DANTE

Deutschsprachige Anwendervereinigung TEX e.V.

Die TEXnische Komödie

Heft 2/1993 5. Jahrgang September 1993

Impressum

Beiträge sollten in Standard-LATEX-Quellcode an untenstehende Anschrift geschickt werden (entweder per e-mail oder auf Diskette). Sind spezielle Makros oder Stylefiles dafür nötig, so müssen auch diese mitgeliefert werden. Außerdem müssen sie auf Anfrage Interessierten zugänglich gemacht werden.

Diese Ausgabe wurde mit Hilfe von `PubliC TeX--XeT, V. 3.141--1.1/DOS-TP 1.4`, LATEX 2.09 v. 25.3.1992, `dviscr` v. E. Mattes (für die Bildschirmdarstellung), `dvihplj 1.4d` v. E. Mattes (für die Korrektur) und `dvips 5.518` (für die endgültige Belichtung) fertiggestellt.

Erscheinungsweise: vierteljährlich
Erscheinungsort: Heidelberg
Auflage: 3000

Herausgeber: DANTE, Deutschsprachige Anwendervereinigung TEX e.V.
Postfach 10 18 40
69008 Heidelberg

Tel.: 06221/2 97 66
Fax: 06221/16 79 06
e-mail: `dante@vm.urz.uni-heidelberg.de`

Belichtung: Jürgen Glöckner
In der Hessel 23
69168 Wiesloch

Druck: VOD Vereinigte Offsetdruckereien Mannheim Heidelberg
Handelsstr. 13
69214 Eppelheim

Redaktionsschluß für Heft 3/1993: 30.9.1993

Editorial

Liebe Leserinnen und Leser,

nachdem ich beim letzten Mal allen den Mund wäßrig gemacht habe, indem ich eine Vielzahl unterschiedlicher Beiträgen versprochen hatte, muß ich das nun doch für diese Ausgabe zurücknehmen. Die Beiträge sind zwar äußerst interessant, aber nicht zahlreich. Das kommt aber nicht etwa daher, daß zu wenige vorlägen, sondern ist vielmehr dadurch bedingt, daß die veröffentlichten Artikel sowohl ungewöhnlich lang sind als auch sinnvollerweise in dieser Ausgabe am Stück veröffentlicht werden sollten. Deshalb habe ich mich entschieden, diesmal nur drei größere Arbeiten abzudrucken und dafür in der nächsten Ausgabe wieder mehr (und) kürzere. Ich möchte alle Autoren um Verständnis bitten, deren Artikel erst das nächste Mal erscheinen werden. Ihre Einsendungen sind nicht vergessen!

Da das die erste Ausgabe nach Umstellung der Postleitzahlen ist, kam auf mich die undankbare Aufgabe zu, diese Umstellung auch bei den hier enthaltenen Adressen vorzunehmen. Ich hoffe, ich habe alle entdeckt und richtig ersetzt. Leider ging das nicht vollautomatisch, aber was tut man nicht alles für das allgemeine Wohl ...

In diesem Sinne verbleibe ich

Ihre Luzia Dietsche

Hinter der Bühne

Vereinsinternes

Grußwort

Joachim Lammarsch

Ich möchte, wie so oft an dieser Stelle, kurz die wichtigsten Gegebenheiten zusammenfassen, ohne der Mitgliederversammlung in Kaiserslautern vorgreifen zu wollen.

Hatte ich mich in der letzten Ausgabe darüber gefreut, daß uns die Hilfskräfte in einer Anzahl zur Verfügung stehen, wie wir uns dies schon immer gewünscht haben, hat dieser Zustand leider nicht lange angehalten. Tinka Reichmann hat Heidelberg verlassen und steht daher unserem Verein nicht mehr zur Verfügung. Das ist ein großer Verlust, trotzdem wünschen wir ihr alles Gute für die weitere Zukunft. Auch die Urlaubszeit geht nicht ohne Einschränkungen an uns vorüber. Das macht sich vor allem durch Verzögerungen beim Versand von Büchern und Software bemerkbar. Ab dem kommenden Semester arbeitet glücklicherweise Susanne Knab (nach zwei Auslandssemestern) wieder für uns, so daß die Durststrecke durch den Ausfall von Frau Reichmann nur von kurzer Dauer ist.

Die Geschäftsräume von DANTE e.V. sind mittlerweile eingerichtet und Ehrenfried Just, einer unserer freiwilligen Helfer, baut gerade die Telefonanlage ein. Stellvertretend für alle, die in ihrer Freizeit geholfen haben und noch helfen werden, die Räume funktionstüchtig zu gestalten, gebührt ihm unser Dank. Es geht, wenn auch langsam, so doch stetig voran.

Die bei Springer-Verlag erscheinenden TEX-Bücher sind nun auch über den Verein erhältlich. Leider unterliegen aber auch die in englischer Sprache erscheinenden Titel der Preisbindung des deutschen Buchhandels, wie sich nach Rücksprache beim Verlag herausstellte. Der Verlag sagte nun zu, ab 1994 wie bereits Addison-Wesley Verlag Deutschland und International Thomson Publisher in die höchste Beitragsgruppe („Firmen, die an TEX verdienen") zu wechseln. Genauso wie die TEX-Bücher von Springer-Verlag ist das LATEX-Buch vom Vieweg-Verlag über DANTE e.V. erhältlich. Leider lehnte es dieser Verlag jedoch ab, durch eine Mitgliedschaft unseren Verein zu unterstützen, was ich sehr bedaure.

Die Erfassung der Mitgliedsbeiträge zur Unterstützung der TUG gestaltete sich problematischer als je angenommen. Der Grund war zum einen, daß die Zuordnung der Gelder noch schwieriger war, als bei den Mitgliedsbeiträgen für DANTE e.V., zum anderen daran, daß es durch den Umzug der TUG von der Ost- an die Westküste zu Problemen bei der Übermittlung der Namen gab. Da die TUG ungeachtet dieser Probleme ihre Mitgliederzeitung, das „TUGboat", ohne Rückfrage bei uns verschickte, war ein Durcheinander unvermeidbar. Zusätzlich kam hinzu, daß all diejenigen, die der TUG mitgeteilt hatten, sie hätten gezahlt, weiter als Mitglieder geführt wurden. Es gilt auch hier, wie so oft, daß aller Anfang schwer ist.

Weitere Informationen über die Sitzungen des *Board of Directors* der TUG werde ich während der Mitgliederversammlung in Kaierslautern mitteilen. Diejenigen, die nicht dabei sind, können das Gesagte im Protokoll nachlesen. Ich freue mich aber wie immer, möglichst viele Mitglieder dort wiederzutreffen bzw. neu kennenzulernen. Bis dahin wünsche ich Ihnen alles Gute, Ihr

Joachim Lammarsch
(Präsident DANTE e.V.)

Fonds zur Unterstützung von Mitgliedern

Joachim Lammarsch

Es hat sich in der Vergangenheit gezeigt, daß einige Mitglieder trotz ermäßigtem Mitgliedsbeitrag den Beitrag nicht bezahlen konnten und deshalb ausgetreten sind. Dank verschiedener großzügiger Spenden ist es jetzt möglich, diese Mitglieder zu unterstützen (siehe Protokoll der letzten Mitgliederversammlung).

Jedes Mitglied, das auf Grund widriger Umstände nicht in der Lage sein wird, den Mitgliedsbeitrag für 1994 aufzubringen, kann in einem formlosen Schreiben an das Präsidium[1] die Gründe dafür erläutern und um Unterstützung bitten. Ein Gremium von fünf Personen wird dann über die Anfragen entscheiden. Beachten Sie bitte, daß der Antrag auf Unterstützung bis zum 31. Oktober 1993 beim Präsidium eingegangen sein muß (das Datum des Poststempels zählt).

[1] Stichwort „Fonds", Postfach 10 18 40, 69008 Heidelberg. Das Stichwort nicht vergessen, dann wird die Post auf jeden Fall direkt an das Präsidium weitergeleitet.

TEX-Theatertage

Bericht von der 14. Tagung der TUG

Ulrik Vieth

Die 14. Jahrestagung der TEX Users Group (TUG '93) stand in diesem Jahr unter dem Motto *'A World-Wide Window on TEX'* und fand wohl auch aus diesem Grunde erstmals in Europa statt. Veranstaltungsort war die Aston University in Birmingham, UK, die einigen wohl als Standort des Archivs der UK-TEX Users Group bekannt sein dürfte.

Die Teilnehmerliste verzeichnete insgesamt 162 Teilnehmer aus 23 Nationen. Interessanterweise bildeten die deutschen Teilnehmer die drittstärkste Gruppe gleich hinter denen aus Großbritannien und den USA. Erfreulich war auch die Zahl der Teilnehmer aus Osteuropa, speziell aus Polen, der Tschechischen Republik und aus Rußland, die zahlenmäßig nahezu mit denen aus Kanada, Frankreich, den Niederlanden oder Japan mithalten konnten.

Die Atmosphäre auf der Tagung entsprach dem, was man von anderen europäischen oder deutschen TEX-Tagungen kennt, d.h. man saß im Anschluß an das eigentliche Tagungsprogramm noch bis spät in die Nacht zusammen und unterhielt sich über alle nur denkbaren TEXnischen oder nicht-TEXnischen Dinge.

Zu den Besonderheiten englischer TEX-Tagungen gehört allerdings, daß um 23 Uhr in den Pubs *"Last Orders!"* ausgerufen und anschließend innerhalb einer halben Stunde geschlossen wird. Dies hatte zur Folge, daß sich das Geschehen danach auf die Etagen-Küchen des Studentenwohnheims verlagerte, wo sich bei einer mitternächtlichen Tea-Party die Gespräche häufig auf die nicht-TEXnischen Themen verlagerten, so z.B. über den britischen Humor, über Science-Fiction-Literatur, über die britische Nationalsportart Cricket oder auch über die Sorgen und Nöte unseres Vereins.

Schließlich war da auch noch der letzte Abend, an dem PHILIP TAYLOR begleitet von BOGUSŁAW JACKOWSKI und polnischem Wodka zur Gitarre griff und seine Sangeskünste demonstrierte. Wahrlich eine bemerkenswerte Darbietung, deren Fortsetzung bei der nächsten EuroTEX-Tagung schon geplant ist! Wer hätte gedacht, welche Talente sich unter uns befinden?

Das Tagungsprogramm füllte die Woche vom 26. bis 30. Juli 1993 und resultierte in einem 143 Seiten starken Buch, das noch nicht alle der eingereichten Beiträge zu den *'Proceedings'* der Tagung umfaßte. Die endgültige Fassung wird im *'TUGboat'* erscheinen und sicher noch umfangreicher werden. Im folgenden soll auf die einzelnen Beiträge genauer eingegangen werden.

Montag, 26. Juli 93

Tutorien

Der erste Tag begann mit Tutorien. In einer Parallelsitzung gaben SEBASTIAN RAHTZ und MICHEL GOOSENS eine Einführung in LaTeX, während CHRIS ROWLEY unter dem Titel *'Flavours of TeX: A brief tour'* einen Überblick über die Stärken und Schwächen der vielen Varianten von TeX-Formaten gab, darunter PLAIN TeX, LaTeX, AMS-TeX, AMS-LaTeX, LAMS-TeX und *e*-PLAIN.

In der zweiten Sitzung des Vormittags befaßte sich ALLEN REESE unter dem Thema *'Getting TeX'* mit den Problemen, mit denen sich ein Anfänger bei der Installation von TeX auf einem neuen System konfrontiert sieht. In der daran anschließenden Diskussion kam der Einwand, daß nicht die Vielzahl der Files an sich das Problem ist, sondern eher die Zahl der Files, die manuell installiert oder bei der Installation angepaßt werden müssen. Eine automatische Installationsroutine könnte viele Probleme beseitigen helfen. Aus Zeitmangel konnten Vorschläge zu einer Gliederung der TeX-Distribution in einen verbindlichen, einen empfohlenen und einen optionalen Teil nicht weiter ausdiskutiert werden.

Nach der Mittagspause gab dann MAREK RYĆKO unter dem Titel *'TeX from* `\indent` *to* `\par`*'* einen kleinen Einblick in die Geheimnisse der inneren Abläufe von TeX am Anfang und am Ende eines Absatzes. Damit zeigte er, daß es auch für fortgeschrittene TeX-Benutzer noch einiges zu lernen gibt.

Keynotes

Im weiteren Verlauf der ersten Nachmittagssitzung sprach dann die derzeitige Präsidentin der TUG, CHRISTINA THIELE, über die Zukunft von TeX und der TUG. Sie betonte, wie wichtig es für den erfolgreichen Fortbestand der TUG ist, daß sich alle Mitglieder über aktuelle Entwicklungen auf dem laufenden halten und an der weiteren Entwicklung mitarbeiten. Dazu zählt auch die Arbeit der 'Technical Working Groups', die im Laufe der Woche ihre Berichte abgaben.

Nach der Pause berichtete dann JOACHIM LAMMARSCH unter dem Titel *'A new typesetting system: Is it really necessary?'* über die Beweggründe, die zur Initiierung des NTS-Projekts durch DANTE eV. geführt haben. Er kam zu dem Schluß, daß NTS die einzige Chance für die Zukunft ist, die wir als TEX-Anwender haben. Verglichen mit der Zahl der Anwender anderer DTP-Produkte sind wir nämlich nur eine kleine Minderheit.

LATEX-*Projekt*

Der interessanteste Teil des Nachmittags war für viele sicherlich der Bericht des LATEX3-Entwickler-Teams, vertreten durch FRANK MITTELBACH, RAINER SCHÖPF und CHRIS ROWLEY. Zunächst gab es ein Announcement über ein neues Release des derzeitigen LATEX2.09 unter dem Arbeitstitel LATEX2e. Dieses ist für Herbst des Jahres vorgesehen und wird standardmäßig NFSS2 enthalten, welches bereits jetzt auf den File-Servern erhältlich ist.

In Zukunft wird es dann zweimal jährlich ein vollständiges Release aller Files zu festen Terminen geben, auch wenn sich an einzelnen Files nichts geändert haben sollte. Damit soll der verwirrenden Vielzahl nebeneinander bestehender Versionen entgegengewirkt werden.

Künftig wird es auch eine offizielle email-Adresse für Bug-Reports geben. Zu LATEX2e gehört ein File `latexbug.tex`, das alle dafür benötigten Daten in ein ASCII-File ausgibt. Wenn die benutzte LATEX-Version zu alt ist, wird die Empfehlung ausgegeben, zunächst das letzte Update zu installieren. Aus dem Publikum kam dann der Vorschlag, in diesem Fall auch gleich einen freundlichen Brief an den Systemverwalter zu generieren.

Zu den Neuerungen gehört auch, daß das offizielle LATEX-Manual von LESLIE LAMPORT derzeit überarbeitet wird, um einige neu aufgenommene Features zu berücksichtigen. Außerdem wird es im Herbst ein neues Buch von MICHEL GOOSENS, FRANK MITTELBACH und ALEXANDER SAMARIN unter dem Titel *'The LATEX Companion'* geben, das die Details des NFSS2 und einer Vielzahl von LATEX-Styles ausführlich beschreibt.

Über die Arbeit am LATEX3 Projekt wurde später in einem Workshop berichtet. Im letzten Jahr hat sich die Arbeit vor allem auf das NFSS2, das Update des Babel-Systems und die Untersuchung verschiedener Einzelprobleme konzentriert, die auf der 'Volunteer Task List' ausgeschrieben waren. Ein Teil der dabei gefundenen Ergebnisse sowie allgemeine Berichte über LATEX3 sind auf den bekannten File-Servern verfügbar gemacht worden.

Ein Fertigstellungstermin für LaTeX3 konnte nicht genannt werden. Es ist nicht beabsichtigt, Teile des Systems vorab herauszugeben, da das interne Interface noch nicht endgültig feststeht. Wenn das System fertig ist, wird es auf jeden Fall eine vollständige Dokumentation geben, sowohl für das Benutzer-Interface als auch für das interne Interface.

Social Event

Der Abend des ersten Tages klang aus mit einem Willkommensempfang an der Bar der Universität.[1] Da vielen Teilnehmern die „Grundlage" fehlte (es gab freie Getränke, aber nur ein begrenztes Kontingent an Erdnüssen und Kartoffelchips mit Essig), verschwand ein großer Teil des Teilnehmerfeldes auf der Suche nach einem Imbiß. Wie an allen Tagen endete der Abend mit einer Tea-Party, wo man sich dann wieder traf und bis spät in die Nacht zusammensaß.

Dienstag, 27. Juli 93

Multilingual TeX

Der Vormittag des zweiten Tages war ganz dem Thema nationaler Sprachanpassungen gewidmet. Zuerst berichtete Gabriel Valiente Feruglio über die TeX-Anpassung an das Katalanische.

Danach gab Irina Makhovaya einen Bericht über die derzeitige Situation der russischen TeX-Gruppe (CyrTUG).

Unter dem Titel *'Language-dependent ligatures'* berichtete dann John Plaice aus Kanada über eine 16- bzw. 32-bit TeX-Erweiterung unter dem Titel Ω, die das Konzept von 'Character Clustern' benutzt, und damit einige der bei Trennungen und Ligaturen auftretenden Probleme zu lösen versucht. Ω beginnt mit Version π, d.h. dort wo TeX aufhört, und ist sicherlich ein interessanter Beitrag zur Diskussion über NTS.

Nach der Pause berichtete Larry Siebenmann über sein System FORMATDUMPER, das die Umschaltung zwischen verschiedenen Sprachen in einem Text über eine virtuelle Sprache, die er 'EG' nennt, bewerkstelligen soll.

Anschließend gab Yannis Haralambous einen Bericht über die Aktivitäten der Multilingual TeX Working Group. Zu den Ergebnissen gehört eine Festlegung des Minimalumfangs eines TLPs (TeX Language Packages) sowie eine

[1] Bei dieser Gelegenheit habe ich mich dann auch dazu überreden lassen, diesen Artikel zu schreiben. Eine interessante Erfahrung, wie lange das braucht!

Einigung über ein Manifest, das die Ziele der Working Group festlegt. Bestandteil dieses Manifests ist, daß alle Teile eines TLPs *freeware* und alle Sprachen gleichberechtigt sein sollen. Auch wird Kompatibilität zwischen allen TLPs angestrebt. Schließlich gehört auch eine Dokumentation aller Teile in der jeweiligen Sprache und in US-Englisch[2] zum Minimalumfang eines jeden TLPs.

Weiterhin wurde dann eine Übersicht über die Sprachen gegeben, für die TEX-Anpassungen existieren oder in Arbeit sind. In der Liste der zuständigen Koordinatoren wurde für den deutschen Sprachraum BERND RAICHLE genannt, der ja schon seit einiger Zeit Koordinator von DANTE e.V. für den `german.sty` ist.

Bibliographische Tools

Der Nachmittag begann mit NELSON BEEBE, der ein System von BIBTEX-Tools vorstellte, das unter anderem eine verbesserte Fehlerabfrage und eine striktere Syntax vorsieht und dadurch auch die Aufbereitung von BIBTEX-Dateien zur Verarbeitung in anderen Systemen ermöglicht. Es wird mit diesen Tools dann sehr einfach, z.B. eine sortierte Liste aller Einträge zu generieren oder doppelte Einträge herauszufiltern.

Im Anschluß daran berichtete FRANK BENNET über LEXITEX, ein Macro-Paket, das einen bestimmten Stil für Literaturzitate in juristischen Zeitschriften implementiert, wo abgekürzte Zitate in einer bestimmten Form als Fußnoten gesetzt werden sollen.

Horizonte

Nach der Pause zeigte DANIEL TAUPIN, wie man mit TEX und METAFONT komplizierte Landkarten konstruieren kann. Besonders ärgerlich fiel dabei auf, daß viele `DVI`-Treiber unter DOS mit der unnötigen Limitierung auf 64 KB Speicher compiliert sind, was für die hier gezeigten Anwendungen nicht ausreicht.

Unter dem Titel *'Mixing TEX and* SGML*: A recipe for disaster?'* berichte dann PETER FLYNN über die Dokumentstrukturbeschreibungssprache SGML, die von vielen Verlagen bevorzugt wird, da sie vielseitiger verarbeitet werden kann als TEX, wo häufig neben dem Inhalt eines Dokuments auch schon das Aussehen mehr oder weniger (un)flexibel kodiert ist.

Er stellte eine Utility namens `sgml2tex` vor, das ein standardisiertes SGML-Dokument in TEX konvertieren kann und nur noch ein geeignetes Style-File

[2] Einige Sprachen sind eben doch gleicher als andere!

benötigt. In der anschließenden Diskussion erwähnte FRANK MITTELBACH, daß er vor einiger Zeit etwas ähnliches entwickelt hat, was er bei Interesse auch gerne zur Verfügung stellen will.

Workshops, Cultural Event und Bowling

Die Nachmittagssitzung endete mit zwei parallelen Workshops über BIBTEX-Utilities und über Multilingual TEX. Außerdem wurde für Interessierte ein Besuch der 'Barber Collection of Fine Art', einer Kunstausstellung in der Universität, angeboten. Der Abend nahm seinen Ausklang beim traditionellen Bowling, das bei den TUG-Tagungen der letzten Jahre zu einem festen Bestandteil des Programms geworden ist. Auch hier hieß es um 23 Uhr *"Last Orders!"*. Die allabendliche Tea-Party bedarf wohl keiner Erwähnung mehr.

Mittwoch, 28. Juli 93

Zukunft

Nach einer von MARTIN BRYAN gehaltenen Einführung in DSSSL, eine standardisierte Dokumentstilbeschreibungssprache, die eine Lisp-ähnliche Syntax benutzt, folgten eine Reihe von Vorträgen, die sich mit der Zukunft von TEX befaßten.

Unter dem Titel *'NTS: A future to TEX?'* berichtete PHILIP TAYLOR über die technischen Aspekte des NTS-Projekts, dessen Leitung er auf der letzten DANTE-Tagung übernommen hatte. Er betonte, daß es für die Glaubwürdigkeit des Projekts wichtig ist, Ergebnisse zu erzielen. Als geeigneten Weg dazu, sieht er eine stufenweise Verbesserung von TEX unter dem Titel *e*-TEX, die 100-prozentige Kompatibilität für die Benutzer und die Implementatoren bewahren soll.

Er ging auch auf die Diskussion über ein `\system` Kommando ein, das eine saubere Schnittstelle zu Betriebssystem-Routinen wie z.B. `sort` oder `diff` bereitstellen soll. Ein solcher `\system` Befehl birgt einige Risiken, da auf diese Weise auch Benutzernummern „geknackt" oder ganze Verzeichnisse gelöscht werden können. Deshalb ist es erforderlich, daß ein solcher Befehl abgeschaltet oder mit einer Sicherheitsabfrage versehen werden kann. Ein anderer Vorschlag beruht auf der Idee einer Makrobibliothek, die Routinen für benötigte File-Zugriffe in einer kontrollierten Weise ermöglichen soll. Es ist dann Aufgabe des jeweiligen Implementators, entsprechende system-spezifische Routinen zu entwickeln.

PHILIP TAYLOR bedauerte, daß er sich wegen der Organisation der TUG-Tagung in den letzten Monaten leider nicht im erforderlichen Umfang um das NTS-Projekt kümmern konnte. Er hofft aber, nach Abschluß der Tagung wieder einen neuen Anfang machen zu können.

Im weiteren Verlauf der Vormittagssitzung kamen dann auch die Entwickler kommerzieller TEX-Produkte zu Wort: Zunächst stellte RICHARD KINCH eine neue Implementierung von *TurboTEX* für Windows vor.

Anschließend berichtete MICHEL LAVAUD von seinem System A^{S}TEX, das auf dem kommerziellen Produkt *Framework* aufbaut und nun an seine Grenzen kommt, da einige Probleme nur noch durch Änderungen am zugrundeliegenden System zu lösen wären. Er erwägt nun eine Portierung auf *GNU Emacs*, was auf PCs wiederum zu anderen Problemen (Speicherplatz) führt.

Schließlich berichtete ROGER HUNTER über das Produkt *Scientific Word*, das aus einer Windows-Oberfläche und einer kommerziellen TEX-Version (*TurboTEX*) im Hintergrund besteht. Für Windows-Anwender, die sich nicht mit der Syntax der TEX-Kommandos anfreunden können, ist dieses Produkt vielleicht eine Alternative.

Der Vormittag endete mit einer Diskussion zum Thema „Zukunft", in der auch die Software-Entwickler Rede und Antwort standen.

Cultural Event

Der Nachmittag dieses Tages war freigehalten für einen Ausflug nach Stratford-upon-Avon, dem Geburtsort von William Shakespeare, mit einem Besuch des Shakespeare-Museums und einer Aufführung von *'King Lear'*. Bei dieser Gelegenheit war auch das Original-Exemplar von *'King Lear'* zu sehen, das bestimmt nicht für jede beliebige Besuchergruppe zugänglich gemacht wird. Offensichtlich scheinen TEX-Benutzer inzwischen bekannt dafür zu sein, daß sie einen Sinn für wertvolle Bücher haben — oder sind es nur die Organisatoren der Tagungen, die über besondere Kontakte zu den Archivaren verfügen? Es war jedenfalls nicht das erste Mal, daß anläßlich des kulturellen Beiprogramms einer TEX-Tagung Archivare ihre Schätze zeigten!

Andere Konferenzteilnehmer, die auf den Ausflug verzichtet hatten (Kostenpunkt immerhin £ 40), trafen sich am Nachmittag mehr oder weniger zufällig in den Buchläden von Birmingham. Leider befand sich einer der beiden größten Läden am Ort gerade im Umzug und das neue Geschäft mit 4 Meilen(!) Büchern war noch nicht eröffnet.

Gegen Abend fanden sich dann die Zurückgebliebenen, darunter ein großer Teil der deutschen Teilnehmer, nach und nach im `/pub/sacks_of_potatoes` auf dem Campus-Gelände ein, wo sie auch den Rest des Abends bis hin zu den *"Last Orders!"* verbrachten. Wie zu hören war, sollen einige Teilnehmer den freien Nachmittag einfach nur zum Ausschlafen benutzt haben. Andere mußten dagegen frühzeitig ins Bett gebracht werden. Eine solche Konferenz bedeutet eben fast einen 18-Stunden-Tag!

Donnerstag, 29. Juli 93

Fonts

Der Vormittag dieses Tages war dem Thema Zeichensätze gewidmet. MICHAEL DOOB berichtete über den Einsatz von Virtual Fonts bei der Produktion des Journals der Canadian Mathematical Society. Dabei soll der Ausdruck in einem Büro in Toronto mit einem speziellen Symbol-Zeichensatz erfolgen, während der Text unter TEX in den Büros in Winnipeg und anderen Orten bearbeitet wird. Auf Grund der großen Entfernung zum Druckerstandort schied eine Zeichensatzanpassung auf der Druckerseite aus, so daß eine Umkodierung auf der Eingabeseite mittels 'virtual fonts' gewählt wurde. MICHAEL DOOB kam zu den Schluß, daß 'virtual fonts' wirklich von großem Nutzen sein können.

YANNIS HARALAMBOUS stellte anschließend wieder einmal eine neue exotische METAFONT-Familie vor, diesmal die kambodschanische Khmer-Schrift, die eine recht komplizierte Silbenschrift ist, für die es bislang noch keine standardisierte Fonttabelle gibt. YANNIS HARALAMBOUS beabsichtigt, seine Khmer-Schrift im Rahmen eines 'TEX Language Packages' als *public domain software* zur Verfügung zu stellen.

BERTHOLD HORN befaßte sich in seinem Vortrag mit dem Titel *'Scalable outline fonts'* sich mit der Frage, warum es bei über 14000 verschiedenen Zeichensatzfamilien nur ganze vier für TEX geeignete Familien von mathematischen Zeichensätzen im PostScript-Outline-Format gibt. Er berichtete von den Problemen bei der Generierung der Outline-Fonts für die CM-Familie aus den METAFONT-Quellen, die vor allem darauf beruhen, daß in METAFONT die Umrisse der Zeichen in der Regel indirekt erzeugt werden. Ein anderes Problem ist die anspruchsvolle Kodierung der mathematischen Zeichensätze in TEX.

Nach der Pause berichtete dann ALAN JEFFREY über `fontinst`, eine in TEX geschriebene konfigurierbare Installationsroutine für PostScript-Fonts. Bei der Entwicklung stellte sich das Problem, daß die in Cork beschlossene Zeichensatztabelle einige Zeichen vorsieht, die in den meisten PostScript-Fonts nicht

vorhanden sind und in manchen Fällen auch nicht aus Kombinationen anderer Zeichen konstruiert werden können. Andererseits ist aber für zusätzliche vorhandene Zeichen, wie z.B. weitere Ligaturen, kein Platz mehr frei ist.

Unter dem Titel *'A modest proposal'* machte er schließlich den Vorschlag, die Cork-Zeichensatztabelle geringfügig abzuändern und einige Symbole, die nicht direkt zum Textzeichensatz gehören, in einem 'Text Companion Font' zu verlagern, um damit Platz für weitere Ligaturen in den Textzeichensätzen zu schaffen. Leider fehlte es an Zeit, um diesen Vorschlag ausführlicher diskutieren zu können.

Die Serie der Vorträge über Fonts wurde abgeschlossen durch einen Bericht von MINATO KAWAGUTI über einen in Japan entwickelten flexiblen `DVI`-Treiber, der sowohl TEX-Fonts im `PK`-Format als auch PostScript-Fonts verarbeiten kann.

Literate Programming

An Stelle eines ausgefallenen Vortrags über Mathematiksatz berichtete WLODEK BZYL über eine Anwendung von 'Literate Programming' bei der Anpassung der *TUGboat*-Macros für die Bedürfnisse der Zeitung der polnischen TEX-Gruppe (GUST). Durch eine manuelle Konvertierung wurden zunächst die *TUGboat*-Macros an die Konventionen von `FWEB` angepaßt. Anschließend wurden dann die benötigten Änderungen mittels Change-Files vorgenommen. Es wurde der Vorschlag gemacht, die konvertierten `FWEB`-Files zukünftig auch als Quellen für die offiziellen *TUGboat*-Macros zu benutzen.

Mathematiksatz

LARRY SIEBENMANN berichtete schließlich über den Einfluß des Parameters `\mathsurround` auf das Erscheinungsbild mathematischer Texte. Er zeigte, daß der in TEX implementierte Mechanismus unzureichend ist, da ein vergrößerter Zwischenraum zwischen Text und Formeln nicht in allen Fällen sinnvoll ist, nämlich dann nicht, wenn z.B. Satzzeichen oder Bindestriche folgen. Eine intelligentere Lösung erfordert allerdings die Benutzung von `\futurelet`, was zu Problemen führt, wenn nicht gleichzeitig einige Macros in `plain.tex` abgesichert werden, die intern den Mathematikmodus für verschiedene Zwecke mißbrauchen.

Der Vormittag endete mit einem kurzen Bericht über den Stand des Wettbewerbs *'The "A" in* LATEX*'*. BARBARA BEETON und CHRISTINA THIELE gaben bekannt, daß der Einsendeschluß für den Wettbewerb bis zum Jahresende

verlängert wird, da das Heft der *'TEX and TUG News'* mit der Ausschreibung zu spät erschienen ist.

TEX-Archive

Im Anschluß an die Generalversammlung der TUG berichtete am Nachmittag GEORGE GREENWADE über das CTAN (Comprehensive TEX Archive Network) sowie über die Arbeit der zuständigen Technical Working Group. Anschließend folgte eine Diskussion, an der neben ihm noch SEBASTIAN RAHTZ (Aston Archiv), RAINER SCHÖPF (Server von DANTE e.V., Stuttgart) sowie JOACHIM SCHROD (Mirror-Software) als Podiumsteilnehmer beteiligt waren.

Die Diskussion drehte sich zunächst um die Frage, wie man Autoren von TEX-Macros dazu bewegen kann, standardisierte File-Header und `README`-Files mitzuliefern. Weiterhin wurde diskutiert, auf welche Weise Ankündigungen über neue TEX-Pakete am besten verbreitet werden können. Schließlich stellte sich die Frage, ob die Archiv-Manager veraltete oder nicht mehr unterstützte Files von den Servern löschen dürfen. Die Diskussion war kontrovers und führte zu keinen konkreten Ergebnissen.

Workshops und Math Font Encoding

Die Nachmittagssitzung endete wieder mit zwei parallelen Workshops, diesmal über *MakeIndex* und über das neue Encoding der Mathematik-Fonts. Unter dem Vorsitz von BARBARA BEETON von der AMS hat eine Working Group, an der unter anderem FRANK MITTELBACH, JUSTIN ZIEGLER sowie ALAN JEFFREY beteiligt sind, am Rande der Tagung ein Konzept für die Neu-Gruppierung der mathematischen Symbole in insgesamt sechs Zeichensätze erarbeitet. Darunter ist auch ein normaler Text-Zeichensatz im Cork-Encoding enthalten, der für Operatoren wie 'log' benutzt wird.

Grundgedanke dieser Aufteilung war es, sämtliche Abhängigkeiten zwischen den Text-Fonts und den Mathematik-Fonts zu beseitigen. Außerdem sollen alle Zeichen, zwischen denen eventuell Kerning erforderlich ist, also lateinische und griechische Buchstaben sowie Satzzeichen, Akzente und Klammern, in einen einzigen Font zusammengefaßt werden. Neben diesem 'Math Core' Font und einem 'Math Extension' Font (ähnlich wie bisher) soll es insgesamt drei 'Math Symbol' Fonts geben, die je ein Alphabet (in Kalligraphisch, Blackboard Bold und Fraktur) sowie alle übrigen geometrischen Symbole enthalten.

Bei den 'Math Symbol' Fonts soll vermieden werden, daß Zeichen für verschiedene Zwecke mißbraucht werden (z.B. das Minuszeichen für die Konstruktion von Pfeilen), da dies häufig zu Problemen führt, wenn andere Zeichensatzfamilien als die CM-Fonts benutzt werden.

Schließlich soll auch die Zahl der in `\bold` benötigten Versionen klein gehalten werden, wobei zugute kommt, daß schon aus anderen Gründen alle lateinischen und griechischen Buchstaben, die wohl am häufigsten in verschiedenen Stärken benötigt werden, bereits in einem einzigen Font zusammenstehen.

Unabhängig von den Mathematik-Zeichensätzen soll es zu jedem Text-Zeichensatz einen 'Text Companion Font' geben, worin unter anderem alle nicht-mathematischen Symbole und die Oldstyle-Ziffern zusammengefaßt werden. In diesem Font werden auch Plätze für zusätzliche Ligaturen reserviert, allerdings ohne eine genaue Vorgabe, welche diese sein sollen.

JUSTIN ZIEGLER, der seit einiger Zeit speziell an den Mathematik-Zeichensätzen arbeitet, beabsichtigt möglichst bald eine Prototyp-Implementierung, eventuell mittels 'virtual fonts', zu erstellen. ALAN JEFFREY wird ferner eine Mailing-Liste über das Encoding der Mathematik-Fonts einrichten.[3]

Es besteht die Hoffnung, daß in spätestens einem Jahr das Problem der fehlenden Mathematik-Fonts endlich gelöst sein wird und daß einer Umstellung auf die DC-Fonts dann nichts mehr im Wege steht.

Social Event

Am Abend folgte das abschließende Tagungsbankett. Stellvertretend für das Programm-Komitee dankte MALCOLM CLARK den Organisatoren der Konferenz, insbesondere PETER ABBOTT und MAUREEN CAMPBELL. Nicht vergessen werden sollten auch die ungezählten übrigen Helfer sowie STEFFEN KERNSTOCK, der als Redakteur des *'TUGly Telegraph'* für die tägliche Verbreitung aktueller Neuigkeiten sorgte. Nach dem Essen versammelten sich dann noch einmal fast alle Tagungsteilnehmer zu einem Abschiedstrunk im Campus-Pub, bevor sie sich zu den schon an anderer Stelle erwähnten Attraktionen des letzten Abends zusammenfanden.

[3] Diese Mailing-Liste `math-font-discuss@cogs.susx.ac.uk` besteht seit Anfang August und zeugt von erheblichen Aktivitäten. Sie dient in erster Linie der internen Kommunikation der Working Group, ist aber auch für interessierte Benutzer gedacht, um Vorschläge in die Diskussion einzubringen. Interessenten, die diese Mailing-Liste lesen wollen, sollten sich per email an die Adresse `math-font-request@cogs.susx.ac.uk` wenden. Achtung: Dahinter steht kein Listserver!

Freitag, 30. Juli 93

Macros

Der Vormittag des letzten Tages war schließlich den Spezialisten unter den Macro-Programmieren gewidmet. Unter dem etwas mysteriösen Titel *'Syntactic Sugar'* zeigte KEES VAN DER LAAN, wie man Programm-Kontrollstrukturen in TEX programmieren kann.

Dann zeigte JOHNATHAN FINE unter dem Titel *'Galleys, Space and Automata'*, wie man den vertikalen Zwischenraum zwischen verschiedenen Textelementen unter Verwendung des Konzepts endlicher Automaten sinnvoll kontrollieren kann. Leider berücksichtigen die Layout-Vorgaben vieler Verlage dieses Konzept nicht, da sie nur die Zwischenräume vor und nach den Textelementen spezifizieren, ohne dabei den Kontext des vorangegangenen Elements zu beachten.

In einem weiteren Vortrag zeigte KEES VAN DER LAAN dann, wie man Sortieralgorithmen in TEX programmieren kann.

Nach der Pause berichtete DANIEL TAUPIN über den aktuellen Stand von MusicTEX. Gegenüber dem Bericht auf der letzten EuroTEX-Tagung waren keine wesentlichen Neuerungen zu vermelden.

Nach diesen Vorträgen über Macro-Programmierung folgte ein an dieser Stelle etwas deplazierter kurzer Bericht von YURI MELNICHUK über die Benutzung von TEX in der Ukraine. Zu den Problemen gehört, daß die in der Ukraine benutzten Zeichensätze von den in Rußland benutzten etwas abweichen. Deshalb müssen wieder einmal eigene Varianten der Zeichensätze und TEX-Macros entwickelt werden, obwohl es sich in beiden Fällen um Kyrillisch handelt.

Typographie

Der letzte Vortrag des Vormittags und der letzte Nachmittag der Konferenz beschäftigten sich mit dem Thema Typographie.

MARY DYSON berichtete über das DIDOT-Programm, ein von der EG unterstütztes Programm zur Förderung der Ausbildung im Bereich Typographie und Schriftgestaltung. Sie berichtete von verschiedenen Seminaren und einem über mehrere Jahre angelegten Kurs für Design-Studenten, die von DIDOT organisiert wurden. Die Gestaltung eines Unterrichtsprogramms erwies sich schwieriger als erwartet, da die Teilnehmer sehr verschiedene Vorkenntnisse

und Erwartungen haben, je nachdem, ob sie aus den Bereichen Design oder Computer kommen oder irgendwo dazwischen angesiedelt sind.

Neben handwerklichen Kursen, in denen Schriftgestaltung mit Pinsel und Tusche sowie Schriftsatz in Blei vermittelt wurde, wurden auch Seminare veranstaltet, die in die Prinzipien des Computersatzes einführen sollen. Dazu wurde ein Hypertext-System entwickelt, das bedauerlicherweise nicht gut angenommen wurde, da vielfach gedruckter Output bevorzugt wird. Dies führte auch zu der Frage, ob Dokumente auf dem Bildschirm nicht ohnehin anders gestaltet werden müssen als auf dem Papier.

Weiterhin wurden Videos von einigen der veranstalteten Kursen aufgenommen, die eventuell bei zukünftigen Veranstaltungen als Lehrmaterialien verwendet werden können. Obwohl die Förderung des DIDOT-Programms durch die EG bald ausläuft, soll die begonnene Arbeit fortgesetzt werden. In diesem Zusammenhang wurde auch auf eine Konferenz über Electronic Publishing und Typographie hingewiesen, die im nächsten Jahr in Darmstadt stattfindet. Bezeichnenderweise werden Konferenzbeiträge nur im Word-Format und nicht in TEX akzeptiert.

Am Nachmittag stand dann noch ein Vortrag von RICHARD SOUTHALL über *'Document Design'* und eine abschließende Diskussionsrunde über Typographie auf dem Programm, die der Autor dieses Artikels leider versäumen mußte, da er in Unkenntnis des endgültigen Tagungsprogramms schon seinen Rückflug gebucht hatte.

Rückblick und Ausblick

Insgesamt war es eine sehr interessante und vielseitige Tagung, die diesmal auch durch eine Reihe von Ankündigungen der einzelnen Working Groups geprägt war. Es tut sich sehr viel. Allerdings fällt auf, daß ein großer Teil der Entwicklungen aus Europa kommen. Es bleibt zu hoffen, daß sich die Arbeit der Working Groups bald in brauchbaren Standards niederschlägt, die es ermöglichen, trotz der Vielfalt verschiedener Entwicklungen TEX-Dokumente weiterhin weltweit austauschen zu können.

Die nächste Tagung der TEX Users Group (TUG '94) ist für Santa Barbara in Kalifornien vorgesehen, wo die TUG seit etwa einem Jahr mit ihrer Geschäftsstelle angesiedelt ist. Der Zeitpunkt der nächsten TEX Users Group-Tagung in Europa steht noch nicht fest, aber es gibt ja auch noch die EuroTEX-Tagung und die Tagungen von DANTE e.V.

Von fremden Bühnen

Die Zukunft von TEX

Phil Taylor
Walter Obermiller

TEX und die anderen Programme aus Knuths „Computers and Typesetting" Familie gehören fraglos zu den erfolgreichsten Software-Paketen der Welt. Auf fast jedes erdenkliche Betriebssystem portiert, erfreuen sie sich einer manchmal fanatisch anmutenden Fan-Gemeinde. Die Weiterentwicklung dieses Programmpakets wurde 1992 eingestellt und viele am Computersatz Interessierte meinen, daß etwas zu unternehmen sei, um sicherzustellen, daß die Idee und die Philosophie, die TEX verkörpert, weiterentwickelt wird und nicht einfach untergeht. Im folgenden möchte ich Stärken und Schwächen des TEX-Systems in seinem jetzigen Zustand diskutieren und einige Wege aufzeigen, die beschritten werden können, um die TEX-Philosophie auszubauen. Ich stelle eine Vorgehensweise zur Weiterentwicklung von TEX vor, die der Philosophie und Knuths Wunsch Rechnung trägt: Einerseits die Entwicklung einzustellen, aber andererseits sicherzustellen, daß die Ideen und Ansätze von TEX nicht dadurch verloren gehen, daß ihrer Weiterentwicklung künstliche Beschränkungen auferlegt werden.

„Meine Entwicklungsarbeit an TEX, Metafont und Computer Modern ist beendet." Mit diesen Worten teilte D. E. Knuth, „Schöpfer" von TEX, der Welt mit, daß die Entwicklung des vielleicht erfolgreichsten Computersatzprogramms abgeschlossen ist und außer der Behebung von Fehlern keine weiteren Änderungen daran vorgesehen seien. Er verfügte, daß TEXs Versionsnummer sich asymptotisch π nähern solle und nach seinem Tod auf π gesetzt wird. Danach soll das Programm genau so bleiben, wie er es am Ende selbst hinterlassen hat: Als passendes Denkmal für einen der genialsten und produktivsten Computerwissenschaftler (auch Mathematiker und Bibelkenner), den die Welt jemals gesehen hat.

Wenn die Zukunft von TEX also vorgezeichnet ist, warum trägt dieser Aufsatz den Titel „Die Zukunft von TEX"? Einfach weil, wenn wir unseren Ohren und Augen trauen können, die Philosophie, auf die TEX gebaut ist, bereits so veraltet ist wie eine Pferdekutsche: TEX ist das Ergebnis einer vorzeitlichen Evo-

lutionskette, die auf der Erbmasse von Programmen wie Runoff, Nroff, Troff, Ditroff und Scribe aufsetzt, während sich der moderne Steinzeitmensch bereits mit Produkten wie Ventura Publisher, Aldus Pagemaker und Quark Xpress den Planeten untertan macht.

Es scheint, daß die Pionierzeiten längst vorbei sind, d.h. die Zeiten als die folgende Vorgehensweise noch akzeptabel war: Den Text einzugeben, ihn dann per Augenschein (!) auf seine Rechtschreibung zu prüfen, Formatierkommandos hinzuzufügen, das Formatierprogramm auf den Text loszulassen, einen etwaigen Fehler zu identifizieren, den Text zu korrigieren, das Formatierprogramm wieder anzuwerfen, den zweiten Fehler zu identifizieren, ihn dann in Ordnung zu bringen, den Text ein drittes Mal zu formatieren und diese Prozedur so lange zu wiederholen bis alle Fehler korrigiert waren. Zusätzliche Formatierläufe wurden notwendig, um Querverweise aufzulösen, danach erst konnte man das Ergebnis auf dem Bildschirm bewundern. Nun sah man Formatierfehler, die es notwendig machten, den Zyklus aufs neue zu durchlaufen. Die Kollegen hingegen sitzen an ihren Macs, klicken sich an ihren Mäusen wie wahnsinnig die Finger wund und halten uns für „total übergeschnappt". Das müssen wir ja wohl auch sein, da wir diesen *modus operandi* nicht nur genießen, sondern sogar versuchen, die „wahnsinnigen Mausklicker" zu TEX-Benutzern zu konvertieren.

Und warum? Was ist es, das uns TEX-süchtig macht? Vielleicht TEXs beschreibende und buchstabenorientierte Art? Die Tatsache, daß TEX den Benutzer — in direktem Gegensatz zu momentanen Trends — zwingt, darüber nachzudenken, was er erreichen will und diese Gedanken dann als eine Serie von Worten und Symbolen in die Datei zu tippen, anstatt das gleich mit einer Anzahl mehr oder weniger unstrukturierter Mausklicks auf dem Bildschirm zu tun? Vielleicht ist es seine leichte Portierbarkeit, die Tatsache, daß es Implementierungen (meist Public Domain) für fast jedes Betriebssystem auf diesem Planeten gibt? Ist es die Tatsache, daß TEX deterministisch ist, d.h. daß ein Text, der mit TEX-Kommandos formatiert ist, immer *genau* das gleiche Resultat erzeugt? Ist es das (vielleicht zu naive) box- und glue-Modell, das TEX benutzt, um Text auf einer Seite anzuordnen? Ist es die Eleganz, mit der Form und Inhalt sauber voneinander getrennt werden? Vielleicht seine Implementierung als Makrosprache anstatt einer prozeduralen Sprache? (Könnte man eine prozedurale Sprache noch als TEX erkennen?) Könnten es auch die unglaublichen Verkünstelungen sein, die man machen muß, um ein bestimmtes Ergebnis zu erreichen? (Oder gar die unglaubliche Freude, wenn man es endlich erreicht hat?) Wie viele der angesprochenen Elemente könnten entfernt werden, so daß TEX immer noch TEX bleibt? Ich werde auf diese Fragen zurückkommen und versuchen, einige von ihnen in dieser Arbeit zu beantworten.

Wir haben also die Wahl: Wir lassen der natürlichen Selektion freie Bahn, d.h. TEX kann somit (nachdem es seinen Zweck erfüllt hat, uns die Vorteile des „Literate Programming" näherzubringen, während wir gleichzeitig mehr Zeit dazu aufwenden, schöne Arbeiten zu setzen, auf Kosten der Zeit, die für die eigentliche Forschung aufgewandt wird) in den Himmel für Computerprogramme kommen. Oder wir übernehmen gemeinsam die Verantwortung für die Zukunft von TEX und greifen in die natürliche Auslese korrigierend ein. Wir müßten sicherstellen, daß es sich in ein System weiterentwickelt, das weiterhin klar besser ist als die graue Masse der Maus-basierten, menügesteuerten Text- und Bildmanipulatoren, die TEX das Leben so schwermachen. So gut, daß einfach jeder zugeben muß, daß TEX der Schrittmacher der Satztechnologie des 21. Jahrhunderts ist.

Folgende Möglichkeiten bestehen:

1. Wir lassen TEX genau so, wie es jetzt ist: Eine Vorgehensweise, die zu verantworten ist, weil sie exakt dem entspricht, was Knuth selbst mit TEX vorhat; es wäre sehr arrogant zu meinen, wir wüßten mehr als Knuth über diesen Punkt.

2. Wir können an TEX minimale Verbesserungen vornehmen, bis diejenigen von uns, die TEX wirklich verstehen (die Wizards) der Meinung sind, die Schwächen seien ausgemerzt (d.h. es gibt keine einfachen Satzprobleme mehr, mit denen das verbesserte TEX im Gegensatz zu einem erweiterten TEX nicht fertigwerden könnte).

3. Wir können TEX erweitern, indem wir alle Wünsche seiner Benutzer integrieren, es quasi zu einer „eierlegenden Wollmilchsau" machen, und gleichzeitig sein gewohntes „Look and Feel" erhalten.

4. Wir können TEX (wie unter 3.) verbessern, aber uns die Möglichkeit offenhalten, das jetzige „Look and Feel" zu überdenken und es vielleicht substantiell zu verändern.

5. Wir können das Problem angehen, wie Knuth es tun würde, wenn er sich heute zum ersten Mal mit den Problemen des Satzes zu beschäftigen hätte. Sich die besten der heute erhältlichen Satzsysteme anzuschauen (TEX natürlich auch) und dann ein neues Satzsystem zu entwerfen, das weit besser wäre als eine Schnittmenge der heute bestehenden Systeme. Ein System, welches nicht nur den Anforderungen und Möglichkeiten von heute gerecht wird, sondern auch denen der überschaubaren Zukunft. Wir bräuchten allerdings einen Weg, um die Genialität, die Knuths Werk kennzeichnet, in dieses Projekt einzubringen.

Zweifellos wird jeder von Ihnen eigene Gründe dafür haben, warum er die eine oder andere der Optionen für gut oder schlecht hält. Es ist nicht meine Absicht, Sie von der einen oder anderen Lösung zu überzeugen. Aber es wäre auch unverantwortlich nicht zu sagen, daß ich die 3. Option für die schlechteste halte, da im schlimmsten Falle nur neue „Features" hinzugefügt würden. Es stellt sich aber die Frage nach einem roten Faden, der alle Erweiterungen mit dem alten Kern verbindet.

Die 1. Möglichkeit wäre durchaus zu rechtfertigen, ist sie doch Knuths Vorzugslösung. TEX würde dann am Selektionsdruck scheitern, aber es bleibt doch die Möglichkeit, daß TEX sowohl das höchstentwickelte Glied aber zugleich auch das Ende einer Evolutionsreihe bildet und daß zukünftige Satzsysteme eine ganz andere Philosophie haben werden (z.B. Maus-basierte Systeme).

Die zweite Option ist die konservativste und es gibt viel, was zumindest kurzfristig für sie spricht. Das jetzige „Look and Feel" bliebe erhalten und die Kompatibilität mit TEX wäre zwar nicht a priori garantiert, könnte aber durch gutes Design angestrebt werden. Im schlimmsten Fall könnte man die Erweiterung durch eine Option abschalten und man hätte ein „normales" TEX 3. Obwohl diese Lösung nicht im Sinne der Wünsche Knuths ist, so hat er doch klar gemacht, daß er nichts gegen solche Erweiterungen habe, solange das erweiterte System dann nicht TEX heißt. Ich schlage vor, diesen Ansatz „erweitertes TEX" zu nennen, um damit die Erweiterungen zum Ausdruck zu bringen, und viel wichtiger, den Geist und den Wunsch von Knuth zu respektieren.

Option 3 ist bedeutend innovativer, aber das gewohnte „Look and Feel" von TEX bliebe erhalten, allerdings sind Umfang und Art der Modifikationen völlig offen. Hierbei bestände die Möglichkeit tiefgreifender Änderungen (aus oben genannten Gründen scheue ich mich, das Wort „Verbesserungen" zu benutzen...). Wenn die Änderungen wohl durchdacht sind, sollte die Kompatibilität mit TEX 3 kein Problem sein, im schlimmsten Fall könnte man mit Hilfe einer `/noextension`-Option die Änderungen ausschalten. Die Zeit, die eine solche Lösung in Anspruch nähme, ist lang, auch wenn die Änderungen nicht einen Schwarm von „bugs" nach sich ziehen. Es ist nicht klar, wie verhindert werden soll, daß auch diese Modifikationen in absehbarer Zeit veralten. Und: Wenn das neue TEX, ich werde es „endgültiges TEX" nennen, sämtliche Wünsche *aller* Anwender in sich vereinen könnte, welche Verbesserungen blieben dann für die Zukunft?

Option 4 wäre der erste Versuch, TEX von Grund auf zu überarbeiten. Sie gibt uns die Möglichkeit, alle Gesichtspunkte — so auch das „Look and Feel" — zu bewerten und unter Umständen zu überdenken, während viele der zugrunde-

liegenden Algorithmen übernommen werden können. Man könnte dabei an ein System denken, bei dem Markup und Text völlig getrennt werden, unter Benutzung von Pointern, die das Markup und den Text (und umgekehrt?) verbinden. Ein Vorteil einer solchen Lösung ist, daß manche Probleme mit Blanks, Escape-Zeichen und unter Umständen die Probleme mit den category-codes komplett wegfallen. Natürlich ist dies nur eine von vielen Möglichkeiten. Gibt man das gewohnte „Look and Feel" von TEX auf, ist alles möglich. Man könnte diese Variante „Future-TEX" nennen.

Möglichkeit 5 ist mit Abstand die radikalste. Sie stellt nicht nur (zumindest anfangs) TEXs „Look and Feel" in Frage, sondern die gesamte Weisheit, auf der TEX fußt, und stellt die Gretchenfrage: Wie sollte Computersatz in der Zukunft funktionieren? Allerdings glaube ich, daß Knuth mit genau diesem Ansatz an die Entwicklung von TEX78 herangegangen ist. Hinzu kämen die Gedanken, die er sich heute machen würde bei der Aufgabe, eine Photosatzmaschine dazu zu überreden, höchste Qualität zu liefern. Ich denke, es ist wichtig festzustellen, daß es nichts in Option 5 gibt, was zwangsläufig zur Aufgabe der Philosophie und der Ansätze von TEX führen müßte. Es ist durchaus möglich, daß man nach genauem Hinschauen zu dem Schluß kommen muß, daß TEX *weiterhin* den „state of the art" im Computersatz darstellt und wir am besten damit beraten wären, es einfach so zu lassen wie es ist. Oder vielleicht eine gewisse Anzahl Verbesserungen vorzunehmen, aber das Modell des Satzes, das TEX repräsentiert, weiterhin zu benutzen. Andererseits gibt es keine Garantie, daß man *wirklich* zu einem solchen Schluß gelangen würde. Dieses System möchte ich „A New Typesetting System" nennen (um es von „The New Typesetting-System" zu unterscheiden, für das NTS die Abkürzung ist, s.o.).

Die oben aufgeführten Möglichkeiten schließen sich nicht notwendigerweise gegenseitig aus: Option 2 könnte eine Zwischenlösung sein, während wir damit beschäftigt sind, die Ressourcen für Option 5 bereitzustellen. Diese Lösung, könnte auf lange Sicht die beste sein (tatsächlich habe ich für sie gewisse Sympathien). Aber was auch immer der gewählte Weg sein wird, wir müssen einen Schlachtplan entwickeln, um einerseits zu entscheiden, welche Option (möglicherweise auch eine, die ich nicht vorgestellt habe) jetzt wirklich zum Einsatz kommt, andererseits um die Implementation der Verbesserungen zu koordinieren.

Bisher habe ich mich in meinen Ausführungen auf allgemeine Bemerkungen beschränkt. Ich möchte jetzt einige spezifische Probleme aufgreifen, auf die ich oben angespielt habe und meine persönliche Meinung dazu sagen, wie wir zu einer Lösung kommen könnten. Ich schlage vor, zuerst die Frage zu beantworten,

die uns am meisten am Herzen liegt: Was macht TEX zu TEX? Es scheint, daß TEX einige fundamentale und einige periphere Aspekte hat. Zu den fundamentalen gehören seine deskriptive und an Buchstaben orientierte Natur sowie sein deterministisches Verhalten, auch seine Programmierbarkeit (z.B. die Art, wie Schleifen implementiert sind), die Fähigkeit, für verschiedenste Aufgaben (wie Text-, Formel- und Musiksatz) einsetzbar zu sein, ebenso wie die Unabhängigkeit vom Ausgabegerät. Nicht zu unterschlagen ist auch die Tatsache, daß das Ergebnis — wenn man weiß, was man tut (:-)) — vom Gesichtspunkt der Ästhetik nicht von traditionell gesetzten Texten zu unterscheiden ist. Ebenso wichtig ist die Tatsache, daß TEX vollständig dokumentiert ist, beinahe alle Programmfehler beseitigt sind und Fehler, die noch auftreten, vom Autor des Programms sofort korrigiert werden. Zudem existiert für Probleme auf höherer (Makro-)Ebene eine weltweite Gemeinschaft von tausenden TEX-Anwendern, die man konsultieren kann. Knuths selbstlose Haltung, den Quellcode[1] mit einem Minimum von Restriktionen für jeden frei zugänglich zu machen, darf auch nicht vergessen werden. Dieser letzte Punkt ist in Kombination mit den oben angesprochenen entscheidend dafür, daß sich TEX weltweit so schnell verbreiten konnte.

Zu den peripheren Aspekten gehört, daß TEX als Makrosprache implementiert wurde und nicht als prozedurale oder deklarative Sprache, ebenso wie das grundlegende Paradigma von box und glue (Klebstoff). Ich zögere zu behaupten, das box-glue-Paradigma sei *kein* fundamentaler Gesichtspunkt von TEX. Denn eines scheint klar: Wenn ein Abkömmling von TEX statt über die bounding-box genaue Informationen über die Umrisse jeden Buchstabens hätte und er diese Buchstaben auf einem Rechteck-Netz setzen könnte (jetzt ist ein floating space zwischen den Zeilen) und die Buchstaben und Worte durch unterschiedlich dehnbare Leerzeichen getrennt wären, ansonsten aber alle anderen Eigenschaften von TEX hätte, würden die meisten Anwender dieses Programm als einen echten Abkömmling von TEX bezeichnen und nicht als eine „kranke" Mutation.

Ohne bewußt darüber nachzudenken, habe ich TEX jetzt durch seine Stärken charakterisiert, nicht durch seine Mängel. Aber wenn wir in den Prozeß der natürlichen Selektion eingreifen wollen, dann ist es wichtig, sich auch mit den Schwächen von TEX auseinanderzusetzen: Wenn es nämlich keine Schwächen gäbe, wäre die gesamte Frage nach der Zukunft von TEX nie aufgekommen. Aber während es relativ leicht ist, eine Untermenge von Stärken von TEX zu definieren, denen die meisten TEX-Anwender zustimmen könnten, bereitet die

[1] Auch für die TEX und Metafont Bücher, das wird leider oft vergessen...

Definition einer Liste mit den Schwächen von TEX ungleich größere Schwierigkeiten. Aber da müssen wir durch!

Fangen wir am besten damit an, die „impliziten" Kriterien anzuschauen, die Knuth im Kopf gehabt haben muß, als er mit dem TEX-Projekt begann und die noch heute die Funktionalität von TEX prägen. Denken Sie daran, TEX wurde im Jahre 1978 geboren. Damals als der Hauptspeicher eines Computers in Kilobyte angegeben wurde statt in Megabyte. Laserdrucker waren unbekannt und die Rechenleistung eines Großrechners kleiner als die der PCs, die heute bei vielen auf dem Tisch stehen. Das Konzept eines Previewers in Echtzeit erschien als ein phantastischer Traum. Jede dieser Beschränkungen muß in Knuths Denken eine Rolle gespielt haben, wenn nicht bewußt, so doch unbewußt. Wir sind uns der Seltenheit von Laserdruckern im Jahre 1978 nur deshalb so bewußt, weil man sie heute überall herumstehen sieht und wir erkennen den Mangel an Ionenstrahlhyperantrieben heute einfach deshalb nicht, weil Ionenstrahlhyperantriebe noch nicht erfunden sind.

Durch sorgfältige Lektüre des TEX-Buches und von `tex.web` können wir die damaligen Beschränkungen zu verstehen versuchen, die Knuth bei der Entwicklung von TEX beeinflußt haben. Auf Seite 110 kann man zum Beispiel lesen: „TEX benutzt eine spezielle Methode, um die optimalen Trennstellen von Zeilen innerhalb eines Absatzes herauszufinden, aber es versucht nicht, die besten Umbruchstellen für die Seiten eines gesamten Dokuments zu bestimmen. *Der Computer hat nicht genügend schnellen Speicher, um den Inhalt mehrerer Seiten gleichzeitig im Speicher zu halten*, deshalb optimiert TEX die Seitenumbrüche 'lokal' und nicht 'global'" (im Dokument). Man kann wohl mit Fug und Recht sagen, daß der Umbruchalgorithmus ein anderer geworden wäre, hätte man damals genug RAM in den Rechnern zur Verfügung gehabt. Andere Beschränkungen kann man erahnen, wenn man die Liste der numerischen Konstanten auf Seite 336 im TEX-Book ansieht, z.B. ist die Anzahl der Familien der Mathematik-Schriften auf 16 begrenzt. Solche Limitationen sind heute die Quelle von Problemen, wie bei der Implementierung des NFSS (Mittelbach und Schöpfs „New Font Selection Scheme") und auch die 16 Kategorien für Zeichen sind nur scheinbar ausreichend. Eine Folge hiervon ist es, daß das „^"-Zeichen mehrere Bedeutungen hat: Um nicht darstellbare Zeichen zu kodieren (z.B. `^^e4` ist in den dc-fonts ein ß) und als Superscript-Operator in mathematischen Formeln.

Wir können also mit einer gewissen Berechtigung davon ausgehen, daß die Summe der Restriktionen von wenig schnellem RAM, fehlenden Preview-Geräten und damals noch primitiven Druckern das ursprüngliche Design von TEX

einschneidend beeinflußt haben. Diese Einflüsse bekommen wir heute noch zu spüren. Es ist fast tragisch, daß immer noch Systeme verkauft werden, die fundamentale Schwächen in einem dieser Bereiche (schnelles RAM) haben: Ich meine natürlich die Rechner unter MS-DOS (fraglos die erfolgreichsten Computer überhaupt), die teilweise dieselben Beschränkungen in sich tragen wie der Ur-PC mit seinem 8088/8086-Prozessor. Weil diese Rechner so weit verbreitet sind, haben wir eine große Anzahl von Beiträgen zur Diskussionsliste NTS-L (Bitnet)[2] bekommen, in denen wir bekniet wurden, die Limitationen dieser Hardware zu berücksichtigen. Aber obwohl ich selbst ein PC-Benutzer bin, vertrete ich die Auffassung, daß sich die Entwicklung eines neuen Satzsystems *nicht* am 640 KB Limit des Speichermodells von DOS beschränken darf. Ich habe keine besondere Schwäche für die graphisch orientierte Oberfläche von Microsoft Windows, andererseits ist der Wegfall des 640 KB Limits unter Windows ein solch wichtiger Fortschritt, daß ich inzwischen die Meinung vertrete, ein zukünftiges System, das auf PC-Hardware läuft, müsse einfach Windows (oder OS/2) voraussetzen oder die Speicherbegrenzung mit Techniken (DOS-Extender wie im emTEX386 von Eberhard Mattes) umgehen. Wenn wir uns von den Beschränkungen primitiver Systeme wie MS-DOS leiten lassen, wie können wir dann hoffen, ein System für die Zukunft zu entwickeln?

Die Beschränkungen, die oben aufgeführt sind, sind die *historischen* Schwächen von TEX. Sie sind Einflüssen zu verdanken, die Knuth nicht ändern konnte. Aber wenn wir die Schwächen von TEX zusammenstellen wollen, so müssen wir auch die Anforderungen der Benutzer analysieren und herausfinden, in welchen Bereichen TEX diese Erfordernisse nicht erfüllt, gleich aus welchem Grund. Der „Benutzer", den ich hier meine, ist *jeder* TEX-Benutzer, gleich, ob er LATEX-Anfänger, TEX-Guru (jemand der Style-Files schreibt) oder Setzer ist. Die Designer von Style-Files können zwar gewisse Schwächen von TEX verstecken (z.B. die Tatsache, daß es keinen `\loop` Befehl gibt), aber grundsätzliche Probleme werden den Benutzer immer noch plagen, weil sie einfach nicht zu verstecken sind. Gute Einführungen in diese Problematik finden sich [1] und [2]. Etwas schwieriger sind die Beiträge zur `NTS-L` Diskussionsliste zu finden, die sich auch mit diesem Thema auseinandersetzen. Die Beiträge sind mittels `ftp` aus Darmstadt (`ftp.th-darmstadt.de` in `/pub/tex/documentation/nts-l/*`), Stuttgart (`ftp.uni-stuttgart.de` im Verzeichnis `/pub/soft/tex/digest/nts/*`) oder aus Aston (`tex.ac.uk` im Verzeichnis `disk$tex:[tex-archive.nts]nts-l.all`) zu holen.

[2] Abonnieren: Mail an listserv@vm.urz.uni-heidelberg.de mit einer Zeile: SUB nts-l vorname name

Was sind denn nun diese „fundamentalen Schwächen"? Zweifelsohne wird ein jeder seine eigenen Ansichten zu diesem Thema haben. Die Quellen, die ich oben angegeben habe, sind ein guter Ausgangspunkt für all diejenigen, die sich noch keine Gedanken zu diesem Thema gemacht haben. Im folgenden will ich meine sehr persönliche Meinung dazu sagen — was *ich* für wichtig und unwichtig halte. Ich behaupte nicht, an alles gedacht zu haben oder neue Gesichtspunkte ins Spiel zu bringen: einige der Ideen, die ich jetzt beschreibe, finden sich bereits in den oben angegebenen Quellen. Ich hoffe und glaube allerdings, daß meine Ausführungen ein representativer Querschnitt des derzeitigen Standes der Diskussion zu diesem Thema sind. Und nun zur Sache!

1. Fehlende Exception/Fehler-Behandlung: Es ist unmöglich, in TEX Fehler abzufangen. Wenn ein Fehler auftritt, löst er unweigerlich eine Standardmeldung aus. Ist der Fehler ernster als eine Warnung (d.h. mehr als overfull oder underfull boxes), hält TEX an und wartet auf eine Benutzereingabe. Dieses Verhalten erschwert es dem Formatdesigner, auftretende Fehler vom Style selbst behandeln zu lassen und ein benutzerfreundliches Verhalten sicherzustellen. Es ist zum Beispiel nicht möglich, beim Laden einer nicht vorhandenen Schrift, den Fehler zu behandeln und stattdessen einen anderen voreingestellten Font zu benutzen.

2. Es ist unmöglich festzustellen, ob und daß ein Fehler passiert ist: die Kommandos der `\last`-Familie (`\lastbox`, `\lastskip`, `\lastkern`, `\lastpenalty`) sind nicht in der Lage, zwischen einem passenden Wert in der Liste und Null zu unterscheiden. Da ein riesiger Unterschied zwischen einer Penalty von Null und keiner Penalty besteht, geht wichtige Information verloren.

3. Die hierarchische Natur der Zeilen- und Seitenumbruch-Prozedur: Wenn ein Paragraph einmal in Zeilen zerlegt ist, ist es für TEX nahezu unmöglich seine Umbruchentscheidung zu revidieren. In einem Absatz über zwei Seiten werden die Zeilen auf der zweiten Seite Zeilenumbrüche haben, die von den Umbrüchen auf der ersten Seite beeinflußt worden sind. Dies ist ein unhaltbarer Zustand, da sich die Zeilen ja in völlig verschiedenen Umgebungen (= Seiten) befinden. Weiterhin verhindert dieses Verhalten, daß beispielsweise der Anfang einer Seite eine spezielle typographische Behandlung erfahren kann: Soll auf der zweiten Seite eine Abbildung von Text umflossen werden, so wäre nach dem Umbruch des gesamten Absatzes (erste und zweite Seite) keine Möglichkeit mehr dazu. Auf das Problem des Umfließens mit Text komme ich noch zurück. Diese „asynchrone" Umbruchweise macht es auch unmöglich, unterschiedliche Absatzformen (je

nachdem wo der Absatz steht) zu realisieren. Es könnte ein Stil existieren, der verlangt, daß Absätze am Anfang einer Seite nicht eingerückt werden. Die Realisierung einer solchen Vorgabe ist momentan schwierig.

4. Die lokale Natur des Seitenumbruch-Prozesses: Für ein Schriftstück westlichen Zuschnitts ist der rechteckige Satzspiegel die logische Konsequenz. Man würde demnach gern die Möglichkeit haben sicherzustellen, daß verso-recto-Paare (nebeneinanderliegende Seiten) immer die gleiche Tiefe haben, selbst wenn diese Tiefe von Paar zu Paar um eine Zeile abweicht. So wie der jetzige Seitenumbruchalgorithmus von TEX Einfügungen (insertions) und Marken behandelt, ist diese Anforderung nur sehr schwer zu erfüllen. Der „lokale" Seitenumbruch erhöht das Risiko, daß die nächste Seite echt schrecklich wird, da kein „Vorausblick" auf die folgende Seite stattfindet. Man bräuchte einen Mechanismus, der es erlaubt, den Umbruch einer Seite durch die Qualität des Umbruchs der *folgenden* Seite beeinflussen zu lassen.

5. Die Analog-Eigenschaften des Klebstoffs (glue): TEXs fundamentale Weltanschauung, einen Text als ein Gebilde aus Boxen und Klebstoff dazwischen anzusehen, ist ein elegantes, wenn auch etwas vereinfachendes Modell einer Druckseite. Weil der Klebstoff dehnbar ist, ist es zumindest als allgemeine Anforderung unmöglich, Rastersatz (Layout auf einem rechteckigen Gitter, wie im Satzgewerbe üblich) zu bewerkstelligen. Das jetzige box-and-glue Modell könnte durchaus für den Rastersatz geeignet sein, müßte allerdings um eine Art „Schwerkraft" erweitert werden, die die Zeile (baseline) auf die nächste Gitterzeile zieht. Dabei müssen die Beschränkungen des Parameters `\lineskiplimit` berücksichtigt werden, d.h. eine Baseline darf nur nach oben rutschen, wenn sie dadurch nicht in die darüberliegende Zeile ragt.

6. Das Fehlen eines allgemein wirksamen Mechanismus für das Fließen von Text um Boxen. TEX hat nur ein sehr rudimentäres Werkzeug, um das Aussehen von Absätzen zu verändern (`\parshape`). Man könnte sich allerdings ein Kommando wie `\pageshape` oder `\spreadshape` vorstellen, das erlauben könnte, eine Seite oder ein Seitenpaar als Aneinanderreihung diskreter Bereiche zu definieren, in denen der Text frei fließen kann. Der Mechanismus sollte zumindest in einem Kernel-Makro implementiert werden. Er müßte dann die Interaktion von floating objects (Bildern und Tabellen, etc.) mit diesen Primitiven erlauben, um die gleiche Funktionalität zu erreichen, die bereits in Maus-orientierten Systemen anzutreffen ist.

7. Ein zu simples Modell einer Textzeile: Hat TEX einen Paragraph in Zeilen gebrochen, wird diese Zeile in eine `\hbox` eingepackt, deren Höhe die gesamte bounding-box dieser Zeile umfaßt. Wenn zwei Zeilen übereinander plaziert werden, ist der Mindestabstand zwischen den Zeilen durch `\lineskiplimit` definiert. Enthält die obere Zeile einen Buchstaben mit großer Unterlänge und die untere Zeile gleichzeitig einen mit großer Oberlänge, so wird der Zeilenabstand vergrößert, um die Vorgabe des `\lineskiplimit` zu erfüllen. Die Wahrscheinlichkeit, daß der tiefe Buchstabe der oberen Zeile und der hohe Buchstabe der unteren Zeile an derselben Position sind, ist aber sehr gering. Wenn TEX anstelle des jetzigen bounding-box-Modells ein sogenanntes Skyline-Modell einer Zeile verwenden würde, dann hätten solche Zeilen keinen anomal großen Zeilenabstand, es sei denn Buchstaben würden sich wirklich überlappen. Lassen Sie mich noch bemerken, daß TEX für diese Modifikation das genaue Aussehen eines jeden Buchstaben nicht kennen muß, das Modell einer bounding-box für jeden Buchstaben reicht durchaus aus, zumindest für diese Diskussion.

8. Nur teilweise Orthogonalität bei der Behandlung einzelner Objekte: TEX sieht eine halbwegs orthogonale Behandlung für viele seiner Objekte vor (z.B. die Kommandos der `\new...`-Familie), hat aber diese Art von Kommando nicht für alle. Es gibt keinen Befehl um z.B. neue `\marks` zu erzeugen. Man kann `\the` dazu benutzen, den Wert von vielen Parametern zu bestimmen, `\the` funktioniert aber bei der Bestimmung des Wertes von `\parshape` *nicht*. Es ist möglich, `\vbox` (oder `\vtop`) mit `\vsplit` zu zerlegen, aber es gibt kein analoges `\hsplit` um eine `\hbox` kleinzubekommen. Die Zerlegung beliebiger Listen ist unmöglich, weil es nur eine Untermenge der dazu nötigen `\last...`- und `\un...`-Operatoren gibt. Die implizite Multiplikation ohne einen Operator von `<nummer><dimen-or-skip-register>`, bei der ein `<dimen>` erzeugt wird, hat Anlaß zu vielen Mißverständnissen gegeben. Es wäre der Übersichtlichkeit zuträglich, wenn dieses Konzept verallgemeinert würde, so daß `<register>` ein `<register-type>` ergibt. Dieses Problem wirft viele andere Fragen im Zusammenhang mit den arithmetischen Möglichkeiten von TEX auf, die aber im Rahmen dieser Diskussion unwichtig sind. Zusammenfassend meine ich, daß die Konsistenz in diesem Bereich einer nachhaltigen Verbesserung bedarf.

9. Unzureichende Parametrisierung: TEX hat einen sehr ausführlichen Satz von Parametern, die den Satzprozeß steuern, aber diese Parameter sind nicht ausreichend.
   Es existiert der Parameter `\doublehyphendemerits`, ein Maß dafür, wie unerwünscht zwei aufeinanderfolgende Zeilen mit Trennungen sind. Wenn

man schon zwei aufeinanderfolgende Trennungen schlecht findet, dann wird man drei noch schlechter finden. Momentan gibt es aber keine Möglichkeit, darauf Einfluß zu nehmen. Ähnlich sieht es mit dem Befehl `\brokenpenalty` aus, der die Worttrennung mit einem (Zahlen-)Strafwert belegt. Doch ist es durchaus denkbar, daß man auf einer linken Seite gern einen anderen Wert hätte als auf einer rechten Seite, aber es existiert nur *eine* `\brokenpenalty`. Eine einfache, aber auf lange Sicht tödliche Lösung ist die Erhöhung der Anzahl der Parameter. Aber wann ist es dann genug? Flexibler wäre es, Ausdrücke für diese Parameter zuzulassen, z.B. `\brokenpenalty = {\ifrecto 500 \else 200 \fi}`, die verzögert ausgewertet werden.

10. Mangelndes Bewußtsein für Ästhetik: TEX kann in Punkto Ästhetik mit den besten heute bekannten Satzsystemen mithalten. Dennoch können die Ergebnisse bedeutend schlechter ausfallen als beim traditionellen Satz. Dies rührt von der zunehmenden Entkopplung vom Satzprozeß her: Der Mensch füttert jetzt nur noch den Computer, lehnt sich zurück und wartet auf die Resultate. Als man noch von Hand setzte, mußte sich der Setzer nicht nur der übergeordneten Struktur des Textes bewußt sein, sondern auch jeder Nuance, die er nur durch Prüfen der Muster und Formen auf der Seite sehen konnte. Flußläufe, (das sind mehr oder weniger offensichtliche Wortzwischenräume, die durch eine Seite mäandrieren), Wiederholungen (der gleiche Satz neben oder untereinander) und andere ästhetische Probleme sprangen dem Setzer förmlich ins Gesicht. TEX dagegen ist sich solcher Probleme nicht bewußt. Für die Problemlösung wäre es nötig, komplexe Mustererkennung und vielleicht sogar Ansätze von Bildverarbeitung in TEX einzubetten, bevor man in der Lage ist, den gleichen Ästhetikstandard zu erreichen, den der Bleisatz hatte.

Es ist klar, daß man diese „Mängelliste" fast beliebig verlängern könnte. Jedes System — unabhängig von der Komplexität — kann verbessert werden, TEX macht da keine Ausnahme. Ich habe mit voller Absicht Probleme wie gedrehten Text und Boxen, Graphikeinbindung und Farbunterstützung nicht angesprochen, da ich sie in der gegenwärtigen Diskussion für nicht relevant halte. Dies wären echte Erweiterungen von TEX, nicht Mängel, die es zu beseitigen gilt. Aber ich habe aufgezeigt, daß es Bereiche gibt, in denen man TEX verbessern könnte und dabei möchte ich es belassen.

Das bringt uns zum letzten Thema: Wie sollen wir weiter vorgehen? Der Ansatz der Diskussion (via `NTS-L`) ist offensichtlich hilfreich, insofern sie die Teilnahme der gesamten TEX-Gemeinde erlaubt, aber ich sehe dennoch zwei Probleme:

1. Diejenigen, die keinen Email-Anschluß haben, sind praktisch von der Diskussion ausgeschlossen (Knuth hat keine Email). Für dieses Problem gibt es keine einfache Lösung.

2. Die in der Diskussion geäußerten Meinungen sind so unterschiedlich, daß ich mich manchmal frage, ob wir uns jemals auf einen für alle akzeptablen Kompromiß einigen können.

Dieser zweite Punkt ist (abgesehen von Knuth, der keine Email hat) der wichtigere, denn wenn nur ein Teil der Benutzer die Entscheidung mitträgt, könnte sich die Benutzergemeinde spalten und verschiedene Wege gehen. Dies würde zur Verbreitung einer großen Zahl von Über-TEXs führen und einer entsprechenden Spaltung der Interessen. Die „natürliche Auslese" würde dafür sorgen, daß die Über-TEXs wieder von der Bildfläche verschwinden, aber ich mache mir Sorgen um die Auswirkungen einer solchen Entwicklung auf die TEX-Benutzer und TEX selbst. Wenn wir uns nämlich nicht einmal darauf verständigen können, ob es einen Nachfolger für TEX geben soll und welcher Funktionalität dieser Nachfolger bedarf, dann wird die Glaubwürdigkeit und das Ethos, das TEX jetzt genießt, in Frage gestellt. Mir wäre es lieber, wenn dies nicht geschähe.

Wir müssen also irgendwie eine allseits akzeptable Lösung finden. Mein Gefühl sagt mir, daß diese Lösung entweder eine konservative sein wird oder eine ganz radikale. Eine in der Mitte angesiedelte kann es aber nicht sein. Es mag scheinen, als ob ich schon bereit bin, Wetten abzuschließen, aber eines ist klar: Eine Kompromißlösung (die eierlegende Wollmilchsau), die allen Forderungen nachzukommen versucht, muß scheitern.

Wenn wir den konservativen Ansatz wählen, müssen wir zuerst identifizieren, was machbar und wünschenswert ist. Dazu brauchen wir dringend den Rat derer, die sich sehr gut mit `tex.web` auskennen. Ich sehe die konservative Vorgehensweise in der Modifikation von `tex.web` ohne ein Neuschreiben in irgendeinem Sinne.

Sollten wir hingegen die Radikalkur wählen, dann brauchen wir uns wahrscheinlich nicht mit `tex.web` zu beschäftigen — eine weitere Beschäftigung damit könnte sogar von Nachteil sein. Wollen wir wirklich ein *neues* Satzsystem haben, dann würde zu intime Kenntnis existierender Satzsysteme unsere Gedanken in zu enge Schranken weisen. Wir müssen dann vielmehr den Dialog mit Personen außerhalb der TEX-Welt suchen und uns des Rates erfahrener Typographen versichern. Aber vor allem brauchen wir dann Personen mit Vision und klarem Blick, Leute, die sich nicht durch technologische Trends beeinflussen lassen. Menschen, mit denen die Phantasie „durchgehen" kann.

Und zu welchen Ergebnissen könnte eine solche Gruppe dann kommen? Definitionsgemäß haben normale Sterbliche nicht die Vision, eine solche Frage zu beantworten; aber ich habe meine eigene Vision vom Satzsystem der Zukunft, die ich hier nur als Beispiel dafür wiedergebe, wie ein „New Typesetting System" aussehen könnte.

Erstens glaube ich (trotz meiner lächerlichen Vorbehalte gegenüber Window-Systemen), daß es eine Umgebung mit mehreren Fenstern benötigen wird oder es wird diese Fenster-Umgebung mitbringen. Dies bedeutet, daß es an kein bestimmtes Betriebssystem gebunden ist, sondern seine Systemaufrufe mithilfe einer wohldefinierten Schnittstelle macht. Verfügt der Host-Rechner über eine Fensterumgebung wie MS-Windows oder X, so wird NTS diese nutzen, sollte kein Fenstersystem zur Verfügung stehen, ist der Implementor für die Bereitstellung dieser Funktionalität verantwortlich. Ich kann mir vorstellen, daß vielleicht bis zu acht solcher Fenster gleichzeitig gebraucht werden. Gelinkte Grafik und Textfenster, die I/O-fähig sind, mit denen der Benutzer das Layout des Textes festlegen und gleichzeitig im Grafikfenster das Resultat seines Vorgehens beobachten kann. Ein algorithmisches Fenster, in dem der Benutzer definiert, wie Satzentscheidungen getroffen werden. Ein weiteres kann dann interaktiv Entscheidungen vom Benutzer verlangen, wenn das System Hilfe benötigt, weil z.B. eine besonders schwierige Abbildung nicht zufriedenstellend auf der Seite plaziert werden kann. Und ein weiteres Fenster, in dem der Benutzer verfolgen kann, wie das System die Satzentscheidungen trifft, ohne andere Interaktionsfenster mit Meldungen vollzuschreiben. Es ist klar, daß dieses System ein Echtzeitsystem sein muß, d.h. Änderungen im Layout und im Text werden sofort in allen Fenstern wirksam. Ich gehe auch davon aus, daß in den Fenstern auch andere Programme laufen können, die nicht Teil von NTS sind und damit zum Beispiel jeder seinen Lieblingstext- oder Grafikeditor benutzen kann. Natürlich wird nicht jeder Benutzer alle Fenster benötigen. Die Benutzer, die vordefinierte Designs (d.h. style files) einsetzen, brauchen nicht die Erstell-, Trace- und Debug-Fenster für Designentwickler. Sie werden aber sicherlich die Interaktionsfenster zur Eingabe, Setzen und Previewen des Textes verwenden. Aus diesen Gründen wird dieses System auch in der Lage sein, sämtliche Designs (Stile) und Dokumente, die mit diesen Stilformaten erzeugt wurden, als Text zu exportieren, zum Import in andere Systeme.

Und was ist die Basis, die Grundlage dieses Systems? Vielleicht nur eine stark verbesserte Version des TEX-Prozessors; völlig umgeschrieben, vielleicht mit einer prozeduralen Sprache anstatt einer Makrosprache. Warum Prozeduralsprache? Um eine maximale Akzeptanz des neuen Systems zu erreichen. Es gibt viele Menschen, die sich mit einer Prozeduralsprache erheblich leichter tun,

als mit den anderen Arten von Programmiersprachen. Was noch? Zumindest mit den gleichen Verbesserungen, die man für die konservative Interimslösung implementiert hätte, mit einer Unterstützung für Farbgrafik, Rotation von Elementen und so weiter. Das Ganze wird natürlich ein Musterbeispiel für „literate programming", es wird ein Public Domain Programm sein, es wird alte `DVI`-files ebenso schreiben können wie ein erweitertes `DVI`-file-format oder eine PostScript-Datei. Und es wird so fehlerfrei sein, daß sich seine Schöpfer leisten können, eine Belohnung für das Auffinden von Fehlern auszusetzen, die sich bei jedem gefundenen Fehler verdoppelt...

Aber wir brauchen noch ein letztes Element und ich habe es mir absichtlich für den Schluß aufgehoben: Wir brauchen den Rat von Don Knuth selbst. Don hat sich vom TEX-Projekt distanziert, um sich wieder ganz der „*Art of Computer Programming*" zuzuwenden. Seine Distanz ist verständlich — TEX hat immerhin einen wesentlichen Teil seines Arbeits- und wohl auch Privatlebens in Anspruch genommen — und ich hoffe, daß wir alle seinen Wunsch respektieren, sich wieder der Informatik, der Mathematik und dem Studium der Bibel zuzuwenden. Aber ich halte es für undenkbar, seinen Rat zu ignorieren und wenn ich einen Wunsch hätte, wäre es dieser: daß ich mich mit ihm treffen könnte, wann immer er das Gefühl hat, er hätte Zeit übrig, um das gesamte NTS-Projekt zu besprechen. Vor allem würde ich gern wissen, was er an TEX ändern würde, wenn er es *heute* entwickeln müßte. Ich wüßte gern, ob er meine Meinung teilt, daß die Mängel von TEX, die ich aufgezählt habe, echte Mängel sind oder einfach Folge der Tatsache, daß wir die wahren Fähigkeiten von TEX nicht erkannt haben, wie ich manchmal befürchte. Und wie er zur Idee des erweiterten TEX und des „New Typesetting System" steht (ich habe den Verdacht, er wäre mehr für das zweite als das erste). Und wenn ich dann ehrlich bin, würde ich zu ihm sagen: „Danke Don", für die zahllosen Stunden, Tage, Wochen, Monate und wahrscheinlich Jahre der Freude, die ich an TEX bisher gehabt habe.

Epilog

Das vorangegangene ist genaugenommen die Zusammenfassung eines Vortrages, den ich 1992 in Prag gehalten habe; jetzt folgt ein Nachtrag, der den Leser auf den Stand der Dinge (Mai 1993) bringen soll.

Während des Treffens von DANTE e.V. 1992 kündigte Joachim Lammarsch die Bildung einer Arbeitsgruppe namens „NTS" (für New Typesetting System) an. Diese Gruppe hat sich unter Vorsitz von Rainer Schöpf zum Ziel gesetzt, Mittel

und Wege zu finden, die Philosophie und Paradigmen von TEX zu erhalten und zu verbessern.

Neben Rainer Schöpf waren DANTE e.V. und die britische Benutzergruppe UK-TUG vertreten. Der Ausschuß war international besetzt und die Teilnahme nicht (obwohl von DANTE e.V. organisiert) auf deren Mitglieder beschränkt. Im folgenden Jahr beteiligten sich die Mitglieder der NTS-Gruppe an lebhaften Diskussionen der `nts-l`-email Liste.

Bis zur DANTE'93 in Chemnitz mußte Rainer Schöpf erkennen, daß er wegen anderer Aufgaben, z.B. der Arbeit an LATEX3 (aber auch wegen des „eigentlichen" Berufes...) zeitlich nicht mehr in der Lage war, größere Aktionen im Rahmen des NTS-Projektes zu „pushen" und er fragte Phil Taylor (meine Wenigkeit), ob er bereit sei, den Vorsitz zu übernehmen. Die Auflösung der alten NTS-Gruppe und die Bildung der neuen unter meiner Ægide wurden in Chemnitz angekündigt. Zu dieser Zeit war ich auch das einzige Mitglied, mit dem Auftrag andere Interessierte zu berufen.

Das vielleicht größte Problem der NTS-Gruppe ist der Mangel an Vertrauen an der „Basis". Obwohl die Gruppe seit mehr als einem Jahr existiert, hat sie außer Diskussionen nichts bewegt. Aus diesem Grund muß sie, wenn NTS mehr als ein Luftschloß sein soll, innerhalb des folgenden Jahres etwas zustande bringen. Ginge ein weiteres Jahr ohne *sichtbare* Erfolge ins Land, dann wird das Vertrauen in diese Gruppe zweifellos extrem leiden. Mir ist es so wichtig, daß dies nicht geschieht, daß ich nun bereit bin, bei einigen der Ideale, die ich im Prag-Teil dieses Aufsatzes ausgeführt habe, Abstriche in Kauf zu nehmen und mich auf die zweite Option festzulegen. TEX wird soweit geändert, daß es für professionelle Anwender keine der Schwächen mehr hat, die in der jetzigen Implementierungen behoben werden können. Dieses verbesserte TEX soll e-TEX (enhanced) heißen. Als Vertrauensbeweis in diesen Ansatz schlage ich deshalb vor, diese Änderungen in diskreten Schritten zu vollziehen. Das erste Release soll in ca. 12 Monaten realisiert sein und weitere würden in regelmäßigen Zeitintervallen (sagen wir 6 Monate) folgen.

Dies läßt sich in den folgenden „12 Geboten" zusammenfassen. Sie wurden von Diskussionen mit Chris Rowley, Rainer Schöpf und Joachim Schrod beeinflußt.

1. TEX hat offensichtliche Schwächen, teilweise wegen seines Alters (1982) und weil einige Ansätze nicht konsequent bis zum bitteren Ende realisiert wurden (s.o.).

2. Die Stärken überwiegen die Schwächen. TEX ist weit verbreitet und die Anhänger sind „fanatisch".

3. Um die große Benutzerbasis beibehalten zu können, muß der Nachfolger von TEX genauso portabel sein, aber besser und zugleich noch rückwärts kompatibel zu TEX-$\approx \pi$.

4. Es ist unwahrscheinlich, daß der intellektuelle Kraftakt, ein komplett neues Satzsystem zu entwickeln, in einem endlichen Zeitraum durch ein paar TEX-„freaks" zu erbringen ist, die in ihrer Freizeit daran arbeiten. Ein solches Projekt ist vielmehr (wie das ursprüngliche TEX-Projekt in Stanford) ein echtes Forschungsprojekt, das einige Jahre dauert.

5. Das „alte" NTS-Projekt hat dadurch, daß effektiv nichts geschehen ist, viel an Vertrauen verloren.

6. Wenn das „neue" NTS-Projekt erfolgreicher sein soll, muß es nicht nur etwas Sinnvolles tun, sondern muß auch *zeigen*, daß es etwas Sinnvolles tut.

7. Um allgemein akzeptiert zu werden (das mag ein frommer Wunsch sein, aber wir sollten nicht weniger anstreben), muß NTS voll kompatibel zu bisherigen TEX-Implementationen sein, nicht aber bezüglich der Eingabedateien.

8. Es muß deshalb:

   (a) als change-file für das jetzige `tex.web` implementiert und nicht von Grund auf neu geschrieben sein.

   (b) voll rückwärtskompatibel sein (identisches Verhalten, identischer Output) unabhängig davon, ob TEX oder extended TEX (e-TEX) benutzt wird.

9. Diese zwei Anforderungen reichen natürlich nicht aus: Die Funktionalität muß erweitert werden, dabei darf die Philosophie von TEX möglichst nicht verletzt werden. Fragestellungen wie z.B. die Einbindung von Grafik würden diesen Rahmen sprengen und sollten in diesem Zusammenhang nicht weiter verfolgt werden.

10. Ein klarer Mangel von TEX ist das Fehlen eines Interfaces zum Betriebssystem. Eine lange Diskussion entbrannte darüber, wie diese Funktionalität durch Erweiterung von Befehlen wie `\open{in|out}`, `\read`, `\write` etc. hinzugefügt werden könnte. Diese Ergänzungen haben überraschenderweise den Segen von Don Knuth.

11. Einige (u.a. ich) sind der Auffassung, daß die Ergänzung der jetzigen TEX-Semantik ein Sakrileg ist; eine schlampige Zwischenlösung an einer Stelle, wo eigentlich eine saubere Lösung von Grund auf nötig wäre.

12. Die neuerlich wiedergegründete NTS-Gruppe soll sich deshalb primär mit der Identifizierung von „echten" Schwächen von TEX befassen, dann Prioritäten für deren Beseitigung vergeben und einen Zeitplan für die schrittweise Erstellung von Change-files festlegen. Dann kann durch gründliches α- und β-Testen in unterschiedlichsten Rechnerumgebungen verhindert werden, daß sich neue Fehler in den e-TEX-Code einschleichen.

Die neue NTS-Gruppe soll also *keine* esoterische Forschungsgemeinschaft in Sachen Computersatz des 21. Jahrhunderts sein, sondern die realen Probleme benennen, die auftreten, wenn man TEX bis an seine Grenzen strapaziert. Sind diese identifiziert, wird die NTS-Gruppe diese Probleme schrittweise mit Hilfe eines master-change-files lösen. Dabei wird sie mit den Änderungen beginnen, die bei kleinstem Aufwand die größte Wirkung haben. Durch die Benutzung von Werkzeugen wie z.B. TIE, WEB-Merge und Patch-WEB sollte die Gruppe die Entwicklung und das Testen von Plattform-spezifischen TEX-Implementationen unterstützen. Wenn dann sichergestellt ist, daß sich keine Fehler eingeschlichen haben und der neue Code Ergebnisse erzielt, die identisch mit denen des „alten" sind, dann erst kann e-TEX auf den Rest der TEX-Welt losgelassen werden. Die Akzeptanz von e-TEX muß dann genau beobachtet werden. Fehlt sie trotz Rückwärtskompatibilität zum alten TEX, dann wird die NTS-Gruppe wohl ihre Arbeit einstellen. Das wäre zwar eine Schande, aber immer noch besser als viel Anstrengung und Arbeit an ein Projekt zu verschwenden, an dem sowieso niemand interessiert ist. Wenn das Echo auf e-TEX andererseits positiv ist und die TEX-Benutzer froh sind von TEX auf e-TEX umstellen zu können, dann muß die NTS-Gruppe den Dialog mit der TEX-Welt weiter gezielt ausbauen, um die Fehler, die e-TEX dann noch hat, zu identifizieren und mittelfristig auszumerzen.

So hoffe ich, daß die NTS-Gruppe auf diesem Wege etwas Sinnvolles für die TEX-Gemeinde tut, aber andererseits dies auch *glaubhaft machen* kann.

Philip Taylor, RHBNC
Koordinator, NTS-Projekt
Chemnitz, 1993

Literatur

[1] Frank Mittelbach: *E-TEX: Guidelines for future TEX*, in *TUGboat*,

Vol. 11, No. 3, pp. 337–345, September 1990.

[2] Michael Vulis: *Should TEX be extended?*, in *TUGboat*, Vol. 12, No. 3, pp. 442–447, September 1991.

Bretter, die die Welt bedeuten

Grafik-Import in LaTeX

Dr. Heinz Werntges

Der folgende Artikel erschien zuerst in der Computerzeitschrift c't 12/92, S. 252–258. Der Abdruck erfolgt mit freundlicher Genehmigung der Verlages. Aktualisierende Korrekturen wurden bei den Literaturangaben vorgenommen.

Zusammenfassung

TeX/LaTeX ist hervorragend geeignet zur Erstellung von technisch-wissenschaftlichen Dokumenten wie Studien- und Diplomarbeiten, Dissertationen und Veröffentlichungen — solange es sich um Text, Formeln und Tabellen handelt. Sollen aber Grafiken und Abbildungen nicht nur eingeklebt, sondern Teil des TeX-Dokuments werden, finden die Anwender statt systematischer Erklärungen nur zu oft ein Dickicht aus sich teilweise ausschließenden Tips und Tricks vor. Dabei gibt es inzwischen einige sehr hilfreiche Programme zur Grafikintegration.

Einleitung

Es ist schon verwunderlich: Zwar sind neue LaTeX-Anwender trotz der zunächst unübersichtlich wirkenden Fülle von TeX/LaTeX-Kommandos rasch in der Lage, auch komplizierte Formeln und Tabellen zu generieren, aber wenn sie dann etwas scheinbar so „Einfaches" wie eine Grafik im TIFF-Format einbauen möchten, gibt's erst mal lange Gesichter, denn für TeX sind Grafiken unbekannt.

Wen der daraufhin meist einsetzende Spott aus dem WYSIWYG-Lager nicht abschreckt, sondern wer diese Bit-Pfriemler gelassen ihre Formeln, Querverweise, Inhaltsverzeichnisse, Literaturhinweise usw. auf die mühsame Tour basteln und ständig aktualisieren läßt und auf Auswege sinnt, wird meist erfahren, daß er sich vor Möglichkeiten kaum retten kann, aber auch (meist viel später), daß jede dieser Methoden zur Grafikerzeugung in LaTeX ihre Grenzen hat. Dieser Artikel erläutert die Grundlagen zur Grafikgenerierung mit LaTeX, zeigt die wichtigsten Methoden anhand von Beispielen auf und diskutiert ihre Vor- und Nachteile.

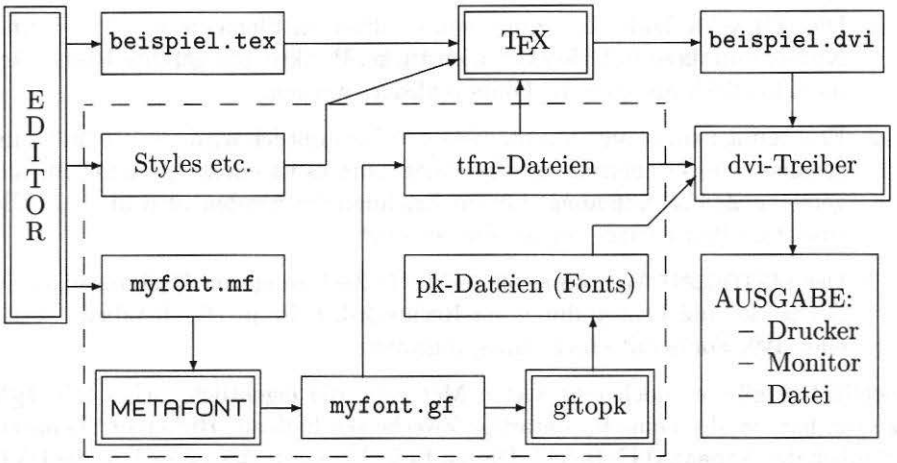

Abbildung 1: Das TEX-System im Überblick: Die Teile im gestrichelten Rahmen werden nur selten benutzt. Diese Zeichnung ist gleichzeitig ein gutes Beispiel für die eingebauten LaTeX-Grafikmöglichkeiten: Mit Hilfe von TEXcad entstand sie innerhalb von etwa einer halben Stunde.

Überblick

Um die verschiedenen Methoden zur Realisierung von Grafiken in TEX zu verstehen, ist eine Zusammenfassung der Funktionsweise des gesamten TEX-Systems hilfreich (Abb. 1):

Eine TEX-Quelldatei wird von TEX, dem Satzprogramm, bzw. LaTeX, dem darauf aufbauenden Layout-Makropaket, eigentlich nur in viele kleine Rechtecke verwandelt und zu Zeilen und schließlich zu Seiten montiert. Dazu muß TEX die Größe und den Bezugspunkt dieser Rechtecke kennen, was in den TFM-Dateien steht, nicht aber deren Inhalte. Diese Inhalte, also die zahlreichen Font-Dateien, werden erst von den (gerätespezifischen) DVI-Treibern verwendet. Wer schon einmal mit einem Previewer eine DVI-Datei anzeigen ließ, ohne alle Font-Dateien installiert zu haben, kennt die erwähnten Rechtecke, die dann anstelle der fehlenden Zeichen erscheinen.

Was hat dies mit Grafiken zu tun? Nun, da TEX die Inhalte dieser Rechtecke nicht kennen muß, kann man Grafiken durch folgende Methoden erzeugen:

1. Die LaTeX-Methode: Zerlegung von Grafiken in Elemente wie Linien und Kreise, und diese in kleine Geradenstücke, Punkte und gefüllte Rechtecke, die schließlich als spezielle Fonts realisiert werden.
2. Erzeugung neuer Fonts mit METAFONT. Bekanntlich wird METAFONT zum „Malen" von Zeichensätzen verwendet, aber es ist ebenso geeignet für allgemeine Zeichenaufgaben. Ganze Zeichnungen werden also als spezielle (meist großformatige) Buchstaben erzeugt.
3. Der METAFONT-Weg ohne METAFONT: Zerlegung von Rastergrafiken in Rechtecke und Umwandlung der Rechteck-Inhalte in „Buchstaben" spezieller TeX-Fonts mit einem Hilfsprogramm.

Schließlich gibt es noch eine vierte Methode, die eigentlich nichts mit TeX zu tun hat, sondern eine für derartige Zwecke geschaffene „Hintertür" benutzt, nämlich den `\special{}`-Befehl. Diesen Befehl kopiert TeX einfach in die DVI-Datei und überläßt es den DVI-Treibern, ihn zu interpretieren.

Diese vier Methoden werden nun auf ihre Eignung

- zur Erzeugung von Zeichnungen,
- zum Einbinden vorhandener Rastergrafiken
- und zum Einbinden von Vektorgrafiken

geprüft und bewertet, wobei der Akzent auf den letzten beiden Punkten liegt.

Zeichnen mit LaTeX & Co.

Bekanntlich [11] bietet LaTeX Makros zur Erzeugung einfacher Zeichnungen. Abbildung 1 ist mit dieser Technik generiert worden. Die Einbindung in den TeX-Quelltext (Abb. 2) ist denkbar einfach.

Die erwähnte Beschränkung von TeX auf das Setzen von Rechtecken hat hier zur Konsequenz, daß nur senkrechte und waagerechte Linien ohne Einschränkungen generiert werden können. Diagonal verlaufende Linien werden durch Überdeckung mit kleinen Rechtecken realisiert, deren Inhalte (kurze Geradenstücke) wie Buchstaben, also als Fontelemente, behandelt werden. Da nicht beliebig viele Fonts existieren, können Geraden nur in wenigen diskreten Steigungen dargestellt werden. Ähnliche Überlegungen führen zu Einschränkungen bei der Realisierbarkeit von Kreisen etc.

Schon bald entstanden „Styles" mit leistungsfähigen Makros, die durch Setzen vieler kleiner Punkte bzw. winziger Rechtecke realisiert wurden. Dazu gehören

```
\begin{figure}
   \centering
   \input{tex_sys.pic}
   \caption[]{Das \TeX{}-System ... }
\end{figure}
```

Abbildung 2: So einfach wird die -Datei `tex_sys.pic` aus Abb. 1 in den Text eingebunden. Alle Größenangaben, Positionierungen und Platzreservierungen sind bereits berücksichtigt.

Makropakete wie `epic.sty` und `bezier.sty` ebenso wie das sehr umfangreiche PiCTeX-Paket [1].

Alle diese Methoden besitzen gravierende Nachteile: Sie

- kosten sehr viel TeX-Rechenzeit,
- verursachen Kapazitätsprobleme von TeX,
- sind für extern generierte Grafiken schlecht geeignet,
- bieten nicht immer überzeugende Bildqualität
- und sie sind umständlich zu handhaben.

Auch wenn der letzte Punkt seit Verbreitung von CAD-artigen PD-Programmen zur Erzeugung von LaTeX-Grafik-Quellcode wie `xfig` [2] oder `TeXcad` [3] etwas entkräftet wurde — selbst bei relativ schlichten Grafiken wird TeX einfach überfordert, denn es wurde zum Setzen von Buchstaben konzipiert und von denen passen in der Regel bedeutend weniger auf eine Seite als die Tausende kleiner Punkte, aus denen die o.g. Makros eine Zeichnung aufbauen. Zudem muß TeX eventuell notwendige Fließkommaberechnungen umständlich emulieren.

Die hier genannten Techniken beruhen alle auf Methode 1. Wer den großen Vorteil dieser Methode, nämlich Portabilität der Grafikerzeugung durch Beschränkung auf reine TeX-Features, nicht missen mag, findet ausreichende Dokumentation in den angegebenen Quellen, so daß hier auf weitere Beispiele verzichtet werden kann. Potentielle Anwender(innen) sollten aber ein BigTeX und einen schnellen Rechner benutzen, sonst sind die Grenzen nur allzu früh erreicht.

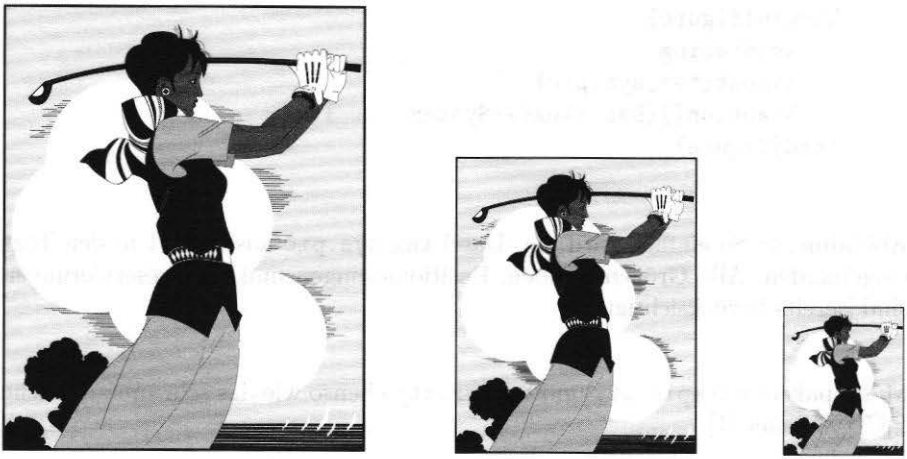

Abbildung 3: Das Beispiel der Datei `golfer.ps` aus dem GhostScript-Paket zeigt eine Stärke des PostScript-Weges: Einfache Bildgrößenanpassung.

Der Weg über PostScript

Manche TeX-Fans schauten in der Vergangenheit neidisch auf die PostScript-Anwender, und mit gutem Grund:

1. PostScript (PS) besitzt alle Sprachelemente zur Erstellung von Abbildungen, auch von sehr komplexen.
2. PS-Abbildungen sind geräte- und größenunabhängig definierbar.
3. Eine Vielzahl von Programmen beherrscht heute die Ausgabe im PS-Format.
4. Mit GhostScript [4] ist ein leistungsfähiges PD-Programm verfügbar, mit dem PS-Ausgabe auch auf *low-cost* Systemen realisierbar ist.
5. Schließlich gibt es einen eingebauten Mechanismus (über *encapsulated PostScript*/EPS) für den Import extern generierter PS-Dateien wie Grafiken in andere PS-Dokumente.

Seit Erscheinen des ausgezeichneten PD-Programms `dvips` [5], das DVI-Dateien in PS-Dateien wandelt, stehen den TeX-Anwendern diese Vorteile von PostScript ebenfalls zur Verfügung. `dvips` importiert Grafiken, indem es EPS-Dateien über Methode 4, den `\special{}`-Befehl, einliest.

```
\begin{figure}[htb]
  \mbox{\epsfysize=60mm\epsffile{golfer.eps}}\hfill%
  \mbox{\epsfysize=40mm\epsffile{golfer.eps}}\hfill%
  \mbox{\epsfysize=20mm\epsffile{golfer.eps}}
  \caption[]{Das Beispiel ...}
\end{figure}
```

Abbildung 4: Der TeX-Quelltext zur Abb. 3. Entweder die Bildbreite *oder* die Bildhöhe können (mit `\epsfxsize` bzw. `\epsfysize`) gewählt werden. Die jeweils andere Abmessung berechnet TeX aus dem Seitenverhältnis der Abbildung laut *bounding box* in der EPS-Datei.

Im `epsf.sty` werden TeX-Befehle definiert, mit denen das Einbinden und die Größenanpassung von Abbildungen sehr einfach möglich sind (Abb. 4). TeX erhält die natürliche Bildgröße durch Suche in der EPS-Datei nach der sogenannten *bounding box*, die in der Nähe des Dateianfangs stehen sollte. Dadurch ist eine automatische Platzreservierung für die Abbildung durch TeX kein Problem.

Wenn auch oft die Meinung vertreten wird, daß dem Weg über PostScript die Zukunft gehört — einen Schönheitsfehler hat er: Die Grafiken sind im DVI-Preview nicht sichtbar. Kein Wunder, denn ein DVI-Previewer, der das können sollte, müßte gleichzeitig ein vollwertiger PostScript-Interpreter sein! In der Praxis ist dieses Manko ziemlich lästig, denn der Entwicklungszyklus „Editieren ⇝ TeX-Lauf ⇝ DVI-Preview ⇝ Drucken" verlängert sich zu „Editieren ⇝ TeX-Lauf ⇝ `dvips`-Lauf ⇝ PS-Preview ⇝ PS-Druck", und PS-Previewer sind wesentlich langsamer und unhandlicher als DVI-Previewer. Wer sich z.B. an das mühelose und rasche Blättern, Suchen und Zoomen mit `dviscr` aus dem emTeX-Paket [6] gewöhnt hat, wird das strikt seriell ablaufende, nur ganze Seiten im Überblick anzeigende Previewing mit GhostScript nicht akzeptieren. Selbst an einer schnellen UNIX-Workstation empfinde ich das PS-Previewing als unbefriedigend und nur für letzte Kontrolle vor dem Ausdruck geeignet.

Einbau von Rastergrafiken ...

Noch häufiger als EPS-Dateien begegnen uns Rastergrafik-Dateien, zum Beispiel im BMP-, PCX-, TIFF-, GIF- oder einem der vielen anderen Formate. Sie stammen aus sehr verschiedenen Quellen wie etwa von Bildschirmkopien, Scannern oder Mal- und Zeichenprogrammen und können von einfachen s/w-

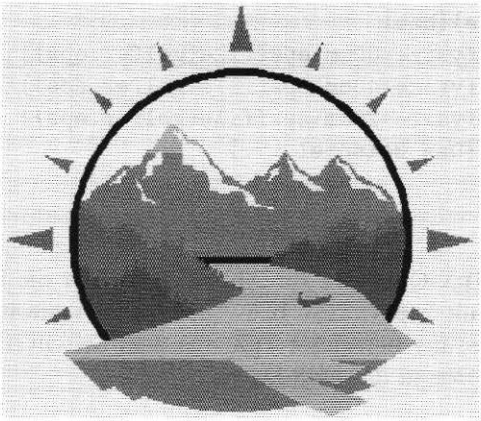

Abbildung 5: Eine BMP-Farbvorlage, mit `bm2font` gedithert und als Font (also preview-fähig) in TeX eingebaut nach Methode 3.

Daten bis zu aufwendigen *true color*-Daten (etwa: Scans von Farbfotos) sehr verschiedene Wiedergabequalitäten erfordern.

... durch Fonterzeugung ...

Bis vor etwa zwei Jahren gab es keine Hilfsmittel, um diese vielleicht wichtigste Klasse von Abbildungen in TeX zu importieren, aber inzwischen sind zumindest zwei PD-Programme verfügbar. Beide arbeiten nach Methode 3, sie erzeugen also Font-Dateien, die ja auch nicht viel mehr sind als Instanzen spezieller Rastergrafik-Formate. Zusätzlich teilen sie TeX die Abmessungen der Zeichnungselemente über TFM-Dateien mit. Dieser Weg ist der von METAFONT selbst vorgezeichnete, sozusagen der „kanonische". Er gewährleistet die Portabilität der Ergebnisse und fügt sich nahtlos in die übrigen Elemente des TeX-Systems ein.

Das Programm `rumgraph` [7] beschränkt sich auf Schwarz/Weiß-Dateien im ADI- oder PCX-Format und generiert TFM- und PXL-Dateien. In TeX aktiviert man an geeigneter Stelle einen neuen Font und setzt z.B. einen bestimmten Buchstaben aus diesem Font an die gewünschte Stelle — fertig, denn der „Buchstabe" ist tatsächlich die Zeichnung. Wir wollen hier kein Beispiel angeben, da inzwischen mit `bm2font` [13, 8] ein noch flexibler einsetzbares PD-Programm vorliegt.

```
\input{bach.tex}
\begin{figure}[htb]
  \centering
  \setbach
  \caption[]{Eine BMP-Farbvorlage ...}
\end{figure}
```

Abbildung 6: Der TeX-Quelltext zur Abb. 5. Alle Arbeit erledigt das automatisch generierte Macro `\setbach`. Die Zeichnung wird wie eine `\parbox` behandelt und ist frei positionierbar.

`bm2font` versteht die meisten gängigen Rastergrafik-Formate, verarbeitet auch Grauwert- und Farbbilder, beherrscht ein gut entwickeltes und über mehrere Parameter anpaßbares Dithering und erzeugt Fonts im besser komprimierten PK- statt im alten PXL-Format. Im Unterschied zu `rumgraph` zerlegt `bm2font` Rastergrafiken in viele kleine statt in wenige große Rechtecke, damit auch unvorsichtig programmierte DVI-Treiber garantiert damit zurechtkommen — aber keine Sorge: Für jede Abbildung erzeugt `bm2font` automatisch eine TeX-Datei mit Makros, die die Rechtecke, also die „Buchstaben" der generierten Fonts, wieder zum Gesamtbild zusammensetzen. Abbildung 6 zeigt, was in einer LaTeX-Datei zu tun ist, um Abbildung 5 zu erzeugen. Wer Einzelheiten zu diesem Weg wissen möchte, kommt an der Lektüre des Manuals von `bm2font` nicht vorbei. Es enthält viele Beispiel-Grafiken und natürlich die Beschreibung der zahlreichen Optionen.

Bietet der Einsatz von `bm2font` nur Vorteile? Die Kombination vieler guter Eigenschaften wie Preview-Fähigkeit der Abbildungen unter Beibehaltung der Wahl des Ausgabeformates (incl. PostScript), Portabilität der Ergebnisse (von `bm2font` erzeugte Dateien sind normale Font- bzw. Style-Dateien), Verfügbarkeit des C-Quellcodes, Unterstützung bei der Anpassung der Bildgröße und -auflösung ist schon recht attraktiv. Nachteilig sind nur gewisse Details in der Anwendung:

1. Die Benutzung eines separaten Programms ist lästiger als eine schlichte Dateispezifikation innerhalb der TeX-Quelle.
2. TeX ist für Einsteiger schon kompliziert genug. Den Umgang mit Fonts und Utilities möchten nicht alle Anwender auch noch erlernen müssen.

3. Nicht alle Namen von Grafikdateien sind geeignet: Weil `bm2font` z.B. aus `foo.pcx` das TeX-Kommando `\setfoo` baut, beschwert sich TeX, wenn statt „foo“ etwa „bild_1a“ eingesetzt wird: _ ist in TeX ein reserviertes Zeichen und Ziffern sind Terminatoren in Kommandonamen. Ferner verfügt `bm2font` über den achten Buchstaben des Dateinamens, um mehrere Fonts pro Datei benennen zu können. In solchen Fällen wird ein zulässiger Ersatzname per `bm2font`-Option angegeben.

4. In Rechnernetzen ist die Installation eigener Fonts nicht immer zulässig. Lokale Installation und moderne Treiber mit Suchpfadunterstützung für Fonts helfen da weiter.

5. In Dokumenten mit sehr vielen Abbildungen kann die Maximalzahl erlaubter Fonts überschritten werden. Dann hilft aber ein Zerlegen des Dokuments weiter, z.B. unter Verwendung von `\include`.

Zudem erzeugt `bm2font` oft mehrere Fonts pro Abbildung, was nicht immer notwendig ist, so daß die maximale Fontanzahl noch schneller erreicht wird. Der Autor von `bm2font`, Friedhelm Sowa, will diesen Punkt bald (durch Unterstützung virtueller Fonts) verbessern. Die Maximalzahl möglicher Abbildungs-Fonts hängt von der TeX-Implementation und von der Zahl der für andere Zwecke bereits vergebenen Fonts ab. In der Regel wird dieses Limit aber nur bei sehr großen Dokumenten erreicht.

... *oder „Spezial“-Lösungen*

Einfachere, aber weniger allgemeingültige Methoden zum Import von Rastergrafiken sind ebenfalls verfügbar. Sie verwenden den `\special{}`-Befehl (Methode 4), überlassen die Arbeit also den DVI-Treibern. Da es keine standardisierten `\special{}`-Befehle gibt, bietet sich deren Einsatz nur in einer hinreichend homogenen Umgebung an. Beispielsweise kann das Ausdrucken einer daheim auf einem PC geschriebenen Semesterarbeit auf dem Instituts-Laserdrucker zum Fehlen der Grafiken führen, wenn dieser über andere DVI-Treiber bedient wird.

Wegen ihrer großen Verbreitung sollen hier die DOS- und OS/2-basierten emTeX-Treiber als Beispiel dienen. Sie können Rastergrafiken direkt einlesen, beherrschen aber nur wenige Formate und sind am besten für s/w-Vorlagen geeignet. Das Kommando `\special{em:graph foo.bmp}` veranlaßt den emTeX-DVI-Treiber, am momentan gültigen Bezugspunkt die Grafik der TeX-Seite zu überlagern. Die Wahl des Bezugspunktes und die Platzreservierung ist manuell zu besorgen, da TeX vollständig umgangen wird. Das Vorgehen entspricht ei-

```
\unitlength1mm
\begin{figure}[htb]
   \centering
   \begin{picture}(100,50)
     \put(0,50)
      {\special{em:graph foo.pcx}}
   \end{picture}
   \caption[]{...}
\end{figure}
```

Abbildung 7: Die Verwendung von `\special{em:graph}` erfordert eine explizite Platzreservierung für die Grafik. Im Beispiel wird angenommen, daß die Datei `foo.pcx` bei der gegebenen Auflösung, z.B. 300 dpi, 100 mm breit und 50 mm hoch ist. Der Bezugspunkt der emTeX-Treiber ist die *obere* linke Ecke des Bildes. Mit `\put` wird die Grafik an den korrekten Ort gesetzt.

ner elektronischen Variante des Einklebens von Abbildungen. Abb. 7 zeigt ein Quellcode-Beispiel und das Ergebnis.

Wer keinen Wert auf Portabilität legen muß und nur s/w-Vorlagen in der bereits korrekten Größe und Auflösung verwendet, hat mit diesem Weg eine schnelle und einfach hantierbare Methode des Rastergrafik-Imports zur Verfügung. Problematisch wird's natürlich, wenn

- Drucker- und Bildschirmtreiber verschiedene Auflösungen verwenden, denn die Grafik wird nur in einem der Fälle korrekt sein,
- die Grafik nicht in der gewünschten Größe bzw. Auflösung vorliegt,
- oder wenn eine spätere PS-Ausgabe nicht ausgeschlossen werden kann (`dvips` unterstützt zwar viele emTeX-specials, nicht aber `\special{em:graph}`).

Externe Vektorgrafiken

Im technisch-wissenschaftlichen Umfeld, also der „Heimat" von TeX, spielen Diagramme aller Art eine große Rolle. Im Unterschied zu Präsentationsgrafiken ist Hinterlegen mit Grauwert- oder Farbmustern relativ selten gefragt, so daß diese Diagramme am besten als Vektorgrafiken repräsentiert werden. Sie bleiben dadurch skalierbar und auflösungsunabhängig.

Viele dieser Diagramme werden heute durch PostScript realisiert. Da die Verarbeitung von PS-Dateien bereits besprochen wurde, wollen wir uns hier der Integration eines anderen verbreiteten Grafik-Formats widmen, Hewlett-Packards Plottersprache HP-GL [12].

HPGL besitzt elementare `move-` und `draw`-Kommandos, aber auch viel leistungsfähigere Befehle wie etwa für String-Ausgabe, für das Zeichnen von Kreisen und für die Schraffur ganzer Polygonflächen. Der erste Verarbeitungsschritt besteht in der Wandlung dieser abstrakteren Kommandos in Elementarkommandos.

Diese `move-` und `draw`-Anweisungen können dann auf folgende Weisen in TeX-Abbildungen gewandelt werden:

1. Durch Umcodierung in die METAFONT-Syntax und Erzeugung eines neuen Fonts über den in Abbildung 1 dargestellten Weg (Methode 2). Da diese Methode recht umständlich, langsam und von Kapazitätslimits geplagt ist, wird sie hier nicht näher beschrieben.

2. Durch Umcodierung in TeX-eigene `draw`-Macros, etwa `\drawline` aus `epic.sty` (Methode 1). Dabei handelt man sich die bereits früher erläuterten Nachteile ein.

3. Durch Umcodierung in `\special{}`-Anweisungen für DVI-Treiber wie `dvips` oder die emTeX-Treiber, die Linien mit beliebiger Steigung zeichnen können (Methode 4). Hier sind leider Kapazitäts- und Effizienzprobleme auf Treiberseite zu befürchten, da dieser Weg nicht für komplizierte Grafiken vorgesehen worden war.

4. Durch virtuelles Plotten auf einem Raster, welches eine Rastergrafik erzeugt, und dann durch Weiterbehandlung z.B. mit `bm2font`.

Umcodieren und insbesondere Interpretieren von HPGL erfordern ein Hilfsprogramm. Es gibt kommerzielle Wandlerprogramme für HPGL, aber diese sind nicht unbedingt für TeX-Anwendungen optimiert. Alle hier erwähnten Methoden werden aber von dem von mir entwickelten PD-Programm `hp2xx` [9, 14] unterstützt, welches HPGL-Dateien in eine Reihe verschiedener Formate (übrigens auch EPS) wandelt, Bildschirm-Previews gestattet und für die meisten verbreiteten Rechner als Binary mit Dokumentation verfügbar ist.

Nach meinen Erfahrungen sind die ersten drei Wege nur selten praktikabel, aber die Kombination von `hp2xx` mit `bm2font` bzw. mit `\special{em:graph}` hat sich bewährt und soll nun an einem Beispiel vorgestellt werden:

```
Wandlung von HPGL nach PCX:
   > hp2xx -m pcx -d 300
       -w 60 -h 50 -p21232222 foo.hp

Wandlung von PCX nach TFM / PK:
   > bm2font foo.pcx

Einbau in TeX:

   \input{foo.tex}      %  !!!
   \begin{figure}[htb]
     \centering
     \setfoo            %  !!!
     \caption[]{...}
   \end{figure}
```

Abbildung 8: Mit den beiden Utilities `hp2xx` und `bm2font` wird eine HPGL-Datei zunächst passend gerastert und dann in TeX-Fonts verwandelt.

Angenommen, die von einem 3D-Plotprogramm erzeugte Datei `foo.hp` soll in einem TeX-Dokument dargestellt werden. Wir wollen keinen Einfluß auf das Seitenverhältnis der Abbildung nehmen, aber die Abbildung soll in ein 60 mm breites und 50 mm hohes Rechteck passen. Die Auflösung betrage 300 dpi, das Zielformat sei PCX, und verschiedenen Plotterstiften sollen unterschiedliche Stiftdicken (in Pixel-Einheiten) zugeordnet werden. In Abbildung 8 ist zu sehen, wie `hp2xx` und `bm2font` aufgerufen werden müssen und wie die Abbildung im LaTeX-Dokument schließlich realisiert wird. Abbildung 9 zeigt das Ergebnis.

Vergleich der Methoden

Offenbar gibt es keine „beste" Methode, um TeX mit Grafik zu bereichern, wohl aber eine Reihe verbreiteter, aber veralteter Verfahren, als auch mehrere sinnvolle neue Kompromisse, die auf Fortschritten in der Entwicklung von DVI-Treibern und speziellen Utilities beruhen. Tabelle 1 gibt eine Übersicht.

Wegen der sehr langsamen Verarbeitung und des ständigen Risikos, Teile des TeX-Systems zu überlasten, möchte ich vom Gebrauch der seit Jahren verfügbaren Makropakete wie `epic.sty` und PiCTeX genauso abraten wie vom Einsatz von METAFONT zur Generierung von Zeichnungen und vom extensiven Einsatz

	A	B	C
Druckqualität	schlecht	gut	mäßig
Previewfähigkeit	ja	ja	ja
Portabilität	ja	ja	nein
Engpaß bei	TeX	METAFONT	DVI-Treiber
Zuwachs der DVI-Datei	deutlich	deutlich	erheblich
Bildgrößenänderung	leicht	leicht	leicht
Bedienkomfort	mäßig	schlecht	mäßig
Notwendige Utilities	Styles	evtl. `hp2xx`	evtl. `hp2xx`
Geeignet für Farbe?	nein	nein	nein
Größter Nachteil	„TeX capacity exceeded“	`mf, gftopk` überlastet	Treiber überlastet
	D	E	F
Druckqualität	gut	sehr gut	gut
Previewfähigkeit	ja	nein	ja
Portabilität	nein	nein	ja
Engpaß bei	—	—	—
Zuwachs der DVI-Datei	—	—	gering
Bildgrößenänderung	schwer	leicht	in Stufen
Bedienkomfort	hoch	hoch	befriedigend
Notwendige Utilities	keine	keine	`bm2font`
Geeignet für Farbe?	nein	ja	noch nicht
Größter Nachteil:	nicht portabel	nicht portabel	Begrenzte Zahl an Abbildungen

Tabelle 1: Vergleichende Bewertung verschiedener Methoden zur Grafikerzeugung in TeX bzw. LaTeX: A = Einsatz vom Macros wie `epic.sty` und PiCTeX, B = Linien malen mit METAFONT, C = Linien malen mit `\special`, D = Rastergrafik-Einbau mit `\special`, E = Grafiken via PostScript-Treiber, F = Einsatz von `bm2font`.

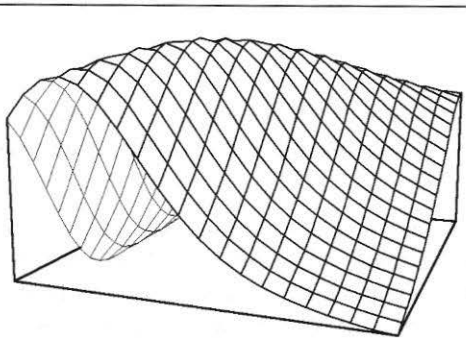

Abbildung 9: Mit den richtigen Utilities ist auch der Einbau einer HPGL-Datei in LaTeX kein Problem.

der `\special{}`-Befehle zum Zeichnen von Linien mit beliebiger Steigung. Die folgenden Methoden sind dagegen auch für komplizierte Abbildungen bauchbar:

1. Wer ohnehin EPS-Dateien verarbeiten muß und somit nicht mehr in den Genuß vom DVI-Preview eingebundener Grafiken kommen kann, mag den PostScript-Weg ruhig beschreiten und die vielen in PS begründeten Vorteile genießen.

2. Sollen nur s/w-Rastergrafiken, die schon in der passenden Größe und Auflösung vorliegen, ohne Rücksicht auf eventuelle spätere Portierungswünsche verarbeitet werden, sind `\special{}`-Befehle zum direkten Einlesen der Grafiken durch geeignete DVI-Treiber wie die emTeX-Serie wegen ihrer einfachen Handhabung und Effizienz sehr hilfreich.

3. Für TeX-Puristen führt dagegen kein Weg an der von `bm2font` benutzten Methode vorbei, auch wenn sie etwas mehr Lernaufwand erfordert. Sie verwendet konsequent die Mechanismen des TeX-Systems, ohne Kompromisse in der Anwendungspraxis einzugehen. Sie bleibt dadurch genauso portabel wie TeX selbst und kommt mit allen DVI-Treibern zurecht. `bm2font` bietet außerdem die derzeit beste Möglichkeit, Bilder mit vielen Farben bzw. Graustufen für TeX aufzubereiten.

Sind schließlich HPGL-Daten die Ausgangsbasis, kann jede dieser Methoden frei gewählt werden, denn mit der Utility `hp2xx` ist jede entsprechende Format-Konvertierung möglich.

Bezugsquellen

Die schönsten Methoden nutzen nichts, wenn die Utilities dazu nicht erreichbar sind. Deshalb folgt nun ein relativ ausführlicher Nachweis der angegebenen Daten und Dokumente.

Praktisch alle angegebenen Daten sind über die Computernetze erhältlich. Wer nicht zumindest email-Zugang hat, kann aber Mitglied von DANTE, Deutschsprachige Anwendervereinigung TeX e.V., Postfach 10 18 40, 69008 Heidelberg, werden. Für Mitglieder von DANTE e.V. gibt es einen Disketten-Versandservice (und die Mitgliedzeitschrift „Die TeXnische Komödie" mit vielen Tips und Tricks).

Die beiden wichtigsten Quellen für TeX-Software sind der Heidelberger Listserv `listserv@vm.urz.uni-heidelberg.de` und der ftp-Server `ftp.uni-stuttgart.de [129.69.1.12]`, der auch einen Mail-Service unter `mail-server@rus.uni-stuttgart.de` bietet. Die einzeilige email „help" an einen der Mail-Server wird mit näheren Instruktionen beantwortet.

Praktisch alle angegebenen Daten befinden sich auf `ftp.uni-stuttgart.de` im Unterdateibaum `/soft/tex`. Dieser wird in den folgenden Zitaten stets als Ausgangspunkt verwendet. Eine Angabe „./utilities" expandiert also zu `ftp.uni-stuttgart.de:/soft/tex/utilities`. Ist keine Quelle angegeben, führt eine Suche ab `/soft/tex` häufig zum Ziel.

Literatur

[1] Die PiCTeX-Makros befinden sich in `./graphics/pictex/`. Dokumentation zu PiCTeX ist in [10] enthalten. Die Original-Dokumentation [15] kann gegen eine Gebühr nur bei der TeX Users Group bezogen werden.

[2] `Xfig` ist ein X11-basiertes CAD-artiges Zeichenprogramm, das u.a. auch LaTeX-Quellcode generiert. ftp-Server lagern es bei X11-Utilities, nicht unbedingt bei TeX.

[3] TeXcad wird innerhalb des emTeX-Pakets vertrieben. Es ist ein *shareware*-Programm, also nicht kostenlos.

[4] GhostScript ist GNU-Software. Sie ist über viele ftp-Server kostenlos erhältlich.

[5] `dvips` gibt es in `./dviware/dvips/`.

[6] Das emTeX-Paket findet man in `./systems/msdos/emtex/`.

[7] `RUMgraph` gibt es als zoo-Archiv in `./support/rumgraph.zoo`.

[8] `bm2font` befindet sich inkl. Handbuch in `./graphics/bm2font/`; aktuelle Version: 2.0.

[9] Standard-Version von `hp2xx` inkl. Dokumentation in `./support/hp2xx/`; aktuelle Version: 3.13.

[10] Helmut Kopka. *LaTeX-Erweiterungsmöglichkeiten.* Addison-Wesley, Bonn; München; Reading, Mass., 1992.

[11] Leslie Lamport. *LaTeX: A Document Preparation System.* Addison-Wesley, Reading, Mass., 1986.

[12] Reinhard Sippel. HP-GL-Standardbefehle. *c't* 1/90, S. 335–342.

[13] Friedhelm Sowa. Grafikintegration mit `bm2font`. *Die TeXnische Komödie*, 3(2), S. 10–13, 1991.

[14] Heinz Werntges. `hp2xx` – Ein Konverter für HPGL-Dateien. *Die TeXnische Komödie*, 3(2), S. 14–21, 1991.

[15] Michael J. Wichura. *The PiCTeX Manual.* TeX Users Group, Providence, RI., 1987.

Was Sie schon immer über TEX wissen wollten

...

Kopfzeilen mal anders

Luzia Dietsche

Immer wieder erreichen mich Anfragen von Mitgliedern, wie man die Kopfzeile eines LaTeX-Dokumentes ändern kann. Der am häufigsten geäußerte Wunsch ist, die Großschreibung in Groß-/Kleinschreibung umzuwandeln. Aber auch andere Schrift, eigene Aufteilung der Informationen in der Zeile oder einen Trennstrich zwischen Kopf und Fließtext sind gefragt.

Es gibt viele Wege, solch eine Modifikation durchzuführen. Eine Möglichkeit bietet L. Lamport selbst über die Anweisung `\pagestyle{myheadings}` und die dazu passenden Anweisungen `\markright{}` bzw. `\markboth{}{}`, die in der Standard-Literatur beschrieben sind. Das Problem hierbei ist aber, daß LaTeX nur begrenzt Eingriffe erlaubt. Die Positionierung der Seitenummer liegt fest und sie erscheint immer. Will man den Inhalt von `\chapter`, `\section` oder einer anderen Überschrift in die Kopfzeile übernehmen, so muß man diese Information zweimal schreiben (für die Überschrift und die Kopfzeile) oder wissen, wie der Mechanismus in LaTeX aussieht.

Eine zweite Möglichkeit hat man, wenn man den verwendeten Style-File modifiziert. Für die Änderung kopiert man sich die betreffenden Original-Zeilen in einen eigenen File mit der Erweiterung `sty` (z.B. `meinkopf.sty`) und bindet diesen neugeschaffenen Style-File in der Optionsklammer der `\documentstyle`-Anweisung ein. Für das folgende Beispiel habe ich die Vorgabe aus `report.sty`[1] kopiert:

```
\if@twoside
\def\ps@headings{\let\@mkboth\markboth
 \def\@oddfoot{}\def\@evenfoot{}%       No feet.
 \def\@evenhead{\rm \thepage\hfil \sl \leftmark}%  Left heading.
 \def\@oddhead{{\sl \rightmark}\hfil \rm\thepage}% Right heading.
 \def\chaptermark##1{\markboth{\uppercase{\ifnum \c@secnumdepth >\m@ne
      \@chapapp\ \thechapter. \ \fi ##1}}{}}%
 \def\sectionmark##1{\markright{\uppercase{\ifnum \c@secnumdepth >\z@
   \thesection. \ \fi ##1}}}}
\else
```

[1] Aus dem Kopf des Files: `Standard Document Style 'report' <14 Jan 92>.`

```
\def\ps@headings{\let\@mkboth\markboth
\def\@oddfoot{}\def\@evenfoot{}%     No feet.
\def\@oddhead{{\sl \rightmark}\hfil \rm\thepage}% Heading.
\def\chaptermark##1{\markright{\uppercase{\ifnum \c@secnumdepth >\m@ne
  \@chapapp\ \thechapter. \ \fi ##1}}}}
\fi
```

Die Anweisung, die die Großschreibung bewirkt, ist das `\uppercase` in der 6ten Zeile. Entfernt man es, wird die Kopfzeile in normaler gemischter Schreibweise gesetzt. Soll die Aufteilung der Information über die Zeile anders aussehen oder die Schriftart geändert werden, so ist dafür die richtige Stelle die Anweisung `\def\@evenhead` bzw. `\def\@oddhead` in Zeile 4 bzw. 5. Bei Bedarf kann man sich auch eine Fußzeile schaffen, indem man die leeren Klammern von `\def\@evenfoot` und `\def\@oddfoot` mit Leben füllt. Der Variantionsmöglichkeiten sind gar viele.

Um die ursprüngliche Definition des Seitenstils `headings` zu bewahren, erfindet man einen neuen Namen für den Stil (z.B. `kopfneu`). Dazu muß `headings` überall durch `kopfneu` ersetzt werden. Aktivieren kann man nun den neuen Seitenstil, wie gewohnt, durch die Anweisung `\pagestyle` (z.B. `\pagestyle{kopfneu}`). Im folgenden habe ich die oben erwähnten Änderungen eingearbeitet und für diese Seite eingeschaltet:

```
\if@twoside
\def\ps@kopfneu{\let\@mkboth\markboth
\def\@oddfoot{\hfil\today}\def\@evenfoot{\today\hfil}% aktuelles Datum
 \def\@evenhead{\sl\thepage\hfil\leftmark\hfil}% Schrift und
 \def\@oddhead{\sl\hfil\rightmark\hfil\thepage}% Aufteilung
 \def\chaptermark##1{\markboth{\ifnum \c@secnumdepth >\m@ne
      \@chapapp\ \thechapter. \ \fi ##1}{}}% \uppercase entfernt
 \def\sectionmark##1{\markright{\ifnum \c@secnumdepth >\z@
   \thesection. \ \fi ##1}}}%                \uppercase entfernt
\else
\def\ps@kopfneu{\let\@mkboth\markboth
\def\@oddfoot{\hfil\today}%
\def\@oddhead{\sl\hfil\rightmark\hfil\thepage}%
\def\chaptermark##1{\markright{\ifnum \c@secnumdepth >\m@ne
  \@chapapp\ \thechapter. \ \fi ##1}}}
\fi
```

Leserbrief(e)

Leserbriefe und Antworten geben die Meinung der Schreibenden wieder und werden ungekürzt veröffentlicht. Diese Rubrik soll auch für allgemeine Anfragen dienen. Bitte Zuschriften für eine Veröffentlichung an dieser Stelle kennzeichnen.

Gegendarstellung zum Artikel „Amiga und TEX“, 1/1993

An und für sich halte ich nicht viel von TEX-Geschwindigkeitsmessungen, weil für mich qualitative Aspekte wie Bedienbarkeit oder Sicherheit weitaus wichtiger sind. Dennoch kann ich die im letzten Heft angegebenen Zahlen nicht unkommentiert lassen.

Zunächst die Zeiten der zwei Atari-TEXs: Die angegebenen Zahlen für *Lkurz* sind für den Vergleich relativ nutzlos, da sie mit einer kaputten LATEX-Installation gemessen wurden. Es wurden für den Test sämtliche Ligaturen ausgeschaltet, die spezielle Umlautbehandlung des `german.sty` (niedrigere Tremata, besser für deutschen Text) ausgeschaltet und der `german.sty` in die Format-Datei eingebunden, das hat beim TEX von Stefan Lindner immerhin 90 Sekunden Unterschied ausgemacht (von 590 auf 500 Sekunden)! Eigene Messungen mit einem richtig installierten LATEX ergaben einen Wert von 640 Sekunden, der inzwischen auf 575 Sekunden gesenkt wurde (die aktuelle Version befindet sich noch in der Testphase), und gleicht damit ungefähr den TEX-Versionen auf dem Amiga.

Noch haarsträubender sind die Zahlen für das TEXbook: So ist das csTEX dort nicht etwa besonders schnell, sondern die damalige Version des csTEX hat die Übersetzung des TEXbooks nach 304 Seiten mit einer Fehlermeldung abgebrochen! Die 1050 Sekunden sind daher völlig irrelevant, da sie die Zeit bis zu Seite 304 darstellen. Eigene Messungen mit dem Lindner-TEX ergeben für die damalige Version 1768 Sekunden, die inzwischen auf 1560 Sekunden gesenkt werden konnten (jeweils mit 16 MHz). Interessant wären noch Messungen mit aktuellen Versionen des csTEX, sowohl mit der PD-Version (4.0) als auch evtl. mit der kommerziellen Version MultiTEX (5.x).

Die ersten zwei Zeilen der Tabelle müßten also eher so aussehen:

TEX-Version	LKURZ (s)	TB (s)
LindnerTEX	640	1768
LindnerTEX 2.4x	575	1560
csTEX 2.0	≈ 560	—

Daß der Mac „so ver... schnell" ist, wie der Autor sich ausdrückte, ist kein Wunder, wurden die Zeiten doch auf einem Mac SE/30 mit einem 68030 bei 16 MHz, 68882 und 8 MB RAM gemessen. Der Hinweis auf den Prozessor 68000 bezog sich einzig und allein darauf, daß die verwendete TEX-Version nicht speziell für den 68030 angepaßt ist, also auch auf jedem langsameren Prozessor laufen würde. Und die Zeiten für den Amiga 4000 zeigen ja auch, daß allein der Prozessortyp schon eine Menge ausmacht.

Zum Schluß möchte ich noch darauf hinweisen, daß in der Anleitung `GermanDoc.tex` der bisherigen METAFONT-Version von Stefan Becker ein böser Fehler enthalten ist: Es wird dort beim Erzeugen der `plain.base` empfohlen, die Datei `cmbase.mf` einzulesen. Das ist grundsätzlich falsch und führt zu den schlimmsten Fehlern (nämlich solche, die fast nicht zu bemerken und schwer zu finden sind, aber falsche Resultate erzeugen). Die Zeile mit dem „`\input cmbase`" ist folglich ersatzlos zu streichen. Falls jemand mit dieser Base-Datei LATEX-Grafikfonts (`circle*`, `lcircle*`, `line*`) erzeugt hat, sind diese definitiv falsch, obwohl METAFONT keine Fehlermeldung ausgegeben hat! Erst die Fonts aus neueren LATEX-Versionen (ab März 1992) fangen diesen Fehler ab, indem sie wenigstens eine Fehlermeldung ausgeben. Die Zeichensätze aus dem Notensatzpaket MuTEX von Schofer und Steinbach lassen sich mit der falschen `plain.base` überhaupt nicht erzeugen, weitere Problemfälle ließen sich sicherlich leicht finden.

Ich schreibe das deswegen so drastisch, weil das Problem schon uralt ist und schon viele Sorgen bereitet hat, aber offensichtlich immer noch nicht ausgerottet ist. Wenn überhaupt, dann darf `cmbase.mf` nur in einer `cm.base` enthalten sein, niemals jedoch in `plain.base`. Wenn denn unbedingt gewollt, muß man also zwei verschiedene Bases bereithalten, `plain.base` *ohne* und `cm.base` *mit* den zusätzlichen Makros. Da das Einlesen von `cmbase.mf` aber sowieso nur sehr kurze Zeit in Anspruch nimmt und es auch mit der `plain.base` vollkommen automatisch erfolgt, aber eben nur dann, wenn die `cmbase` benötigt wird, lassen die meisten METAFONT-Implementationen heutzutage die `cm.base` einfach weg.

In zukünftigen Versionen von Becker-METAFONT, die von DANTE e.V. bezogen werden können, wird das Problem laut Aussage des Amiga-Koordinators, Markus Erlmeier, behoben sein.

Mit freundlichen Grüßen,

Lutz Birkhahn

Fraktur und eqnarray

Sehr geehrte Frau Dietsche,

als LaTeX-Anwender darf ich mich an Sie mit folgenden Problemen wenden; vielleicht weiß ein Leser Rat:

1. Gibt es die Möglichkeit, auch Frakturschrift („deutsche Buchstaben" in Schreib- oder Druckschrift) in LaTeX-Dokumenten zu verwenden? Der Softwareliste entnehme ich, daß mir hier offenbar DISK013 weiterhelfen würde. Wie aber installiert man solche zusätzlichen Tools? Kann auch ich als PC-Unerfahrener damit rechnen, die notwendigen Anpassungen auf meinem PC vornehmen zu können?

2. Wie erreicht man am elegantesten, daß innerhalb einer `eqnarray`-Umgebung eine Zeile anstelle einer in Klammern gesetzten Formelnummer eine andere Markierung erhält, z.B. $(*)$ oder eine Box $\square$ ohne Klammern? Zum Beispiel:

$$\begin{aligned} d(a,b) &\leq d(a,c) + d(c,b) \\ &= 1/k - \delta_1 + d(c,b) \\ &< 1/k - \delta_1 + \delta \qquad\qquad \square \\ &\leq 1/k - \delta_1 + \delta_1 = 1/k \end{aligned}$$

   Der `\hfill`-Befehl bewirkt innerhalb der `eqnarray` Umgebung leider nicht das Gewünschte; und ebenso auch nicht der von H. Kopka in „LaTeX — Eine Einführung" (3. Auflage) in Kapitel 5.4.8. angegebene Trick, zwei Formelgruppen nebeneinander stehen zu lassen, von denen die linke die Formeln und die rechte die $\square$ enthält, denn dann kann die $\square$ nur `[t]`, `[b]` oder vertikal zentriert positioniert werden, aber nicht exakt auf Höhe der gewünschten Formel.

Ich denke, daß meine Fragen im Rahmen der mathematischen Textverarbeitung von allgemeinem Interesse sein könnten, und würde mich über eine Antwort sehr freuen.

Vielen Dank im voraus!

Franz Strobl

Anfängerfragen?

Sehr geehrte Damen und Herren, liebe TEX-Gurus,

ich habe einige Fragen, Anfängerfragen, wie in „Die TEXnische Komödie" angeregt, aber auch wohl etwas für fortgeschrittenere Benutzer.

Promillezeichen

Bisher ist es mir noch nicht gelungen, ein ästhetisch befriedigendes Promillezeichen zu erzeugen. Irgendwie sollte es so ähnlich wie

$$\%_0$$

aussehen. Vielleicht veröffentlichen Sie einmal, da das Promillezeichen ja öfters verwendet wird, einen Vorschlag, wie ein Makro dafür aussehen könnte. Ja, ich bin sogar der Meinung, daß es in LATEX ähnlich wie für das Prozentzeichen einen eigenen Befehl für das Promillezeichen geben sollte. Anregung für die nächste Version von LATEX?

Umgebungen, Erklärungen

Auf den Seiten 10 ff von H. Kopka, LATEX — *Eine Einführung* findet man dazu näheres. Mir ist nicht klar geworden, wann man, wie z.B. bei `quote`, mit `\begin` und `\end` einklammern muß und wann, wie z.B. bei `\bf`, die geschweiften Klammern allein reichen. Gibt es dafür Regeln, oder muß man das einfach auswendig lernen?

Minimal-TEX-System

In unserem Schulrechner — PC — soll ein TEX-System installiert werden; ich selbst besitze einen Mega ST von Atari. Bei beiden Rechnern besteht Platzman-

gel auf der Festplatte. TEX soll nur zur Demonstration und für kleinere Aufgaben wie z.B. die Anfertigung von Arbeitsblättern benutzt werden. Da müßte es doch möglich sein, recht viele Dateien auf der Platte zu löschen: z.B. werden nicht alle Styles gebraucht, die vielen, vielen Fonts sind überflüssig, genauso BIBTEX und einiges mehr. Ich traue mich nicht so recht, mit „Trial and Error" auszuprobieren, welche Dateien für das Arbeiten wirklich unabdingbar sind. Vielleicht könnte doch einmal darüber geschrieben werden, wie die Verzeichnisse unmittelbar nach der Installation aussehen und was verzichtbar ist.

Ein solches Minimalsystem wäre nicht nur bei knappem Festplattenplatz nützlich. Vorstellbar ist, daß zu Beginn einer Rechnersitzung das System in eine *Ram-Disk* geladen und dann bei Bedarf von dort gelesen wird. Dies würde wegen des wesentlich schnelleren Zugriffs eine Beschleunigung der Arbeit bedeuten. Hat man aber nur z.B. 1 MByte Arbeitsspeicher, könnte man dieses Verfahren nur anwenden, wenn das System entsprechend klein wäre.

Preview

Nach der Umsetzung der `.tex`-Dateien in `.dvi`-Dateien kontrolliert man gewöhnlich vor der Druckausgabe sein Werk auf dem Bildschirm. Bei längeren Dokumenten stört, daß man zwar schnell von oberer auf die untere Seitenhälfte umschalten kann, die Ausgabe anderer Seiten wegen der Zeit für die Umsetzung von `.dvi` in Pixelformat aber deutlich verzögert ist. (Bei „normalen" Editoren erscheint die nächste Seite unmittelbar.) Gibt es Systeme, die da schneller sind? Vorstellbar wäre z.B., daß bei Ausgabe der Seite 10 gleichzeitig die Seiten 5 bis 15 im Pixelformat im Speicher bereitgehalten werden. Wird jetzt durch Tastendruck Seite 11 verlangt, steht diese sofort zur Verfügung. Unmittelbar nach der Ausgabe — der Benutzer ist jetzt erst einmal einige Sekunden beschäftigt — würde dann automatisch die Seite 5 aus dem Speicher entfernt und die Seite 16 ins Pixelformat umgesetzt werden. Bei der nächsten Anforderung stünde also die gewünschte Seite schon bereit, und man könnte auch mal vorgreifen und anstelle von Seite 12 gleich Seite 14 anwählen.

Meiner Meinung nach braucht man dafür sogar kein *multitasking*-Betriebssystem, obwohl das natürlich hilfreich wäre: es könnte fortlaufend umgesetzt werden, alle 10 Millisekunden fragt man periodisch die Tastatur ab, ob eine Bedienungsanforderung vorliegt.

Auch hier glaube ich, daß eine solche Beschleunigung der Bearbeitung nicht nur mich, sondern auch andere Benutzer interessiert. Deshalb schlage ich vor, daß dieses Thema in der Mitgliederzeitschrift als Beitrag aufgegriffen wird:

Gibt es solche Systeme schon? Für welche Rechner? Wird an solchen Systemen gearbeitet?

Format der Absätze

Beim Kontrollieren dieses Briefes fällt mir auf, daß er nur mit dem `letter`-Style bei einem neuen Absatz außer der Einrückung eine Leerzeile hat. Mit `article` geschrieben, fehlt diese Leerzeile. Liegt da irgendwie eine Fehlbedienung durch mich vor, oder ist das Fehlen der Leerzeile bei `article` beabsichtigt?

Ich hoffe, daß Sie den Umfang diese Briefes nicht als Zumutung empfinden. Bedanken möchte ich mich dafür, daß Sie bereit sind, in Ihrer Freizeit nicht nur solche Anfragen zu beantworten, sondern allgemein für die Benutzbarkeit und Fortentwicklung von TEX zu arbeiten.

Mit freundlichen Grüßen

Albrecht Mehl

Anmerkung der Redaktion

Ich wäre sehr an den Antworten zu diesen und den Fragen des vorigen Briefes in Form von kurzen (oder auch längeren) Artikeln interessiert, da ich glaube, daß viele Mitglieder die gleichen oder ähnliche Probleme haben. Ich bin leider aus Zeitgründen nicht in der Lage, diese Artikel selbst zu schreiben, obwohl Herr Mehl auf einen Teil seiner Fragen bereits (sehr kurze) Antwort erhalten hat. Vielleicht kann ja auch Herr Mehl selbst eine Zusammenfassung der Antworten zur Veröffentlichung an mich schicken.

Mit freundlichem Gruß

Luzia Dietsche

Spielplan

Termine

23.–24.9.1993 9. Mitgliederversammlung von DANTE e.V.
Kaiserslautern
Kontakt: Klaus Uttler

4.–7.10.1993 CyrTUG-93 — 3. Jahrestreffen
Pereslavl'-Zalesskiĭ, Rußland
Kontakt: Irina Makhovaya

18.10.1993 NTG — 12. Jahrestreffen
OCÉ, Den Bosch
Kontakt: Gerard van Nes

25.–29.10.1993 Typography: tradition and innovation
Bilbao, Spanien
Kontakt: DZ-Centro de Diseño
Industrial S.A.
Sabino Arana, 8
480 13 Bilbao
Tel.: 34/4/4278160
Fax: 34/4/4278005

16.–18.2.1994 DANTE'94 und
10. Mitgliederversammlung von DANTE e.V.
Münster
Kontakt: Wolfgang Kaspar

16.–18.2.1994 RIDT'94 und EP94
Raster Imaging and Digital Typography
Darmstadt
Kontakt: Christine Harms

Stammtische

In verschiedenen Städten im Einzugsbereich von DANTE e.V. finden regelmäßig Treffen von TEX-Anwendern statt, die für jeden offen sind. Im folgenden sind die Daten und Adressen aufgelistet, die an uns weitergeleitet wurden.

12687 Berlin
Horst Szillat
Sella-Hasse-Str. 31
Tel.: 9322496 (Beantworter)
Gaststätte „Bachmann"
Brunnenstraße, nähe Rosenth. Platz
Letzter Donnerstag im Monat, 19.00 Uhr

22527 Hamburg
Reinhard Zierke
Tel.: 040/54715-295
„TEX's Bar-B-Q", Grindelallee 31
20146 Hamburg
Letzter Mittwoch im Monat, 18.00 Uhr

27570 Bremerhaven
Lutz-Peter Kurdelski
Alfred-Wegener-Institut
Am Handelshafen 12
Tel.: 0471/4831-503
`tex@awi-bremerhaven.de`
Erster Dienstag im Monat

28759 Bremen
Martin Schröder
Tel.: 0421/628813
`115d@alf.zfn.uni-bremen.de`
Universität Bremen, MZH 4.St.
gegenüber den Fahrstühlen
Erster Donnerstag im Monat, 18.30 Uhr

42279 Wuppertal
Andreas Schrell
Windhövel 2
Tel.: 0202/66 68 89
`Andreas.Schrell@FernUni-Hagen.de`
Gasthaus Yol, Ernststr. 45
Zweiter Donnerstag im Monat, 19.30 Uhr

47226 Duisburg
Friedhelm Sowa
Rheinstr. 14
„Gatz an der Kö", Königstraße 67
Dritter Dienstag im Monat, 19.30 Uhr

53111 Bonn
Ulrich Wisser
Heerstr. 125
Tel.: 0228/692356
„Anno", Kölnstr. 47
Dritter Montag im Monat, 20.00 Uhr

63589 Linsengericht
Michael Baas
Taunusstr. 4
Tel.: 06051/67 97 9
noch nicht festgelegt

69008 Heidelberg
Luzia Dietsche
Tel.: 06221/29 76 6
`dante@vm.urz.uni-heidelberg.de`
„Nikarklause" (beim Schwimmbad)
Letzter Mittwoch im Monat, 20.00 Uhr

70569 Stuttgart
Barbara Burr
Rechenzentrum
Allmandring 30
Tel.: 0711/68 55 81 1
`zrfn0370@ds0rus54`
Wechselnd

95326 Kulmbach
Martin Leidig
Obere Stadt 3
Tel.: 09221/8 16 28
Fax: 09221/8 44 93
noch nicht festgelegt

Adressen

DANTE,
Deutschsprachige Anwendervereinigung TEX e.V.
Postfach 10 18 40
69008 Heidelberg

Tel.: 06221/2 97 66
Fax: 06221/16 79 06
e-mail: `dante@vm.urz.uni-heidelberg.de`

Konten: Postgiroamt Karlsruhe
BLZ 660 100 75

2134 00-757 für Beiträge
bzw. 2946 01-750 für Bücher und Disketten
bzw. 1990 66-752 für Tagungen

Präsidium:

Joachim Lammarsch	Präsident
Uwe Untermarzoner	Vizepräsident
Friedhelm Sowa	Schatzmeister
Luzia Dietsche	Schriftführerin

TEX Users Group
P.O. Box 869
Santa Barbara, CA 93 102
U.S.A.
e-mail: `tug@tug.org`

Server in Stuttgart:
`ftp.uni-stuttgart.de` [129.69.1.12] (ftp)
`mail-server@rus.uni-stuttgart.de` (e-mail)

Server in Heidelberg:
`listserv@vm.urz.uni-heidelberg.de`

Autoren/Organisatoren

Luzia Dietsche [3,54]
Postfach 10 18 40
69008 Heidelberg
dante@vm.urz.uni-heidelberg.de

Christine Harms [62]
GMD, Schloß Berlinghofen
Postfach 13 16
53757 Sankt Augustin
Tel.: 02241/142473
FAX: 02241/142618
ep94@gmd.de

Wolfgang Kaspar [62]
Westfälische Wilhelms-Universität
Universitätsrechenzentrum
Einsteinstr. 60
48149 Münster
kaspar@dmswwu1a.uni-muenster.de

Joachim Lammarsch [4,5]
siehe Seite 66

Irina Makhova [62]
Executive Director
CyrTUG, Mir Publishers
2 Pervy Rizhsky pereulok
SU-Moscow 129820
Tel.: 286/06/22
cyrtug@mir.msk.su

Albrecht Mehl [59]
Hebbelstr. 42
64291 Darmstadt
Tel.: 06151/373992

G.J.H. van Nes [62]
Postbus 394
NL-1740 AJ Schagen
vannes@ecu.nl

Walter Obermiller [19]
MPI Chemie, Abt. Geochemie
Saarstr. 23
55122 Mainz
walter@sbart.tynet.sub.org

Franz Strobl [58]
Starenweg 12
85356 Freising

Philip Taylor [19]
72 The Course, Eltham
GB-SE 9 London
chaa006@vax.rhbnc.ac.uk

Klaus Uttler [62]
Universität Kaiserslautern
RHRK
Paul-Ehrlich-Str.
67663 Kaiserslautern
uttler@rhrk.uni-kl.de

Dr. Heinz Werntges [38]
c/o Physikalische Biologie
Heinrich-Heine-Universität
40225 Düsseldorf
werntges@convex.rz.
uni-duesseldorf.de

Ulrik Vieth [6]
Am Ostbahnhof 28
40878 Ratingen
vieth@convex.rz.uni-duesseldorf.de

Technischer Beirat

Zuschriften an die Koordinatoren werden in der Regel nur beantwortet, wenn ein ausreichend frankierter und adressierter Rückumschlag mitgeschickt wird. Die Koordinatoren sind nicht verpflichtet, auf jede Frage einzugehen.

Amiga
Markus Erlmeier
Postfach 415
84001 Landshut
Tel.: 0871/77939
Btx: 087177939-0001
Markus_Erlmeier@p21.F6.N246.Z2.FIDONET.ORG
mwe@marie.physik.tu-berlin.de

Atari
Stefan Lindner
Iltisstr. 3
90766 Fürth
Tel.: 0911/7591886 oder

Lutz Birkhahn
Fürtherstr. 6
90556 Cadolzburg
Tel.: 09103/2886
lutz@bisun.nbg.sub.org

BS2000 & Graphik
Friedhelm Sowa
Heinr.-Heine Universität
Rechenzentrum
Universitätsstr. 1
40225 Düsseldorf
Tel.: 0211/3113913
tex@ze8.rz.uni-duesseldorf.de

Macintosh
Lothar Meyer-Lerbs
Am Rüten 100
28357 Bremen
Tel.: 0421/252624
TeXSatz@zfn.uni-bremen.de

MVS
Joachim Lammarsch
Universitätsrechenzentrum
Im Neuenheimer Feld 293
69120 Heidelberg
x92@vm.urz.uni-heidelberg.de

Vertreter:
Dr. Klaus Braune, s. UNIX

NOS/VE & METAFONT
Norbert Schwarz
Ruhr Universität
Rechenzentrum
Universitätsstr. 150
44721 Bochum
Tel.: 0234/700-3940
Norbert.Schwarz@ruba.rz.ruhr-uni-bochum.dbp.de

PC
Dr. Peter Breitenlohner
Max-Planck-Institut für Physik
Postfach 40 12 12
80805 München
peb@dmumpiwh.bitnet

UNIX
Dr. Klaus Braune
Universität Karlsruhe
Rechenzentrum
Zirkel 2
76128 Karlsruhe
Tel.: 0721/608-4031
Braune@rz.uni-karlsruhe.de

VAX/VMS
Peter Saueressig
Philips Kommunikations
Industrie AG, Abt. LD
Thurn-und-Taxis-Str. 10
90411 Nürnberg
Tel.: 0911/5262714
Fax: 0911/5262014
pla_psa@pki-nbg.philips.de

Vertreter:
Gerhard Friesland-Köpke
Universität Hamburg
FB Informatik
Vogt-Kölln-Str. 30
22527 Hamburg
friesland@rz.informatik.
uni-hamburg.d400.de

VM
Dr. Georg Bayer
TU Braunschweig
Rechenzentrum
Postfach 3329
38023 Braunschweig
c0030001@dbstu1.bitnet

Dokumentation[1]
Jürgen Egeling
Klosterweg 28/L 601
76131 Karlsruhe
ry90@dkauni2.bitnet

[1] von TEX Makros, Style Files, etc.

German-Style
Bernd Raichle
Stettener Str. 73
73732 Esslingen
raichle@azu.informatik.
uni-stuttgart.de

Lehrerfortbildung
Werner Burkhardt
Carl-Benz-Schule Mannheim
Neckarpromenade 23
68167 Mannheim

PostScript
Jürgen Glöckner
In der Hessel 23
69168 Wiesloch

Server-Koordination
Dr. Rainer Schöpf
Konrad-Zuse-Zentrum
für Informationstechnik
Heilbronner Str. 10
10711 Berlin
jl2@vm.urz.
uni-heidelberg.de

Treiber
Joachim Schrod
Kranichweg 1
63322 Rödermark-Urberach
schrod@iti.informatik.
th-darmstadt.de

Verlag und Buchhandel
Christa Preisendanz-Loeser
International Thomson Publishing GmbH
Trübnerstr. 38
69121 Heidelberg
Tel.: 06221/400177
Fax: 06221/472909

Inhalt Heft 2/93