hacker should carefully study Knuth's encoding of manmac.

If only there had been a user's guide for manmac, and ipso facto for the other example formats of DEK, then the (LA)TEX world would have looked much different from what it is today.

What is on? In the section Everybody should know about manmac, I give the reasons why, and summarize the funtionalities provided in manmac. The section BLU's needs, provides a user's guide. In The Grandwizard at work, I'll discuss DEK's encodings—the hows-and-whys, and provide now and then some alternative encodings. In the Enhancements section the writing of more general index reminders to a file is elaborated upon, and the inclusion of AMS and non-CM[9] fonts is alluded to. I supplied in Appendix A a listing of manmac.tex—DEK's macros—and in Appendix B the template report based on manmac and Knuth's sample in the TEXbook p. 340–341. I end up with a table of contents.

Notations. `\ea`, `\nx`, and `\ag`, are used as shorthand notations for `\expandafter`, `\noexpand`, respectively `\aftergroup`. ⟨cr⟩ means the ASCII carriage return (and line feed), better known as `^^M` in the TEX-arcana. Some subsections start their title systematically with: mark up, encoding, respectively mod. These stand for

- mark up: how to use it, to insert the mark-up
- encoding: an explanation of how it has been encoded
- mod: concerns a modification or alternative.

1 Everybody should know *about* manmac!

Even the Ben Lee User type. Knuth's approach is that he does not separate user guidelines from explaining the encoding. It is all there, mixed, and too difficult for BLU, I suppose.[10] BLU easily loses the wood for the trees, with the pragmatic but wrong consequence, that *everybody* neglects manmac.

But, . . . also the gurus should exercise patience and study the macros carefully, instead of being carried away by the result of the (fancy) format of their own.

> Manmac embodies a lot of subtle macro writing TEXniques, which can be applied to other contexts too!

I found the encoding of the right-justification of the quotation paragraphs, an eye-opener. Then there are the two-part macros, the alignment display,. . . and even how to shipout a subset of pages.[11]

What is it all about? It is difficult to classify the various items. In want for a better system I like to think in terms of

- logical elements
- page make-up consequences.

Manmac consists of macros on top of plain for

a. loading of fonts
b. size-switching macros
c. mark up of logical elements (chapter, subsection, exercise (and answer), indented displays (math, verbatim), syntax and code charts, . . .)
d. page make-up (A modified OTR, for titlepage handling, and the typesetting of the index. Running heads (no footers))
e. miscellaneous utilities (Shipping out selected pages, some handy macros).

For the beginner there are `\oct#1{...}` and the like. This macro typesets the (number) argument with the octal ´ tag before the number. Other handy macros are

- `\bull`, square bullet (an appropriate `\vrule`)
- `\|`, vertical line (`\char'\|` from `\tt` font)
- `\dn`, downward arrow (`\char'14` from `\tt` font)
- `\up`, upward arrow (`\char'13` from `\tt` font)
- `\<...>`, syntactic quantity[12]
- `\[...]`, keyword (for example: to, by, . . .)
- `\cstok`, control sequence token (as in the TEXbook)
- `\parbreak`, to break a paragraph

[6] Electronic Technical Publishing services company.
[7] They started a workshop with the title 'Modifying Manmac,' at the Boston TUG '91 meeting.
[8] Who likes to understand and customize what is used as tool. For example a user's group with its bulletin.
[9] Computer Modern, DEK's parameterized fonts generated by Metafont.
[10] The great step forward of LATEX is the general available user's guide.
[11] Yes, I know of and use the tool `dvitodvi`. Now I understand the process, and DEK's way is more efficient.
[12] With `\beginsyntax`. . . `\endsyntax` the <, and [, are made active.

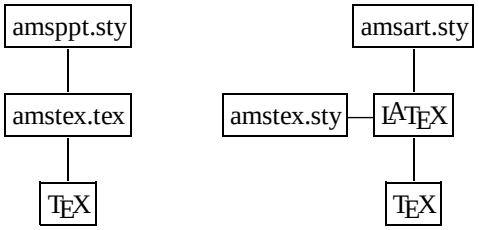
- `\arrows`, ⟵ text ⟶ via `\arrows{2cm}{text}`
- `\hidehrule`, `\hidevrule`, for rules with reference point left invariant
- `\makeblankbox`, puts rules at the end of a blank box whose dimensions are those of `\box0` (assumming nonnegative wd, ht, dp)
- `\samplebox`, the outline of a box, with `\bigdot` as reference point
- `\sampleglue`, makes glue between sample boxes
- `\setcornerrules`, for positioning of paste-up items.

Not in there. There are no facilities provided for automatic symbolic or cross-referencing,[13] handling of graphics (pictures),[14] table of contents preparation (or list of tables, figures, bibliography, ...), floating (island) objects different from plain's `\midinsert`, and `\topinsert`, marginal note facilities different from plain's use of `\vadjust`, and endnote macros. Most of the above functionalities have been encoded already by the TEX community at large and are available from the file servers. For a survey of what is available see Jones' macro index, or Nelson Beebe's TUGlib, for example.

Appetizer. Of the above macros from manmac, `\makeblankbox` and its descendants can be used for simple schemes as shown below.[15] Via

```
$$\hbox{\vbox{%
\element{\fbox{amsppt.sty}}
\vconnector
\element{\fbox{amstex.tex}}
\vconnector
\element{\fbox{\TeX}}
}\qquad\qquad\qquad\vbox{%
\element{\fbox{amsart.sty}}
\vconnector
\element{\llap{\fbox{amstex.sty}---}
 \fbox{\LaTeX}}
\vconnector
\element{\fbox{\TeX}}
}}$$
```

I obtained



with the auxiliaries

```
\def\strut{\vrule height2.5ex depth1ex
 width0pt}
\def\fbox#1{\setbox0\hbox{\strut
 $\;$#1$\,$}\leavevmode\rlap{\copy0}%
 \makelightbox}
\def\element#1{\hbox to15ex{\hss#1\hss}}
\def\vconnector{\element{\strut\vrule}} .
```

2 BLU's needs: a user's guide

A real user's guide needs a more complete and elaborated documentation, with simple examples and an attractive layout. However, the completeness is a difficult issue, because it is so closely connected to plain, which is also deprived from a user's guide.

I like to look at a book as consisting of front matter, main part, and back matter, in the SGML spirit.

2.1 Front matter

Things like cover page, bibliographic data, preface, and table of contents are easily[16] added when the book is finished, either by the author or the publisher. So there will be nothing about that in the user's guide.

2.2 Back matter

Knuth adopted as back matter the appendixes. These appendixes follow the same structure as the chapters, except for Appendix I, the index (fancy two-column intermezzo), and Appendix B (subsections). Appendix A is a (numbered) anthology of the answers of the exercises. Apart from the two-column typesetting the preparation of an index is complex, because of the many tools involved and the special knowledge required. BLU, and even the guru, can better rely on his publisher for this.

[13] It is true, however, that in the code of manmac there are some symbolic names for numbers—for example `\sesame` for the page number of the Sesam street quote—which DEK apparently referred to, abstracting in this way from the places where the references were inserted. Spivak in his LAMS-TEX has provided a nice and general complex for handling cross-references. In my Math into BLUes, I adopted a very simple approach for handling automatic and symbolic cross-referencing.

[14] In the file there are some macros for drawing figures: arrows and frames. Furthermore, there exist various drawing packages, for example PICTEX, to name but one.

[15] Once I realized this, I don't need LATEX's picture environment so much, making my TEXing life simpler.

[16] Provided that the page numbers of the main part are independent of the page numbers of the front matter.

2.3 Main part: The chapters

The book is divided into chapters, all more or less independent. Each chapter consists of a title page,[17] several paragraphs, and some quotations, to end up with.

The copy must obey the structure

```
\beginchapter⟨title matter⟩\par
⟨copy proper⟩
\endchapter
⟨quotation-s⟩
\eject
```

Mark up: `\beginchapter`. The ⟨*title matter*⟩ must comply with the definition

```
\outer\def\beginchapter#1 #2#3. #4\par{...}
```

with

#1	Chapter (Appendix, or ...)
#2#3	ordered characters (12, XI, A, ...)
#4	the title.

(The title page and the running heads are now handled automatically. Note the use of the various parameter separators to terminate the arguments, and the generality of the parameters which make the macro flexible and suitable for handling similar but different situations. But we are preoccupied with strong typing isn't?)

Mark up: ⟨*copy proper*⟩**.** Nearly all chapters are just a chain of paragraphs.[18] There is however a control sequence `\subsection`.[19] Its use must comply with the definition

```
\outer\def\subsection#1.{...}
```

with as argument the title. The numbering is handled automatically, as is the terminating period.[20]

Mark up: fonts. Manmac uses in addition to the fonts which come along with plain the following
tentex the TEX character set as supplied in Appendix C of the TEXbook, p 369 (file: cmtex10)
inchhigh the inch-sized chapter numbers (file: cminch)
tenu the unslanted text italic font (file: cmu10)
manual special (sized) symbols for the METAFONT[21] logo, and the dangerous bend (file: manfnt)
cmman miscellaneous Computer Modern variations (file: cmman).

An example is provided by the mark-up of the first words of the preface

```
\noindent\hang\hangafter-2
\smash{\lower12pt\hbox to 0pt{\hskip-
 \hangindent\cmman G\hfill}}\hskip-16pt%
 {\sc ENTLE} R{\sc EADER}: \strut This...
```

with from Appendix B of the TEXbook

```
\def\hang{\hangindent\parindent}
```

Mark up: paragraphs. In the paragraphs of the chapter we can use all what has been provided by plain and the special macros of manmac.

Provided are the environments

- `|...|` typesets `...` as characters. Its use—as in `tugboat.sty`—turns the special characters (`\`, `{`, `}`, `$`, `&`, `#`, `%`, `~`, `_`, `^`) into normal symbols (category other). A so-called inline verbatim. As example `|\abc|` yields `\abc`.
- `\begindisplay...\enddisplay`, an indented display. For an example of use see the TEXbook, p. 420. `\obeylines` is on. This can be used for non-centered (indented) displayed math.
- `\begintt...\endtt`, for (indented) displays entirely in typewriter type. For an example of use see the TEXbook, p. 420. `\obeylines` is on. (It is much like LATEX's `\begin{verbatim}...\end{verbatim}`. `tugboat.sty` has enlarged its functionality by some options.)
- The `|` is an escape character, and permits escape out of the verbatim mode (actually, it ends the group).[22] Some experienced authors will undoubtedly raise the question: 'How to typeset the `|` itself within such a context?'
  The answer is we can't! But, ... we can end the group, do whatever we have to do, and go back into verbatim mode as follows
  - end the scope of the redefinitions via `|`, which is equal in this context to `\endgroup`
  - typeset `|` via `{\tt\char'174}`, and
  - go back to the inline verbatim situation via `\ttverbatim`, or `|`.[23]
- `\beginlines...\endlines`,
  a (zero-indented) display, which obeys the line structure and inserts a `\hrule` for and aft. For an example of use see the TEXbook, p. 421. `\obeylines` is on.[24]

[17] With a nice lion illustration as running gag.

[18] This makes the work of Wittbecker, 1988, and Eijkhout, 1990, more interesting. Wittbecker surveys the usual paragraph shapes, and Eijkhout supplies another unusual one. DEK's quotations at the end of each chapter are another example of unusual paragraph shapes.

[19] Not to confuse with plain's `\beginsection`, TEXbook, p. 340.

[20] It is first gobbled because of being a parameter separator. Later it is reinserted. Subsection titles ending with a ! or ? need special attention.

[21] See the `\MF` definitions in the manmac.tex file.

[22] `TUGboat.sty` is more general in the sense that it allows for an option to specify the escape character.

[23] The latter character acts like a toggle, similar to $ for math.

[24] Variant `\endlines` are `\weakendlines`, `\finalendlines`. The former uses a `\medskip` instead of a `\medbreak`.

- `\beginmathdemo...\endmathdemo`, (and variants) indented display for math (Chapters 16–19).[25] They are used to typeset the marked up copy and the typeset result side-by-side (in the one-column format). The user does not have to bother about the template for the used `\halign`, as alignment display. The example on TEXbook p. 132, can be obtained via

  ```
  \beginmathdemo|$x+y-z$|&
          x+y-z\cr\endmathdemo
  ```

  The functionality provided is similar to the temporarily switching from one-column into two-column format. But, ... I guess that the above macro is much simpler and more efficient, because it does not entail the complicated OTR processing.

  > (LA)TEX history has it that the manmac gem has been neglected, especially by the LATEX community. This is understandable given the widespread LATEX's user's guide, in contrast with the lack of a 'manmac' user's guide.

  If only I had been aware of this macro, the mark-up of my Table Diversions paper would have been much simpler.
- `\beginsyntax...\endsyntax`, for formal syntax (Chapters 24–26). The syntax elements are specified enclosed by the brackets `<...>`. The alternatives are separated by `\alt`. Reserved words are enclosed by **[** and **]**. The diagrams allow for index reminder mark-up via carets,[26] as does other structures except when in math mode.
- `\beginchart...\endchart`, for the font tables of Appendixes C and F, etc.[27] For its use see the ASCII table example with the data separately provided in manmac via the replacement text of `\def\normalchart{...}`. The invokation reads[28]

  ```
  \beginchart{}\normalchart\endchart
  ```

  The result is shown on p. 367. So nice to have the data separated from the encoding of the table, and the template hidden.[29]
- `\exercise...`, `\dangerexercise...`, or `\ddangerexercise`, with immediately after the exercise the `\answer...`. A blank line terminates the answer. The latter is stored.[30] How to verify the typesetting of the answers is shown in the 'galley,' TEXbook, p. 425.

Mark up: ⟨*quotations*⟩**.** The quotations are supplied as paragraphs with the specification of the author in compliance with the definition

```
\def\author#1(#2){...}
```

with as parameters

#1 the author name
#2 the year.

In the mark-up of the TEXbook the quotations have been separated by a `\medskip`.

Mark up: index reminders. Knuth implemented the handling of the following 5 kinds of index entries.

Mark up	Typeset in copy*	In `index.tex`
`^{...}`	...	`... !0` ⟨*page no*⟩
`^^{...}`	'silent'	`... !0` ⟨*page no*⟩
`^|...|`	`|...|`	`... !1` ⟨*page no*⟩
`^|\...|`	`|\...|`	`... !2` ⟨*page no*⟩
`^|<...>|`	⟨...⟩	`... !3` ⟨*page no*⟩

* `|...|` denotes manmac's, inline verbatim.

From this table we distill that DEK first prepared the copy, TEXed it, proofed it, and not until the endproofs phase inserted the ^ (circumflexes) to mark up the index reminders.

To mark up for writing to the index file there is the circumflex, ^. Silent index reminders—which are not typeset in the copy—take a double circumflex, `^^{...}`.[31] The difficulty is not the writing to the file, but the selection, post-processing and final typesetting. The latter is dealt with in the grandmaster chapter of the TEXbook, p. 261–264.

2.4 Shipping out selected pages

The advantage of this efficient macro is that a smaller `.dvi` file will result, as opposed to the interactive post-processing utility `dvitodvi`, which selects the pages from the complete `.dvi` file. In order to use this facility one simply has to supply the required page numbers in the file `pages.tex`, each page number on a line. That is all. A handy tool!

That is about all from BLU's viewpoint. The structure of the mark-up of a chapter has been given in the TEXbook, p. 418, and how to process each chapter separately is supplied on p. 425. The complete TEXbook copy—one big example of how to mark up copy—is available from the file servers.

[25] There is only one second part macro for all these cases. Its replacement text reads `\egroup$$`.
[26] According to the TEXbook p369, we should talk about 'circumflex' or 'hat,' because the caret is a larger symbol.
[27] The macro is also used in `testfont.tex` which comes along with Metafont.
[28] Remind the parameter of `\beginchart`.
[29] In my Table Diversions paper, EuroTEX '92, I have also adopted the approach to separate the encoding of the macro from the specification of the data, but not in this TEX-specific way. Another eye-opener!
[30] The answer is not typeset at the place corresponding to where it occurs in the compuscript!
[31] Note that it is not possible to write an index reminder within math mode!

3 The Grandwizard at work

To begin with Knuth knew what he was up to. He does not need safeguarding goodies, like a warning when a chapter is not closed by an `\endchapter`, and the like. Preventive safeguards are not implemented in manmac.

> Although the TEXbook is full of dangerous bends there is no safety-rail along the road through it.

Are there debugging tools provided? Yes, there are! And very fine ones too, once you have learned how to use them. I quite often use `\tracingmacros=1`, to find out what actually has parsed as arguments to the macros.

3.1 Encoding: `\beginchapter`

The numbering of subsections and exercises has been done automatically and relative to the (user) provided chapter 'number.' Peculiar is that for this chapter number two parameters are used.[32] This is done to account for the spacing in between, when typsetting the huge two digits on the title page. The title is typeset via an `\halign`, which allows for a title consisting of a dynamical number of (right-justified) lines.[33] In the mark up each line has to be ended by `\\`. Also heavily used is TEX's redefinition capability, to let control sequences execute context dependent actions. An example at hand is `\\`, which acts like a ␣ in the running heads—therefore an (expanded) `\xdef\rhead{...}` was needed—and as a `\cr` within the `\halign`. This context dependent-ness entailed to abstract from `\cr` into `\\`.

3.2 Encoding: `\endchapter`

This is a nice example of the second part of a two-part macro. We are used to providing the chapter contents via the argument of a routine when nowadays high-level programming languages are used for text processing.

> Two-part macros behave differently. The first part sets up the environment, then the contents is provided, and finally the invokation of the second part finishes the environment, en-passant doing whatever has to be done.

For the TEXbook the purpose of `\endchapter` is to take care of proper typesetting of the quotations. First, the paragraphs of the ⟨*copy proper*⟩ are ended. Then for an even-numbered page a `\vfill\eject\null` is invoked followed by opening a group and a `\vfill`, in order that the quotations will be typeset at the bottom of the odd-numbered pages. Then the parameters for the quotation paragraphs are adjusted such that each paragraph is typeset, right-justified. The latter has been realized via

```
\leftskip0pt plus 40pc minus\parindent
```

together in conjunction with `\parfillskip0pt`.[34] Concise and elegant!

Because of the `\obeylines`, the input can be natural here, line by line. (No terminating `\\` or so, just the ⟨cr⟩.)

The specification of the author is again natural: the name with the year in parentheses, in compliance with the author definition given above. The justification is handled automatically, because the group is closed by `\eject`, which has been redefined to do so within `\endchapter` via `\def\eject{\endgroup\eject}`.[35] Neat!

3.3 Encoding: `\subsection`

Nothing special. It closes a paragraph (via `\medbreak`), handles automatically the numbering and typesets the title as the beginning of a new paragraph. That is all.

Similar is the typesetting of the 'title' of each answer. The answer numbers are maintained automatically and typeset as the beginning of the answer.

3.4 Encoding: `\begindisplay...\enddisplay`

Again a nice example of encoding two-part macros. `\obeylines` is on. `\halign` is used as an alignment display, TEXbook, p. 190. An alignment display is different from the use of `\halign` for tables. Note that despite the `$$`-s an alignment display is not processed within math mode! The opening `$$`⟨*assignments*⟩`\halign\begingroup...` is separated from its closing `\endgroup$$`, via the two-part macro TEXnique.

The parameter for the `\startdisplay` macro is passed on in a TEX peculiar way. It looks like that `\begindisplay` does not take parameters, and indeed it doesn't, but `\startdisplay` does, and this macro is invoked at the end of the replacement text of `\begindisplay`. Because `\startdisplay` uses the ⟨cr⟩ as end separator, all that occurs on the line after `\begindisplay` will become the argument of `\startdisplay`. This argument is reinserted between `$$` and `\halign{...` and has the syntactical meaning of 'assignments,' of the alignment display, see the TEXbook, p. 190.

`\enddisplay` inserts `\crcr`—allowing users to supply or forget about the ending `\cr`—ends the group and the alignment display environment.

[32] That is a maximum of two digits is accounted for.

[33] In actual use the number of title lines varied from 1 to 3.

[34] There is a funtional similarity here with how the title lines are typeset.

[35] Note that this is not an (infinite) recursive definition. After ending the group the original meaning of `\eject` is on, no recursion at all. Contexts!

3.5 Encoding: \begintt...\endtt

The vertical display skips (\abovedisplayskip, \belowdisplayskip) are on via $$. \begintt invokes \ttverbatim, which changes the catcodes of some (the special) characters into category 12—other—and invokes \obeyspaces, \obeylines, and \tt. The things one would expect to be set for verbatim mode.

The vertical bar is made into the escape character (category 0). To allow for overflow at the right \rightskip-5pc is adjusted.[36] The worker \ttfinish takes the first line and the rest as arguments and ends the group and the (math) display. The second argument (=rest) is put in a vbox, and therefore can't be split over a page boundary.

3.6 Encoding: \beginlines...\endlines

\obeylines is on. The idea is that the copy, with a \hrule for and aft, is typeset line by line. Care has been taken to prevent page breaks after the opening \hrule and before the closing one, via the combined use of \nobreak and \medbreak. Again very instructive.

3.7 Encoding: \beginmathdemo...\endmathdemo

TEXnically an alignment display is used with as assignment the enlargement \advance\baselineskip2pt.
In the alignment this enlargement is undone for the first time, so the above vertical space comes only from one of the \abovedisplayskip-s. The \halign template allows for two columns. The first is an \hbox, rather wide, with the argument flushed left. The second is set in math mode, again flushed left. The variant encodings differ with respect to the size of the \hbox, and which style is set for the second parameter: (\textstyle (default) versus \displaystyle).

3.8 Encoding: \beginsyntax...\endsyntax

The first part of the macro inserts vertical space and starts a group. The catcodes of < and **[** are changed into active. (Now these symbols can do more than just be typeset. They have the function to indicate the meta-linguistic variables—beginning with <— and the reserved words—beginning with [.) The provision of natural input is encouraged furthermore by \obeylines and giving the ASCII end-of-line the meaning of the replacement text of \endsyntaxline. The latter macro looks ahead and selects via \syntaxswitch the appropriate action. The actions encoded are

- \syntaxrule, a new one has started
- \alt, alternative separator
- \continuerule, and
- \endsyntax.

No parameter parsing. Just processing line after line until the \endsyntax token is encountered.

The \syntaxrule invokes \xref, which is also used in determining what kind of index entry is written to the file index.tex. The invokation of \xref determines the type of element that comes next, and typesets it.

3.9 Encoding: \beginchart...\endchart

This is a marvel of how to encode tables via the two-part macro TEXnique. Again the alignment display is used, with the parameter of \beginchart as the assignments of the alignment display.

The \cr-s of \halign are abstracted into \evenline, respectively \oddline, to handle the row spans in the last column, and the alternating size of the horizontal rules.

\nointerlineskip switches off the interline skip.

The horizontal rule in \oddline is drawn via \multispan{19}\hrulefill[37], and the box with the hexadecimal number is \smashed. The horizontal rule in \evenline is table (and page) wide; just the \hrule.

New for me is the lowering of \null to get enough separation between the text and the horizontal rule below it. Also new is the encoding of the table strut: a lowered \vbox to14pt{}.

The main data can be supplied after \beginchart{}. An example of data specification is given for the ASCII font table in the definition of \normalchart.

\endchart takes care of the last row of the table.

> In essence DEK showed us how to encode a (page-wide) bordered table, with the data separately provided, and the template hidden.

The data specification has to be done with the use of \beginchart...\endchart in mind. No data specification independent from the formatting, however.

The :-definition hides the numbering of the table entries. The automatically maintained entry number is converted into its ASCII character via the TEX primitive \char. Elegant it is!

3.10 Encoding: \exercise

A counter, \exno, is maintained for the exercise number. The triangle is typeset in the margin followed by the word EXERCISE, the chapter number, a period

[36] This has also the effect of non-centering—effectively shifting to the left—of the verbatim text. A very concise way of overriding the centering of math displays!

[37] The font charts logically have 10 columns. The number 19 accounts also for the columns which typeset the vertical rules.

and the exercise number. No parameter and no two-part ending. After the keyword the exercise text has to be supplied. Similar are `\dangerexercise`, and `\ddangerexercise`, which insert dangerous bends.

3.11 Encoding: `\answer`

Each answer consists of the copy between `\answer` and the first blank line. The answer is copied line-by-line and with the catcode of all special characters changed into category 12 (other) to the file `answers.tex`. A nice example of a line-by-line FIFO process. In the notation of FIFO an alternative encoding reads[38]

```
\def\copytoblankline{\begingroup\setupcopy
 \fifol}
{\obeylines\gdef\fifol#1
 {\ifx\empty#1\empty\lofif\fi
  \processl{#1}\fifol}}
\def\lofif#1\fifol{\fi\endgroup}
\def\processl#1{\immediate\write\ans{#1}}
```

The above given alternative encoding of `\copytoblankline` works with `\answer` as encoded by DEK.

3.12 Encoding: shipout selected pages[39]

`\shipout` is redefined such that only pages with page numbers as supplied in the file `pages.tex` will be written to the `.dvi` file. First, plain's `\shipout` is copied under name `\Shipout`. Second, `\shipout` is redefined with the function to `\Shipout`—the original meaning—the page if the number agrees with the specified number. Otherwise the page is put in a garbage box,[40] and TEX goes on, that is, formats the rest of the copy and invokes the output routine for the next page, et cetera.

3.13 Encoding: writing index reminders

Knuth used 5 types of index reminders as displayed in the earlier given table. The encoding for taking care of the various possibilities for writing to `index.tex` is complicated. Many things are intertwined. For a better understanding, I will prune non-essentials, and start with a bare-to-the-bones encoding of the writing of ordinary index reminders to the file.

An ordinary index reminder is a $\langle word \rangle$ which should be typeset in the copy and written to the file `index.tex` with the code 0, and the page number attached to it. The syntax is

$\langle word \rangle$␣!0␣$\langle page\ number \rangle$.

The page number is attached, in the shipout process of the OTR.[41] In order to simplify matters further, I adopted the control sequence `\ir`[42] instead of the `^`-symbol and the catcode changes. With these simplifying choices the encoding for writing an ordinary index reminder to a file comes down to

```
\def\ir#1{#1\write\inx{#1 !0 \the\pageno.}}
```

For verification

```
\newwrite\inx
\immediate\openout\inx=index
\ir{aap}\vfil\eject
\ir{noot}\vfil\eject
\ir{mies}       \bye
```

yields in the file `index.tex`

```
aap !0 1.
noot !0 2.
mies !0 3.
```

Knuth's writing to a file. Let's now go back and see how DEK did it. It is important to realize that Knuth used the parameters

- `\text`, to store the $\langle word \rangle$-part of the index reminder, to be written to `index.tex`,
- `\next`,[43] to store the marked up index reminder to be typeset in the copy, with the mark up for `|`, `\<` and the required font, next to
- `\xreftype`, to store the code number of the type of the index reminder.

With more than one reminder on a page[44] the `\text` parameter must be saved from being overwritten. Knuth did this via the partial expanded definition[45]

```
\xdef\writeit{\write\inx{\text\space
 !\xreftype\space\noexpand\number%
                              \pageno.}}
\writeit.
```

Let us go through the encoding step by step. DEK first checks—in `\specialhat`, which has been `\let`-equal to `^`—whether TEX is in math mode or not. If so, nothing is written as index reminder, and the typesetting of the math just goes on. Otherwise, the next token is looked ahead in `\beginxref` and `\beginxrefswitch` is invoked. The latter macro determines whether a silent ref is the case—the switch `\silentref` is set accordingly—and `\xref` is invoked. The latter branches on to the next character and invokes

[38] The test `\ifx\empty#1\empty` can be replaced by `\def\next{#1}\ifx\next\empty` ....

[39] To remind you the default `\shipout` writes its argument to the `.dvi` file.

[40] With en-passant maintaining `\deadcycles` and reading the number of the next page to be shipped out from `pages.tex`.

[41] Remember that the Output Routine, when handling `\write`-s, knows the page number. This makes me ponder about writing the index entry and code number to an array and let the OTR add the page number to the relevant array elements. This approach would make the writing to a file superfluous, especially because sorting within TEX can be done, as exposed in my Sorting in BLUe. More on that in Two-pass BLUes jobs, to come.

[42] Mnemonics: Index reminder.

[43] This control sequence is also used for storing a 'next' symbol in between. A little confusing.

[44] Which must be accounted for!

[45] Note that the other approach of an `\immediate...` write would yield the wrong page number now and then.

`\xrefswitch`. This macro selects, and aftergroups, the process to be applied: `\vxref`, `\anglexref`, or `\normalxref`. Quite something isn't it?

In the normal case the index reminder is assigned to `\text` (and `\next`), and the type of the reminder is stored in `\xreftype`, after which `\makexref` is invoked for the typeset-and-write-to-file process.

The other cases also end up in invoking `\makexref`, with the parameters `\next`, `\text`, and `\xreftype` appropriately defined.

Intermezzo (*Overloading of the circumflex*)
By the way this is a nice example of overloading the circumflex. It can be applied to the underscore too. All that is needed is[46]

```
\def\specialhat{\ifmmode\def\next{^}\else
 \let\next\yourprocess\fi\next}
\def\yourprocess{%overloaded hat
 ...}
\catcode`\^=\active\let ^=\specialhat .
```

Intermezzo (*Recognition backslash*)
En-passant DEK coded the recognition of the backslash. In `\vxref` the catcode of the backslash is changed into `\active`. The look ahead via `\futurelet\next\...` can store in `\next` just a backslash. Because this active control sequence is left undefined it is equal to `\empty`. After this the test

```
\ifx\next\empty...
```

will yield true in this case. Handy to know of this TEXnique!

Alternative encoding. Because I don't use a `\text` parameter in the simplified encoding, I can also get rid of the induced expansion. I don't need the various look-ahead-s, encoded via `\futurelet\next...`. I just can go ahead and encode for what has to be done.[47] From the earlier given table we can easily account for the specific typesetting in the copy, while writing the $\langle word \rangle$ along with the appropriate code to the file. I had to think of another four names for the macros. Here we go.

```
\def\sir#1{%Silent Index reminder
 \write\inx{#1 !0 \the\pageno.}}
\def\ttir#1{{\tt#1}%Set in \tt
 \write\inx{#1 !1 \the\pageno.}}
\def\csir#1{|\#1|%Set as control sequence
 \write\inx{#1 !2 \the\pageno.}}
\def\bkir#1{\<#1>%Set <...>
 \write\inx{#1 !3 \the\pageno.}}
```

In my opinion the above is simpler than Knuth's encoding. The same (basic) functionalities have been preserved, at the expense of a slightly more laborous user mark up. Not just the insertion of circumflexes.

3.14 Encoding: font selection

In plain DEK provides fonts to be used in horizontal mode, and organized them into 8 families for use in math mode. The size is basically 10pt.

In Manmac the need arose apparently to provide for 8pt, and 9pt as well.[48] Knuth introduced therefore the size-switching macros: `\tenpoint`, `\ninepoint`, and `\eightpoint`. For the listing of these macros see the file manmac.tex in Appendix A.

Explanation. In the size-switching macros

- `\rm` is defined as assigning the value zero to the parameter `\fam` and the invokation of `\tenrm`
- `\textfont`, `\scriptfont`, and `\scriptscriptfont` with 'indexes' 0, 1, ... 7, are assigned to the appropriate fonts[49]
- `\tt`[50] is invoked and `\ttglue` is assigned the appropriate value
- `\normalbaselineskip` gets its value
- `\MF`, `\sc`, `\big`, `\strutbox`, some size-dependent control sequences, are defined
- `\normalbaselines`, and `\rm` are invoked.

The template to use is there. For the Fraktur example see the enhancements section.

4 BLU's manmac

Let us assume that the TUGboat, AMS(ppt/book/...), or... styles are not available and that your publisher does not provide a (LA)TEX style. Then your customization of manmac most likely will concern modifications of the encoding of the layout macros for the pages,[51] that is

- the title page, and in general
- every other page (header, footer, size, number of columns).

[46] Realize that the definition of the `\specialhat` should precede the `\catcode` change.

[47] It is true, that I don't write in the margin, and that I also don't take care of the `\ifproof` status. Those can be added easily in `\ir`, however.

[48] Not yet full-fledged!

[49] Note the various assignments with *ten*ex.

[50] This is done because the context dependent value of em in the `\ttglue` is needed.

[51] The use of fonts different from the Computer Modern family occurs too, for example Times Roman, Postscript fonts, ... That is not a modification but an enhancement. Another enhancement is the preparation of bibliographies in cooperation with a bibliographic database. That is beyond the scope of this paper.

4.1 Mod: the layout of the title

Let us assume that we wish to adapt the manmac style into a house-style for (one-column) reports. Up till now reality has it that the cover will be prepared and printed separately, with logo, text for the back of the issue and that kind of things, more efficiently handled by traditional means.

What remains is that each chapter becomes a chapter of the report. And because it is a report the title pages can become much more modest, even absent as such. Let us assume that the title of each chapter will precede the text of the first page of each chapter.

To remind you the typesetting of the title is done in `\beginchapter`, via

```
\halign{\line{\titlefont\hfil##}\\#4
 \unskip\\}
```

with the vertical positioning handled by `\vfill\eject`.[52]

First we can provide another titlefont via redefining `\titlefont`. Horizontal centering can be obtained via inserting `\hfil` after the ## in the template. In order to continue on the same page replace `\vfill\eject` by for example `\bigskip`. We also have to deactivate the setting of the (title) corner rules via `\def\setcornerrules{}`.

The modified part of `\beginchapter` then reads

```
%mods in manmac
\halign{\line{\titlefont\hfil##\hfil}\\%
        #1 #2#3 #4\unskip\\}
\titlepage\bigskip
%changes at the beginning of copy
\font\titlefont=cmssdc10 at 12pt%
\def\setcornerrules{}
```

4.2 Mod: lay-out headers

Most likely also another format of the headlines is wanted, and we like footers too. This can be done via redefining `\leftheadline`, and `\rightheadline`. In the headings use is made of the def-s `\rhead`, `\folio`, and the dimension variable `\pagewidth`. The first contains as replacement text the contents of the running head (the chapter title), the second takes care of typesetting the page number, and the last contains the size of the page width. Because the OTR of manmac lacks footers it is simplest to fall back on plain's OTR and just fill the token variables `\headline` and `\footline`. See TEXbook, p. 364. For example, begin your compuscript with

```
\headline{\issue\hfil\tenit\rhead}
\footline{\fiverm\rlap{\the\year}\hfil
 --\folio--\hfil\llap{\copyright cgl}}
\def\issue{MAPS93.1}
\output{\plainoutput}
```

[52] By the way, `\unskip` gobbles the optional space.

[53] In being silent about it and just use `\vfil\eject`.

4.3 Mod: odd-numbered pages

Suppose we like to start each chapter just on the next page. To achieve this omit the first line `\ifodd\pageno...\fi`, in `\endchapter`.

4.4 Mod: sober chapter endings

Suppose we don't want to end each chapter of our report with quotations. Then we can simplify[53] `\endchapter` into

```
\outer\def\endchapter{\vfill\eject}
```

If we like to end each chapter with for example a section called 'What have we learned?' or 'Summary' just insert a subsection with that title.

4.5 Mod: page layout parameters

The dimensions in charge are

- `\hsize`, with its value copied in `\pagewidth`,
- `\vsize`, with its value copied in `\pageheight`, and
- the size of the indentation in `\parindent`.

Note that in the running heads `\pagewidth` rather than `\hsize` is used. In the two-column layout of the index, `\hsize` is halved and `\vsize` doubled. The page width value for the headers and footnotes remains unchanged thanks to the stored values in `\pagewidth` and `\pageheight`. Each page is of size (3pc + `\pageheight`) $*$ `\pagewidth`.

For example a 24-by-17.5 page size can be obtained via inclusion of the following at the beginning of your compuscript

```
\hsize17.5cm\pagewidth\hsize
\vsize24cm\pageheight\vsize .
```

4.6 Mod: two-column

It is not true that switching from one-column format into two-column format and vice versa can be attained via a pre-programmed 'switch' (read: using a different OTR). No, also BLU's (math) markup is subject to change.

If we favor a two-column format the Output Routine has to be adapted, as shown for Appendix I. That change does not account for footnotes. Instead of writing another variant, start with borrowing TUGboat's OTR, which is in the public domain.

4.7 In summary

A customized one-column format, which for each

- chapter: starts with the chapter heading and continues the chapter on the same page,
- page: typesets our own running head and footer, with header suppressed on each 'titlepage'

can be obtained on top of manmac, by starting your compuscript with

Manmac customization additions:[54]

```
\input manmac
\font\titlefont=cmssdc10at12pt
%
\outer\def\beginchapter#1 #2#3.#4\par{%
 \def\chapno{#2#3}\global\exno0 \subsecno0
 \def\hl{\gdef\hl{\issue\hfil\it\rhead}}
 \headline{\hl}
 \def\\{ }\xdef\rhead{#4}
 {\let\\\cr\halign{\line{\titlefont
  \hfil##\hfil}\\\#1 #2#3 #4\unskip\\}}
 \bigskip\tenpoint\noindent\ignorespaces}
%
\footline{\fiverm\rlap{\the\year}\hfil
 --\folio--\hfil\llap{\copyright cgl}}
%
\outer\def\endchapter{\vfill\eject}
%
\output{\plainoutput}
%
\hsize17.5cm\pagewidth\hsize
\vsize24cm \pageheight\vsize
\def\issue{MAPS93.1}
```

Simple isn't it?

What remains are the handy macros, the suitable environments, the powerful `\exercise`, `\answer`, and the simple mark up and writing to a file of index reminders, next to the shipping-out of selected pages.

5 Enhancements

As enhancements to manmac I will discuss: writing more detailed index reminders to the file `index.tex`, and the use of AMS and non-CM fonts.

5.1 Refined index reminders

Knuth specified the index reminders some 10 years ago. Since then development towards automatic generation of indexes has taken place. The task of the author can be alleviated a little more. In the sequel I'll consider as an IR an accented word, a word which allows for comments, and a (sub)structured word.

Accented words. These do occur in indexes, as can be seen from the TEXbook. My sorting program in TEX allows for accented words too. So, an extension for writing accented words to a file would be nice. The problem with getting the page number from the OTR is that we don't know when it will be invoked, and that we don't know of the environment which will be active then.

In typesetting an accented index entry in the copy we need the normal meaning of the accents. In writing to the file `index.tex` the normal expansion must be suspended. My choice for coping with this is to write `\string...`-s. The encoding reads

```
\def\irdefs{%Accent cs-s as strings
 \def\'{\string\'}\def\`{\string\`}
 \def\^{\string\^}\def\c{\string\c\space}
 \def\"{\string\"}}
\def\ir#1{#1{\irdefs\xdef\txt{#1}
 \ea\write\ea\inx\ea{\txt\space!0
                        \the\pageno.}}}
```

With the above `\^ab\'e` yields in the file `index.tex`

```
\^ab\'e !0 1.
```

Comments. As can be seen from the index in the TEXbook it is handy to allow for comments as part of the index reminder. These comments are silent except for being a comment to what will be typeset in the index. As example of these one can think of `see...`, and `see also...` additions. My choice for the mark up of comments is to enclose them by

```
\co...\oc
```

Here, as with accents, we have to write the names (`\co`, respectively `\oc`) to the `index.tex` file and not their expansions. Again, I adopted the mechanism to convert the control sequences into strings.

For example, with

```
\def\co{\string\co\space}
\def\oc{\string\oc\space}
```

added to the replacement text of `\irdefs`, then

```
\ir{\^ab\'e \co comment\oc}
```

yields in the file `index.tex`

```
\^ab\'e \co comment\oc !0 1.
```

A special situation arises when in the comment a control sequence is supplied. For example the name `\TeX` itself. Then the expansion of that control sequence must be suspended too until the typesetting of the index. In order to achieve this, precede each control sequence by `\string`, when specifying it in the comment. Thus `\co\string\TeX\oc`.[55] This is not too far from natural behaviour, because we like to pass through the string and to typeset it nicely later.[56]

Substructuring. The first obvious thing is to allow for more words, and to have them separated by a ␣ or so.

I also like to handle subentries and subsubentries in this way. The mechanism I adopted is to surround the entries by

```
\se...\es
```

for subentries, and nested within these, enclose the subsubentries by

[54] For my personalized template, see the Appendix B: The file `manmac.tem`.

[55] When text follows the control sequence insert again `\space`.

[56] A more automatic and natural way of doing is to look for control sequences in the comment automatically. This looking ahead for a control sequence has been encoded by DEK already, burried in `\vrefswitch`, TEXbook, p. 424.

`\sse...\ess`
For example after having extended `\irdefs` by
```
\def\se{\string\se\space}
\def\es{\string\es\space}
```
then the TEXbook entry
```
\ir{spaces\se as active characters\es}
```
yields in the file `index.tex`
```
spaces\se as active characters\es !0 1.
```

Mark up: ^ and `\irdefs`. I asked myself the question

Do I really need `\ir` and its descendants?

Looking more closely at DEK's encodings revealed that if the one unnecessary expanded `\edef\text{...}` is changed into `\def\text{...}`, then my ideas can be used along with DEK's macros. I have to modify `\makexref`, however. If we assume that `\irdefs` provides the definitions for the context of writing to the file, and `\tirdefs` provides the definitions for typesetting in the copy, then my redefinition of `\makexref` comes basically down to
```
\makexref{{\tirdefs\next\irdefs
 \xdef\text{\text\space!\xreftype\space}
 \ea\write\ea\inx\ea{\text\the\pageno}}} .
```

In summary. To allow for control sequences in the $\langle word \rangle$ entails

- the index reminder word must be typeset with the substructures neglected: the structuring and comment commands must eat their arguments
- while writing to `index.tex` the control sequence must be retained and not expanded
- the control sequences must be accounted for while sorting,[57]
- the control sequences must be expanded while typesetting the index, taking account of comments, subentries and the like.

Quite some context dependency!

5.2 Fraktur fonts

Suppose we wish to use the AMS Fraktur fonts along with manmac. This is detailed with in the AMSfonts user's guide and included here for completeness. After having copied the `.pk`, and `.tfm` files extend your manmac with
```
\font\teneufm=eufm10
\font\seveneufm=eufm7
\font\fiveeufm=eufm5
\newfam\eufmfam
\textfont\eufmfam=teneufm
\scriptfont\eufmfam=seveneufm
\scriptscriptfont\eufmfam=fiveeufm
\def\frak#1{{\fam\eufmfam\relax#1}}
```
For the worked out examples with respect to the inclusion of modern bold italic or other AMS fonts, see the AMSfonts user's guide.

Explanation. This works because letters are of class 7—variable—and take the actual family value. Note that the above use is just for one argument of `\frak`; no changing of the class for the time being.

5.3 Non-CM fonts

The major stream of late is the inclusion of Postscript and the use of the accompanying fonts.[58] Next to the inclusion of fonts[59] via font definitions, we must create 'families,'[60] especially for size-switching within math mode, via either

- copying DEK's way to include font famillies,[61] or
- adopt a font selection scheme à la Mittelbach and Schöpf.[62]

A font which lacks a `.tfm` file and the like, should first be converted. See for those tools Vens (1992). Virtual fonts is all about the merging at the dvi-level, and is beyond the scope of this paper, though very important.

5.4 Miscellaneous enhancements

These are the tools provided by the TEX community at large. With respect to (simple) tables I would use my bordered table macro as given in Table Diversions. In that paper I have also discussed other table macros/packages, and mentioned the macros for handling tables which extend the page, and have to be rotated or split over pages.

For the inclusion of figures reality has it that the local—non-portable—facilities are used. Encapsulated Postscript is the Hi-TEX way to paste up the various parts at the driver level.

Very promising is Hoenig's work with respect to interfacing TEX and Metafont. That is a way to solve the high-quality and portability issues of graphics as part of a document.

The typesetting of a bibliography, given a literature database, will be treated in AMS BLUes, as alternative to the TEXniques provided by AMS.

Epilogue

Manmac has been summarized, a user's guide has been given, and the encodings have been explained.

I have touched upon modifying manmac for formatting reports. I enhanced manmac by allowing for writing more general index reminders to `index.tex`. The know-how for using AMS and non-CM fonts has been referred to.

[57] See Indexing with TEX, to come.
[58] See Anita Hoover's compilation, Hoover (1991).
[59] Fonts are supplied via `.pk`—the bitmaps—and `.tfm`—the despritions—files.
[60] Remember that only 16 families are allowed and that plain comes already with seven!
[61] As done in manmac.
[62] This issue will return in AMS BLUes, respectively TUGboat BLUes, when I'll explain how it has been incorporated there.

Acknowledgements

Ronald Kappert is kindly acknowledged for proofing the article.

TEXniques used by Knuth

Handy `\backslash`.
Recognition of \.
Coping with a dynamical number of digits.
Handling a dynamical number of lines (title lines, quotations).
Automatic typesetting a paragraph ragged-left and each line right-justified.
Two-part macros.
Separating data specifications from the macros.
Alignment displays.
Preventing unwanted pagebreaks.
Non-centered (math) displays (indented).
TEX input and formatted results side by side.
Natural input, syntax diagrams.
Bordered table (font charts).
Row spans in tables.
Writing strings (index reminders) and lines (answers to exercises) to a file.
Reading answers from file.
Special paragraph shapes, `\hang`, `\hangafter`.
Use of special fonts.
Size-switching macros.
`\ignorespaces`.
`\obeyspaces`, `\obeylines`.
Separate processing of document parts (chapters).

References

[1] AMSfonts User's Guide 2.1. AMS.

[2] Beebe, N.H.F (1991): The TUGlib server. MAPS91.2, 117–123. (Has also appeared in TEXline 11.)

[3] Beeton, B.N, R.F Whitney (1989): TUGboat Author's Guide. *TUGboat* 10, no. 3, 378–385. (The `TUGboat.sty`-les have entailed descendants like the `tugproc.sty`, and `eurotex.sty` formats. The LATEX and TEX user interfaces don't provide of yet the same typeset results, when similar control sequences are used in the mark up. Furthermore, the user interfaces don't provide the same functionalities. For example, in `tugboat.sty` one can include a file verbatim, have the lines numbered and the like, not so of yet in `ltugboat.sty`. I will come back to it in more detail in TUGboat BLUes. There I will enumerate and compare the functionalities and show how easy it is to customize the (plain) `tugboat.sty` once manmac is understood.)

[4] Eijkhout, V (1990): Unusual paragraph shapes. *TUGboat* 11, no. 1, 51–53. (Also MAPS89.2)

[5] Hoenig, A.J.J (1992): When TEX and Metafont work together. Proceedings EuroTEX '92, 1–19. (To appear in MAPS93.1.)

[6] Hoover, A (1991): Report on Workshop Getting PostScript into TEX and LATEX documents. MAPS91.2, 111–116.

[7] Jones, D.M (1992): Macro Index. (From the file server.)

[8] Knuth, D.E (1984): The TEXbook. Addison-Wesley.

[9] Laan, C.G van der (1992): Table Diversions. Proceedings EuroTEX '92, 191–211. (Also in MAPS92.2, 115–129.)

[10] Laan, C.G van der (1993): Sorting in BLUe. MAPS93.1 (Submitted to TUG '93.)

[11] Lamport, L (1986): LATEX user's guide and reference manual. Addison-Wesley.

[12] Mittelbach, F, R. Schöpf (1990): A new font selection scheme, for TEX macro packages – The basic macros. *TUGboat* 10, no. 2, 222–238. (Also elaborated for LATEX in *TUGboat* 11, no. 2, 297–305.)

[13] Quin, L.R.E (1990): Summary of Metafont fonts available. TEXHaX, 4, 6. (Also in MAPS91.1, 93—98.)

[14] Salomon, D (priv. comm.)

[15] Spivak, M.D (1986): The Joy of TEX. AMS. ISBN 0-8218-2999-8. (second printing of 1990).

[16] Spivak, M.D (1989): LA$\mathcal{MS}$-TEX The Synthesis. TEXplorators. 3701 W. Alabama, Suite 450-273, Houston, TX 77027.

[17] Vens, E.J (1992): Incorporating PostScript fonts in TEX. Proceedings EuroTEX '92, 173–181. (Also in MAPS92.2.)

[18] Wittbecker, A (1988): Making paragraphs. *TUGboat* 9, no. 3, 272–276.

Appendix A: The file manmac.tex

```
% Macros for The TeXbook (cgl, March 93)

\catcode'@=11 % borrow the private macros
              % of PLAIN (with care)

\font\tentex=cmtex10

\font\inchhigh=cminch
\font\titlefont=cmssdc10 at 40pt

\font\ninerm=cmr9
\font\eightrm=cmr8
\font\sixrm=cmr6

\font\ninei=cmmi9
\font\eighti=cmmi8
\font\sixi=cmmi6
\skewchar\ninei='177
\skewchar\eighti='177
\skewchar\sixi='177

\font\ninesy=cmsy9
\font\eightsy=cmsy8
\font\sixsy=cmsy6
\skewchar\ninesy='60
\skewchar\eightsy='60
\skewchar\sixsy='60

\font\eightss=cmssq8

\font\eightssi=cmssqi8
```

```
\font\ninebf=cmbx9
\font\eightbf=cmbx8
\font\sixbf=cmbx6

\font\ninett=cmtt9
\font\eighttt=cmtt8

\hyphenchar\tentt=-1 % inhibit hyphenation
                     % in typewriter type
\hyphenchar\ninett=-1
\hyphenchar\eighttt=-1

\font\ninesl=cmsl9
\font\eightsl=cmsl8

\font\nineit=cmti9
\font\eightit=cmti8

\font\tenu=cmu10 % unslanted text italic
\font\magnifiedfiverm=cmr5 at 10pt
\font\manual=manfnt % font used for the
                    % METAFONT logo, etc.
%Mod because  cmman not available at RUG
%\font\cmman=cmman % font used for
%miscellaneous Computer Modern variations

\newskip\ttglue
\def\tenpoint{\def\rm{\fam0\tenrm}%
  \textfont0=\tenrm \scriptfont0=\sevenrm
               \scriptscriptfont0=\fiverm
  \textfont1=\teni \scriptfont1=\seveni
                \scriptscriptfont1=\fivei
  \textfont2=\tensy \scriptfont2=\sevensy
               \scriptscriptfont2=\fivesy
  \textfont3=\tenex \scriptfont3=\tenex
                \scriptscriptfont3=\tenex
  \def\it{\fam\itfam\tenit}%
  \textfont\itfam=\tenit
  \def\sl{\fam\slfam\tensl}%
  \textfont\slfam=\tensl
  \def\bf{\fam\bffam\tenbf}%
  \textfont\bffam=\tenbf
   \scriptfont\bffam=\sevenbf
   \scriptscriptfont\bffam=\fivebf
  \def\tt{\fam\ttfam\tentt}%
  \textfont\ttfam=\tentt
  \tt \ttglue=.5em plus.25em minus.15em
  \normalbaselineskip=12pt
  \def\MF{{\manual META}\-{\manual FONT}}%
  \let\sc=\eightrm
  \let\big=\tenbig
  \setbox\strutbox=\hbox{\vrule height8.5pt
   depth3.5pt width\z@}%
  \normalbaselines\rm}

\def\ninepoint{\def\rm{\fam0\ninerm}%
  \textfont0=\ninerm \scriptfont0=\sixrm
              \scriptscriptfont0=\fiverm
  \textfont1=\ninei \scriptfont1=\sixi
               \scriptscriptfont1=\fivei
  \textfont2=\ninesy \scriptfont2=\sixsy
              \scriptscriptfont2=\fivesy
  \textfont3=\tenex \scriptfont3=\tenex
               \scriptscriptfont3=\tenex
  \def\it{\fam\itfam\nineit}%
  \textfont\itfam=\nineit
  \def\sl{\fam\slfam\ninesl}%
  \textfont\slfam=\ninesl
  \def\bf{\fam\bffam\ninebf}%
  \textfont\bffam=\ninebf
   \scriptfont\bffam=\sixbf
   \scriptscriptfont\bffam=\fivebf
  \def\tt{\fam\ttfam\ninett}%
  \textfont\ttfam=\ninett
  \tt \ttglue=.5em plus.25em minus.15em
  \normalbaselineskip=11pt
  \def\MF{{\manual hijk}\-{\manual lmnj}}%
  \let\sc=\sevenrm
  \let\big=\ninebig
  \setbox\strutbox=\hbox{\vrule height8pt
   depth3pt width\z@}%
  \normalbaselines\rm}

\def\eightpoint{\def\rm{\fam0\eightrm}%
  \textfont0=\eightrm \scriptfont0=\sixrm
              \scriptscriptfont0=\fiverm
  \textfont1=\eighti \scriptfont1=\sixi
               \scriptscriptfont1=\fivei
  \textfont2=\eightsy \scriptfont2=\sixsy
              \scriptscriptfont2=\fivesy
  \textfont3=\tenex \scriptfont3=\tenex
               \scriptscriptfont3=\tenex
  \def\it{\fam\itfam\eightit}%
  \textfont\itfam=\eightit
  \def\sl{\fam\slfam\eightsl}%
  \textfont\slfam=\eightsl
  \def\bf{\fam\bffam\eightbf}%
  \textfont\bffam=\eightbf
   \scriptfont\bffam=\sixbf
   \scriptscriptfont\bffam=\fivebf
  \def\tt{\fam\ttfam\eighttt}%
  \textfont\ttfam=\eighttt
  \tt \ttglue=.5em plus.25em minus.15em
  \normalbaselineskip=9pt
  \def\MF{{\manual opqr}\-{\manual stuq}}%
  \let\sc=\sixrm
  \let\big=\eightbig
  \setbox\strutbox=\hbox{\vrule height7pt
   depth2pt width\z@}%
  \normalbaselines\rm}

\def\tenmath{\tenpoint\fam-1 } % use after $
                    % in ninepoint sections
\def\tenbig#1{{\hbox{$\left#1\vbox to8.5pt{}%
                         \right.\n@space$}}}
\def\ninebig#1{{\hbox{$\textfont0=\tenrm
 \textfont2=\tensy
 \left#1\vbox to7.25pt{}\right.\n@space$}}}
\def\eightbig#1{{\hbox{$\textfont0=\ninerm
 \textfont2=\ninesy
 \left#1\vbox to6.5pt{}\right.\n@space$}}}

% Page layout
\newdimen\pagewidth \newdimen\pageheight
\newdimen\ruleht      % Corner rules
\hsize=29pc  \vsize=44pc  \maxdepth=2.2pt
\parindent=3pc \ruleht=.5pt
\pagewidth=\hsize \pageheight=\vsize
\abovedisplayskip=6pt plus 3pt minus 1pt
\belowdisplayskip=6pt plus 3pt minus 1pt
\abovedisplayshortskip=0pt plus 3pt
\belowdisplayshortskip=4pt plus 3pt

%\newinsert\footins
\def\footnote#1{\edef\@sf{\spacefactor
 \the\spacefactor}#1\@sf
 \insert\footins\bgroup\eightpoint
 \interlinepenalty100 \let\par=\endgraf
 \leftskip=\z@skip \rightskip=\z@skip
 \splittopskip=10pt plus 1pt minus 1pt
 \floatingpenalty=20000
 \smallskip\item{#1}\bgroup\strut
 \aftergroup\@foot\let\next}
%
\skip\footins=12pt plus 2pt minus 4pt
% space added when footnote is present
%\count\footins=1000
% footnote magnification factor (1 to 1)
```

```
\dimen\footins=30pc
% maximum footnotes per page

\newinsert\margin
\dimen\margin=\maxdimen
%\count\margin=0 \skip\margin=0pt
% marginal inserts take up no space

\newif\iftitle
\def\titlepage{\global\titletrue}
% for pages without headlines
\def\rhead{} % contains the running headline

\def\leftheadline{\hbox to \pagewidth{%
 \vbox to 10pt{}% strut to position the
                % baseline
 \llap{\tenbf\folio\kern1pc}% folio to
                            % left of text
 \tenit\rhead\hfil% running head flush left
}}
\def\rightheadline{\hbox to \pagewidth{%
 \vbox to 10pt{}% strut to position the
                % baseline
 \hfil\tenit\rhead\/% running head flush
                    % right
 \rlap{\kern1pc\tenbf\folio}% folio to
                            % right of text
}}

\def\onepageout#1{\shipout\vbox{ % here we
               % define one page of output
 \offinterlineskip % butt the boxes together
 \vbox to 3pc{ % this part goes on top of
               % the 44pc pages
  \iftitle % the next is used for title pages
   \global\titlefalse % reset the titlepage
                      % switch
   \setcornerrules % for camera alignment
  \else\ifodd\pageno \rightheadline\else
                     \leftheadline\fi\fi
 \vfill} % this completes the \vbox to 3pc
 \vbox to \pageheight{
  \ifvoid\margin\else%marginal info is present
   \rlap{\kern31pc\vbox to\z@{\kern4pt%
   \box\margin \vss}}\fi
  #1 % now insert the main information
  \ifvoid\footins\else%footnote info is present
   \vskip\skip\footins \kern-3pt
   \hrule height\ruleht width\pagewidth
   \kern-\ruleht \kern3pt
   \unvbox\footins\fi
   \boxmaxdepth=\maxdepth
 } % this completes the \vbox to \pageheight
}\advancepageno}

\def\setcornerrules{\hbox to \pagewidth{%
 \vrule width 1pc height\ruleht
    \hfil \vrule width 1pc}
 \hbox to \pagewidth{\llap{\sevenrm(page
    \folio)\kern1pc}%
 \vrule height1pc width\ruleht depth\z@
    \hfil \vrule width\ruleht depth\z@}}

\output{\onepageout{\unvbox255}}

\newbox\partialpage
\def\begindoublecolumns{\begingroup
 \output={\global\setbox\partialpage=
  \vbox{\unvbox255\bigskip}}
 \eject
 \output={\doublecolumnout}
 \hsize=14pc \vsize=89pc}
\def\enddoublecolumns{\output=
 {\balancecolumns}\eject \endgroup
 \pagegoal=\vsize}

\def\doublecolumnout{\splittopskip=\topskips

 \splitmaxdepth=\maxdepth \dimen@=44pc
 \advance\dimen@ by-\ht\partialpage
 \setbox0=\vsplit255 to\dimen@
 \setbox2=\vsplit255 to\dimen@
 \onepageout\pagesofar
 \unvbox255 \penalty\outputpenalty}
\def\pagesofar{\unvbox\partialpage
 \wd0=\hsize \wd2=\hsize \hbox to
 \pagewidth{\box0\hfil\box2}}
\def\balancecolumns{\setbox0=\vbox{\unvbox255}
 \dimen@=\ht0
  \advance\dimen@ by\topskip
  \advance\dimen@ by-\baselineskip
  \divide\dimen@ by2 \splittopskip=\topskip
 {\vbadness=10000
 \loop \global\setbox3=\copy0
  \global\setbox1=\vsplit3 to\dimen@
 \ifdim\ht3>\dimen@
  \global\advance\dimen@ by1pt
 \repeat}
 \setbox0=\vbox to\dimen@{\unvbox1}
 \setbox2=\vbox to\dimen@{\unvbox3}
 \pagesofar}

% To produce only a subset of pages,
% put the page numbers on separate
% lines in a file called pages.tex
\let\Shipout=\shipout
\newread\pages \newcount\nextpage
\openin\pages=pages
\def\getnextpage{\ifeof\pages\else
 {\endlinechar=-1\read\pages to\next
  \ifx\next\empty % in this case we should
                  % have eof now
  \else\global\nextpage=\next\fi}\fi}
\ifeof\pages\else\message{OK, I'll ship only
 the requested pages!}\getnextpage\fi
%
\def\shipout{\ifeof\pages
 \let\next=\Shipout\else
 \ifnum\pageno=\nextpage
  \getnextpage\let\next=\Shipout
 \else\let\next=\Tosspage\fi\fi \next}
%
\newbox\garbage
\def\Tosspage{\deadcycles=0\setbox\garbage=}

% Chapter formatting
% The preface and table of contents are
% formatted in place, not here

\newcount\exno % for the number of exercises
               % in the current chapter
\newcount\subsecno % for the number of
  % subsections in the current chapter

\def\beginchapter#1 #2#3. #4\par{\global
 \exno=0 \subsecno=0 \def\chapno{#2#3}
 \ifodd\pageno
  \errmessage{You had too much text on
       that last page; I'm backing up}
  \advance\pageno by-1
 \fi \titlepage
 \def\\{ } % \\'s in the title will be
           % treated as spaces
 \message{#1 #2#3:} % show the chapter
                 % title on the terminal
 \def\MF{{\manual 89:;<=>:}} % slant the logo
 \xdef\rhead{#1 #2#3: #4\unskip}
 {\def\TeX{T\kern-.2em\lower.5ex%
```

```
          \hbox{E}\kern-.06em X}
  \def\MF{{\vbox to30pt{}\manual ()*+,-.*}}
  \def\\{#3}
  \ifx\empty\\
   \rightline{\inchhigh #2\kern-.04em}
  \else\rightline{\inchhigh #2\kern-.06em%
                  #3\kern-.04em}%
  \fi \vskip 1.75pc
  \baselineskip 36pt \lineskiplimit
  \titlelsl \lineskip 12pt
  \let\\=\cr%now the \\'s are line dividers
  \halign{\line{\titlefont\hfil##}\\#4%
   \unskip\\}\vfill\eject} % output the
                  % chapter title page
  \tenpoint\noindent\ignorespaces}% the first
%paragraph of a chapter is not indented

\newdimen\titlelsl \titlelsl=1pt

\outer\def\endchapter{\ifodd\pageno\else
 \vfill\eject\null\fi
 \begingroup\bigskip\vfill % beginning of
                           % the quotes
 \def\eject{\endgroup\eject}
 \def\par{\ifhmode\/\endgraf\fi}\obeylines
 \def\TeX{T\kern-.2em\lower.5ex\hbox{E}%
  \kern-.000em X}
 \def\MF{{\manual opqr}\-{\manual stuq}}
 \eightpoint \let\tt=\ninett
 \baselineskip 10pt
 \parfillskip \z@
 \interlinepenalty 10000
 \leftskip \z@ plus 40pc minus \parindent
 \let\rm=\eightss \let\sl=\eightssi
 \everypar{\sl}}
%
\def\author#1(#2){\smallskip\noindent\rm---
 #1\unskip\enspace(#2)}

\def\dbend{{\manual\char127}}
% dangerous bend sign
\def\d@nger{\medbreak\begingroup
 \clubpenalty=10000
 \def\par{\endgraf\endgroup\medbreak}
 \noindent\hang\hangafter=-2
 \hbox to0pt{\hskip-\hangindent\dbend
  \hfill}\ninepoint}
\outer\def\danger{\d@nger}
\def\dd@nger{\medbreak\begingroup
 \clubpenalty=10000
 \def\par{\endgraf\endgroup\medbreak}
 \noindent\hang\hangafter=-2
 \hbox to0pt{\hskip-\hangindent\dbend
  \kern1pt\dbend\hfill}\ninepoint}
\outer\def\ddanger{\dd@nger}
\def\enddanger{\endgraf\endgroup} % omits
                          % the \medbreak

\outer\def\subsection#1. {\medbreak
 \advance\subsecno by 1
 \noindent
 {\it \the\subsecno.\enspace#1.\enspace}}
\def\ansno#1.#2:{\medbreak\noindent
 \hbox to\parindent{\bf\hss#1.#2.\enspace}
 \ignorespaces}

% Composition macros
\hyphenation{man-u-script man-u-scripts
 ap-pen-dix xscaled}

\def\AmSTeX{$\cal A\kern-.1667em\lower
 .5ex\hbox{$\cal M$}\kern-.075em S$-\TeX}
\def\bull{\vrule height .9ex width .8ex%
 depth -.1ex } % square bullet

\def\SS{{\it SS}} % scriptscript style
\def\|{\leavevmode\hbox{\tt\char'\|}}
 % vertical line
\def\dn{\leavevmode\hbox{\tt\char'14}}
 % downward arrow
\def\up{\leavevmode\hbox{\tt\char'13}}
 % upward arrow
\def\]{\leavevmode\hbox{\tt\char'\ }}
 % visible space

\def\pt{\,{\rm pt}} % units of points,
                    % in math formulas
\def\em{\,{\rm em}} % units of ems,
                    % in math formulas
\def\<#1>{\leavevmode\hbox{$\langle$#1%
 \/$\rangle$}} % syntactic quantity
\def\oct#1{\hbox{\rm\'{}\kern-.2em\it
 #1\/\kern.05em}} % octal constant
\def\hex#1{\hbox{\rm\H{}\tt#1}}
 % hexadecimal constant
\def\cstok#1{\leavevmode\thinspace\hbox{%
 \vrule\vtop{\vbox{\hrule\kern1pt\hbox{%
 \vphantom{\tt/}\thinspace{\tt#1}%
 \thinspace}}\kern1pt\hrule}\vrule}%
 \thinspace} % control sequence token

{\obeyspaces\gdef {\ }}
\def\parbreak{\hfil\break\indent\strut}
\def\stretch{\nobreak\hskip0pt plus2pt\relax}

% macros for non-centered displays
\outer\def\begindisplay{\obeylines
 \startdisplay}
{\obeylines\gdef\startdisplay#1
 {\catcode'\^^M=5$$#1\halign\bgroup\indent
  ##\hfil&&\qquad##\hfil\cr}}
\outer\def\enddisplay{\crcr\egroup$$}

% (the following \begin...\end-type
%  macros do not appear in Appendix E)
% macros for demonstrating math constructions
\outer\def\beginmathdemo{
 $$\advance\baselineskip by2pt
 \halign\bgroup\indent\hbox to 160pt{##\hfil}&
  $##$\hfil\cr\noalign{\vskip-2pt}
}
\outer\def\begindisplaymathdemo{
 $$\advance\baselineskip by15pt
 \halign\bgroup\indent\hbox to 160pt{##\hfil}&
  $\displaystyle{##}$\hfil\cr
  \noalign{\vskip-15pt}
}
\outer\def\beginlongmathdemo{
 $$\advance\baselineskip by2pt
 \halign\bgroup\indent\hbox to 210pt{##\hfil}&
  $##$\hfil\cr\noalign{\vskip-2pt}
}
\outer\def\beginlongdisplaymathdemo{
 $$\advance\baselineskip by15pt
 \halign\bgroup\indent\hbox to 210pt{##\hfil}&
  $\displaystyle{##}$\hfil\cr
  \noalign{\vskip-15pt}
}
\outer\def\endmathdemo{\egroup$$}

% macros for font tables
\def\oddline#1{\cr
 \noalign{\nointerlineskip}
 \multispan{19}\hrulefill&
 \setbox0=\hbox{\lower 2.3pt\hbox{%
        \hex{#1x}}}\smash{\box0}\cr
 \noalign{\nointerlineskip}}
\def\evenline{\cr\noalign{\hrule}}
\def\chartstrut{\lower4.5pt\vbox to14pt{}}
```

```
\def\beginchart#1{%
           $$\postdisplaypenalty=-10000
 \global\count@=0 #1
 \halign to\hsize\bgroup
  \chartstrut##\tabskip0pt plus10pt&
          &\hfil##\hfil&\vrule##\cr
  \lower6.5pt\null
  &&&\oct0&&\oct1&&\oct2&&\oct3&&\oct4&&
     \oct5&&\oct6&&\oct7&\evenline}
\def\endchart{\raise11.5pt\null&&&\hex 8&&
 \hex 9&&\hex A&&\hex B&&\hex C&&\hex D&&
 \hex E&&\hex F&\cr\egroup$$}
\def\:{\setbox0=\hbox{\char\count@}%
 \ifdim\ht0>7.5pt\reposition
 \else\ifdim\dp0>2.5pt\reposition\fi\fi
 \box0\global\advance\count@ by1 }
\def\reposition{\setbox0=\hbox{$\vcenter
 {\kern2pt\box0\kern2pt}$}}
\def\normalchart{%
 &\oct{00x}&&\:&&\:&&\:&&\:&&\:&&\:&&
  \:&&\oddline0
 &\oct{01x}&&\:&&\:&&\:&&\:&&\:&&\:&&
  \:&\evenline
 &\oct{02x}&&\:&&\:&&\:&&\:&&\:&&\:&&
  \:&&\oddline1
 &\oct{03x}&&\:&&\:&&\:&&\:&&\:&&\:&&
  \:&\evenline
 &\oct{04x}&&\:&&\:&&\:&&\:&&\:&&\:&&
  \:&&\oddline2
 &\oct{05x}&&\:&&\:&&\:&&\:&&\:&&\:&&
  \:&\evenline
 &\oct{06x}&&\:&&\:&&\:&&\:&&\:&&\:&&
  \:&&\oddline3
 &\oct{07x}&&\:&&\:&&\:&&\:&&\:&&\:&&
  \:&\evenline
 &\oct{10x}&&\:&&\:&&\:&&\:&&\:&&\:&&
  \:&&\oddline4
 &\oct{11x}&&\:&&\:&&\:&&\:&&\:&&\:&&
  \:&\evenline
 &\oct{12x}&&\:&&\:&&\:&&\:&&\:&&\:&&
  \:&&\oddline5
 &\oct{13x}&&\:&&\:&&\:&&\:&&\:&&\:&&
  \:&\evenline
 &\oct{14x}&&\:&&\:&&\:&&\:&&\:&&\:&&
  \:&&\oddline6
 &\oct{15x}&&\:&&\:&&\:&&\:&&\:&&\:&&
  \:&\evenline
 &\oct{16x}&&\:&&\:&&\:&&\:&&\:&&\:&&
  \:&&\oddline7
 &\oct{17x}&&\:&&\:&&\:&&\:&&\:&&\:&&
  \:&\evenline}

% (now Appendix E resumes again)
% macros for verbatim scanning
\chardef\other=12
\def\ttverbatim{\begingroup
 \catcode`\\=\other
 \catcode`\{=\other
 \catcode`\}=\other
 \catcode`\$=\other
 \catcode`\&=\other
 \catcode`\#=\other
 \catcode`\%=\other
 \catcode`\~=\other
 \catcode`\_=\other
 \catcode`\^=\other
 \obeyspaces \obeylines \tt}

\outer\def\begintt{$$\let\par=\endgraf
 \ttverbatim \parskip=\z@
 \catcode`\|=0 \rightskip-5pc \ttfinish}
{\catcode`\|=0 |catcode`|\=\other
 % | is temporary escape character
 |obeylines % end of line is active
 |gdef|ttfinish#1^^M#2\endtt{#1|vbox{#2}%
 |endgroup$$}}

\catcode`\|=\active
{\obeylines \gdef|{\ttverbatim \spaceskip
 \ttglue \let^^M=\  \let|=\endgroup}}

% macros for syntax rules (again, not in
%                           Appendix E)
\def\[#1]{\silenttrue\xref|#1|\thinspace
 {\tt#1}\thinspace} % keyword in syntax
\def\beginsyntax{\endgraf\nobreak\medskip
 \begingroup \catcode`<=13 \catcode`[=13
 \let\par=\endsyntaxline \obeylines}
\def\endsyntaxline{\futurelet\next
 \syntaxswitch}
\def\syntaxswitch{\ifx\next\<\let\next=
 \syntaxrule\else
 \ifx\next\endsyntax\let\next=\endgroup
 \else\let\next=\continuerule
 \fi\fi \next
}
\def\continuerule{\hfil\break\indent\qquad}
\def\endsyntax{\medbreak\noindent}
{\catcode`<=13 \catcode`[=13
 \global\let<=\< \global\let[=\[
 \gdef\syntaxrule<#1>{\endgraf\indent
 \silentfalse\xref\<#1>}}
\def\is{\ $\longrightarrow$ }
\def\alt{\ $\vert$ }

% macros to demarcate lines quoted from
% TeX source files
\def\beginlines{\par\begingroup\nobreak
 \medskip\parindent\z@ \obeylines
 \hrule\kern1pt\nobreak \everypar{\strut}}
\def\endlines{\kern1pt\hrule\endgroup
 \medbreak\noindent}
\def\weakendlines{\kern1pt\hrule\endgroup
 \medskip\noindent}
\def\finalendlines{\kern1pt\hrule\endgroup
 \medbreak}

\outer\def\exercise{\medbreak
 \global\advance\exno by 1
 \noindent\llap{\manual\char'170
        \rm\kern.15em}% triangle in margin
 {\ninebf EXERCISE \bf\chapno.\the\exno}\par
 \nobreak\noindent}
\def\dexercise{\global\advance\exno by 1
 \llap{\manual\char'170
  \rm\kern.15em}% triangle in indented space
 {\eightbf EXERCISE \bf\chapno.\the\exno}\hfil
  \break}
\outer\def\dangerexercise{\d@nger\dexercise}
\outer\def\ddangerexercise{\dd@nger\dexercise}

\newwrite\ans
\immediate\openout\ans=answers
 % file for answers to exercises
\outer\def\answer{\par\medbreak
 \immediate\write\ans{}
 \immediate\write\ans{\string\ansno
  \chapno.\the\exno:}\copytoblankline}
\def\copytoblankline{\begingroup\setupcopy
 \copyans}
\def\setupcopy{\def\do##1{\catcode`##1=
 \other}\dospecials
 \catcode`\|=\other \obeylines}
{\obeylines \gdef\copyans#1
 {\def\next{#1}%
  \ifx\next\empty\let\next=\endgroup
  \else\immediate\write\ans{\next}%
   \let\next=\copyans
```

```
  \fi\next}}

% Macros for drawing figures (not in
%                        Appendix E)
\def\hidehrule#1#2{\kern-#1\hrule
 height#1 depth#2 \kern-#2 }
\def\hidevrule#1#2{\kern-#1{\dimen0=#1
 \advance\dimen0 by#2\vrule width\dimen0}%
 \kern-#2 }
% \makeblankbox puts rules at the edges of
% a blank box whose dimensions are those
% of \box0 (assuming nonnegative wd,ht,dp)
% #1 is rule thickness outside,
% #2 is rule thickness inside
\def\makeblankbox#1#2{\hbox{\lower\dp0
 \vbox{\hidehrule{#1}{#2}%
  \kern-#1% overlap the rules at the corners
  \hbox to\wd0{\hidevrule{#1}{#2}%
  \raise\ht0\vbox to #1{}% set the vrule height
  \lower\dp0\vtop to #1{}% set the vrule depth
  \hfil\hidevrule{#2}{#1}}%
  \kern-#1\hidehrule{#2}{#1}}}}
\def\maketypebox{\makesblankbox{0pt}{1pt}}
\def\makelightbox{\makeblankbox{.2pt}{.2pt}}

% \box\bigdot is a null box with a bullet
% at its reference point
\newbox\bigdot \newbox\smalldot
\setbox0=\hbox{$\vcenter{}$}
% \ht0 is the axis height
\setbox1=\hbox to\z@{$\hss\bullet\hss$}
% bullet is centered on the axis
\setbox\bigdot=\vbox to\z@{\kern-\ht1
 \kern\ht0 \box1 \vss}
\setbox1=\hbox to\z@{$\hss\cdot\hss$}
% cdot is centered on the axis
\setbox\smalldot=\vbox to\z@{\kern-\ht1
 \kern\ht0 \box1 \vss}

% \arrows makes things like <--- text --->
\def\arrows#1#2{% #1=width, #2=text
{\setbox0=\hbox{$\mkern-2mu\mathord
                -\mkern-2mu$}
 \hbox to #1{\kern-.055556em$\leftarrow
  \mkern-6mu$\cleaders\copy0\hfil
  \kern.4em #2\kern.4em\cleaders\copy0\hfil
  $\mkern-6mu\rightarrow$\kern-.055556em}}}

% \samplebox makes the outline of a box,
% with big dot at reference point
\def\samplebox#1#2#3#4{% #1=ht, #2=dp, #3=wd,
% #4=text
 {\setbox0=\vtop{\vbox to #1{\hbox to #3{}
  \vss}\nointerlineskip\vbox to #2{}}
  % now \box0 has the desired ht, dp, and wd
 \hbox{\copy\bigdot
  \vrule height.2pt depth.2pt width#3%
  \kern-#3\makelightbox\kern-#3%
  \raise#1\vbox{\hbox to #3{\hss#4\hss}
   \kern 3pt}}}}

% \sampleglue makes glue between sample boxes
\newdimen\varunit
\varunit=\hsize
\advance\varunit by-2\parindent
\divide\varunit by 58 % illustrations in
                      % Chapter 12
\def\sampleglue#1#2{% #1=width, #2=text
 \vtop{\hbox to #1{\xleaders\hbox to
 .5\varunit{\hss\copy\smalldot\hss}\hfil}
 \kern3pt
 \tabskip \z@ plus 1fil
 \halign to #1{\hfil##\cr#2\cr}}}

% Indexing macros
\newif\ifproofmode
\proofmodetrue % this should be false when
               % making camera-ready copy
\newwrite\inx
\immediate\openout\inx=index % file for
               % index reminders
\newif\ifsilent
\def\specialhat{\ifmmode\def\next{^}\else
 \let\next=\beginxref\fi\next}
\def\beginxref{\futurelet\next
 \beginxrefswitch}
\def\beginxrefswitch{\ifx\next\specialhat
   \let\next=\silentxref\else
 \silentfalse\let\next=\xref\fi \next}
\catcode`\^=\active \let ^=\specialhat
\def\silentxref^{\silenttrue\xref}

\def\marginstyle{\vrule height6pt depth2pt
 width\z@ \sevenrm}

\chardef\bslash=`\\
\def\xref{\futurelet\next\xrefswitch}
\def\xrefswitch{\begingroup
 \ifx\next|\aftergroup\vxref
  % case 1 or 2, |arg| or |\arg|
 \else\ifx\next\<\aftergroup\anglexref
   % case 3, "\<arg>" means angle brackets
  \else\aftergroup\normalxref
 \fi\fi\endgroup} % case 0, "{arg}"
\def\vxref|{\catcode`\\=\active
 \futurelet\next\vxrefswitch}
\def\vxrefswitch#1|{\catcode`\\=0
 \ifx\next\empty\def\xreftype{2}%
  \def\next{{\tt\bslash\text}}% type 2,
                              % |\arg|
 \else\def\xreftype{1}\def\next{%
       {\tt\text}}\fi % type 1, |arg|
 \edef\text{#1}\makexref}
{\catcode`\|=0 \catcode`\\=\active |gdef\{}}
\def\anglexref\<#1>{\def\xreftype{3}%
 \def\text{#1}\def\next{\<\text>}\makexref}
\def\normalxref#1{\def\xreftype{0}%
 \def\text{#1}\let\next=\text\makexref}
\def\makexref{\ifproofmode\insert\margin
  {\hbox{\marginstyle\text}}%
  \xdef\writeit{\write\inx{\text\space!%
   \xreftype\space\noexpand\number\pageno.%
  }}\writeit
 \else\ifhmode\kern\z@\fi\fi
 \ifsilent\ignorespaces\else\next\fi}
% the \insert (which is done in proofmode
% only) suppresses hyphenation, so the
% \kern\z@ is put in to give the same effect
% in non-proofmode.

% Internal cross references that may change
\def\sesame{61}
 % page number for Sesame Street quote
\def\bmiexno{20}
 % exercise number for bold math italic
\def\punishexno{1}
 % exercise number for `punishment'
\def\fracexno{6}
 % exercise number for `\frac'
\def\vshippage{31}
 % error message from `\vship'
\def\storypage{24}
 % listing of story.tex
\def\metaT{4}
 % exercise number for T of METAFONT
\def\xwhat{2}
 % exercise number for x3:=whatever
\def\Xwhat{2}
```

```
 % exercise number for whatever itself

\def\checkequals#1#2{\ifnum#1=#2\else
 \errmessage{Redefine \string#1 to be
  \the#2}\fi}

% Things for The METAFONTbook only
\ifx\MFmanual\!\else\endinput\fi

\def\!{\kern-.03em\relax}

\def\frac#1/#2{\leavevmode\kern.1em\raise
 .5ex\hbox{\the\scriptfont0 #1}\kern-.1em/%
 \kern-.15em\lower.25ex\hbox{%
 \the\scriptfont0 #2}}

\outer\def\displayfig #1 (#2){$$
 \advance\abovedisplayskip by 3pt
 \leftline{\indent\figbox{#1}{3in}{#2}%
 \vbox}$$}
\def\rightfig #1 (#2 x #3) ^#4 {%
%#2 wide and #3 deep, raised #4
 \strut\vadjust{\setbox0=\vbox to 0pt{\vss
  \hbox to\pagewidth{\hfil
   \raise #4\figbox{#1}{#2}{#3}\vtop \quad}}
  \dp0=0pt \box0}}
\def\figbox#1#2#3#4{#4to#3{%
% makes a box #2 wide and #3 deep
 \ifproofmode\kern0pt\hrule\vfill
  \hsize=#2 \baselineskip 6pt
  \fiverm\noindent\raggedright
  (Figure #1 will be inserted here;
   too bad you can't see it now.)
  \endgraf\vfill\hrule
 \else\vfill\hbox to#2{}\fi}}

\def\endsyntax{\begingroup\let\par=\endgraf
 \medbreak\endgroup\noindent}

\let\BEGINCHAPTER=\beginchapter
\def\beginchapter{\titlelsl=1pt \BEGINCHAPTER}
\def\beginChapter{\titlelsl=2pt \BEGINCHAPTER}

\def\decreasehsize #1 {\advance\hsize-#1}
\def\restorehsize{\hsize=\pagewidth}

\catcode`\@=\active
\catcode`\"=\active
\def\ttverbatim{\begingroup
 \catcode`\@=\other \catcode`\"=\other
 \catcode`\\=\other \catcode`\{=\other
 \catcode`\}=\other \catcode`\$=\other
 \catcode`\&=\other \catcode`\#=\other
 \catcode`\%=\other \catcode`\~=\other
 \catcode`\_=\other \catcode`\^=\other
 \obeyspaces \obeylines \tt}
\def\setupcopy{\def\do##1{\catcode`##1=
  \other}\dospecials
 \catcode`\|=\other \catcode`\@=\other
 \catcode`\"=\other \obeylines}
\def\_{\leavevmode \kern.06em
 \vbox{\hrule width.3em}}
\def@#1@{\begingroup\def\_{\kern.04em
 \vbox{\hrule width.3em height .6pt}%
 \kern.08em}%
 \ifmmode\mathop{\bf#1}\else
  \hbox{\bf#1\/}\fi\endgroup}
\def"#1"{\hbox{\it#1\/\kern.05em}}
 % italic type for identifiers
\def\xrefswitch{\begingroup
 \ifx\next|\aftergroup\vxref
  % case 1, |arg| or |\arg|
 \else\ifx\next@\aftergroup\boldxref
  % case 2, "@arg@" means boldface
 \else\ifx\next"\aftergroup\italxref
  % case 4, ""arg"" means boldface
 \else\ifx\next\<\aftergroup\anglexref
  % case 3, "\<arg>" means angle brackets
 \else\aftergroup\normalxref
 \fi\fi\fi\fi\endgroup} % case 0, "{arg}"
\def\boldxref@#1@{\def\xreftype{2}%
 \def\text{#1}\def\next{@\text@}\makexref}
\def\italxref"#1"{\def\xreftype{4}%
 \def\text{#1}\def\next{"\text"}\makexref}

\def\pyth+{\mathbin{++}}
\def\0{\raise.7ex\hbox{$\scriptstyle\#$}}
\def\to{\mathrel{\ldotp\ldotp}}
\def\dashto{\mathrel{\hbox{-\thinspace
 -\kern-.05em}}}
\def\ddashto{\mathrel{\hbox{-\thinspace
 -\thinspace-\kern-.05em}}}
\def\round{\mathop{\rm round}}
\def\angle{\mathop{\rm angle}}
\def\rmsqrt{\mathop{\rm sqrt}}
\def\reverse{\mathop{\rm reverse}}
\def\curl{\mathop{\rm curl}}
\def\tension{\mathop{\rm tension}}
\def\atleast{\mathop{\rm atleast}}
\def\controls{\mathop{\rm controls}}
\def\and{\,{\rm and}\,}
\def\cycle{{\rm cycle}}
\def\pickup{@pickup@ \thinspace}
\def\penpos#1{\hbox{\it penpos}_{#1}}
\def\pentaper#1{\hbox{\it pentaper}_{#1}}

\chardef\hexa=1  % first hex
\chardef\hexb=2 % top and bot adjusted
\chardef\hexc=3 % same, bold
\chardef\hexd=4 % same, confined to box
\chardef\hexe=5 % penstroked hex
\chardef\Aa=6 % stick-figure A, golden ratio
\def\sevenAs{\char7\char8\char9\char10
 \char11\char12\char13} % same, variants
\chardef\Az=14 % same, with crooked bar
\chardef\Ab=15 % \Aa with rectilinear
               % elliptical pen
\chardef\Ac=16 % same, with the ellipse
               % tilted
\chardef\beana=17 % kidney bean, default pen
\chardef\beanb=18 % same, twice as bold
\chardef\beanc=19 % same, rectilinear
                  % elliptical pen
\chardef\beand=20 % same, with the ellipse
                  % tilted
\chardef\niba=21 % 10x rectilinear ellipse
\chardef\nibb=22 % same, with the ellipse
                 % tilted
\chardef\nibc=23 % same, 90 degrees titled
\chardef\IOT=24 % Ionian T
\chardef\IOS=25 % Ionian S
\chardef\IOO=26 % Ionian O
\chardef\IOI=27 % Ionian I
\chardef\cubea=28 % possible cube
\chardef\cubeb=29 % impossible cube
\chardef\bicentennial=30
 % star with overlapping strokes
\chardef\oneu=31 % 1/4 of uuuu ornament
\chardef\circa=32 % quartercircle
\chardef\circb=33 % filled quartercircle
\chardef\circc=34 % rotated quartercircle
\chardef\circd=35 % cone
\chardef\circe=36 % concentric circles
\chardef\circf=37 % concentric diamonds
\chardef\fouru=38 % uuuu ornament
\chardef\fourc=39 % same, rotated
\chardef\seventh='140 % 1/7, to go with
                      % cmssqi8
```

```
\newdimen\apspix
\apspix=31448sp % 8 APS pixels = 52413.64sp,
                % and I'm taking 60% of this
% to crude approximation, there are about
% 2\apspix per pt
\newdimen\blankpix \newdimen\Blankpix
\setbox0=\hbox{\manual P} \blankpix=\wd0
% approximately 1pt blank pixel
\setbox0=\hbox{\manual R} \Blankpix=\wd0
% approximately 3pt blank pixel

\def\leftheadline{\hbox to \pagewidth{%
 \vbox to 10pt{}% to position the baseline
 \llap{\tenbf\folio\kern1pc}% to left of text
 \def\MF{{\manual 89:;<=>:}}% slanted 10pt
  \tenit\rhead\hfil% running head flush left
 }}
\def\rightheadline{\hbox to \pagewidth{%
 \vbox to 10pt{}% to position the baseline
 \def\MF{{\manual 89:;<=>:}}% slanted 10pt
 \hfil\tenit\rhead\/% running head flush r.
 \rlap{\kern1pc\tenbf\folio}% to right of text
 }}
\def\ttok#1{\leavevmode\thinspace\hbox{%
 \vrule\vtop{\vbox{\hrule\kern1pt
 \hbox{\vphantom{\tt(j}\thinspace{\tt#1}
  \thinspace}}
 \kern1pt\hrule}\vrule}\thinspace} % token

\newdimen\tinypix
\setbox0=\hbox{\sixrm0} \tinypix=5pt
\newdimen\pixcorr
\pixcorr=\tinypix \advance\pixcorr by-\wd0
\def\pixpat#1#2#3#4{\vcenter{\sixrm
 \baselineskip=\tinypix
 \hbox{#1\kern\pixcorr#2}
 \hbox{#3\kern\pixcorr#4}}}

\font\rand=random
```

Appendix B: The file manmac.tem

A little mixing of the functionalities provided by manmac and DEK's sample, supplied in the Appendix B of the TEXbook, yields the following proposal for NTG's MAPS Special report series. The proposal is molded into a template, which makes the total structure clear. An author can reuse the template and just fill-in the copy.

For convenience the auxiliary macro files are appended after the `\bye`.

Just a few macros on top of manmac and 80% or so of your report layout is there, ready for your customization. Run it, and I hope you will be surprised too. I welcome your comments.

```
%Template for manmac based MAPS report
\def\issue{%
MAPS Special 93.x            %issue
}\def\title{%
MAPS Special Template        %title
}\def\abstract{%
A template for MAPS Special is provided.
}\def\keywords{%
manmac, MAPS, NTG            %keywords
}
\input manmac                %to be
\input manmac.mod            %combined
\input ntglogo               %in one file
\input cover                 %mapsspecial
%\input toc%table of contents
%Copy proper
\beginchapter {} {}1. First\par
%{} is keyword, after. the title
%In original and my mod the . disappears,
%making it flexible with respect to
%unnumbered chapters.
\beginsection 1. Introduction\par
...%copy proper introduction
\beginsection 2.Real first section\par
...%copy proper real first section
\endchapter
%et cetera
\beginchapter Bibliography {}{}.{}\par
\item{[1]} Knuth, D.E (1984):
 The \TeX book. Addison-Wesley.
\endchapter
Further an index or so, and
on inside back cover the NTG info sheet.
%\input ntg.info
\bye                         %cgl, March 93


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Auxiliary files
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%File: manmac.mod        %modifications
\font\titlefont=cmbx12\relax
%First manmac's mod-s
\outer\def\beginchapter#1 #2#3.#4\par{%
 \def\chapno{#2#3}\global\exno0 \subsecno0
 \def\hl{\gdef\hl{\issue\hfil\it\rhead}}
 \headline{\hl}
 \def\\{ }\xdef\rhead{#4}
 {\let\\\cr\halign{\line{\titlefont
  \hfil##\hfil}\\#1 #2#3 #4\unskip\\}}
 \bigskip\tenpoint\noindent\ignorespaces}
%
\outer\def\endchapter{\vfill\eject}
%
\output{\output}
%
\hsize17cm\pagewidth\hsize
\vsize24cm\pageheight\vsize
\def\date{\ifcase\month\or Jan\or Feb\or
 March\or April\or May\or June\or July\or
 Aug\or Sept\or Oct\or Nov\or Dec\fi
 $\,$'93}
\endinput                    %cgl, March 93

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%File: ntglogo.tex
\font\calx=cmsy10 scaled \magstep 4
\font\rmx =cmr10  scaled \magstephalf
\def\NTG{{\rmx\noindent\vtop to0pt{%
 \hbox{{\calx N}ederlandstalige}
 \hbox{\hskip1em\relax{\calx T}\raise
  -.5ex\hbox{E}X}
 \vskip0.1ex
 \hbox{\hskip2em{\calx G}ebruikersgroep}
 \vss}}}
\font\hmx=cmr10 scaled\magstep0
\def\NTGADR{{\hmx\noindent\vtop to 0pt{%
 \hbox{Postbus 394}
 \hbox{1740 AJ Schagen}\vss}
}}
\def\NTGKOP{\NTG\hfill\NTGADR\quad}
\endinput                     %cgl March 93

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%File: cover.tex
\nopagenumbers%Cover and inside cover
\NTGKOP\vskip4.5cm
```

```
\centerline{\titlefont\title}
\medskip\centerline{by}\medskip
\centerline{\bf C.G.\ van der Laan%
 \footnote*{Hunzeweg 57, 9893PB,
  Garnwerd, The Netherlands.
  email: cgl@rug.nl.}}
\vskip3cm
\setcornerrules
\vskip9cm%paste-up illustration
\vfill\centerline{\issue}\vfill
\eject
%inside cover
\null\vfill
\centerline{\bf Abstract}
\par{\smallskip\noindent\abstract
    \smallskip}\par\vfill
\noindent{\bf Keywords}: \keywords.
\vfill\eject
%
\pageno1
\footline{{\fiverm\rlap{\date}\hfil
 --\folio--\hfil\llap{\copyright cgl}}}
\endinput               %cgl, March 93
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

A table of contents can be made via extracting the chapter and section titles via the use of a programmable editor: extract the lines which contain `\beginchapter...`, and `\beginsection...`, store these, and add

```
\bgroup\beginchapter{} {}{}.Contents\par
\def\beginchapter#1 #2#3. #4\par{\par
 \noindent#2#3 #4}
\def\beginsection#1\par{\item{--}#1}
\input toc
\egroup
```

The above requires that the titles are ended by `\par`, and all one-liners. To circumvent extra editing there should be no copy on the same line after the `\par`. The contents as supplied below has been created in this way.

When page numbers are also wanted, we have to write the entries to a file, and let the OTR do the work.

Appendix C: Contents

AMS BLUes

professionals at work

Kees van der Laan

Hunzeweg 57
9893 PB, Garnwerd, The Netherlands
cgl@rug.nl

Abstract

The significance of the American Mathematical Society for the TEX community at large, and more general the leading role of the AMS in the area of professional computer-assisted typesetting, is praised.
$\mathcal{AMS}$-TEX, $\mathcal{AMS}$-LATEX, their accompanying styles `amsppt.sty`, respectively `amsart.sty`, as well as AMSfonts are discussed. AMS provides excellent user's and installation guides along with the software and fonts, all in the public domain. Despite the quality, an alternative approach—and in detail some alternative encodings—are provided.
A publisher is strongly encouraged to take notice of the computer-assisted publishing activities of this pace-setting society.
A new procedural idea with respect to specifying and formatting bibliographies, given a background file of all the references an author is familiar with, is proposed to suit the author and the publisher.

Keywords: Computer-assisted typography, math, bibliography, mark up, typesetting, $\mathcal{AMS}$-(LA)TEX, (math and cyrillic) fonts, plain TEX, macro writing, education.

Introduction

This is the second paper in the series about general, public domain macro collections to format complex math-oriented documents. $\mathcal{AMS}$-TEX has been developed by Michael Spivak. Frank Mittelbach, Rainer Schöpf, and Michael Downes from AMS, continued the project with $\mathcal{AMS}$-LATEX, in the early nineties.

Next to these basic macro collections some styles have been designed to mimic the layout of the various AMS publication series. For an author these are even more important.

For a survey of how the production process goes at AMS, see Youngen, 1991. At the moment it is difficult to tell what belongs to a certain style and what is part of the underlying $\mathcal{AMS}$-TEX. It is alive, and. . . boundaries are moving.[1]

Reality has it, that the AMS has succeeded in maintaining the high-quality of the appearance of their publications[2] while adopting new TEXnology. My wishful thinking is

> To maintain the same appearance in print, independent of whether the copy has been marked up via `amsppt.sty`, that is $\mathcal{AMS}$-TEX, or via `amsart.sty`, that is $\mathcal{AMS}$-LATEX.[3]

There are two author interfaces

> TEX- vs. LATEX-oriented.

Although both are aimed at mimicking AMS traditional layout, they yield different results in print, especially the amount of white space, and the numbering of theorems and the like (see the templates in the Appendixes B and C). They don't provide the same functionalities either. That LATEX's goodies are taken for granted is understandable. It is curious why the underlying `amstex.tex` etc. formatting has not been used for the other parts. For the mark-up of bibliographies BiBTEX can be used, next to LATEX's bibliography tool, instead of interfacing to `amstex.tex`'s `\Refs`. . . `\endRefs` et cetera.

That the similarity has not been attained completely is understandable. LATEX has its conventions, which

[1] As a rule of thumb the following from `amsppt.doc`: 'For the most part the internals of `amsppt.sty` aren't closely related to `amstex.tex`: `amstex.tex` handles mathematics, `amsppt.sty` handles visual design and overall structure of documents.' LATEX too did not define clearly what should be part of the sty-file and what not. Indeed sty-files are provided but also files which handle the sizes of the fonts.

[2] The contents is not the issue here.

[3] At the moment this is not the case.

are hard to supersede, I suppose. A balance had to be found, which does justice to TEX and LATEX. It is subject to change into a more abstract and more functionally oriented author interface. The latter is perhaps my wishful thinking.

For the author the `amsppt.sty`, respectively `amsart.sty`, has emerged as the style to work with. AMS staff will take care of the details needed to typeset the electronically prepared manuscript via the other styles.

In the sequel I'll treat

- What is provided by AMS? The tools, the documentation, the services, and the philosophy, next to the hardware requirements
- $\mathcal{AMS}$-TEX, `amsppt.sty`, the Joy of TEX's math, and how to access fonts
- $\mathcal{AMS}$-LATEX, `amsart.sty`, some remarks about LATEX's math, and how to access fonts
- AMSfonts, what is provided in this collection, how to load and access
- Appendixes.
  A: How to get it?
  B: A rough template of an author script with respect to `amsppt.tex`.
  C: A rough template of an author script with respect to `amsart.sty`.
  D: Contents (the table of contents of this article).

It is in the nature of this paper that it is hard to read, because of the many issues treated—plain, $\mathcal{AMS}$-TEX, LATEX, BiBTEX, $\mathcal{AMS}$-LATEX, AMSfonts, NFSS—on the one hand, and because of the necessary amount of detail on the other hand. The parts $\mathcal{AMS}$-TEX and $\mathcal{AMS}$-LATEX are independent from each other; in both the use of the AMSfonts collection is discussed. A BLU type reader is encouraged to grasp the essentials from the abstract, the conclusion, or the titles and examples of the main sections.

> If only BLUe would start reading one of the AMS Guidelines, or gaze at the templates provided in Appendix B and C, I have reached my goal already.

What is provided by AMS?

Apart from the general goodies of publishing with the AMS, they provide TEXnically

- Software.
  As basis collections there are $\mathcal{AMS}$-TEX, and $\mathcal{AMS}$-LATEX. The next layer is formed by the styles `amsptt.sty`, respectively `amsart.sty`, with variants for books, bulletins, journals,... . One-column format is used throughout. The encodings are highly systematic and consistent.
- Documentation.
  Especially useful are the 'Guidelines for Preparing Electronic Manuscripts.' One for $\mathcal{AMS}$-TEX-oriented authors, and a similar one with respect to $\mathcal{AMS}$-LATEX. The main purpose of these booklets is to provide authors with know-how about mark-up and to inform them about procedural aspects, such as submission rules. These papers contain much worthwhile to know for preparing electronic manuscripts in general—if not for the samples in the appendixes— although biased towards $\mathcal{AMS}$-TEX. Then there are the user's and installation guides. All very-well done!
- Fonts.
  The collection is called AMSFonts, version $\geq$ 2.1. There is a user's and an installation guide.
- Support.
  The collections are professionally maintained. No volunteers! email: tech-support@math.ams.org.

Hardware requirements. Hardly none. Just some ten-s of MBytes for the fonts. $\mathcal{AMS}$-LATEX works best with a big TEX, because of the many control sequences in use.

Not in there. Remarkable is that after so many years of TEX being around, AMS is modest and realistic in its use of it. AMS does not deal yet with hard issues like cost-effective electronic inclusion of graphics, and the preparation of tables. It is left to the author to do it by (LA)TEX, or by other means. Indexing tools are not the issue for the `amsppt` and `amsart` styles. There are no facilities to include files verbatim in $\mathcal{AMS}$-TEX (Not so much needed, given the context.) In $\mathcal{AMS}$-LATEX an option is available which provides Schöpf's `verbatim.sty`.

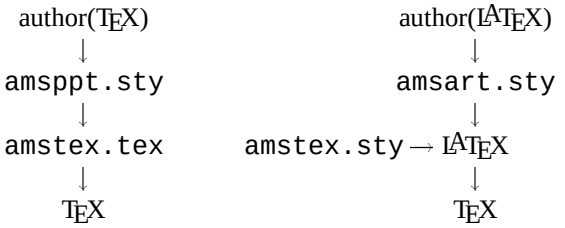
Believe it or not. $\mathcal{AMS}$-(LA)TEX, especially its accompanying customized `amsppt.sty`, is the 'camino royal' for every AMS author.

> Reality[4] has it, that `amsppt.sty`, respectively `amsart.sty`, gets more and more the role of the mark-up tool for each and every AMS author, with the AMS staff taking care of the finishing touch.

The latter involves solving hard mark-up problems, or more commonly, the handling of the details required by the style for the concrete publication series, or laying hands on a special symbol provided within the AMSfonts collection.

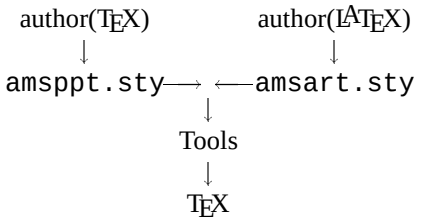
[4] To be honest my wishful thinking.

This approach can be illustrated by the following scheme[5]



Alternative approach

In my opinion a better company policy would have been to allow for a LaTeX author interface, next to the TeX one, as depicted by the following scheme.



I omitted on purpose `amstex.tex`, because I consider the math mark-up as proposed there, as syntactic sugar of what has already been provided by plain. The tools are the macros provided by the TeX community at large, for example PiCTeX, (Bordered) Table macros, Commutative diagrams macros,. . . , see the Jones index for a survey.

Notations `\ea`, `\nx`, and `\ag`, are used as shorthand notations for `\expandafter`, `\noexpand`, respectively `\aftergroup`. Now and then 'we' is used, especially when text has been borrowed from documentation provided by AMS. Read in those cases 'AMS.'

$\mathcal{AMS}$-TEX

If only the reader would now turn over to the excellent 'Guidelines for preparing electronic manuscripts—$\mathcal{AMS}$-TeX,' much would have been gained.

1 The $\mathcal{AMS}$-TEX package

The $\mathcal{AMS}$-TeX package consists of

- amstex.tex, amstex.doc
- some styles, notably amsppt.sty, amsppt.doc
- amssym.def, amssym.tex, with symbol definitions

along with the documentation[6]

- $\mathcal{AMS}$-TeX User's Guide
- $\mathcal{AMS}$-TeX Installation Guide
- Guidelines for preparing electronic manuscripts—$\mathcal{AMS}$-TeX.

In this part I'll treat

- amsppt style
- some math mark up details, and
- the use of AMSfonts.

1.1 Styles

A document is thought to consist of[7]

- preliminary part,
- top matter, and
- the document.

The preliminary part contains

```
\input amstex
<declarations and font loading>
\documentstyle{amsppt}
```

The top matter consists of the title, author and the like information. See the user's guide.

The document itself consists of `\head...\endhead`-s. Nested therein the `\subhead...` and `\subsubhead...`-s, and paragraphs. Appendixes are just `\head... \endhead`-s. At the paragraph level the plain and $\mathcal{AMS}$-TeX (math) constructs can be used. The section about formatting references is special.

Strong points

- Structuring macros have explicit ending macros[8]
- Very-well developed top matter material
- A fancy bibliography tool[9]
- Elaborated context related error detection, correction, and messages
- Access to the rich AMSfonts collection
- The syntax-checking-only run mode has been made available, via `\syntax`, with confirmation interactively.

Not in there

- No automatic cross-referencing
- No full-fledged 14pt for title matter
- No automatic numbering schemes.

1.2 Encoding

With respect to the encoding I will discuss the systematic use of control sequences `\nofrills`, `\title`,

[5] Neglecting the AMSfonts files.

[6] In booklets and also available in electronic form.

[7] A more detailed structure is provided in the template in Appendix B: `amst-art.tem`.

[8] In the spirit of SGML's complete mark-up.

[9] The marked up fields can be supplied in arbitrary order, preceded by just an opening tag and no explicit closing tag, in the spirit of SGML's minimal mark-up.

\head, \theorem and the like,[10] and the mark up of the list of references. Furthermore, I will discuss the used font selection scheme.

Encoding: \nofrills

Purpose. \nofrills is a general optional parameter to overrule the default typesetting. From the example in the user's guide it turns out that the user *needs* it for appendixes, where \chapter has to be used and a caption different from CHAPTER is needed, via for example

```
\chapter\nofrills{Appendix B}...
```

or in general when captions in another language are wanted, for example

```
\Refs\nofrills{Literatuur}
```

For several macros, e.g., \title, we want to allow the user the following two options

```
\title ...\endtitle
\title\nofrills ...\endtitle .
```

In the first case, we want the text of the title to be stored in a box for later use. In the second case, we want certain parts of the typesetting to be omitted when constructing the box. Internally this is done by prefixing them with \frills@, which in the normal cases is simply an identity function (\let\frills@\identity@), whereas in the nofrills case it gobbles its argument (\let\frills@\eat@).

In order to test for \nofrills we have to use \futurelet (because if \nofrills is absent, we don't want to do anything that might affect the actual text of the title, such as stripping braces or adding an unwanted extra pair around the first token or group).

Use. \nofrills must immediately follow the control sequence it applies to. For example

```
\title\nofrills...\endtitle .
```

Typical use: if we want to execute a macro called \title, but check to see if \nofrills is coming up next, we do this via

```
... \nofrillscheck\title
```

If indeed the next thing is \nofrills, then we will execute \nofrills@@ before calling \title again. This involves some shuffling around in order for us to get back the 'real' definition of \title afterwards and yet still be able to use a control sequence named \title to read the argument.

> This is important in giving the user a useful error message if they accidentally omit or misspell the matching \endtitle or whatever.

We could use instead the trick used with \align[11] of adding a space onto the \csname with which we read the argument, but that would increase the use of hash size.

Design. The general encoding mechanism for \nofrills, to be applied to the generic \⟨*tag*⟩, reads

\def\⟨*tag*⟩{\let\savedef@\⟨*tag*⟩
\def\⟨*tag*⟩⟨*parameters*⟩{\let\⟨*tag*⟩\safedef@
⟨*code of tag for default formatting*⟩}
\nofrillscheck\⟨*tag*⟩} .

The effect of the above template is that the user's invoke of \⟨*tag*⟩ is replaced by

\nofrillscheck\⟨*tag*⟩

with \⟨*tag*⟩ temporarily redefined. In this redefinition use is made of

- \next equals the token after \⟨*tag*⟩
- \next@ equals \⟨*tag*⟩, and
- \frills@ equals either \eat@ or \identity@

with \eat@ the function to gobble its argument and \identity@ the identity function.

The code. From the file amsppt.doc I borrowed the following.

```
\def\nofrillscheck#1{\def\nofrills@{%
 \nofrills@@{#1}}\futurelet\next\nofrills@}
%
\def\nofrills@@#1{%
%This extra step in defining \next@
%is a precaution in case #1
%might be more than one token.
%Parameters use:
%\next contains the 'token after the
% item token,' e.g. \nofrills
%\next@ contains the item token, e.g.
% \title
%\frills@ is either \eat@ or \identity@
 \DN@{#1}%\def\next@{#1}
 \ifx\next\nofrills \let\frills@\eat@
   \ea\ea\ea\next@\ea\eat@
 \else\let\frills@\identity@\ea\next@\fi}
```

Explanation. The purpose of the redefinition of \⟨*tag*⟩ is to supply as argument of \frills@ the process to be executed, and to give \frills@ the appropriate meaning.[12]

\savedef@ stores the original \⟨*tag*⟩ definition. The next redefinition reestablishes the original definition, and defines \frills@ either as \eat@ or \identity@, given the values of \next and \next@. At the end it invokes \nexta. \nofrills is eaten when present. And that is about it.

[10] With \add@missing—the former \runaway@—as the general error detecting (and correcting) macro.

[11] Spivak, 1986.

[12] This is a discipline for the encoders and not at all for the authors. The authors are ignorant about this. The encoding team creates in the style file from the template versions for every control sequence which allows for an optional \nofrills.

1.3 Encoding: \title

The title control sequence is typical for the top matter material. Once it is understood, the other control sequences won't provide mysteries anymore.

Purpose. The purpose is to typeset the title, default in upper case.

The code. From amsppt.doc the following.

```
\newbox\titlebox@
%
\def\title{\let\savedef@\title
%We use a \vtop here because we want to
%know the height of the first line of
%the title when we start typesetting
%the topmatter. In order to get the
%correct sinkage from the top of the page.
 \def\title##1\endtitle{\let\title
  \savedef@\global\setbox\titlebox@
  \vtop{\tenpoint\bf\raggedcenter@
%Increased \baselineskip is because of the
%uppercasing.  We do it like this instead
%of putting it inside the \uppercasetext@
%macro, because it only applies where an
%entire paragraph is made up of
%upper case text.
  \baselineskip1.3\baselineskip
  \frills@\uppercasetext@{##1}\endgraf}%
%Check to see if right and left hand
%running heads have been already assigned
%by the user---if so, don't override.
 \ifmonograph@\edef\next{%
                     \the\leftheadtoks}%
  \ifx\next\empty@\leftheadtext{##1}\fi
 \fi
 \edef\next{\the\rightheadtoks}
  \ifx\next\empty@
                \rightheadtext{##1}\fi}%
 \nofrillscheck\title}
```

Explanation. For completeness I also borrowed from amsppt.doc the following explanation. We store the title in a box using a \setbox. It could also be done by defining \thetitle@.

> One advantage of \setbox is that syntax errors within the title (say for math) are reported immediately instead of during the processing of \endtopmatter.

The title box is always put on the page by \endtopmatter, even if it's empty. (With other pieces of the topmatter we check first to see whether there's any text to typeset and skip to the next item if not.) The text of the title is stored up for the running heads, unless \rightheadtoks is nonempty—then presumably the user has already used \rightheadtext to set the right-hand running head, in which case we leave it unchanged. The default for titles is uppercasing. From a design standpoint it might be preferable to do titles in 14-point text with initial caps, but since titles may potentially contain any kind of math, doing this would require a full-fledged \fourteenpoint analogous to \tenpoint. So to give the title extra prominence we use uppercasing instead. The uppercasing can be removed by \nofrills.

Alternative encoding. For the title part the use of \nofrills comes down to override the default (upper case) typesetting. But if that is all what has to be done, we can simpler provide the author with the possibility to provide the name (and eventually mark-up) in \titlename. If the latter control sequence remains empty, then the user apparently accepts the default. The encoding is simple: just test for the emptiness of \titlename.

Furthermore there is the general question about end punctuation.

> Why not parameterize over the end punctuation as well? For example via \titlepunctuation?

If the user wants his own end punctuation, the latter control sequence can just be redefined. This could have been a general design choice—parameterizing over the end punctuation—eventually in a global way per level.

Remark. Practice has it that publications take long titles, despite the advice to keep them short. In the headlines of the paper a so-called (short) running title is supplied. In $\mathcal{AMS}$-TEX this is done for the right pages.[13] AMS provides however the possibility to supply next to the title, an optional short title in \righthandtext, and ipso facto for author names in \lefthandtext.

1.4 Encoding: \chapter

Nested within the title part we can have \chapter, again with the \nofrills option. For example

```
\title\chapter\nofrills{Appendix B} The
 Poisson Integral\endtitle .
```

Especially when the publication is a monograph.

In amsppt.sty an appendix heading is marked up via

```
\head*{} Appendix*.
```

This shows that it is at least confusing to treat amsppt.sty along with the book style in one booklet, and also to have one code which accounts for both.

Purpose. The function is to typeset the chapter keyword—or the one supplied by the user (Appendix, for example)—and to typeset the supplied chapter 'number.' That is all, and one can question whether it can't be attained in a simpler way.[14]

[13] The author name(s) are used in the running heads for the left pages.

[14] There is no \endchapter, but as used it is not a structuring control sequence.

The code. From `amsppt.doc` the following.

```
\def\chapter{\let\savedef@\chapter
 \def\chapter##1{\let\chapter\savedef@
 \leavevmode\hskip-\leftskip
%Put the \chapter stuff in an \rlap so
%it doesn't affect centering of the title,
%and in an "uplap" so it is placed above
%where we want it.
 \rlap{\vbox to\z@{\vss
  \centerline{\eightpoint
%We do a baselineskip of 2pc from the
%base of the "CHAPTER" banner to the base
%of the first title line. (The baseline
%of the \null will coincide with the
%baseline of the first title line.)
   \frills@{CHAPTER\space\afterassignment
    \chapterno@\global\chaptercount@=}##1%
    \unskip}\baselineskip2pc\null}}%\vbox
  \hskip\leftskip}%end \rlap
 \nofrillscheck\chapter}
```

Explanation. Let us trace what happens after `\chapter\nofrills....`

First, the chapter definition is copied via

`\let\savedef@\chapter`

then `\chapter` is redefined, with a parameter, with the function to provide the default typesetting of the chapter keyword and the (supplied) number. The check for frills is done by the invocation of

`\nofrillscheck\chapter`.

Because our copy contained `\chapter\nofrills... \next` is defined as `\nofrills`, and `\nofrills@@{ \chapter}` is invoked. In the latter macro `\DN@{\chapter}` is processed, and then via an ingenious use of `\ea`

- `\nofrills` will disappear (is eaten),
- `\next@` is invoked (which has been `\let`-equal to `\chapter`), and
- `\frills@` will become `\eat@`,

and therefore the provided chapter-marked-data (read: alternative keyword) will be typeset in the copy. Quite something isn't it?

Alternative. At the basis of the alternative encoding for `\chapter` are the following questions.

> Why not have a control sequence `\chapterno`, that provides the contents (that is the number or something similar), and a control sequence `\chpkeyword`, for the keyword?

This could lead to the following alternative encoding, without the use of the optional parameter mechanism as encoded via `\nofrills`[15]

```
\def\chapter{\leavevmode\hskip-\leftskip
 \rlap{\vbox to\z@{vss\centerline{%
  \chapterkeywordandno}%
  \baselineskip2pc\null}}\hskip\leftskip}
%
\def\chapterkeywordandno{\eightpoint
 \chpkeyword\space\chapterno\unskip}
%
\def\chpkeyword{CHAPTER}
\def\chapterno{11}
```

In the above alternative the inner details of the chapter keyword and number encoding can be overridden by the author, for example another typeface can be used.[16] Of course one could decide to freeze the used typeface too. The point I like to make is, that the above is at least as clear—if not clearer—with the same functionality.[17] It is true however, that for an encoding-team manager it is more difficult to verify whether the same encoding discipline has been adhered to by every team member. Perhaps it must be emphasized stronger what should be the function, along with (inescapable) guidance about the discipline to be adhered to.

1.5 Encoding: `\head`

The special thing here is to prevent page breaks between `\head` and an immediate following `\subhead`, or between a `\subhead` followed by a `\subsubhead`, and the like. It is in general still an unsolved problem, as follows from the following from `amsppt.doc`.

In accordance with conventional design principles, the space below headings is not given any stretchability or shrinkability. Since we often want to do a penalty and vskip, and since there are extra complications involved if there is a preceding vskip from something else, we define a macro to do it. Normally it is used while we're working on the main vertical list, so we have to use `\removelastskip` (which does a negative skip) rather than `\unskip` (which *really* removes the last skip). If the last thing on the main vertical list is anything other than a vskip, say a penalty from `\pagebreak`, the value returned by `\lastskip` will be 0, but the potential complications are a whole other subject. This macro handles the straightforward cases.

In some cases we may not want to put a penalty at all. We refrain from doing any penalty if the first argument of `\penaltyandskip@` is 0; the essential effect of an explicit `\penalty0` can be gotten by doing a penalty of 1 instead.

We use `\penalty@` instead of `\penalty`, so that we can redefine `\penalty@` in `\nobreak`, to prevent page breaks between certain pairs such as `\head \subhead` or `\head \proclaim` or `\nopagebreak \proclaim` —for example, in

```
... which leads to this theorem:
\proclaim{Theorem 8.2} ...
```

[15] I understand that some meta-ness of encoding has been striven after. When it has to be changed only the metacode needs adaptation. I doubt it, whether it has worked out this way in reality. It is so intertwined.

[16] The end punctuation in other cases.

[17] By the way, it is such a simple task, just the keyword and the number! If that has to be handled in such a complex way, we are on the wrong track.

a page break after the colon would be bad, and a user might want to be able to add a \nopagebreak.

The sequence

```
\subhead Text...\endsubhead
\subsubhead Text...\endsubsubhead
```

will still have the weakness of allowing a pagebreak between the two headings, because \endsubhead doesn't do a \nobreak (since it's a run-in heading). If the next piece of the document after \nobreak is something like \subhead or \proclaim that calls \penaltyandskip@, then \penalty@ will reset itself in the way that we want. But if not, then we still want to reset \penalty@; so we use \everypar.

> There are probably some unusual cases that will still have problems, *but at the moment this is the best solution we have.*

We equate the old form of the headings to the new form, for backward compatibility. It's easier to do this now rather than later because \head and \subhead are going to be outer. The purpose of \restoredef@ is to work around problems caused by the outerness of things like \subhead. \relax at the beginning prevents the \savedef@ (which may be \outer) from being read prematurely in certain kinds of expansion.

```
\def\restoredef@#1{\relax\let#1\savedef@
 \let\savedef@\relax}
```

\subhead and \subsubhead are simpler, though they allow \nofrills. (\head, being centered, does not have automatic punctuation put in at the end and so \nofrills doesn't have anything to do.) The syntax of \subhead is changed from \subheading{...} (version 1) to \subhead...\endsubhead (version 2). This was done for the following reasons

- to be consistent with \head...\endhead (we could have changed \head instead of \subhead, but the \x...\endx syntax is the one currently in use in AMS production);
- if someone (perhaps us) ever wants to do something tricky with the headings, having the \end... may help them avoid technical complications.

1.6 Encoding: theorems, proofs, definitions, remarks

These structures all allow for the same mark-up scheme
\⟨tag⟩{⟨heading text⟩}
⟨general text⟩
\end⟨tag⟩

With for example as heading text
Lemma 1. in case of ⟨tag⟩ equals proclaim,
Proof in case of ⟨tag⟩ equals demo,
Example 5 in the case of ⟨tag⟩ equals example,
and the like.

Alternative

I find it very unnatural to have to provide

```
\demo{Proof}...
\endemo
```

and the like. Why not provide the structures, the user is familiar with, by their names? For example

```
\proof...
\endproof
```

The endcoding can be parameterized over \proofkeyword{Proof}, or something like that? Eventually as alias? This should all be done behind the scenes, and not bother the author.

Intermezzo *(Error detection and correction)*

For these environments a general error detection scheme has been encoded, based upon the concept of an environment stack. This stack keeps track of which environment the formatting process is in.

The environment stack \revert@ is a toks variable. \environ@stack stacks its argument in \revert@ and also as replacement text of \environ@end.[18] When an environment is ended the invocation of \revert@envir, with the appropriate argument, restores the previous \envir@end—pops up the stack—via the invocation \the\revert@.

The code. From amsppt.doc I borrowed the following. \revert@ and \envir@stack are for use by any environments that don't enclose their text in a group, e.g., \proclaim, \definition, \roster.

```
\newtoks\revert@
%
\def\envir@stack#1{\toks@\ea
 {\envir@end}
 \edef\next@{\def\noexpand
  \envir@end{\the\toks@}\revert@
  {\the\revert@}}%
 \revert@\ea{\next@}%
 \def\envir@end{#1}}
%
\gdef\revert@envir#1{\ea
 \ifx\envir@end#1\the\revert@
 \else\ifx\envir@end\enddocument
          \Err@{Extra \string#1}%
      \else\ea\add@missing
       \envir@end\revert@envir#1%
      \fi
 \fi}
```

Explanation. The expansion in \envir@stack is difficult to understand. So let us go through it step by step, where I use \end⟨tag⟩ as a generic name for the end tag. The replacement text of

```
\envir@stack\<endtag>
```

[18] The programmer can easily verify in what environment he is in, by inspection of \environ@end.

after expansion is

```
\toks@{\<current endtag>}
\def\next@{%
        \def\envir@end{\<current endtag>}
 \revert@{\<current revert@}}
%Substituting \next@ yields
\revert@{\def\envir@end{\<current endtag>}
 \revert@{\<current revert@}}%(1)
\def\envir@end{\<endtag>}    %(2)
```

This can be read more easily as

1. `\revert@` has as replacement text the definition of the ⟨*current endtag*⟩ together with the (current) stack assignment
2. `\envir@end` has as replacement text the `\end`⟨*tag*⟩.

The curious thing is that the `\end`⟨*tag*⟩ is stored, and the *definition* is stacked, next to the stacking of the `\revert@` assignment. The *separation of concerns* principle of programming has it, that code is easier to read and maintain, when the different tasks are separated. For the concrete situation at hand this means that the stacking and (re)definition can better be separated.

To complete our understanding, let us follow the 'pop-up' process step by step. After the above, the replacement of

```
\revert@envir{\<endtag>}
```

after expansion reads

```
\def\envir@end{\<current endtag>}
\revert@{\<current revert@>}
```

For arguments different from `\end`⟨*tag*⟩ an error message will be supplied and the missing `\end`⟨*tag*⟩ will be inserted via the invocation of `\add@missing{\<endtag>}`.

The error detecting (and correcting) macro reads

```
\def\add@missing#1{\ea\ifx\envir@end#1%
 \Err@{You seem to have a missing or
  misspelled \ea\string\envir@end ...}%
%It is useful to supply the necessary
%missing piece, especially in the case of
%\endref.
 \envir@end
\fi}
```

Alternative. At the heart of the alternative lies the question
'Why not use DEK's general `\leftappend` and `\lop`, TEXbook p.378, instead of `\envir@stack`, respectively `\revert@envir`?'

I'm not saying that DEK's macros are easier to understand. But, once mastered, they are more general—a double ended queue, de-queue for short, has been implemented, instead of a stack—and therefore can be used for other cases as well. Of course DEK's macros can be tailored for this special situation. This, together with not stacking the definition, yields as alternative

```
\newtoks\revert@%the stack variable
\revert@{\empty}\def\envir@end{\empty}
%To push
\def\envir@stack#1{\ea\ea\ea\revert@
 \ea\ea\ea{\ea\envir@end\the\revert@}
 \def\envir@end{#1}}
%To pop up
\def\revert@envir#1{\ea\pop\the\revert@.}
\def\pop#1#2.{\revert@{#2}%
        \def\envir@end{#1}}
```

Explanation. In pushing the stack `\revert@` is left appended by the replacement text of `\envir@end`, and the argument of `\envir@stack` is stored as replacement text of `\envir@end`. In popping up the stack the first token of `\revert@` is stored as replacement text of `\envir@end`, and the rest, also called tail, is (re)stored in the stack toks variable `\revert@`.

I have used the same names as those of AMS, but general names, related to a stack of tokens, would have been more appropriate.

End intermezzo

The code. After the above intermezzo we can understand the code, and the error detection and correcting parts of it, more easily. As usual with AMS the other environments are similarly encoded.
From `amsppt.doc` I borrowed the following as example.

```
\outer\def\proclaim{%
 \let\savedef@\proclaim\let\proclaim\relax
 \add@missing\endroster
 \add@missing\enddefinition
 \add@missing\endproclaim
 \envir@stack\endproclaim%push on the stack
%penalty-100 is the penalty amount used by
%plain.tex's \medbreak.
 \def\proclaim##1{\restoredef@\proclaim
  \penaltyandskip@{-100}\medskipamount
  \varindent@
  \def\usualspace{{\proclaimheadfont@
   \enspace}}\proclaimheadfont@
   \ignorespaces##1\unskip\frills@{.%
   \enspace}%
  \proclaimfont\ignorespaces}%
 \nofrillscheck\proclaim}
```

Explanation. First, the usual alias is created. Then, because of the outerness, it is `\let`-equal to `\relax`. Then via the various `\add@missing\...` it is verified whether we are still in one of those environments. The name `\envir@end` contains the \end⟨*tag*⟩ of the current environment, and therefore it can be compared with the argument of `\add@missing`.

`\endproclaim` ends the paragraph, switches back to `\rm` and adds spacing. (This means that if, for some strange reason, a whole section of text happens to be in italics, then the user must type `\it` again after each `\endproclaim`, but that seems too special a case to need providing for.) The penalty of 55 is just the plain.tex penalty for `\endproclaim`, carried over without change. Version 2.1 change: removed `\outer` prefix, to simplify some programming related to `\add@missing` and `\revert@envir`.

```
\def\endproclaim{%
         \revert@envir\endproclaim
 \par\rm\penaltyandskip@{55}%
                   \medskipamount}
```

1.7 References

The encoding is complicated, but the mark-up possibilities are rich. The macros are powerful and ingenious.

Purpose. The mark-up of bibliographic information in an SGML-like way.

Use. All the references have to be enclosed by `\Refs...\endRefs`. Within these each reference starts with `\ref` and ends with `\endref`.

Between the latter tags the (marked up) detail fields can be supplied in any order. The detail fields don't take an explicit closing tag! Some special tags are `\bysame`, when the reference has the same authors as the previous one, and the `\...info` fields.

The *end* punctuation, whether supplied or not, is taken care of by the macros, unless `\nofrills` is used to override the default formatting. The kind of label is implicitly specified via the use of the tags

- `\no`—the supplied number as label—respectively,
- `\key`—the supplied label, usually letters.

The formatting of the labels—enclosed by square brackets—is done by the macros. The width of the label can be adjusted via `\widestnumber`.

A typical example of mark up is

```
\Refs%\nofrills{Your heading}
\ref\no 1
\by D.E. Knuth
\book The \TeX book
\publisher Addison-Wesley
\yr 1984
\endref
...
```

A glimpse at the code. The encoding with comments included, is some 500 lines. Therefore, I'll only deal with the outer `\Refs`, `\ref`, `\by`, and the TEXnical `\makerefbox`.

```
\outer\def\Refs{\add@missing\endroster
 \add@missing\endproclaim
 \let\savedef@\Refs
 \let\Refs\relax %because of \outer-ness
 \def\Refs##1{\restoredef@\Refs
%For a monograph where the title of the
%References section is done using \title,
%we want to omit the normal "References"
%heading and the vertical skips above and
%below.  This can be accomplished using
%\Refs\nofrills{}.  As long as the vskip
%at the end of \endtopmatter is not less
%than \aboveheadskip and \belowheadskip,
%this will be accomplished by the fact
%that \penaltyandskip@ doesn't add to a
%previous larger vskip, and the ragged
%center part will simply vanish if #1 is
%empty.
 \if\notempty{##1}\penaltyandskip@{-200}%
  \aboveheadskip
  \begingroup \raggedcenter@\headfont@
     \ignorespaces##1\endgraf\endgroup
  \penaltyandskip@\@M\belowheadskip
 \fi
 \begingroup\def\envir@end{\endRefs}%
  \refsfont@\sfcode'\.\@m}%
%This line here is a little tricky. If a
%\nofrills is found when we look ahead,
%then \frills@ will become equal to \eat@
%and it will eat "References" before \Refs
%is allowed to read its argument.
%Also we have to use a \csname trick to
%get around the outerness of \Refs.  MJD
 \nofrillscheck{\csname Refs\expandafter
  \endcsname\frills@{{References}}}}
%
\def\endRefs{\par % This will check for a
 \endgroup}       % missing \endref, also
%
\def\ref{\par
 \begingroup \def\envir@end{\endref}%
%Start the reference.
 \noindent\hangindent\refindentwd
%Change \par so that it will supply a
%(presumably) missing \endref, with an
%error message.
 \def\par{\add@missing\endref}%
%\nofrills@list should always be assigned
%globally, to conserve save stack.
 \global\let\nofrills@list\empty@
%Change \linebreak and \mathbreak to work
%properly in the special ref environment.
 \refbreaks
 \procpaper@false \book@false \moreref@false
%Start an initial box, to match up properly
%with the first upcoming \makerefbox; this
%will be discarded.
 \def\curbox{\z@}\setbox\z@\vbox\bgroup
}
%
\def\endref{%
%To wind up the preceding box it is
%convenient to call \makerefbox again;
%it will also open a new box, however,
%so we give it arguments \thr@@ and
%\endgraf\egroup that will cause the new
%box to be closed immediately and discarded.
%But we must first make sure box 3 is void
%or we'll trigger an error message. This
%is done by dumping the current contents
%of (global register) box 3 into (local
%register) box 2; the \box command always
%makes its argument globally void.
%(Because box 0 is used heavily in
% \makerefbox, it's easier to just use
% box 2 here, rather than try to verify that
% using box 0 would be safe in all cases.)
 \setbox\tw@\box\thr@@
 \makerefbox?\thr@@{\endgraf\egroup}%
%Then we call \endref@ to take all the saved
%material and combine it into a paragraph,
%adding punctuation to separate pieces.
 \endref@
%The \endgraf is done here rather than in
%\endref@ because in \moreref or \transl
%cases \endref@ shouldn't do the \endgraf.
 \endgraf
%Finally, we need to close the group that
%was started by \ref. This has the effect
%of killing the current definition of
%\envir@end, among other things.
 \endgroup\keyhook@
 \global\let\keyhook@\empty@
```

```
%\global to conserve save stack
}
%
\def\by{\makerefbox\by\bybox@\empty@}
\newbox\bybox@
%
%\makerefbox takes three arguments:
%- the first is the name of the calling
%  macro, for use in error messages;
%- argument 2 is the box used for storing
%  data (note: some boxes are shared
%        by more than one calling macro);
%- argument 3 is additional material
%  (optional---may be empty) that may be
%  used to affect the contents of the box.
\def\makerefbox#1#2#3{\endgraf
%Set box0 to the just-completed line of
%text.
 \setbox\z@\lastbox
%Although \holdoverbox will usually be void
%it still doesn't hurt to \unhbox it here
%in every case, which simplifies the
%programming.
 \global\setbox\@ne\hbox{\unhbox\holdoverbox
%\ifvoid test is necessary here to prevent
%a \linebreak at the end of a field from
%being removed by the \unpenalty.
   \ifvoid\z@\else\unhbox\z@\unskip\unskip
                  \unpenalty\fi}%
 \egroup%ends the group from the previous
%If box 1 is empty (width <= 0pt) then set
%the current box to void (it might still
%have a baselineskip glue or something in
%it at this point, for one thing).
%Otherwise set it to box 1.
 \setbox\curbox\box\ifdim\wd\@ne>\z@ \@ne
                   \else\voidb@x\fi
%That finishes the previous box.
%Now let's start a new one using
%the box given as arg 1 of \makerefbox.
%But first check to see if it's void and if
%not give an error message.
 \ifvoid#2\else\Err@{Redundant \string#1;
  duplicate use, or mutually exclusive
  information already given}\fi
 \def\curbox{#2}\setbox\curbox\vbox\bgroup
  \hsize\maxdimen \noindent#3%
}
```

Explanation. `\ref` opens a (zero) box, which is immediately closed in the replacement text of any field tag. Each field is stored in a box. But not via simply enclosing the contents between braces or so. `\makerefbox` opens the box. The copy is processed further—thus processed in the box—and upon the invocation of the next tag, the box is closed and a box corresponding to this new tag is opened. And so on. In between many subtle details have been taken care of, which makes the code hard to understand, even the main lines.

> This boxing via the *chain* of `\makerefbox`-es is an ingenious application of the two-part macro TEXnique.

In `\endref` the macro `\endref@` is invoked, which winds up all the preceding boxes, and closes in a tricky way the last opened one, via `\endgraf\egroup` as third parameter of `\makerefbox`.

En-passant error detection is taken care of: a blank line or `\par` will entail a message: `\end...` tag is missing.

Alternative. There are alternatives available but they are generally overdone. I consider it simpler to work with a basic TEXnique, which I completely understand and which is sufficient and handy.

Personally, I experienced the inconvenience to select and update a set of references from my collection of literature references, every time I prepare a paper.

To overcome this extra and error-prone work, I decided to mold my file of (annotated) literature references into a sequence of TEX definitions.

> The idea is that while preparing a publication, I just have to supply the *list of names* of the definitions, appropriately separated, with the typesetting done automatically, eventually after having sorted the list.[19]

The automatic suppression of identical author names in subsequent references is nice and context sensitive. The substitution of the author name part by a rule is done by the AMS, but there it has to be marked up for, while it can be automated easily.[20] Also context sensitive are the suffixes in the year of publication, and these are handled in a similar way.[21]

There you go. The specification of my references section comes down to

```
\input references
\head*References*%TUGboat.sty's \head
\ls\knuthdea
\ls\knuthdeb
\ls\laancgvanderd
\ls\lamportla
```

My file of references contains among others

```
\def\knuthdea{Knuth, D.E (1973):
 The Art of Computer Programming.
 {\it Sorting and Searching.}
 Addison-Wesley.}
%
\def\knuthdeb{Knuth, D.E (1984):
 The \TeX book. Addison-Wesley.}
%
\def\laancgvanderd{Laan, C.G van der (1992):
```

[19] The approach is similar to my enhancements of Amy Hendrickson's sorting of address labels, as detailed in Sorting in BLUe.

[20] In keeping these kinds of things behind the scenes it does not not burden an author at all.

[21] It must be confessed, that I did not look ahead much, and therefore the first year is there and subsequent years replaced by a rule and a, b, etc. The latter is not common. Generally, the first year starts with the suffix a. My approach is simple to encode and also unambiguous. Of course, I can make it in agreement with current practice—the first year takes an a, by the same name-year references—by storing the reference, and to look ahead appropriately.

```
 Syntactic Sugar. MAPS92.2, 130--136.
 (Also GUST Bulletin 1993, and submitted
  TUG\,'93.)}
%
\def\lamportla{Lamport, L (1986):
 \LaTeX\ User's Guide \& Reference Manual.
 Addison-Wesley.}
```

To typeset this nicely, I use as list separator

```
\def\ls#1{\ea\bibitem#1}
%with
\newcount\bcnt \newcount\suffixcnt
\let\lstnme\relax \let\lstyear\relax
%
\def\bibitem#1(#2){\global\advance\bcnt1
 \def\authornme{#1}\def\authoryear{#2}%
 \ifx\lstnme\authornme
  \def\authornme{------}%
  \ifx\lstyear\authoryear
   \global\advance\suffixcnt1
   \def\authoryear{--}%
  \else\let\lstyear\authoryear\suffixcnt0
  \fi
 \else\let\lstnme\authornme
      \let\lstyear\authoryear\suffixcnt0
 \fi
 \item{[\the\bcnt]}\authornme\,(\authoryear
  \suffix)}%end \bibitem
%
\def\suffix{\ifcase\suffixcnt\or a\or b\or
 c\or d\or e\or f\or g\or h\or i\or j\or
 k\or l\or m\or n\or o\or p\or q\or r\or
 s\or t\or u\or v\or w\or x\or y\or z\fi}
```

The above yields

[1] Knuth, D.E (1973): The Art of Computer Programming. *Sorting and Searching.* Addison-Wesley.
[2] ——(1984): The TEXbook. Addison-Wesley.
[3] Laan, C.G van der (1992): Syntactic Sugar. MAPS92.2, 130–136. (Also GUST Bulletin 1993, and submitted TUG '93.)
[4] Lamport, L (1986): LATEX User's Guide & Reference Manual. Addison-Wesley.

On second thoughts. Should we embarrass authors who publish with AMS, with the details of the mark-up of references at all? Isn't it sufficient for an author just to supply entries to the database of literature references available at AMS? I mean it is already there and the only important thing for an author is

> To know and supply the relevant labels for the entries.

Then the problem of consistent formatting of the references has been transformed into the logistics of providing each author with access to the bibliographic database, especially for the labels to be provided. Ça va sans dire, that it should be possible for an author to copy the references as well, in order to get more complete proofs. This is similar to copying macros from the file servers, which is current practice nowadays.

The above can be applied even if authors like to format their own references. The publisher just replaces the list of marked up references by the pointers to the preformatted set of references, and select the appropriate ones. The best of both worlds!

1.8 Fonts

An a priori point to realize is that authors only need awareness of fonts to pick the special symbols from in their detailed math mark-up. In structure commands it should be hidden: the author just asks for `\title`, `\head`, and the like where the used font is left to the style. This assumption entails that math authors should be aware of what symbols are available and how to access them. The AMSfonts user's guide is very good at this point in providing tables with the symbols and their control sequences. For fonts which are not standard loaded under amsppt the accessing of symbols is more down to earth via the use of `\mathchar`.

What is available is treated in the section AMSfonts. Here we deal with how to access the special alphabets and symbols, that is the loading and the mark-up for special symbols in the copy.

Loading. The problem is that only 16 font families are allowed, so we have to be careful with what to load.[22] To a lesser extent the fonts take memory and slow down proofing. For the latter a `\syntaxonly` option is available which does not format, and can save some 30% in speed.

Fonts loaded with amsppt automatically. For general use: cmcsc8, cmex8, cmex7 (similar to plain but in additional sizes)
Math fonts: msam, msbm, (extra symbols),[23] and eufm (medium-weight Euler Fraktur).

Special load commands.[24]
`\loadbold`: cmmib (bold math italic), cmbsy (bold math symbols).
For this class access macros are supplied as detailed below. For the following classes the accessing of symbols has to be done via `\mathchar`. The classes can be derived from the font names by the suffix `fam`.
`\loadeufb`: eufb (bold Fraktur)
`\loadeusm`: eusm (medium-weight script)
`\loadeusb`: eusb (bold script)
`\loadeurm`: eurm (medium-weight 'cursive roman')
`\loadeurb`: eurb (bold 'cursive roman').

[22] Note that seven are already in use before $\mathcal{AMS}$-TEX begins.
[23] $\mathcal{AMS}$-TEX provides `\newsymbol` to attach a name to the 4-positional specified character for msam, msbm, not much different from the use of `\mathchardef`, except for the error detection.
[24] Control sequences to be supplied in the preamble area.

Mark up in math part of copy

Bold and other symbols. Basically, one has to apply the basic TEXnique[25] to access a math character from a font via the use of `\mathchar`, similar to the use of the `\char` primitive. However, the difference is that for math characters one has to specify also for the 'class' and the family. Because of that complication $\mathcal{AMS}$-TEX provides the following for the bold cmmib, cmbsy, and eufb.

There are the control sequences, of which each takes one argument

- `\bold`, for bold letters (bold text, but in contrast with plain's `\bf` only for one symbol, the argument. Plain's `\bf` remains available)
- `\boldkey`, for symbols that actually appear on the keyboard (letters are in bold italic)
- `\boldsymbol`, for symbols specified by a single control sequence
- `\frak`, for a Fraktur letter
- `\Bbb`, for 'blackboard bold' (upper case only).[26]

An example mark-up reads

```
$\bold x \boldsymbol\in \boldsymbol
\varGamma$.
```

For the specification of the control sequences see the $\mathcal{AMS}$-TEX user's guide, or the file `amssym.def` and `amssym.tex`.

For accessing symbols from eufb, eusm, eurb, eurm, with classes `\eufbfam`, `\eusmfam`, `\eurbfam`, `\eurmfam`, we have to rely on `\mathchar` as shown in the following example for the letter 'a' (neglecting the catcode changes for @, in order to make the class a hexadecimal number)

```
\mathchar"0\eurmfam@69
```

with `\eurmfam@` the hexadecimal value of `\eurmfam`.

The user's guide is somewhat complicated, because it deals also with the situation when you are short of memory.

1.9 Math mark-up

I don't like the use of the heavy braces on among others the pages 103, 104, 106, 110, 111, 116, of Spivak, 1986.

Syntactic sugar. Much functionality of (plain) TEX has been renamed on the one hand, while on the other hand some functionality has been altered under the same name.

> Therefore, I consider the math mark-up syntactic sugar of plain's macros.[27]

This makes an $\mathcal{AMS}$-TEX script incompatible with plain.[28]

- Instead of plain's infix command `\over`, there is the prefix command `\frac`[29]
- `\align`, `\aligned`, `\alignat`, `\split`, `\multiline`, and `\gather` with their `\end...` endings, add to plain's `\eqalign`, `eqalignno`, and `\displaylines`
- `\tag` supersedes `\eqno`
- Instead of `\matrix`, and `\pmatrix`, the substitutes `\bmatrix`, `\vmatrix`, `\pmatrix`, `\Vmatrix`, and `\smallmatrix`[30]
- At the definition front there is `\define`, `\redefine`, and `\predefine` as substitutes for `\def` and `\let`. (No replacement of the TEXnical `\futurelet`, of course.)
- `\edef` etc. disappeared as such. In `\accentedsymbol` it is used from the application viewpoint. This holds for some other TEX macros (or control sequences) too: `\overfullrule=0pt`, `\cr`, `\openup`, `\noalign`, `\phantom`, `\atop` and the like, `\vbox{\hsize=...}`, `\cal`, `\dots`, `\oldstyle`, `\hoffset`, `\voffset`, `\vadjust`, and the abbreviations period.

Next, some examples of the same names but with altered functionality

- `\item` has been redefined within the `\roster` environment[31]
- different use of `\footnote`
- different use of `\proclaim`,[32] and
- different attitude with respect to font changes.

[25] TEXbook chapter 17.

[26] Spivak, 1986, also provided some poor man's bolds via `\pmb`.

[27] I agree, however, that `\text{...}` is more natural than the use of `\hbox{...}`, and that the whole business of extra fonts is really something, not to forget the `\widehat`, `\widetilde`, the multiple integrals and the triple dots of the Schwarz derivative, to name but a few.

[28] Agreed, the reader is warned for those occurrences in Appendix C of the Joy of TEX.

[29] I know that the philosophy behind it is to have opening and closing braces not too far apart. But that could have been attained via defining substructures, which is a good habit anyway.

[30] The powerful and practical `\bordermatrix` disappeared from the stage, I mean is not mentioned at all.

[31] I love plain's `\item`. Happily, Appendix C reassures me that it can still be used as such.

[32] In the mean time `\proclaim` and its variants have moved into `amsppt.sty`.

$\mathcal{AMS}$-LaTeX

If only the reader would now turn over to the excellent 'Guidelines for preparing electronic manuscripts—$\mathcal{AMS}$-LaTeX,' much would have been gained.

2 The $\mathcal{AMS}$-LaTeX package

Because $\mathcal{AMS}$-LaTeX is biased towards LaTeX, Spivak's 'The Joy of TeX' is not quite appropriate as manual. The user's guide has been designed as a self-contained booklet. Alas, the booklet is not sufficient. The accompanying 'Guidelines for preparing electronic manuscripts—$\mathcal{AMS}$-LaTeX,' is not only handy but necessary.

The $\mathcal{AMS}$-LaTeX package consists of[33]

- `amstex.sty`, an extensive modification of `amstex.tex` that allows it to be used in LaTeX as documentstyle option
- `amsart.sty`, `amsbook.sty`, to parallel LaTeX's article and book styles
- some options: `amscd.sty` (commutative diagrams), extra math style options, for example left or right placement of equation numbers, verbatim, theorem, ...
- `amsfonts.sty`, `amssymb.sty`, and the definition files `fontdef.ori`, `fontdef.max`, and `fontdef.ams`

along with the documentation[34]

- $\mathcal{AMS}$-LaTeX User's Guide
- $\mathcal{AMS}$-LaTeX Installation Guide
- Guidelines for preparing electronic manuscripts—$\mathcal{AMS}$-LaTeX

and some sample files

- `testbook.tex`
- `pref.tex`
- `chap1.tex`
- `chap2.tex`
- `app.tex`
- `testbook.bbl`.

In this part I'll treat

- `amsart.sty`
- some math mark-up details, and
- the use of fonts from the AMSfonts collection.
- syntax checking run mode can be selected via `\syntaxonly`.

2.1 Styles: amsart.sty

A document is thought to be marked up via the LaTeX structure[35]

`\documentstyle{amsart}`,
$\langle preamble \rangle$,
`\begin{document}`
$\langle top\ matter \rangle$
`\maketitle`
$\langle document\ proper \rangle$
`\end{document}`.

Strong points

- Inheritance of some LaTeX goodies: picture environment, the automatic and symbolic cross-referencing, automatic numbering schemes
- A very-well developed title part
- Access to the AMSfonts collection via incorporation of NFSS[36]
- A fancy bibliographic tool with an option to include BiBTeX files
- `\syntaxonly` option.

Not in there

- to supply for pre- or suffixes, easily, in the automatically maintained number schemes
- the Babel option as such (There are no hard-wired names. The language determined keywords are parameterized)
- no general policy with respect to $\backslash\langle tag \rangle \ldots$ `\end`$\langle tag \rangle$, as has been adopted in $\mathcal{AMS}$-TeX
- no special provision for setting up marginal notes or two-column format
- 11pt and 12pt options have been reduced to a minimal kernel
- changed function of star-ed forms of `\chapter`, `\section`, and the like
- numbers and punctuation in italic text is set in upright font.

Encoding. With respect to encoding, I will discuss

- the systematic use of LaTeX's optional parameter mechanism,
- some structuring control sequences: `\title`, `\chapter` (c.q. `\section` and `\appendix`)
- math mark-up, and
- how to supply bibliographic items.

Encoding: Optional parameter mechanism. LaTeX's optional parameter mechanism consists of providing after the keyword the optional parameter, enclosed by the brackets, [ ]. The use of the (optional) parameters are detailed with in the LaTeX reference manual and user's guide, LRM for short.

Encoding: `\title`. For the `\title` the optional parameter has the function to supply a short title to be used in the running heads.

[33] The style files are part of the corresponding `.doc` files.
[34] In booklets and also available in electronic form.
[35] For a more complete, and detailed, structure see the template in Appendix B: amsl-art.tem.
[36] The New Font Selection Scheme, as detailed in the user's guide, and in the publications of Mittelbach and Schöpf.

Encoding: `\chapter`, and non-English keywords The keywords needed in `\abstract`, `\chapter`, and the like, make use of `\⟨tag⟩name`, which contain respectively `Abstract`, `Chapter`, and can be redefined.[37]

2.2 Math mark-up

When I started to use TEX,[38] I marked up my math, tables, and diagrams via LATEX. Since then I started to use more and more plain TEX, culminating in Math into BLUes, and Table Diversions. From working on those projects, I concluded—as detailed in my Syntactic Sugar—that the variant mark-up of math and tables—in contrast with the positioning of the element within the context—is just syntactic sugar of what plain already provides, except for a macro or two for special cases! A matter of taste.

The Math facilities provided by LATEX are augmented with some from $\mathcal{AMS}$-TEX.

However, the lack of mentioning—in the LRM—the concept of formula classes is an oversight.

Encoding: Theorems and the like. In order to customize these structures, a `theorem.sty` option has been developed. It is advised to use this one instead of LATEX's `\newtheorem` environment. This option is based upon Mittelbach, 1989.

Three typographically different environments are considered: plain, definition, and remark. Via `\newtheorem` quite some environments have been created. I don't like that the descendants within a class share the same counter, although this can be undone.

2.3 References

There are two possibilities

- direct via the `\begin{thebibliography}{10}`... `\end{thebibliography}`, or
- after the use of the BiBTEX tool.

For more details, see the examples in the user's guide and the LRM.

An alternative to two-pass bibliography typesetting. It is curious that the placement of the bibliography items *in the manuscript* has to be at the end. This entails the forward referencing problem with its (costly) two-pass mechanism.[39] There is no good reason for not supplying the bibliography at the beginning of the compuscript, set it in 'boxes' (giving the formal labels their values), and typeset it at the end. The forward referencing problem is then no longer there. Active documents, aha!

A similar idea holds for the typesetting of table of contents. Why not typeset the toc and the end of the job with the page numbers set appropriately, I mean they usually take i, ii, ... After printing the toc pages can just be put in front. No two-pass job needed!

2.4 Fonts

Invariantly, we have to deal with the loading of the fonts to be used—special alphabets, like Fraktur, or extra math symbol tables—and the mark-up in the copy for the right character, or symbol.

Mittelbach and Schöpf state that typographical tradition has it, to characterize fonts by the attributes

> shape, series, size and family.

This is another naming scheme than used by DEK. They developed this approach as the NFSS. An author using NFSS, has to be aware of

- the attributes and their accompanying control sequences: `\shape`, `\series`, `\size`, `\family`[40]
- the font change control sequence is `\selectfont` (apply this after having specified the attributes)
- the fontdef concept, and the realization in fontdef files (essentially these are the 'tables' to associate the combination of the attributes to the available fonts)
- the various shorthand notations, which do some of the actions of the above under the same name a LATEX author is used to.[41]

It must not be forgotten, that no more fonts are accessed as the $\mathcal{AMS}$-TEX author can select already. Just the way of talking about it and the naming is different. The NFSS is essentially a mapping from the

> imaginary 4-dimensional space onto the 1-dimensional space

of available fonts. I'm still pondering about the role the virtual font concept can play herein. I mean what is the difference between a variant of Euler Fraktur and the one provided by another firm, not belonging to the CM-family? Both are brands of the same essentially. To quote DEK

> 'The idea behind VF files is that a general interface mechanism is needed to

[37] Curious though, because I guess AMS is not publishing in languages different from English. On the one hand the styles are targeted at AMS publications and on the other hand flexibility is strived after. This is a little contradictory. For an AMS author there should be just the styles to format the publication, or even better the generic preprint styles. With as simple documentation as possible, not too much dealing with side-track issues, however important these issues are in general.

[38] De Bruin et al., 1988.

[39] Forward referencing means that the reference to the item precedes the occurrence of the item in the copy, and if this is done via symbolic names then this name is not yet defined.

[40] Especially the family concept is totally different from DEK's.

[41] But, sometimes it works differently, so the author has to delve into the matter. It is not completely upwards compatible and therefore error-prone.

> switch between the myriad font layouts provided by different suppliers of typesetting equipment. Without such an equipment people must go to great lengths writing unscrutinable macros whenever they want to use typesetting conventions based on one font layout in connection with actual fonts that have another layout.'

At the lower level the same names are still there. Abstraction has its features, though.

I would have welcomed the classical font tables of DEK with the fontdef-s in a direct way correlated to them. It is true that from the fontdef files, it is clear what you can ask for in your mark-up and what you get. For example

```
\family{cmr}\shape{it}\series{m}
\size{14}{18pt}\selectfont
```

yields

```
cmti12 at14.4pt
```

which can be looked up in the font table.

Loading. $\mathcal{AMS}$-LaTeX provides the files `fontdef.ori`, `fontdef.max`, and `fontdef.ams`.[42] My wishful thinking is that the loading will be done via the inclusion of the fontdef file. Not so! Appendix A5 of the user's guide explains how to make a new format file via iniTeX. Not easy at all!

Mark-up of math copy. Basically one needs from the user's guide

- Table 3: Font commands used in text, and
- Table 4: Font commands used in math (these are the same as within $\mathcal{AMS}$-TeX: `\bold`, `\boldsymbol`, `\pmb`, `\cal`, `\frak`, and `\Bbb`. `\mathrm` is usually hidden in `\text`{...}.).

> For the moment, I expect that an AMS-LaTeX author will be deprived from the wealth provided by AMSfonts.

I like the TeXnical way the attributes have been parameterized.

2.5 Some LaTeX drawbacks

All those (semi-)automatic features provided for example by LaTeX are dangerous, and paradoxically error-prone. The $\mathcal{AMS}$-LaTeX user's guide for example, demonstrates how easy it is to forget for an unnumbered section in the table of contents.[43]

Another hindering thing is that counters with value 0 are ignored. If this is hidden in examples, without mentioning, I feel myself unhappy: LRM p.93, C.6 p.197.

AMSfonts

It is true that math mark-up requires some extra alphabets—for example Fraktur—and extra symbols, next to what is already provided by plain. The AMS did a great job in filling up this gap.

If only the reader would now turn over to the excellent user's guide AMSfonts, much would have been gained.

3 The AMSfonts package

The package consists of the

- various font files (see below)
- the user's and installation guide, also provided electronically.

3.1 Contents of the AMSFonts collection[44]

The AMSFonts collection contains the following fonts, in the sizes indicated:

- The Euler family, all but `EUEX` in 5, 6, 7, 8, 9, and 10 point:
  - Fraktur (German), medium-weight and bold (`EUFM` and `EUFB`)
  - "Roman" cursive, medium-weight and bold (`EURM` and `EURB`)
  - Script, medium-weight and bold (`EUSM` and `EUSB`)
  - Euler-compatible extension font (`EUEX`), in 7, 8, 9, and 10 point
- Additional sizes of some Computer Modern math fonts (the 10-point fonts are included in standard TeX distributions):
  - bold math italic (`CMMIB`), in 5, 6, 7, 8, and 9 point
  - bold math symbols (`CMBSY`), in 5, 6, 7, 8, and 9 point
  - math extension font (`CMEX`), in 7, 8, and 9 point (the 10-point font is included in standard TeX distributions)
- Extra math symbols, in 5, 6, 7, 8, 9, and 10 point:
  - first series, medium-weight (`MSAM`)
  - second series, including Blackboard Bold, medium-weight (`MSBM`)
- Cyrillic, developed at the University of Washington
  - lightface (`WNCYR`), in 5, 6, 7, 8, 9, and 10 point
  - bold (`WNCYB`), in 5, 6, 7, 8, 9, and 10 point
  - italic (`WNCYI`), in 5, 6, 7, 8, 9, and 10 point
  - caps and small caps (`WNCYSC`), in 10 point
  - sans serif (`WNCYSS`), in 8, 9, and 10 point

[42] It is advised to customize form these files your own fontdef file.

[43] The reference section is missing, while it is not at the end of the booklet, and therefore it should be listed in the table of contents.

[44] Borrowed from the AMSfonts user's guide.

- Computer Modern caps and small caps (`CMCSC`), in 8 and 9 point (the 10-point font is included in standard TEX distributions)
- The "dummy font," used in $\mathcal{AMS}$-TEX for syntax checking, exists only as metrics (`dummy.tfm`)
- Other files needed to use these fonts:
  - `amssym.tex`, a file defining the symbols in fonts `MSAM` and `MSBM`
  - `amssym.def`, a file that loads the fonts `MSAM`, `MSBM` and `EUFM` and defines some control sequences required by `amssym.tex`
  - `cyracc.def`, a file containing definitions needed for proper access to characters in the cyrillic fonts
- Other useful files:
  - `userdoc.tex`, the source file for the AMSfonts user's guide
  - `userdoc.cyr`, the source file for the table showing cyrillic input conventions, input by `userdoc.tex`
  - `userdoc.fnt`, the source file for the tables of the principal 10-point fonts in the AMSFonts collection, input by `userdoc.tex`; this file may also be TEXed by itself
  - `userdoc.def`, the macros used to format the AMSfonts user's guide
  - `userdoc.ins`, the source file for the appendixes to the AMSfonts user's guide, input by `userdoc.tex`; this file may also be TEXed by itself.

Each font at a particular size is provided in seven standard TEX magnifications, magsteps 0 through 5, including magstephalf. The AMSFonts package for the IBM PC and compatibles includes all magnifications. For use with *Textures* on the Macintosh, the Standard AMSfonts package includes only magsteps 0 and 1; the Extended AMSFonts package includes all seven magnifications. All instances of every font have been newly generated for this release of the AMSFonts collection.

Epilogue

The most difficult thing is to know when to stop, to keep the right balance, not to sub-optimize. The American Mathematical Society is doing a tremendous job, for sure. However, I believe that

> with a little less automation nearly the same results can be attained, . . . with less energy.[45]

Perhaps all the energy spent is the price the AMS is willing to pay for

> the benefit of their members, and the Math, TEX, and publishing scientific community at large,

in name of. . . progress.

TEXniques used

- Mark up of top matter in the SGML spirit
- Minimal mark-up style for references
- `\csname...` for using `\outer` control sequences non-outer
- Near generic encoding of `\proclaim`, `\proof`, `\demo`, `\example`, `\remark`, and the like
- General optional parameter TEXniques, to override the default formatting
- An environment stack to keep track of the environment the formatting process is in
- Font selection via the NFSS
- `\add@missing`, a general error detecting (and correcting) mechanism.

Conclusion

$\mathcal{AMS}$-TEX together with the AMSfonts collection is a very good, and rich, extension of TEX for math authors, although I consider Spivak's math mark-up commands mainly as syntactic sugar of what has already been provided by plain.

Because $\mathcal{AMS}$-TEX facilities have been dropped in favor of facilities provided by LATEX, I find myself in good company with the conclusion that the math mark-up via $\mathcal{AMS}$-TEX, or via LATEX, are both syntactic sugar of what plain already provides.

It is confusing that the `amsppt.sty` and some of the other styles have been mixed in the booklet (and in the code).

$\mathcal{AMS}$-LATEX's user's guide is confusing in introducing a new font selection scheme—which is not more powerful—but makes perhaps the *talking* about it easier. It is symptomatic that the source file of the user's guide does not run with the NFSS.

> For the moment, I expect that an AMS-LATEX author will be deprived from the wealth provided by AMSfonts.

General. As can be seen from the Appendices B and C the mark-up via `amsppt.sty`, looks quite different from the mark up via `amsart.sty`. Furthermore, the results in print differ, as can be seen from the samples provided in the Guidelines for preparing electronic manuscripts. This is due to the different tools: TEX vs. LATEX, JoT vs. LATEX math, `\Refs` vs. LATEX's bibliography environment or BiBTEX, DEK's font selection and switching vs. NFSS.[46]

[45] An example of the famous 80%-20% results-investments rule, better known as the law the reduced gains.

[46] Not so much a problem for a specific TEX, respectively LATEX, author, but it complicates and makes the maintenance costly. I myself use (LA)TEX whenever suitable, and then it is confusing.

> I would welcome `amsppt.sty`, and eventually a `amsart.sty`! as *the* generic author interfaces for a TEX, respectively LATEX-oriented author.

Also handy are templates to start from, next to the worked out samples, as supplied in the Guidelines ... But should not they be part of the user's guides?

Acknowledgements

The American Mathematical Society, especially Regina Girouard, Ralph Youngen, and Kevin Curnow, are kindly acknowledged for providing the information, the booklets and the files.

References

[1] AMS (1991a): A look inside the AMS. (A nice brochure of what AMS is all about.)

[2] AMS (1991b): Think about publishing with the AMS. (Another nice brochure about the merits of publishing with AMS: effective marketing, extensive promotion, world-wide distribution, better sales, longer life of book, royalties, support worthwhile non-commercial activities for the benefit of the scientific community at large, for example the TEX project.)

[3] AMS (1993a): $\mathcal{AMS}$-TEX User's Guide 2.1.

[4] AMS (1993b): $\mathcal{AMS}$-TEX Installation Guide 2.1.

[5] AMS (1993c): $\mathcal{AMS}$-LATEX User's Guide 1.1.

[6] AMS (1993d): $\mathcal{AMS}$-LATEX Installation Guide 1.1.

[7] AMS (1993e): AMSfonts. User's Guide 2.1.

[8] AMS (1993f): AMSfonts. Installation Guide 2.1.

[9] AMS (1993g): Guidelines for preparing electronic manuscripts.
$\mathcal{AMS}$-TEX (booklet, 52p), and the mirrored one $\mathcal{AMS}$-LATEX (booklet, 58p).
(The first is very well-done. I have not seen a guideline of similar quality of yet! Simply the best available. Much experience is embodied to learn from. The second is verbose, incomplete, and deals at length with issues an author should not be bothered with.)

[10] Braams, J.L (1991): Babel, a multilingual style-option system for use with LaTeX's standard document styles. *TUGboat* 12, no. (2), 291–301.

[11] Bruin, R de, C.G van der Laan, J.R Luyten, H.F Vogt (1988): Publiceren met LATEX. CWI Syllabus 19. (Dutch. Useful additional remarks have been published in MAPS92.1.)

[12] Knuth, D.E (1984): The TEXbook. Addison-Wesley.

[13] Knuth, D.E (1990): Virtual Fonts: More Fun for Grand Wizards. *TUGboat* 11, no. (1), 13–23.

[14] Laan, C.G van der (1990): SGML(,TEX and ...). Proceedings EuroTEX '90. *TUGboat* 12, no. (1), 90–104. (Has also appeared in GUTenberg Cahiers 5, and MAPS90.2.)

[15] Laan, C.G van der (1991): Math into BLUes.
Part I: Mourning. Proceedings TUG '91, *TUGboat* 12, no. (4), 485–501.
Part II: Sing your song. Proceedings EuroTEX '91, GUTenberg Cahiers, 10&11, 147–170. (An early version has appeared in MAPS91.1.)

[16] Laan, C.G van der (1992a): Syntactic Sugar. MAPS92.2, 130–136. (To appear GUST Bulletin. Submitted TUG '93.)

[17] Laan, C.G van der (1992b): FIFO & LIFO sing the BLUes. MAPS92.2, 139–144. (To appear *TUGboat* 14, no. (1).)

[18] Laan, C.G van der (1992c): Spivak's Œuvre. MAPS92.1. 139–142.

[19] Laan, C.G van der (1992d): Table Diversions. Proceedings EuroTEX '92, 191–211. (Also in MAPS92.2.)

[20] Laan, C.G van der (1993): Sorting in BLUe. MAPS93.1. 21p. (Submitted TUG '93.)

[21] Lamport, L (1986): LATEX, user's guide & reference manual. Addison-Wesley.

[22] Mittelbach, F (1989): An extension of the LATEX theorem environment. *TUGboat* 10, no. (3), 416–426.

[23] Mittelbach, F, R Schöpf (1989a): A new font selection scheme for TEX macro packages—the basic macros. *TUGboat* 10, no. (2), 222–238.

[24] Mittelbach, F, R Schöpf (1989b): The new font family selection—user interface to standard LATEX. *TUGboat* 11, no. (2), 297–305.

[25] Schöpf, R (1989): A new implementation of the LATEX `verbatim` and `verbatim*` environments. *TUGboat* 11, no. (2), 284–296.

[26] Spivak, M.D (1986): The Joy of TEX—$\mathcal{AMS}$-TEX. AMS. ISBN 0-8218-2999-8. (Updated 2^{nd} printing 1990.)

[27] Spivak, M.D (1989): LA$\mathcal{MS}$-TEX—The Synthesis. TEXplorators.

[28] Youngen, R (1989): Computers and Mathematics. Notices AMS. (Discusses TEX, LATEX and $\mathcal{AMS}$-TEX, summarizing also the relative advantages. Since the publication of this note $\mathcal{AMS}$-LATEX has been released, and Spivak has provided LA$\mathcal{MS}$-TEX. Of course these are not dealt with.)

[29] Youngen, R (1991): Typesetting with TEX at the AMS. 4p. (A nice survey of why-and-what AMS is using TEX for.)

[30] Youngen, R (1992): TEX-based production at AMS. MAPS92.2, 63–68.

Appendix A: How to get it?

NTG is discussing with AMS the possibility of NTG being a redistributor of AMS TEX formatting tools ($\mathcal{AMS}$-TEX, $\mathcal{AMS}$-LATEX, styles, and corresponding documentation), off-the-shelf.

The files are in the public domain and can be obtained from the AMS—as detailed in their note below—and also from the file servers, for example Piet van Oostrum's server at RUU.

AMERICAN MATHEMATICAL SOCIETY
P.O. Box 6248, Providence, RI 02940[47]

This is a guide for accessing TEX macro packages and AMSFonts from the Society's public domain archive on the Internet node `e-MATH.ams.org`.

If you wish to obtain an entire package from the archive, such as the entire $\mathcal{AMS}$-TEX or $\mathcal{AMS}$-LATEX package, you should create a directory structure on your local machine which parallels the directories from which you get the files on the host. In this way you will be able to transfer the entire contents of a directory to your local machine, and you will best be able to use the instructions for installing the package on your local machine.

The current it directory tree structure on the host is (Note added: not the files)

```
ams (50.6 MB total)
  amsfonts (21 MB total)
       doc (115K)
       pk-files (19 MB total)
          118dpi (2.1 MB)
          180dpi (2.9 MB)
          240dpi (3.4 MB)
          300dpi (4.5 MB)
          400dpi (6.1 MB)
       sources (2 MB total)
          cyrillic (359K)
          euler (1.3 MB)
          extracm (98K)
          symbols (282K)
  amslatex (1.5 MB total)
       doc (518K)
       fontsel (237K)
       inputs (330K)
       latex (347K)
  amstex (410K total)
       doc (245K)
       author-info (1.2 MB total)
          guidelines (261K)
          sty-files (961K)
       macintosh (3.8 MB total)
       tfm-files (150K)
```

There is a file TeXfiles on the /ams directory that contains a list of ALL the files on the directory, with size and last-change date that is suitable for downloading.

The host is a Unix environment. File and directory names are *case-sensitive.* (Note that directory names are lower case.) The basic FTP commands you will need to know are listed below.[48]

```
cd (directory name)
 connect to a directory on the remote host
cd ..
 connect to the parent of current directory
 on the host
lcd (directory name)
 locally connect to a directory on your
 computer
pwd
 ask for name of current directory on the
 host
lpwd
 ask for name of current directory on your
 computer
dir
 list contents of current directory on the
 host
ldir
 list contents of current directory on your
 computer
get (file name)
 get a single file from the host
mget (file specifications)
 get multiple files from the host
binary
 when you are about to transfer binary
 files, such as tfm files or pk files
ascii
 when you are about to transfer text files
 (most of the files on the host are text
  files)
exit
 leave FTP and return to your system
```

A1. How to log on

First, type

```
ftp e-math.ams.org   <return>
      or
ftp 130.44.1.100     <return>
```

When you see a message indicating that a connection has been opened, you need to log in using the username `anonymous`. If your version of FTP is now prompting you for a Name or Username, simply type `anonymous` (no quotes) and ⟨*return*⟩. Otherwise, type `login anonymous` and hit ⟨*return*⟩.

You will be asked to enter a password; enter your name here and press ⟨*return*⟩.

A2. Which directories do you need

(Users of TEXtures on a Macintosh will find everything they need in the directory /ams/macintosh. As the READ.ME file in that directory explains, files stored here are BinHex'ed StuffIt archives of entire distributions. These are all ascii files. If you are a TEXtures user, you may skip ahead to section 3. below.)

Whether you are going to use $\mathcal{AMS}$-TEX, AMSFonts, $\mathcal{AMS}$-LATEX, or any combination of the three, you will need to have the TFM files for AMSFonts 2.1. To get them, retrieve all the files in the directory /ams/tfm-files using the instructions in Section 3. (These are binary files.)

If you are going to use $\mathcal{AMS}$-TEX 2.1, you need to retrieve all of the files in the directories

[47] JULY 1990 [updated Dec. 22, 1992; NGB]

[48] Some implementations of FTP function slightly differently. For example, you may have to use 'quit' instead of 'exit'. If you have any problems, contact the support people at your site.

`/ams/amstex`

and

`/ams/amstex/doc`

(These are all ascii files.)

If you are going to use $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-LaTeX 1.1, you need to retrieve all of the files in the directories

```
/ams/amslatex
/ams/amslatex/doc
/ams/amslatex/fontsel
/ams/amslatex/inputs
```

(These are all ascii files.)

If you are going to use AMSFonts 2.1, you need to retrieve all of the files in the directories

```
/ams/amsfonts
/ams/amsfonts/doc
```

(All of these are ascii files.) In addition:

- If you are using a 300dpi printer, you need the files from the directory /ams/amsfonts/pk-files/300dpi. (These are binary files.)
- If you are using a screen previewer which uses 118dpi resolution, you need the files from the directory /ams/amsfonts/pk-files/118dpi. (These are binary files.)
- If you have an implementation of Metafont and are able to create your own raster images of fonts from Metafont source files, you may wish to retrieve the files from the directory /ams/amsfonts/sources and some or all of its subdirectories. (These are ascii files.)

If you want to retrieve the Guidelines for Preparing Electronic Manuscripts, you need to retrieve the files in the directories[49]

```
/ams/author-info
/ams/author-info/guidelines .
```

(All of these are ascii files.)

A3. How to get the files from a directory

In the instructions below, when you are directed to retrieve the files from a directory or subdirectory on the host, perform each of the following steps

1. Connect to the directory on the host, using the `cd` command (You can do it in one command, e.g.: `cd /ams/amsfonts/pk-files/300dpi`; or a series of commands, e.g.: if you are already connected to the directory /ams/amsfonts, enter `cd pk-files` followed by `cd 300dpi`)
2. Enter `pwd` to make sure that step (1) was successful.
3. Enter `dir` to see how many files are in the directory. (optional)
4. Connect to the directory on your computer where the files will go, using the `lcd` command.[50]
5. Enter `lpwd` to make sure that step (3) was successful.
6. Enter either `ascii` (for text files) or `binary` (for tfm or pk files).
7. Enter `mget *.*`
8. Enter `ldir` to be sure that all the files were copied. (optional)

A4. How to end the ftp session and log off

To exit FTP simply type `exit` and $\langle return \rangle$.

A5. What to do on your own system

BEFORE attempting to install any of these macro or font packages, READ the appropriate READ.ME files which were copied to your system when you typed `mget *.*`.

Any questions concerning the TeX-ware products can be sent to the Internet address

`Tech-Support@math.ams.org.`

Reports of problems in accessing the FTP node itself should be sent to the Internet address

`support@e-math.ams.org`

Appendix B: amst-art.tem

For the complete sample, see the Guidelines for preparing electronic manuscripts—$\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TeX.

```
%Stripped and modified version of:
% AMS-TeX 2.0+ file for a sample article
% for electronic submission.
%by C.G. van der Laan,
%in order to stress the main structure,
%and to provide more or less a template.
%
\input amstex
\documentstyle{amsppt}
%...
\topmatter
\title Sample \AmSTeX{}et cetera\endtitle
\rightheadtext{SAMPLE \AmSTeX{} ...}
\author BLUe\endauthor
\address Department of ...\endaddress
\email BLUe\@rug.nl\endemail
\keywords AMSppt.sty,...\endkeywords
\subjclass Primary 54C40, 14E20;
 Secondary 46E25, 20C20\endsubjclass
\abstract This paper is ...\endabstract
\thanks The author ...\endgraf\endthanks
\endtopmatter
%
\document
%
\head 1. Introduction%bold, centered;
\endhead%don't type final punctuation
This sample paper illustrates ...

\subhead Top matter\endsubhead
The input format ...

\head 2. Theorems, ...\endhead
%
Theorems and lemmas are varieties of
```

[49] Note added: To get the sample article from the guidelines directory, say `cd /ams/author-info/guidelines` followed by `get amst-art.tex`, respectively `get amsl-art.tex`.

[50] The format in which you type the directory name should be the normal format in which you type it on your operating system.

```
\proclaim{Lemma 1} Let $f, g\in  A(X)$...
\roster
\item"(a)" If $f$ is $E$-regular...
\item"(b)" If $f$ is $E$-regular...
\item"(c)" If $f(x)\ge c>0$ for all...
\endroster
\endproclaim

\demo{Proof}
\roster\runinitem "(a)" Obvious.
\item"(b)" Let $h, k\in A(X)$ ...
\endroster
\enddemo

\definition{Definition}
For $f\in A(X)$, we define
$$...$$
\enddefinition

\head 5. Figures\endhead
Figures are handled as inserts,...

\example{Example 5}
For the link in Figure 5a,...
\endexample
% art work measures 11.5pc for figure 5a
\topinsert\vskip 11.5pc
\botcaption{Figure 5{\rm a}}\endcaption
\endinsert

\Refs%\nofrills{Your heading}
\ref\no 1
\by     V. L. Arnol\cprime{}d,...
\book   Singularities of ... . \rom{I}
\publ   ``Nauka''
\publaddr Moscow \yr1982 \lang Russian
\transl English transl.
\publ Birkh\"auser\publaddr Basel\yr1985
\endref
\ref\no 2
\bysame
...
\endref
\endRefs
\enddocument
```

Appendix C: amsl-art.tem

For the complete sample, see the Guidelines for preparing electronic manuscripts—$\mathcal{AMS}$-LaTeX.

```
%Stripped and modified version of
% AMS-LaTeX 1.1 file for a sample
% article for electronic submission.
%by C.G. van der Laan,
%in order to stress the main structure,
%and to provide a template.
\documentstyle{amsart}
\makeatletter
\def\LaTeX{\leavevmode L\raise.42ex%
 \hbox{\kern-.3em\size{\sf@size}{0pt}%
 \selectfont A}\kern-.15em\TeX}
\makeatother
\newcommand{\AMSLaTeX}{\protect
 \AmS-\protect\LaTeX}
\newcommand{\BibTeX}{{\rm B\kern-.05em{\sc
 i\kern-.025emb}\kern-.08em\TeX}}
\newtheorem{thm}{Theorem}[subsection]
\newtheorem{lem}[thm]{Lemma}

\theoremstyle{definition}
\newtheorem{defn}{Definition}[subsection]
\newtheorem{exmp}{Example}[subsection]
\newcommand{\subfig}{a}
\makeatletter
\def\alphenumi{%
  \def\theenumi{\alph{enumi}}%
  \def\p@enumi{\theenumi}%
  \def\labelenumi{(\@alph\c@enumi)}}
\makeatother

\begin{document}

\title[Sample \AMSLaTeX...]%
 {Sample \AMSLaTeX{}...}
\author{BLUe}
\address{Department of...}
\email{BLUe@@rug.nl}
\thanks{The author ...}

\keywords{amsart.sty, amstex.sty,...}
\subjclass{54C40, 14E20;
 Secondary 46E25, 20C20}

\maketitle

\begin{abstract}This paper...
\end{abstract}

\section{Introduction}
This sample paper...

\subsection{Top matter}
The input format ...

\section{Theorems,...}
Theorems and lemmas and similar...

\begin{lem}
Let $f, g\in  A(X)$...
\begin{description}
\item[(a)] If $f$ is $E$-regular...
\item[(b)] If $f$ is $E$-regular...
\end{description}
\end{lem}

\begin{pf}
\begin{enumerate}
\alphenumi % Defined in the preamble.
\item Obvious.
\item Let $h, k\in A(X)$...
\qed
\end{enumerate}
\renewcommand{\qed}{}
\end{pf}

\begin{defn}
For $f\in A(X)$,...
\begin{equation}...\end{equation}
\end{defn}

\section{Figures}
Figures are handled as inserts,...
\begin{exmp}...\end{exmp}
% art work measures 14.5pc for figure a
\begin{figure}[t]
  \makeatletter\makeatother
  \renewcommand{\thefigure}%Look at this!
    {\arabic{figure}\rom{\subfig}}
  \vspace{14.5pc}
  \caption{}  % figure a
  \label{f:afig}
\end{figure}
% When BibTeX is used... (see guidelines)
\makeatletter \renewcommand{\@biblabel}%
 [1]{\hfill#1.}\makeatother
\begin{thebibliography}{10}
```

```
\bibitem{AVG1}
 V.~L. Arnol$'$d,...
 {\em Singularities of...\rom{I}},
 ''Nauka'', Moscow, 1982 (Russian), English
   transl., Birkh\"auser, Basel, 1985.
\bibitem{AVG2}
 \bysame,
 {\em Singularities of...},
...
\end{thebibliography}
\end{document}
```

Appendix D: Contents

The 14th Annual TEX Users Group Meeting

A World-Wide Window on TEX

Aston University, Birmingham, UK

July 26th – 30th, 1993

Sebastian Rahtz

spqr@minster.york.ac.uk

1 First announcement

The TEX Users Group is pleased to announce its fourteenth annual meeting, entitled *A World-Wide Window on TEX*, to be held at Aston University (Birmingham, U.K.), between July 26th – 30th, 1993. This meeting is the first annual TUG meeting to be held outside the North American continent, and the TUG Board and Conference Committee extend a special invitation to TEX users in Europe and world-wide who may previously have been unable to attend a TUG annual conference. If you would like to attend the Conference, and/or any of the courses which will be run during the week preceding and the week following the Conference proper, please complete the accompanying form and return it as soon as possible.

Payment may be made by cheque or by company order, but a surcharge will be levied on payment by company order; payment by credit card (Mastercard, Visa, and affiliated cards) and debit card (Delta, Switch) can also be accepted. Cheques should *either* be drawn in £ sterling and made payable to TUG'93, *or* drawn in US dollars and made payable to TEX Users Group; those wishing to pay by credit or debit card are requested to provide their card details. The prices are quoted in these notes in £ sterling, but the booking form[1] also lists the appropriate US dollar rates. All prices inclusive of tax.

The conference is a formal TUG meeting, and the prices quoted therefore give full discount to TUG members (£ 100.00). Those who are paid-up members of other TEX user groups benefit from a 50% discount (£ 120.00); if they wish to also take out TUG membership at this time, the additional cost is £ 20.00, *but this is optional*. Those who belong to no user group pay full price (£ 140.00), and also become full TUG members for the current year. Full-time students pay 50% of the TUG member rate, but do not become members. Applications received after May 10th will be subject to a surcharge of £ 15.00.

Various combinations of booking options are available, and whilst it is hoped that the accompanying form is reasonably straightforward, some explanation may be in order.

The conference fee is obligatory for all participants other than (i) accompanying persons and (ii) those solely interested in attending one or more courses.

Accommodation options include: *none* (for local residents and those wishing to make their own accommodation arrangements; as there are no family rooms available in the standard accommodation options, family parties may wish to take this option); *basic* (which is bed and breakfast in student residences) at £ 16.00 per night; *business* (which is bed and breakfast in the Aston Business School) at £ 50.00 per night; and *executive* (as business, but with *en suite* facilities, telephone and television) at £ 70.00 per night. Although we are unable to offer assistance to those wishing to make their own reservations with local hotels, a special arrangement has been made with the nearby Royal Angus Thistle Hotel whereby family rooms may be booked at £ 29.80 per night during weekends and £ 53.14 per night during the week; to qualify for these rates, it is *essential* to mention both the TUG'93 conference and the name of either Peter Abbott or Maureen Campbell; the hotel's telephone number is +44 21 236 4211.

Meals include the conference banquet, lunches, afternoon tea and morning coffee. Apart from the banquet, no dinners are offered, to allow participants the opportunity to sample some of Birmingham's many splendid restaurants, including the inexpensive ethnic establishments for which the City and its environs are justly famous. The conference banquet is £ 22.00, and coffee/tea/lunch is charged at £ 8.00 per day.

Courses are offered during the week preceding and the week following the conference proper, and are charged at a fixed rate depending upon the duration (and therefore regardless of level); the charges are: 1 day: £ 75.00; 2 days: £ 140.00; 3 days: £ 195.00; 4 days: £ 240.00; and 5 days: £ 275.00. These prices include lunch and refreshments for the duration of the course, and also include a TEX or LATEX manual for the intensive-introductory courses (students are advised to bring their own manuals for the other courses).

Social functions include the reception; the conference

[1] Ask NTG secretary for booking form.

banquet; and a cultural event for those wishing to combine TEX with the arts; this last will include a coach trip to Stratford and seats at a performance of *King Lear*, and will be charged at £ 40.00. Other social events may be organised between the publication of this announcement and the start of the conference, and these will be clearly notified to participants in their 'welcome pack', which will also include a guide to Aston University, Birmingham, the Black Country and the Midlands.

Finally, there is one feature of the accompanying form which the organisers hope will attract widespread support: travellers will be aware how biased the exchange rate is in favour of Western countries; because of this disparity, participants from Central and Eastern Europe and from other financially disadvantaged countries are expected to experience considerable difficulty in raising sufficient funds to enable them to attend TUG'93. The organisers ask those from more financially advantageous backgrounds to make a donation to assist others who, for financial reasons, might otherwise be unable to attend; the organisers undertake to distribute all monies so raised among deserving applicants, and to carry forward any unspent monies to boost a similar fund for next year's and future conferences; many national and language-based TEX users groups are also being approached to contribute to this fund. There is, of course, provision on the accompanying form for individuals to make application for a bursary from this fund, as well as to contribute: a separate bursary application form will be sent to those seeking sponsorship under this scheme.

TUG'93 Conference Office
Information Systems,
Aston University,
Aston Triangle,
Birmingham B4 7ET, UK
fax: +44 21 359 6158
phone: +44 21 359 5492
email: `tug93-enquiries@vax.rhbnc.ac.uk`

2 Second announcement

The committee of the TUG '93 conference would like to bring you up to date with plans for this summer's big TEX meeting at Aston University, Birmingham, UK. Booking forms were mailed to all TUG members in mid March, and are now being sent to other user groups. Readers with Internet access who have not received a form can fetch the file 'booking.tex' using ftp from directory `pub/tug93` on `ftp.TeX.ac.uk`, or may request it by email from `tug93-enquiries@vax.rhbnc.ac.uk`.

The five magnificent days of TEX conference will include:

- A first day which includes an introduction TEX to newcomers; bring your partner and get them to learn your obsession! Plus tutorials and special sessions for experienced users.
- A day on typography and design, in conjunction with the DIDOT project.
- A detailed workshop on LATEX3.
- A session on language and font issues.
- A session on TEX standards and the future developments of TEX and related software.
- Discussions of TEX archives.
- A 'problem-solving' panel.
- Birds-of-a-feather groups.

Invited speakers include Christina Thiele, the TUG president, the President of the German TEX User Group, DANTE, and Boguslav Jackowski, the winner of the 'best paper' award at recent conferences in Prague and Karlsruhe.

Some highlights from the accepted papers by speakers from around the world are:

- The Khmer Script tamed by the Lion (of TEX): Yannis Haralambous
- Virtual fonts in a production environment: Michael Doob & Craig Platt
- Readibility of math typesetting: David Murphy
- Russian TEX issues; Looking about and outlook: Irina A. Makhovaya
- Beginners Guide to DSSSL: Martin Bryan
- A PostScript font installer written in TEX: Alan Jeffrey
- A versatile TEX device driver: Minato Kawaguti
- Typesetting Catalan TEXts with TEX: Gabriel Valiente Feruglio
- Bibliography Prettyprinting and Syntax Checking: Nelson Beebe
- Syntactic Sugar: Kees van der Laan

There is a full week of varied TEX courses both before and after the conference proper, of which full details can be found on the booking form.

Socially, we can promise delegates that there will be no dull moments at TUG '93, with

- A half-day trip to Shakespeare country, ending with a performance of King Lear by the Royal Shakespeare Company
- Evening trips to visit a Chocolate Experience or take part in a bowling tournament
- A TEX-related competition with prizes
- Receptions and banquets
- The TEX Users Group annual general meeting

Booking forms returned by May 15th can save 15 pounds ($22.00), and guarantee a place on the trip to Stratford-upon-Avon.

The first draft programme, with names and titles of all speakers, will be issued on May 15th, and will be circulated widely on networks and by electronic mail.

3 Courses

1a20–24 July	5 days	Michael Doob	**TEX for PC users**
1b20–24 July	5 days	To be announced	**TEX for Macintosh users** Prerequisites: knowledge of use of Macintosh, incl editors. Students will get a copy of The TEXBook (or similar) to keep. TEX is the same whatever computer it is run on, but those who prefer to use Macintoshes will choose 1b, while those who are happier with PCs will no doubt select 1a.
2a2–5 August	4 days	To be announced	**LATEX for PC users**
2b31 July–3 August	4 days	Malcolm Clark	**LATEX for Macintosh users** Prerequisites: none. Students will get a copy of LATEX book (or similar) to keep. LATEX is the same whatever computer it is run on, but those who prefer to use Macintoshes will choose 2b, while those who are happier with PCs will no doubt select 2a.
320–24 July	5 days	Chris Rowley	**More TEX—Macros** Prerequisites: knowledge of TEX as covered by intensive TEX Students to bring their own copy of TEXBook.
42–4 August	3 days	Philip Taylor	**More TEX—output routines** Prerequisites: knowledge of TEX as covered by intensive TEX and TEX Macros. Students to bring their own copy of TEX Book.
531 July–1 August	2 days	Sue Brooks	**LATEX 2.09 Style files** How to write your own LATEX style files and modify those written by others. Prerequisites: knowledge of LATEX as covered by intensive LATEX and some knowledge of TEX. Students to bring both TEX and LATEX books.
621–23 July	3 days	To be announced	METAFONT**—logos**
72–6 August	5 days	Yannis Haralambous	METAFONT**—fonts** Prerequisites: none
831 July	1 day	Rosemary Bailey	**Using the TEX family for setting maths (including use of AMS-LATEX)** Prerequisites: knowledge of LATEX; reasonable knowledge of producing LATEX documents and an understanding of mathematical typography conventions.
922–23 July	2 days	Philip Taylor	**Book design in TEX**
1024–25 July	2 days	Goossens/Mittelbach	**Doing more with LATEX 2.09** Participants will learn about important advanced features not normally covered in books about LATEX, such as defining and changing commands, environments, counters, lengths, arithmetic calculations, and the structure of the basic formatting tools. Also covered will be the New Font Selection Scheme, and questions of layout modifications.
1122–23 July	2 days	Yannis Haralambous	**Beyond Computer Modern — using other fonts in TEX** How TEX handles fonts; virtual fonts; actual fonts other than CM; font selection schemes. Prerequisites: knowledge of TEX and some understanding of fonts.
1222–23 July	2 days	Sebastian Rahtz	**TEX and PostScript** How to make the most of your PostScript printer when using TEX; including the use of PostScript fonts, the inclusion of PostScript graphics files; achieving special effects with PostScript 'specials'; and drawing pictures in TEX with PostScript commands. Prerequisites: knowledge of TEX, basic PostScript.

Please note that any course which fails to attract a minimum number of students will be cancelled. In such case, if students cannot be enrolled for an acceptable alternative course, their fee will be refunded in full.

4 Third announcement

Conference themes

- **Multilingual TEX:**
  In addition to many reports on the spreading use of TEX to typeset an ever-increasing variety of languages and scripts, there will be important announcements from TUG's technical working group.
- **TeX Integration:**
  Where does TEX sit in the world of ever-increasing integration of software? windowing systems, document-oriented systems, document interchange and processing standards, active documents, multimedia. . .
- **TEX Futures:**
  Whither TeX?; TEX–XET-TEX, VTEX, the NTS project, . . . , what next?
- **TEX Archives:**
  The Comprehensive TEX Archive Network (CTAN); what it is and what can it do for you?

Conference papers (provisional titles)

- Graham Asher: *Tension*
- Nelson Beebe: *Bibliography Prettyprinting and Syntax Checking*
- Martin Bryan: *Beginner's Guide to DSSSL*
- Michael Doob & Craig Platt: *Virtual fonts in a production environment*
- Gabriel Valiente Feruglio: *Typesetting Catalan Texts with TEX*
- Jonathon Fine: *Fundamental TEX macros for processing structured documents*
- Peter Flynn: *Mixing TEX and SGML: a recipe for disaster?*
- George Greenwade: *The comprehensive TEX archive network – CTAN*
- Yannis Haralambous: *The Khmer Script tamed by the Lion (of TEX)*
- Berthold & Blenda Horn: *Scalable outline fonts*
- Alan Jeffrey: *A PostScript font installer written in TEX*
- Minato Kawaguti: *A versatile TEX device driver*
- Kees van der Laan: *Syntactic Sugar*
- Kees van der Laan: *Sorting in BLUe*
- Michel Lavaud: *Future TEX again*
- Irina A Makhovaya: *Russian TEX issues: Looking about and outlook*
- David Murphy: *Readability of mathematical typesetting*
- John Plaice: *Language-dependent ligatures*
- David Rhead: *Some suggestions on how LATEX3 might approach biblographic references*
- Larry Siebenmann: *A format compiLaTion framework for European languages*
- Larry Siebenmann: *The spacing around mathematics: a quick trip to the limits of TEX*
- Daniel Taupin: *Maps in Metafont*
- Philip Taylor: *A future to TEX*
- Christina Thiele: *The future of TEX and TUG*
- Michael Vulis: *Building a future for TEX*
- Xinxin Wang & Derick Wood: *An Abstract Model for Tables*
- Eddy Zedlewski: *A beginner's guide to the Internet*

Events

- Panel discussions: many of these subjects will be covered by 'question times' with a galaxy of distinguished panelists; contact the organisers if you want to make short contributions:
  - Futures (chair: Philip Taylor)
  - LATEX3 (chair: Rainer Shoepf)
  - Multilingualism (chair: Yannis Haralambous)
  - Archives (chair: George Greenwade).
- Tutorials (in parallel on the morning of Monday 26th, each approximately 1 hour long):
  - Introduction to LATEX: what it is and what it is not, Flavours
  - of TEX: a brief tour of Plain TEX, eplain, LATEX, $\mathcal{AMS}$-TEX, AMS-LATEX, LA$\mathcal{MS}$-TEX, etc.
  - Getting TEX: how to set up and maintain a TEX system for yourself and for your friends
  - Fonts for TEX: how fonts are accessed by TEX itself and the many possibilities for getting 'typeset output'.
- Workshops: informal tutorials and discussions of a range of subjects, including:
  - Virtual fonts (Michael Doob),
  - TEX and multilingualism (Yannis Haralambous),
  - Future of BIBTEX and Makeindex (David Rhead, Joachim Schrod),
  - LATEX3 (Frank Mittelbach),
  - Problem-solving (Philip Taylor).
- Special interest groups (re-named bofs). There will also be opportunities for other special interest groups to meet.
- Welcome party: evening of Monday 26th.
- Orientation: a short introduction to England's 'Second city'.
- Conference Banquet: evening of Thursday 29th
- Either Bowling (10-pin), or visit to 'The Chocolate Experience': evening of Tuesday 27th.
- TUG general meeting: scheduled for the start of the afternoon session on Thursday 29th.
- We also hope to attract exhibits and presentations covering many activities in the area of Electronic Publishing, such as: Bibliographic software, SGML-reLaTed software, Structured-document editors, TEX User Groups from around the world. . .

Table of Contents TUGboat

Volume 13.4

December 1992

TUGboat table of contents files are on `math.utah.edu` in `pub/tex/pub/tugboat`, also accessible via `tuglib@math.utah.edu` server by 'send index from tex/pub/tugboat'.

TUGboat 13.4 (December 1992)