

ソフトウェアアーキテクチャ – 手引き –

萩野達也

2007年 春学期・4回

4 文書清書システム

今回は文書清書システムとして有名な $\text{T}_\text{E}\text{X}$ について見てみよう。

4.1 $\text{T}_\text{E}\text{X}$ の始まり

文書清書システムは計算機システムの初期の段階から存在する。UNIX 上では `roff` というシステムが有名である。UNIX のオンラインマニュアルは `roff` を使っすべて書かれている。`roff` は Multics の `runoff` の後継として UNIX の開発者である Dennis Ritchie が UNIX 上に実装したもので、UNIX の特徴を利用し、テーブルや数式は別のプリプロセッサで処理し、パイプで渡すなどの処理をしている。`roff` 自身は清書システムの総称で、`nroff`, `troff`, `groff` などが実際のプログラム名である。

$\text{T}_\text{E}\text{X}$ は Donald Knuth によって作成された文書清書システムである。Knuth は自分の著書「The Art of Computer Programming (基本算法)」の中でさまざまな数式を組版する必要があり、その校正などに手間取ったことから、著者が最終的な組版までを制御できるようなシステムを作ることにしたのが $\text{T}_\text{E}\text{X}$ である。組版だけではなくさらに文字フォントも自身で作成できるように METAFONT と呼ばれるシステムも作ってしまった。

また $\text{T}_\text{E}\text{X}$ のプログラムは単にプログラムとして書かれているのではなく、プログラムとそのプログラムを説明するドキュメントを合わせて WEB という形式で書いている。プログラムを作成してからドキュメントを作成するのではなく、同時に作成する。プログラムへの単なるコメントでもなく、文書とプログラムが織り重なったもので、Knuth 自身は「Literate Programming (文芸的プログラミング)」と読んでいる。

```
1. 中心アルゴリズム
do_something() は中心となる処理内容である .
<a routing> ==
    item.do_something().

2. メインループ
あるコレクションの全ての内容を<中心アルゴリズム>で処理する .
<main> ==
    for item in collection
    <a routing>
```

(Wikipedia から引用)

を処理すると

```
for item in collection
    item.do_something().
```

というプログラムを得ることができる。

ドキュメント ← WEB → プログラム

4.2 $\text{T}_\text{E}\text{X}$ の例

$\text{T}_\text{E}\text{X}$ は WYSIWYG (What You See Is What You Get) ではない。ソースを用意しておき処理することによって清書される。

$\text{T}_\text{E}\text{X}$ のソース → DVI → 印刷

$\text{T}_\text{E}\text{X}$ 文書のソースはテキストとマクロの集まったものである。マクロは「`\`」で始まる。

```
This is a {\it very} {\bf simple} \TeX{}
source file. We can write equation like
$a x^2 + b y + c = 0$ and
$$1+2^2+3^3+\cdots+n^2 = \sum_{i=1}^n i^2
= \frac{n(n+1)(2n+1)}{6}$$
easily.
```

清書すると次のようになる。

```
This is a very simple  $\text{T}_\text{E}\text{X}$  source file. We can
write equation like  $ax^2 + by + c = 0$  and

$$1 + 2^2 + 3^3 + \cdots + n^2 = \sum_{i=1}^n i^2 = \frac{n(n+1)(2n+1)}{6}$$

easily.
```

4.3 文書清書に機能

文書清書は文書をきれいに印刷することだが、そのためには次のような機能が必要である。

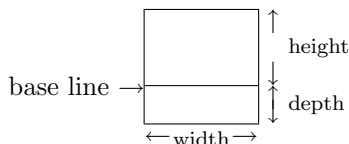
- 文字フォントの種類指定 (明朝, ゴシック, イタリアンなど)
- 文字の大きさの指定
- 行の右寄せ, 左寄せ, センタリング, 均等割付け
- 1行をはみ出す場合に, 複数行に分ける (禁則処理, ハイフネーション)
- 1ページをはみ出す場合に, 複数ページに分ける
- ページに図や表を入れる
- ページ番号, ヘッダなどを付ける
- 章や節の番号をふる
- 目次の作成
- 参考文献, 索引の作成とその引用

練習問題 4-1

普段使っている文書清書のシステムの機能にどんなものがあるか調べなさい。

4.4 箱

TeX では箱 (box) を横や縦に積み重ねながら清書をしていく。一番小さな箱が文字である。箱には width と height と depth がある。

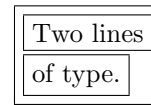


箱を横につなげる時には、ベースラインのところまでつなぎ合わされる。英文字には depth があるものがあるが、漢字にはないので、英文字と漢字のつながり方は良く見ると変である。特に全角の ‘ (’ と半角の ‘ (‘ の違いはこの点から来ている。全角の方が縦に短い。

箱は横につなげた箱 hbox と縦につながる箱 vbox がある。

```
\vbox{\hbox{Two lines}\hbox{of type.}}
```

は次のように箱が縦につながる。



4.5 グルー

箱とはこの間にはグルー (glue) とよばれる伸び縮みする接着剤でくっつけられる。グルーはもともとの幅である space と伸びる量 stretch と縮む量 shrink からなり、行やページを構成する時に伸び縮みさせられる。たとえば、横方向のグルーは次のようになる。

```
\hskip 10mm plus 8mm minus 3mm
```

これはもともとの大きさは 10mm であるが、必要に応じて 7mm から 18mm になってもかわらないことを意味している。均等割りつけの場合には、伸び縮み部分が均等に行なわれるように個々のグルーの伸び縮みが調整される。

また、無限に伸びるグルーも用意されている。マクロ `\hfil` は

```
\hskip 0pt plus 1fil
```

の意味で、横方向に無限に伸びる。これを使って左揃え、右揃え、センタリングが可能になる。

```
\line{左寄せ\hfil}  
\line{\hfil 右寄せ}  
\line{\hfil センタリング\hfil}
```

無限にも `fil`, `fill`, `filll` の 3 種類が用意されていて、後のものほど強力な無限になり、それより小さな無限は無視される。

練習問題 4-2

無限に複数の種類が用意されているのはどうしてか?

4.6 行への分割

入力の記事は改行あるなしに位置に関係なく横につながれていく。改行が英文字間の場合には空白に変換され、漢字の間では除去される。段落は空行 (改行が 2 つ以上連続) によって区切られる。TeX は段落を構成する文字と空白 (グルー) を読み込んだ後に、段落の横幅に合わせて行に分割していく。

行に分割する際には、その段落を構成する行の badness がもっとも小さくなるように分割する。

- 1 行の badness は、その行に含まれるグルーが伸び縮みしないといけな割割合の 3 乗に 100 をかけたものとする。10000 を越えた場合には 10000 とする。
- 行に分割するばあいには、分割点によって penalty が課せられている場合がある。行の badness を b 、分割点の penalty を p としたとき、この分割のデメリット d は次のように計算される。

$$d = \begin{cases} (l+b)^2 + p^2 & (0 \leq p < 10000) \\ (l+b)^2 - p^2 & (-10000 < p < 0) \\ (l+b)^2 & (p \leq -10000) \end{cases}$$

ここで l は `\linepenalty` で与えられた行のデフォルトの badness である。初期値は 10 となっている。大きくすると、なるべく少ない行数にしようとする。

badness の合計が最小になるように行を分割する問題は、一つの最小化問題であり、あらゆる分割を考えてそれから最小のものを見つけようとすると計算量が大きくなる。TeX では動的計画法 (Dynamic Programming) によって解決している。

4.7 ページの分割

行の集まりをページに分割するのも行の分割と基本的に同じである。badness と penalty からページのコストを計算し、コストが小さくなる点で分割する。行と異なり文書全体を見てページ分割のコストを最小化するのではなく、局所的な最小化を行なう。

TeX は `\vsize` で指定された高さの箱が出来上がった段階で、`\output` を呼び出す。出来上がった行は `\vbox255` に入れられているので、ヘッダやフッタを付加して `\shipout` すれば良い。もっとも単純な `\output` は次のようになる。

```
\def\output{\shipout\box255}
```

出来上がったページをヘッダなどを付けずにそのまま DVI として出力する。図や表がページの一部に入る場合には、`\vbox255` の行全てを出力できないため、一部を次のページに送ることもできる。

4.8 マクロ

TeX では出力の時のヘッダやフッタの指定、図や表の挿入、章や節の番号付けなど、プログラムに近い複

雑なことを行なわなくてはいけない。このため、マクロ (macro) と呼ばれる方法でプログラムを書くことができるようになっていく。

もっとも単純なマクロは、たんに文字列に名前を付けるだけである。

```
\def\sfc{湘南藤沢キャンパス}
```

この定義以降、`\sfc` とすると「湘南藤沢キャンパス」に置き換わる。

マクロには引数を与えることも可能である。

```
\def\sfc#1#2{SFC#1 棟#2 階}
```

`\sfc A3` とすると「SFC A 棟 3 階」となる。引数が 1 文字でないばあいには、`{ }` で囲う必要がある。`\sfc{中高}{2}` は「SFC 中高棟 2 階」となる。

マクロの本体の中で別のマクロを呼び出すことも可能で、再帰的に呼び出すことによってループを作ることにもできる。たとえば、次のマクロでは与えられた数字を 3 桁毎に、`'` で区切るものである。

```
\def\money#1{{\ifnum#1<0$\triangle$\count3=-#1
\else\count3=#1\fi\count4=0\mloop}}
\def\mloop{{\count0=\count3 \divide\count3 by 10
\advance\count4 by 1
\ifnum\count4=3 \count4=0\fi
\ifnum\count3>0 \mloop\ifnum\count4=0 ,\fi\fi
\count2=\count3 \multiply\count2 by -10
\advance\count0 by\count2 \number\count0}}
```

4.9 L^AT_EX

この資料は L^AT_EX を使って書かれているが、L^AT_EX は TeX に便利なマクロを追加したものである。TeX ではマクロを定義することによって全てをユーザが作る必要がある。例えば、章や節の扱いや、ページのヘッダやフッタなどもすべて自分で定義しておく必要がある。これは使いたくないものにとっては不便である。そのため、L^AT_EX では本やレポートなどの標準的な文書に関してスタイルのためのマクロを他数定義している。また、環境という使い方を導入し、

```
\begin{itemize}
\item 最初の項目
\item 2 番目の項目
\end{itemize}
```

のように、始まりと終りを `begin` と `end` で囲む。HTML におけるタグ付けと類似し、構造化された文書を書き易くしている。

4.10 DVI と PS と PDF

T_EX (および L^AT_EX) の出力形式は DVI (DeVice Independent) であるが、DVI を直接入力として受け付けるプリンタは少ないため、通常 PS (Postscript) や PDF (Portable Document Format) にさらに変換して使うことが多い。

PostScript は Adobe が作ったページ記述言語であり、実際には描画命令が集まったプログラムである。

```
%!
RRECT { newpath 4 copy pop pop moveto dup 0 exch
  rlineto exch 0 rlineto neg 0 exch
  rlineto closepath pop pop } def

100 100 100 150 RRECT
.5 setgray
fill

100 300 moveto
/Helvetica findfont
12 scalefont
setfont
.5 0 .5 0 setcmykcolor
(test string) show

showpage
```

(Wikipedia から引用)

PDF も Adobe が定義したものであるが、PostScript とは異なりプログラムではなくオブジェクトの集まりである。このため、処理を行なうのは PostScript よりも単純である。

練習問題 4-3

T_EX や L^AT_EX を使って文書を書き、それを処理して印刷したり PDF を作成したりしなさい。

参考文献

- 「T_EX ブック 改定新版」Donald E. Knuth, 斎藤信男監修, 鷺谷好輝訳, アスキー出版局, 1992 年, ISBN 4-7561-0120-8
- 「文芸的プログラミング」Donald E. Knuth, 有澤誠訳, アスキー出版局, 1994 年, ISBD 4-7561-0190-9