Zpravodaj Československého sdružení uživatelů TeXu

Jan Šustek

Sazba odstavců do textových oblastí Typesetting Paragraphs into Text Regions

Zpravodaj Československého sdružení uživatelů TeXu, Vol. 19 (2009), No. 3, 124–137

Persistent URL: http://dml.cz/dmlcz/150086

Terms of use:

© Československé sdružení uživatelů TeXu, 2009

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

Sazba odstavců do textových oblastí

Jan Šustek

Abstrakt

Při sazbě textu do textových oblastí vznikají problémy v případě, že oblasti mají různou šířku a v textu se vyskytují pružné vertikální mezery. Tento článek popisuje jedno z možných řešení uvedeného problému.

Klíčová slova: Textové oblasti, sazba, `\vsplit`.

1. Popis problému

Jedna z nejdůležitějších činností TEXu při sazbě textu je nalezení nejlepšího řádkového zlomu. Při lámání horizontálního seznamu odstavce do řádků TEX zpracovává najednou celý odstavec, případně jeho části oddělené

display matematikou.

TEX najde všechny vhodné řádkové zlomy. U každého potenciálního řádku spočítá, jak moc je nutné roztáhnout nebo stáhnout mezery. Poté spočítá badness tohoto řádku jako celé číslo b blízké číslu

$$100\left|\frac{s}{f_0}\right|^3, \qquad (1)$$

kde f_0 je přirozená a s je požadovaná pružnost mezer. Pokud je $b > 8\,191$, položí se $b := \infty$. Pokud se jedná o stažení a $b > 100$, položí se $b := 10\,000$. Na základě hodnoty badness se vypočítá demerits řádku podle vzorce

$$\bigl(\min\{l+b, 10\,000\}\bigr)^2 + p^2 \operatorname{sgn} p + d_0\,,$$

kde l je hodnota `\linepenalty`, p je penalta za zlom v daném místě a d_0 je přidaná hodnota demerits. Člen s penaltou se nepřičítá, pokud $p \le -10\,000$. Ze všech vhodných řádkových zlomů pak vybere ten s nejmenším součtem demerits.

Při kompletování odstavce TEX do vertikálního seznamu vkládá kromě samotných boxů s řádky také meziřádkové mezery, penalty a materiál z `\vadjust`.

Jako se horizontální seznam láme do řádků, láme se hlavní vertikální seznam na stránky. Pokud by se ale měly hledat všechny vhodné stránkové zlomy hlavního vertikálního seznamu celého dokumentu, bylo by to velmi náročné jak časově, tak paměťově. Proto TEX hledá možné zlomy pouze pro aktuální stranu. Opět pro každý zlom spočítá badness podle vzorce (1). Na základě hodnoty badness se spočítá cena daného zlomu podle vzorce

$$c = b + p\,,$$

*Práce na článku byla podpořena grantem 201/07/0191 Grantové agentury ČR.

kde p je penalta za zlom v daném místě. Člen s penaltou se nepřičítá, pokud $p \leq -10\,000$. Pokud $b = 10\,000$, položí se $c := 100\,000$. Do hledání vhodného zlomu dále mohou zasáhnout různé inzerty, jako například poznámky pod čarou nebo plovoucí objekty. Hledání končí v momentě, kdy je cena některého zlomu nekonečná. Výsledná stránka, která bude poslána výstupní rutině, bude ta, jejíž zlom měl nejmenší cenu. Podobně se vertikální seznam láme do `\vbox`ů primitivem `\vsplit`.

Uvedený algoritmus lze najít v [3], jeho zjednodušenou verzi s komentářem v [2] nebo [4]. Plyne z něj důležitá skutečnost. V době zpracovávání textu na aktuální straně není a nemůže být známo, kolik řádků textu na dané straně bude. Tato informace je známa až po zpracování odstavce majícího tolik řádků, že jej nebude možné celý umístit na aktuální stranu.

Problém nastává v případě, že řádkový zlom závisí na stránkovém zlomu. Pokud je šířka textu na následující straně jiná než na současné straně, pak problém dělá odstavec na zlomu strany. Kdybychom při sazbě odstavce věděli, že například první tři řádky budou na jedné straně a budou mít šířku 108 mm a zbývající řádky budou na druhé straně a budou mít šířku 170 mm, pak bychom použili `\parshape 4 0mm 108mm 0mm 108mm 0mm 108mm 0mm 170mm` a TEX by provedl řádkový zlom a nalámané řádky by poslal algoritmu stránkového zlomu. Jak již bylo poznamenáno, uvedenou informaci nemáme a získáme ji až po vložení řádků odstavce do vertikálního seznamu a po jeho rozlámání na stránky.

Stejný problém nastává při sazbě textu do textových oblastí. Mějme na straně dáno n obdélníkových oblastí. Úkolem je do těchto oblastí „nalít“ souvislý text. V článku popíšu možné řešení uvedeného úkolu. Výsledek je vidět na stránkách, které právě čtete.

V tomto řešení je odstavec při své sazbě vložen do `\vbox`u, a proto jsou všechna přiřazení uvnitř odstavce lokální a mají-li být globální, musí být použit prefix `\global`. Není možné, aby nějaká skupina začala uvnitř jednoho odstavce a končila uvnitř druhého. Při lámání odstavce se neprovádí expanze a odstavec je jako celek poslán algoritmu řádkového zlomu. Proto může obsahovat například příkazy pro změnu fontu nebo matematické vzorce. Ty navíc mohou být zlomeny, případně může být použito makro `\opakuj` z [1]. Dále je možné používat některé registry ovlivňující řádkový zlom, jako jsou `\leftskip`, `\rightskip`, `\parfillskip` nebo `\looseness`. Naopak nelze použít registry `\hangindent`, `\hangafter` a `\parshape`, protože parametry `\parshape` jsou programem předefinovány. Pokud jsou zaplněny všechny oblasti na stránce, začne se plnit další

strana se stejně rozmístěnými oblastmi. Za určitých okolností je možné, aby oblasti na další straně byly rozmístěny jinak. V tom případě je třeba ve vhodný čas znovu použít makro `\threadshape`.

Pokud jsou textové oblasti umístěny vedle sebe, může nastat kolize s typografickým pravidlem, které říká, že při sazbě do sloupců musí být dodržen řádkový rejstřík. To je způsobeno pružnými vertikálními mezerami, které mohou být vloženy například display matematikou.

Použití maker ze souboru OBLASTI3.TEX na sazbu tohoto článku je následující. Rámečky kolem textu byly nakresleny v METAPOSTu.

```
\documentclass ...
\input oblasti3
\begin{document}
...
\threadshape 3
 0cc 0cm 10cc 6cm
 12cc 0cm 15cc 6cm
 0cc 65mm 27cc 5cm
 0cc 65mm 27cc 43mm
 0cc 113mm 15cc 6cm
 17cc 113mm 10cc 6cm
...
\addonepar
Uvedený algoritmus ...
...
\shipoutlastpage
\section{Definice maker}
...
\end{document}
```

Pokud bychom pracovali v plainu a chtěli celý článek vysázet do oblastí, stačilo by napsat následující text.

```
\pageboxheight115mm
\addonepar
Jedna z~nejdůležitějších ...

\addonepar
Při kompletování odstavce ...
...
\threadshape 5
 0cc 0cm 10cc 6cm
 12cc 0cm 15cc 6cm
\input oblasti3
\threadshape ...
\addonepar
Jedna ...

\addonepar
Při kompletování ...
...
\bye
```

Význam jednotlivých maker je popsán v následující sekci. Celý soubor OBLASTI3.TEX je umístěn na [5].

2. Definice maker

2.1. Úvodní makra

2.1.1. Definice zkratek

Pro často používané konstrukce jsou nadefinovány tyto zkratky.

```
\let\ea\expandafter
\def\name#1{\csname#1\endcsname}
\def\namedef#1{\ea\def\csname#1\endcsname}
```

2.1.2. Alokace registrů

Čtyři čítače pro obecné výpočty není nutné speciálně pojmenovávat.

```
\newcount\tmpcounta
\newcount\tmpcountb
\newcount\tmpcountc
\newcount\tmpcountd
```

V čítači `\threadpart` je uloženo číslo aktuálně sázené oblasti. Počet všech oblastí je uložen v čítači `\threadparts`.

```
\newcount\threadpart
\newcount\threadparts
```

Čítače mající v názvu `SPpass` se používají při počítání průchodů makra `\splitparagraph`. Čítač `\SPpass` označuje číslo aktuálního průchodu. Čítač `\maxSPpasses` označuje maximální počet průchodů. Po počtu průchodů daném čítačem `\softSPpasses` se makrem `\setsoftpenalties` sníží některé penalty.

```
\newcount\SPpass
\newcount\maxSPpasses \maxSPpasses16
\newcount\softSPpasses \softSPpasses8
```

Pro správné vertikální mezery mezi odstavci je hodnota `\prevdepth` pro předchozí sázený odstavec uložena do registru `\prevprevdepth`.

```
\newdimen\prevprevdepth
```

V LaTeXu se oblasti sázejí do `\vbox`u výšky `\pageboxheight`. Implicitně je tato hodnota nastavena na 297 mm.

```
\newdimen\pageboxheight \pageboxheight297mm
```

Následující boxy se používají při práci. Ve `\vbox`u `\region` je uložena aktuáně sázená oblast. `\vbox \pagebox` obsahuje všechny dosud vysázené oblasti na stránce. `\vbox`y `\beginbox` a `\endbox` používají při lámání odstavce do oblastí.

```
\newbox\region
\newbox\pagebox
\newbox\beginbox
\newbox\endbox
```

2.1.3. Test formátu

Program funguje ve formátech plain a LaTeX. Některá makra se v jednotlivých formátech liší. Ke zjištění, v jakém formátu je program spuštěn, se používá následující test.

```
\def\ifplain{\ifx\documentclass\undefined}
```

2.1.4. Informace o průběhu

TeX může vypisovat informace o průběhu programu. Tyto informace může vypisovat na terminál nebo do LOG souboru. Podle toho se nastaví jeden z následujících řádků.

```
\def\showinfo#1{} % no info
\let\showinfo\wlog % info > log
\let\showinfo\message % info > terminal
```

2.2. Nastavení vlákna

Všechny textové oblasti jako celek budu nazývat vlákno. Tvar vlákna se nastaví makrem `\threadshape`, které má syntaxi podobnou primitivu `\parshape`. Po použití

$$\texttt{\threadshape}\ n␣x_1␣y_1␣w_1␣h_1␣\ldots␣x_n␣y_n␣w_n␣h_n␣$$

se bude vlákno skládat z n oblastí a i-tá oblast bude začínat na pozici (x_i, y_i) a bude mít šířku w_i a výšku h_i. Pozice $(0\,\mathrm{cm}, 0\,\mathrm{cm})$ odpovídá v plainu levému hornímu rohu stránky a v LaTeXu levému hornímu rohu `\vbox`u vkládaného na stranu.

```
\def\threadshape#1 {\threadparts#1
  \threadpart0
  \readthreadshape}
```

Tvar jedné oblasti se načte pomocným makrem `\readthreadshape`. Jednotlivé souřadnice se uloží do maker `\name{tsx`i`}`, `\name{tsy`i`}`, `\name{tsw`i`}` a `\name{tsh`i`}`, kde i je číslo oblasti.

```
\def\readthreadshape#1 #2 #3 #4 {\advance\threadpart1
  \namedef{tsx\the\threadpart}{#1}
```

```
\namedef{tsy\the\threadpart}{#2}
\namedef{tsw\the\threadpart}{#3}
\namedef{tsh\the\threadpart}{#4}
\ifnum\threadpart<\threadparts
  \ea\readthreadshape
\else
  \threadpart1
  \prevprevdepth0pt
\fi}
```

2.3. Sazba odstavce

2.3.1. Rozhraní

Sazba odstavce do aktuální oblasti je řešena makrem `\addonepar`.[1] Makro vloží text odstavce pro další použití do makra `\paragraphtext`. Dále opakovaně provede makro `\splitparagraph`. Po jeho ukončení odstavec naostro vysází s poslední hodnotou `\linesinregions`.

```
\def\addonepar#1\par{
  \def\paragraphtext{#1\unskip}
  \sejmiprvnislovo#1 \par
  \hsize\name{tsw\the\threadpart}
  \initLIR
  \getnormalpenalties
  \SPpass0
  \splitparagraph
  \setbox\endbox=\vtop{\unvcopy\region
    \prevdepth\prevprevdepth
    \theparshape\paragraphtext\par
    \global\prevprevdepth\prevdepth}
  \loop
    \setbox\region=\vsplit\endbox to\name{tsh\the\threadpart}
  \ifdim\ht\endbox>0pt
    \addtopagebox
  \repeat
  \setnormalpenalties}
```

Pokud je třeba vložit jiný vertikální materiál, například vertikální mezeru, použije se makro `\addtoregion`.

```
\def\addtoregion#1{\setbox\region=\vbox{\unvbox\region#1}}
```

[1] V tuto chvíli je program implementován tak, že se sekvence `\addonepar` musí napsat na začátek každého odstavce. Řešení tohoto problému pomocí `\everypar` není jednoduché, protože se odstavce sází opakovaně. Zatím se takové řešení autorovi nepodařilo najít.

2.3.2. Zlom do oblastí

Jádrem makra `\addonepar` je makro `\splitparagraph`. Toto makro se vykonává opakovaně, dokud se hodnota `\linesinregions` neustálí. Pokud se tak nestane během `\maxSPpasses` průchodů, vysází se odstavec podle poslední hodnoty `\linesinregions`, což bude pravděpodobně špatně. Po `\softSPpasses` průchodech se makrem `\setsoftpenalties` sníží hodnoty některých penalt.

Během jednoho průchodu makro `\splitparagraph` pokusně vysází odstavec při vypočtené hodnotě `\linesinregions` a `\parshape` a naláme jej do jednotlivých oblastí, počínaje (již částečně zaplněnou) aktuální oblastí. Na základě toho znovu spočítá `\linesinregions`. Pokud se hodnota `\linesinregions` liší od předchozí hodnoty, provede se další průchod makra `\splitparagraph`.

```
\def\splitparagraph{\advance\SPpass1
  \let\prevlinesinregions\linesinregions
  \genparshape
  \tmpcounta\threadpart
  \clearLIR
  \setbox\endbox=\vtop{\unvcopy\region
    \prevdepth\prevprevdepth
    \theparshape\paragraphtext\par
    \vfil}% to avoid underfull vboxes
  \loop
    \setbox\beginbox=\vsplit\endbox to\name{tsh\the\tmpcounta}
    \linesinbox\beginbox\tmpcountb
    \addtoLIR{\the\tmpcountb}
    \linesinbox\endbox\tmpcountb
  \ifnum\tmpcountb>0
    \advancemodTP\tmpcounta
  \repeat
  \addtoLIR{-1}
  \countlinesofparagraph
  \showinfo{^^JParagraph \prvnislovo ..., pass
    \the\SPpass: [\linesinregions]}%
  \ifx\linesinregions\prevlinesinregions
    \let\next\relax
  \else
    \ifnum\SPpass=\maxSPpasses
      \let\next\relax
      \message{^^J\string\splitparagraph does not converge!^^J}
    \else
      \ifnum\SPpass=\softSPpasses
        \setsoftpenalties
```

```
      \fi
      \let\next\splitparagraph
    \fi
  \fi
  \next}
```

Při výpisu činnosti programu je aktuální odstavec identifikován svým prvním slovem. Toto slovo se získá pomocným makrem `\sejmiprvnislovo`.[2]

```
\def\sejmiprvnislovo#1 #2\par{\def\prvnislovo{#1}}
```

2.4. Tvar odstavce

2.4.1. Počty řádků

Počty řádků aktuálního odstavce, které se vysází do jednotlivých oblastí, jsou uloženy ve frontě `\linesinregions`. Pokud je číslo aktuální oblasti i a `\linesinregions` je definováno jako

$$r_0; r_1; \ldots; r_k; -1;$$

znamená to, že aktuální odstavec má být rozdělen do oblastí $i, i+1, \ldots, i+k$, přičemž v oblasti $i+j$ bude jeho r_j řádků. Hodnota -1 slouží jako oddělovač.

Pro práci s frontou `\linesinregions` jsou definována následující čtyři makra.

```
\def\getoneLIR#1;#2\endLIR{\tmpcounta#1 \def\tmpLIR{#2}}
\def\clearLIR{\def\linesinregions{}}
\def\addtoLIR#1{\edef\linesinregions{\linesinregions#1;}}
```

Aby mělo makro `\splitparagraph` vždy alespoň dva průchody, je na začátku fronta `\linesinregions` nastavena na nesmyslnou hodnotu.

```
\def\initLIR{\def\linesinregions{-2;-1;}}
```

Pro posun na další oblast se používá makro `\advancemodTP`. Makro zvýší daný čítač o jedničku. Pokud by hodnota čítače byla větší než počet oblastí, nastaví se čítač na hodnotu 1.

```
\def\advancemodTP#1{\ifnum#1=\threadparts
  #1=1 \else \advance#1by1 \fi}
```

2.4.2. Generování tvaru

Řádky odstavce v různých oblastech mají různou šířku. Proto je nutné použít primitiv `\parshape`. Parametry `\parshape` se generují z fronty `\linesinregions` pomocí makra `\genparshape`. Po použití tohoto makra bude primitiv `\parshape` se všemi parametry uložen do makra `\theparshape`.

[2] Proč jsou názvy maker na řádku 91 v češtině, zjistí čtenář, pokud si je přeloží do angličtiny.

```
\def\genparshape{\def\theparshape{}
  \tmpcountb0
  \tmpcountc\threadpart
  \edef\tmpLIR{\linesinregions}
  \GPSloop}
```

Makro `\GPSloop` tvoří hlavní cyklus pro makro `\genparshape`.

```
\def\GPSloop{\ea\getoneLIR\tmpLIR\endLIR
  \ifnum\tmpcounta=-1
    \ea\GPSfinal
  \else
    \tmpcountd0
    \loop
    \ifnum\tmpcountd<\tmpcounta
      \advance\tmpcountd1
      \edef\theparshape{\theparshape0pt
        \name{tsw\the\tmpcountc} }
    \repeat
    \advance\tmpcountb\tmpcounta
    \advancemodTP\tmpcountc
    \ea\GPSloop
  \fi}
```

Makro `\GPSfinal` zakončuje generování `\parshape`.

```
\def\GPSfinal{%
  \edef\theparshape{\noexpand\parshape
    \the\tmpcountb\space \theparshape}}
```

2.5. Počítání řádků

2.5.1. Řádky v boxu

Makro `\linesinbox` spočítá počet řádků ve `\vboxu #1` a výsledek uloží do čítače `#2`. Pokud se ve `\vboxu` vyskytuje display matematika, počítá se příslušný řádek jako tři řádky.

```
\def\linesinbox#1#2{\setbox0=\vbox{\global#2=0
    \unvcopy#1
    \loop
      \unskipdisplay{#2}\unskipdisplay{#2}
      \setbox2=\lastbox
      \unskipdisplay{#2}\unskipdisplay{#2}\unskipdisplay{#2}
      \unpenalty
```

```
    \ifhbox2
      \global\advance#2by1
    \repeat}
  \ifdim\ht0>0pt
    \message{^^JERROR in \string\linesinbox^^J}
    {\showboxbreadth10 \showbox0}
  \fi}
```

Test na přítomnost display matematiky se provádí porovnáním velikosti vertikální mezery v boxu s velikostí mezery vkládané nad rovnici, která je při `\predisplaypenalty=10000` vždy přítomna, na rozdíl od mezery pod rovnicí. Aby nedošlo k záměně s jinými mezerami, jsou tyto mezery zvětšeny o 0,001 pt. V LaTeXu je nutné provést toto zvětšení až po `\begin{document}`.

```
\ifplain
  \advance\abovedisplayskip0.001pt
  \advance\abovedisplayshortskip0.001pt
\else
  \AtBeginDocument{\advance\abovedisplayskip0.001pt
    \advance\abovedisplayshortskip0.001pt}
\fi
```

Makro `\unskipdisplay` pracuje jako `\unskip` s testem na přítomnost display matematiky.

```
\def\unskipdisplay#1{
  \ifdim\lastskip=\abovedisplayskip \global\advance#1by2 \fi
  \ifdim\lastskip=\abovedisplayshortskip\global\advance#1by2\fi
  \unskip}
```

2.5.2. Řádky v odstavci

Při počítání řádků aktuálního odstavce v jednotlivých oblastech (`\vboxech`) bude první položka nesprávná, protože oblast bude obsahovat i řádky předchozích odstavců. Tuto první položku opraví makro `\countlinesofparagraph`.

```
\def\countlinesofparagraph{
  \setbox0=\vbox{\theparshape\paragraphtext\par
    \global\tmpcountc\prevgraf}
  \ea\getoneLIR\linesinregions\endLIR
  \let\linesinregions\tmpLIR
  \tmpcountb0
  \CLOPloop}
```

Makro `\CLOPloop` tvoří hlavní cyklus pro makro `\countlinesofparagraph`.

```
\def\CLOPloop{\ea\getoneLIR\tmpLIR\endLIR
  \ifnum\tmpcounta=-1
    \ea\CLOPfinal
  \else
    \advance\tmpcountb\tmpcounta
    \ea\CLOPloop
  \fi}
```

Makro `\CLOPfinal` zakončuje práci makra `\countlinesofparagraph`. Pokud by vyšlo `\tmpcounta<0`, znamená to, že se předchozí odstavec při zpracovávání současného odstavce rozdělil do více oblastí, než byl rozdělen původně. Tomu je třeba zabránit vložením `\penalty-10000`.

```
\def\CLOPfinal{\tmpcounta\tmpcountc
  \advance\tmpcounta-\tmpcountb
  \edef\linesinregions{\the\tmpcounta;\linesinregions}%
  \ifnum\tmpcounta<0
    \addtoregion{\penalty-10000 }
  \fi}
```

2.6. Nastavení penalt

Může se stát, že ani po několika průchodech makra `\splitparagraph` se hodnota `\linesinregions` neustálí. To může nastat, pokud se blízko hranice oblasti nachází display matematika nebo hranice odstavce a pokud se do zlomu do oblastí zapojí některá z níže uvedených penalt. Na to zareaguje makro `\splitparagraph` tím, že tyto penalty sníží. Tím se sníží typografické požadavky za cenu konvergence algoritmu.

```
\newcount\oriclubpenalty
\newcount\oriwidowpenalty
\newcount\oridisplaywidowpenalty
\newcount\oribrokenpenalty
```

Makro `\getnormalpenalties` zjistí původní hodnoty penalt.

```
\def\getnormalpenalties{\oriclubpenalty\clubpenalty
  \oriwidowpenalty\widowpenalty
  \oridisplaywidowpenalty\displaywidowpenalty
  \oribrokenpenalty\brokenpenalty}
```

Makro `\setnormalpenalties` nastaví původní hodnoty penalt.

```
\def\setnormalpenalties{\clubpenalty\oriclubpenalty
  \widowpenalty\oriwidowpenalty
```

```
  \displaywidowpenalty\oridisplaywidowpenalty
  \brokenpenalty\oribrokenpenalty}
```

Makro `\setsoftpenalties` sníží penalty.

```
\def\setsoftpenalties{\clubpenalty0
  \widowpenalty0
  \displaywidowpenalty0
  \brokenpenalty0 }
```

2.7. Umístění na stranu

Jednotlivé `\vbox`y s oblastmi jsou na správné pozici umístěny v boxu `\pagebox`. Do tohoto boxu se box s aktuální oblastí vloží makrem `\addtopagebox`. Pokud jsou zaplněny všechny oblasti na straně, vysází se `\pagebox` na aktuální stranu a začne se plnit další strana.

```
\def\addtopagebox{\setbox\pagebox=\vtop{\unvbox\pagebox
  \dimen2=\name{tsy\the\threadpart}
  \vskip\dimen2
  \vtop to 0pt{
    \moveright \name{tsx\the\threadpart} \box\region
    \vss}
  \vskip-\dimen2 }
  \showinfo{Exporting region \the\threadpart^^J}
  \advancemodTP\threadpart
  \ifnum\threadpart=1
    \forceoutput
  \fi}
```

Makro `\forceoutput` vysází `\pagebox` na aktuální stranu. V plainu zároveň zastupuje výstupní rutinu a stranu vloží do DVI/PDF souboru. V LaTeXu vloží `\pagebox` do `\vbox`u výšky `\pageboxheight`, ten vloží na aktuální stranu a vyvolá výstupní rutinu.

```
\ifplain
  \def\forceoutput{\shipout\vbox{\vskip-1in
      \moveright-1in\box\pagebox}%
    \global\advance\pageno1 }
\else
  \def\forceoutput{\vbox to \pageboxheight{\box\pagebox\vss}
    \pagebreak}
\fi
```

Zbývá vyřešit sazbu poslední strany. V plainu program očekává, že se celý dokument sází do oblastí. Proto je sazba poslední strany řešena předefinováním

makra `\bye`. Aktuální oblast se vloží do boxu `\pagebox` a tento box je vložen do DVI/PDF souboru. V případě `\threadpart=1` byla aktuální oblast poslední na straně a makro `\forceoutput` již bylo zavoláno z makra `\addtopagebox`.

```
\ifplain
  \let\oribye\bye
  \def\bye{%
    \setbox\region=\vtop{\unvbox\region}%
    \addtopagebox
    \ifnum\threadpart>1 \forceoutput\fi
    \name{oribye}}
```

V LaTeXu je vložení strany do DVI/PDF souboru záležitostí výstupní rutiny a očekává se, že dokument bude obsahovat i text, který se nesází do oblastí. Poslední strana s oblastmi se vysází makrem `\shipoutlastpage`. Pokud se má potlačit stránkový zlom za poslední stranou sázenou do oblastí, stačí před použitím `\shipoutlastpage` lokálně definovat `\let\pagebreak\relax`.

```
\else
  \def\shipoutlastpage{%
    \setbox\region=\vtop{\unvbox\region}%
    \addtopagebox
    \ifnum\threadpart>1 \forceoutput\fi}
\fi
```

Závěr

V článku je řešen problém sazby textu do textových oblastí. Pokud se v textu vyskytují pružné vertikální mezery, není dopředu známo, kolik řádků odstavce v jednotlivých oblastech bude. Toto je známo až po vysázení odstavce, jehož řádkový zlom ovšem závisí na informaci o počtu řádků. Popisovaná makra uvedený problém řeší iteračně – odstavec zkusmo vkládají do oblastí tak dlouho, až se jeho zlom do oblastí ustálí. Poté vysází odstavec naostro. V případě, že jsou zaplněny všechny oblasti na straně, začne se plnit nová strana se stejně umístěnými oblastmi.

Seznam literatury

[1] Horák, Karel. Sazba matematiky v českých textech. [Typesetting Mathematics in Czech Texts.] *Zpravodaj Československého sdružení uživatelů TEXu* [The Bulletin of the Czechoslovak TEX Users Group], 11(1–3):136–148, 2001. ISSN 1211-6661. Dostupné na `http://bulletin.cstug.cz/pdf/bul_0113.pdf`, `ftp://ftp.cstug.cz/pub/tex/local/cstug/horak/slt98/OPAKUJ.TEX`

[2] Knuth, Donald Ervin. *The TEXbook* (Computers and Typesetting, Vol. A). Reading, Massachusetts: Addison-Wesley, 1984. ISBN 0-201-13448-9.

[3] Knuth, Donald Ervin. *TEX: The Program* (Computers and Typesetting, Vol. B). Reading, Massachusetts: Addison-Wesley, 1986. ISBN 0-201-13437-3. Dostupné z FTP serveru `ftp://tug.ctan.org/pub/tex-archive/systems/knuth/dist/tex/tex.web`

[4] Olšák, Petr. *TEXbook naruby*. [TEXbook Inside Out.] 2. vyd. Konvoj 2001, ISBN 80-7302-007-6.

[5] Šustek, Jan. *Sazba odstavců do textových oblastí – zdrojový kód, verze 3*. [Typesetting Paragraphs into Text Regions – Source Code, Version 3.] [on-line] [cit. 8. 7. 2009] Dostupné na `http://sustek.wz.cz/TeX/oblasti3.tex`

Summary: Typesetting Paragraphs into Text Regions

When typesetting paragraphs of text into text regions some problems arise in cases where the regions have different widths and rubber vertical spaces are used. In this paper one possible solution to the problem is described.

Key words: Text Regions, Typesetting, `\vsplit`.

Jan Šustek, jan.sustek@osu.cz
Ostravská univerzita, Přírodovědecká fakulta
Katedra matematiky, 30. dubna 22, Ostrava
CZ-701 03, Czech Republic