Zpravodaj Československého sdružení uživatelů TeXu

Vít Zýka

Používáme pdfTeX IV: mikrotypografické rozšíření

Zpravodaj Československého sdružení uživatelů TeXu, Vol. 14 (2004), No. 2, 47–53

Persistent URL: http://dml.cz/dmlcz/149949

Terms of use:

© Československé sdružení uživatelů TeXu, 2004

Institute of Mathematics of the Czech Academy of Sciences provides access to digitized documents strictly for personal use. Each copy of any part of this document must contain these *Terms of use*.

This document has been digitized, optimized for electronic delivery and stamped with digital signature within the project *DML-CZ: The Czech Digital Mathematics Library* http://dml.cz

Používáme pdfTEX IV: mikrotypografické rozšíření

Vít Zýka

Tento článek se věnuje dvěma mikrotypografickým rozšířením odstavcového zlomu TEXu, které Hàn Thế Thành implementoval do pdfTEXu jako svou doktorskou práci [5]. Prvním rozšířením je prostrkání okrajů. *Jde o zobecnění visící interpunkce na libovolný znak a na obě hrany bloku textu. Cílem je korigovat optické výchylky okrajů textu dané různou světlostí kresby jednotlivých znaků. Druhým rozšířením je hz-*algoritmus. *Jde o možnost horizontálního zúžení nebo rozšíření písma o několik procent; tím se umožní vyšší variabilita algoritmu řádkového zlomu a vyšší pravděpodobnost akceptovatelného vzhledu odstavce. Změna písma musí být taková, aby byla okem nepostřehnutelná. Oba algoritmy navrh věhlasný německý typograf Hermann Zapf [6]. Zaměříme se zde pouze na technickou část použití obou algoritmů; typografickou analýzu a volbu vhodných hodnot parametrů může čtenář najít v publikacích [5, 4, 3, 2].*

Předcházející odstavec je vysázen standardním LaTeXovým nastavením parametrů odstavcového zlomu, především `\tolerance`=200, `\emergencystretch`=0pt. Použijeme-li *hz*-algoritmus s fonty v rozmezí ±3 % a algoritmus prostrkání levého okraje pro znaky *T* a pravého okraje pro tečku, čárku a rozdělovací znaménko, dostaneme náledující výsledek:

Tento článek se věnuje dvěma mikrotypografickým rozšířením odstavcového zlomu TEXu, které Hàn Thế Thành implementoval do pdfTEXu jako svou doktorskou práci [5]. Prvním rozšířením je prostrkání okrajů. *Jde o zobecnění visící interpunkce na libovolný znak a na obě hrany bloku textu. Cílem je korigovat optické výchylky okrajů textu dané různou světlostí kresby jednotlivých znaků. Druhým rozšířením je hz-*algoritmus. *Jde o možnost horizontálního zúžení nebo rozšíření písma o několik procent; tím se umožní vyšší variabilita algoritmu řádkového zlomu a vyšší pravděpodobnost akceptovatelného vzhledu odstavce. Změna písma musí být taková, aby byla okem nepostřehnutelná. Oba algoritmy navrh věhlasný německý typograf Hermann Zapf [6]. Zaměříme se zde pouze na technickou část použití obou algoritmů; typografickou analýzu a volbu vhodných hodnot parametrů může čtenář najít v publikacích [5, 4, 3, 2].*

Prostrkání okrajů (margin kerning)

Začneme algoritmem, jehož použití je jednodušší. Pro zapnutí prostrkání okrajů[1] stačí přiřadit kladnou hodnotu parametru `\pdfprotrudechars` a nastavit každému znaku, který má být prostrčen, velikost přesahu. Parametr má tři polohy:

$\left\{\begin{array}{ll} \leq 0 & \text{Algoritmus je neaktivní, realizuje se standardní zlom TEXu (implicitní hodnota).} \\ 1 & \text{Řádky jsou nalámány standardním algoritmem a pak je aplikován algoritmus prostrkání konce řádků na každý řádek.} \\ \geq 2 & \text{Přesah každého znaku vstupuje do celkového optimalizačního algoritmu zlomu odstavce, takže může dojít i k jinému řádkovému zlomu.} \end{array}\right.$

Registr pracuje jako odstavcové parametry, tj. význam má pouze jeho hodnota při ukončení odstavce. Z toho plyne, že algoritmus prostrkání okrajů nemůžeme aplikovat jen na část odstavce, ale vždy na celý najednou.

Prostrkáním je myšlena míra, o kolik daný znak přesahuje z levého, respektive pravého, okraje odstavce. Levý přesah se nastavuje pomocí

`\lpcode`⟨`font`⟩⟨`8-bit number`⟩⟨`equals`⟩⟨`hodnota`⟩

a pravý pomocí

`\rpcode`⟨`font`⟩⟨`8-bit number`⟩⟨`equals`⟩⟨`hodnota`⟩

kde ⟨`8-bit number`⟩ je ASCII hodnota znaku a ⟨`hodnota`⟩ přesahu je vztažena k velikosti písma, tj. ⟨`hodnota`⟩=1000 značí přesah o 1 em, ⟨`hodnota`⟩=200 určí přesah o ⅕ em. Přesah může nabývat i záporných hodnot.

Příklad: (hodnoty použité při druhém vysázení abstraktu)

```
\def\setupprot{\pdfprotrudechars=2
  \lpcode\font‘\T=80
  \rpcode\font‘\.=70 \rpcode\font‘\,=70
  \rpcode\font\hyphenchar\font=100 }
\itshape \setupprot Tento článek se věnuje ...
```

Hz-algoritmus (font expansion)

Typografické pravidlo říká, že odstavec působí esteticky a nenamáhá při čtení, pokud je v něm tiskařká barva rozprostřena rovnoměrně, takže při pohledu z dálky tvoří jednotnou šedou plochu. Tuto jednotnost mohou narušovat příliš stažené

[1] V příspěvku na SLT 2002 v Seči [7, 8] jsem tento algoritmus nazval *visící znaky* jako zobecnění termínu visící interpunkce. Pojmenování *prostrkání okrajů* však lépe vystihuje důvody zavedení tohoto algoritmu.

nebo příliš roztažené mezislovní mezery, zvláště při sazbě do úzkého bloku (do 40 znaků na řádce).

Použití *hz*-algoritmu[2] je trošku komplikovanější a vyžaduje kromě nastavení na úrovni makrojazyka pdfTEXu i přípravu expandovaných fontů[3].

Úroveň makrojazyka

Algoritmus se zapíná registrem `\pdfadjustspacing`. I on má tři polohy:

- ≤ 0 — Algoritmus je neaktivní, realizuje se standardní zlom TEXu (implicitní hodnota).
- 1 — Řádky jsou nalámány standardním algoritmem a pak při potřebě řádek stáhnout nebo natáhnout pruží vedle mezislovních mezer i jednotlivé znaky (na rozdíl od mezer však jen v diskrétních krocích).
- ≥ 2 — Možnost expanze každého znaku vstupuje do celkového optimalizačního algoritmu zlomu odstavce a je tak ovlivněn i řádkový zlom TEXu.

Registr je odstavcovým parametrem a jeho hodnota je brána v potaz jen při provedení povelu `\par`.

Druhým primitivem pro nastavení *hz*-algoritmu je

`\pdffontexpand`⟨`font`⟩⟨`stretch`⟩⟨`shrink`⟩⟨`step`⟩⟨`scalefactor`⟩

který fontu ⟨`font`⟩ přidělí maximální hodnoty roztažení ⟨`stretch`⟩ a stažení ⟨`shrink`⟩. Dále specifikuje diskrétní krok ⟨`step`⟩, s jakým má expanzi generovat. Tyto tři parametry se zadávají v promilích velikosti písma, tj. v tisícinách jednotky em.

Zatímco parametry ⟨`stretch`⟩, ⟨`shrink`⟩ a ⟨`step`⟩ určují, jaké metriky může pdfTEX při zlomu použít (tím se ovlivní, kolik horizontálního místa budou znaky zabírat), poslední parametr ⟨`scalefactor`⟩ na výpočet zlomu nemá vliv. Jde o koeficient, kterým se mění jen šířka kresby znaku. Hodnota 0 znamená, že znaky fontu budou vykresleny v původní neexpandované šířce a budou prostrkány tak, aby zabíraly zvolenou expandovanou šířku. Hodnota 1000 nastaví šířku kresby na expandovanou velikost. Hodnoty mimo interval 0 až 1000 nemají význam. Pro většinu případů je vhodné použít hodnotu 1000. Je-li výstup do DVI (`\pdfoutput=1`), není možné použít jinou hodnotu než ⟨`spacefactor`⟩=1000.

[2] V příspěvku na SLT 2002 v Seči jsem tento algoritmus nazval *horizontální zvětšování/zmenšování znaků*. Vycházel jsem z překladu termínu *font expansion*. Později jsem objevil termín *hz*-algoritmus. Protože je je tento název odvozen podle jeho tvůrce a je používán i mimo komunitu pdfTEXu [1], rozhodl jsem se tento název respektovat.

[3] Slovo *expandovaný* budeme dále používat jak ve významu horizontálního rozšíření, tak i stažení (expanze se zápornou hodnotou).

Pokud chceme nějaký znak horizontálně expandovat méně než bude vybraná expanze celého fontu, použijeme primitivu

`\efcode⟨font⟩⟨8-bit number⟩⟨equals⟩⟨hodnota⟩`

analogicky jako `\lpcode`. Znaku s ASCII kódem ⟨`8-it number`⟩ lze přiřadit relativní hodnotu v rozmezí 0 (znak expandovat nebude) až 1000 (bude expandovat jako celý font, implicitní hodnota).

Příklad:

```
\pdfadjustspacing=2
\pdffontexpand\tenrm 30 10 10 1000
```

Zapli jsme *hz*-algoritmus s globální optimalizací pro font `\tenrm`, s možností rozšíření fontu o 3 %, stažení o 1 % s krokem 1 %. Kromě základní metriky budeme tedy ještě potřebovat metriku `-10`, `+10`, `+20` a `+30`. Kresba se bude expandovat stejně jako vypočtená šířka, tj. nedojde *k mikroprostrkání*.

```
\pdffontexpand\font 15 0 5 0
```

Zde jsme nepovolili aktuálně nastavené písmo stahovat, ale pro roztažení jsme zvolili jemnější krok 0,5 %. Kresbu písma nechceme expandovat (znaky budou zprava prostrkány).

```
\pdffontexpand\font 15 0 5 500
\efcode\font‘\W=800
```

Kromě toho, že znaky budou expandovány jen na polovinu potřebného expandované místa (zbytek bude doplněn prostrkáním), jsme ještě u znaku *W* omezili expanzi o 20 %.

Příprava rozšířených metrik a bitmapových fontů

Má-li pdfTEX použít expandovaný font, potřebuje znát jeho metriku. Syntaxe názvu metriky je následující: `fontname-shrink.tfm` nebo `fontname+stretch.tfm`, takže například metrika $\mathcal{CS}$fontu Computer Modern Roman o velikosti 10 pt zúžená o 2 % musí být uložena v souboru s názvem `csr10-20.tfm`. Příprava metrik bude závislá na typu fontu. Pro některé z nich, zde uvádím návod:

METAFONT**ové fonty Computer Modern**

Knuth vytvořil Computer Modern pomocí parametrických souborů. Parametr `u#` určuje základní šířku znaků. Pro expandovaný font o -2 % stačí do parametrického souboru `cmr10-20.mf`, který vznikl přejmenováním `cmr10.mf`, připsat za řádek

```
u#:=20/36pt#;       % unit width
```

řádek

```
u#:=u#-20/1000u#;
```

Spustíme-li METAFONT na takto vytvořený soubor

```
mf \mode:=ljfour; mag:=1; input cmr10-20.mf
```

dostaneme jak metriku fontu `cmr10-20.tfm`, tak bitmapovou kresbu[4] `cmr10-20.600gf`. Komprimovanou bitmapu `cmr10-20.pk` získáme příkazem

```
gftopk cmr10-20.600gf
```

Soubory `.tfm` a `.pk` pak uložíme do náležitých adresářů (pro WEB2C instalaci `$TEX/texmf-local/fonts/ftm/` a `$TEX/texmf-local/fonts/pk/`) a přebudujeme databázi souborů (`mktexlsr`).

Pro české $\mathcal{CS}$fonty si zkopírujeme mírně upravený soubor `cscode.mf`, např. do adresáře `$TEX/texmf-local/fonts/source/public/cs/`. Pak už stačí jen připravit soubor s obsahem

```
input cscode
use_driver;
```

a názvem podle požadovaného fontu, např. `csr10-20.mf`. Dále postupujeme analogickým voláním

```
mf \mode:=ljfour; mag:=1; input csr10-20.mf
gftopk csr10-20.600gf
```

Pozor! Předpokladem je, že jsme předem připravili parametrický soubor originálního fontu `cmr10-20.mf`.

Type 1, bez překódování (např. anglické bez potřeby počeštění nebo Štormovy)

Pro PostScriptové Type 1 fonty nepotřebujeme generovat expandované kresby znaků (`.pfb`, `.pfa`), protože takový font si vytvoří pdfTEX automaticky pomocí transformační matice. Je pravda, že takovým způsobem se expanduje i šířka tahů a protože se tak mění světlost písma, není to teoreticky žádoucí postup. Protože však Type 1 fonty nemají parametrizaci oddělující šířku znaku a jeho tahů, nezbývá nám se s tímto nedostatkem smířit. Praxe ukazuje, že při malých hodnotách expanze fontu je změna tahů neznatelná[5].

Vysvětlili jsme si, že kresbami se zabývat nemusíme. Zbývá vytvořit metriky. Napsal jsem pro tento účel skript v Perlu `exppl.pl`, který využívá TEXových metrik (`.tfm`) neexpandovaného fontu. Expandovanou metriku připraví pronásobím horizontálních rozměrů `CHARWD`, `KRN`, `SPACE`, `STRETCH`, `SHRINK`, `QUAD`, `EXTRASPACE` uvnitř původní metriky. Nejdříve však musí metriku převést z komprimovaného tvaru do textové formy (Property List) pomocí programu `tftopl`. Po přepočítání rozměrů provede opačnou konverzi pomocí `pltotf` a soubor přejmenuje do požadovaného tvaru.

[4] Zde je použit `mode` pro laserovou tiskárnu LaserJet 4, 600 dpi, viz soubor `modes.mf`.

[5] Tento problém umožňují správně řešit Multiple Master fonty. Vzhledem k tomu, že Adobe ukončila jejich podporu a existuje velmi málo rodin písem v tomto formátu, návod na jejich použití zde neuvádím, ačkoliv pdfTEX jejich expandované použití umožňuje.

Příklad: Štormův font Preissig Antikva Roman rozšířený o 0,5 %, který se skládá ze dvou metrik:

```
fn=spar
exp=+5
tftopl ${fn}8z.tfm ${fn}8z.pl
tftopl ${fn}6s.tfm ${fn}6s.pl
exppl.pl ${fn}8z $exp
exppl.pl ${fn}6s $exp
pltotf ${fe}8z$exp.pl ${fe}8z$exp.tfm
pltotf ${fe}6s$exp.pl ${fe}6s$exp.tfm
```

Opět nezapomeňme vegenerované metriky umístit do vhodných adresářů a přegenerovat databázi. Expanzi fontu pak musíme zapnout pro obě metriky `8z` a `6r` třeba takto:

```
\input spreiss
\setfonts[PreissigAntikva/]

\newcount\N
\def\resetefcode#1{\N=0
  \loop\efcode#1\N=1000\advance\N by 1 \ifnum\N<256 \repeat}

\def\setupfont#1{\pdfadjustspacing=2
  \def\fotenc{6s}\setfonts[/]
  \resetefcode\font \pdffontexpand\font #1 #1 5 1000
  \def\fotenc{8z}\setfonts[/]
  \resetefcode\font \pdffontexpand\font #1 #1 5 1000 }

\setupfont{20} Text...
```

Používáme-li místo `L2` kódování `T1`, musíme metriku `8z` zaměnit za `8t`.

Virtuální fonty a překódování

Pokud potřebujeme generovat expandované metriky přímo z PostScriptových metrik (`.afm`), lze použít makrobalík `fontinst`. Skripty vytvořil Hàn Thế Thành. Předpokládají metriku v kódování `8a`, tj. musí se jmenovat *name*`8a.afm`. Pokud se font jmenuje jinak, je nutné vytvořit link s vhodným názvem.

Příklad: (Antykwa Torunska, italika, původní metrika `anttri.afm`)

```
ln -s anttri.afm anttri8a.afm
mktexlsr
mktextfm anttri8z+0
mktexlsr
pdftex.map
pdfcsplain file
```

Automatizované generování fontů

Co všechno k sazbě textu variabilními fonty potřebujeme?

- verzi pdfTEXu alespoň 0,14h z ledna 2001,
- rozšířený skript `mktextfm` a jeho další potomky `mktextfm.ext`, `mktfm8z` a `mktfmexstorm`,
- skript v Perlu `exppl.pl` pro Type 1 fonty,
- upravený soubor `cscode.mf` pro generování variabilních českých Computer Modern fontů a
- balík maker `fontinst` (je standardní součástí `tetexu`, `TEX Live` ap.) pro možnosti generování z `.afm` zdrojů.

S laskavým svolením původního autora většiny skriptů Hàn Thế Thành jsou k dispozici na adrese `ftp://cmp.felk.cvut.cz/pub/cmp/users/zyka/fe`. Poskytovány jsou bez jakýchkoliv záruk a garancí.

Reference

[1] Robert Bringhurst. *The Elements of Typographic Style.* Hartley & Marks, Point Roberts, WA, USA, version 2.4 edition, 2001.

[2] Hans Hagen. Does pdfTEX make things better?, 1999.

[3] Mirka Misáková. Písmo s variantní šířkou: nová naděje pro naše úzké sloupce. *Zpravodaj Československého sdružení uživatelů TEXu*, 8(2):65–81, 1998.

[4] Hàn Thế Thành. Bez názvu. `ftp://ftp.cstug.cz/pub/tex/local/cstug/thanh/hz/description.pdf`.

[5] Hàn Thế Thành. *Micro-typographic extensions to the TEX typesetting system.* PhD thesis, Masarykova univerzita v Brně, fakulta informatiky, 2000. Or: TUGboat 21,4, Dec 2000.

[6] Hermann Zapf. About micro-typography and the *hz*-program. *Electronic publishing*, 6(3):283–288, 1993. `http://cajun.cs.nott.ac.uk/compsci/epo/papers/volume6/issue3/zapf.pdf`.

[7] Vít Zýka. TEX a pdf. In Jan Kasprzak and Petr Sojka, editors, *SLT 2002 – sborník semináře o Linuxu a TEXu*, pages 69–77, Brno, Czech Republic, November 2002. Konvoj, CSTUG, CZLUG.

[8] Vít Zýka. TEX a pdf. *Zpravodaj Československého sdružení uživatelů TEXu*, 12(3–4), 2002. rozšřená verze [7].

Vít Zýka `zyka@cmp.felk.cvut.cz`