

TEX を使用した自動清書システムについて

岡山理科大学 理学部

坂本 薫 (Kaoru SAKAMOTO)

概要

最近,TEX や Jtex 等の清書システムが実用化され, 数式処理システムの一部では結果の数式が TEX に出力可能である。しかし, 数理解析の分野では, 最終結果は数式になる場合が多く, その数式は複雑なものが多く, 既存のシステムでは, 全てを自動的に清書するのは不可能に近い。数式を一般の活版印刷のように清書するのは必要である。しかし, 得られた数式を TEX のコマンドで書き直すのは非常に煩わしい。又, 自動処理化を目指しているものにとって, TEX に引き渡すための変換プログラムは必要である。試みに著者は文字列を処理する関数が豊富な Common Lisp でプログラムを作成した。Common Lisp では, 文字アトム名は小文字, 大文字だけのいずれかの選択性になっており, 大文字, 小文字, 特殊文字が混在する数式の変数名には, 使えないので, 変数名を表すのに文字列が使用された。又, 数式を表すデータはフォトラン文中の数式の書式と同じものと仮定した。

1. 文字列から変数および演算子を要素にするリストへ変換

数式を含むファイル中のデータの一行分取りだし, 文字列として出力させる。得られた文字列を変数, 演算子を分離するために, それらを文字列である要素とするリストに変換する。数式中の括弧対 () は 2 重の括弧対 (()) に変換される。sin 等のような引数を持つ関数は引数からなるリストを数式を表すリストの要素とする。例, "a+b" ==> ("a" + "b"), "(a+b)" ==> (("a" + "b")), "sin(x)" ==> (sin ("x"))。ここで, "+", "sin" 等は後の処理を円滑にするためにそれぞれ +, sin に変換された。又, 数を表す文字列は実数, 整数, 分数の数アトムにそれぞれに変換される。

2. 挿入辞記法から接頭辞記法への変換

得られたリストにたいして, +, * 等の二項演算子は挿入辞記法で書かれているので, 接頭辞記法に変換しなければ, しなければならない。又, 演算子の優先順位については従来通りに仮定する。例えば, ("a" + "b") ==> (+ "a" "b"), ("a" + "b" * "c") ==> (+ "a" (* "b" "c")) 他方, 単項演算子の -, sin 等は直後に引数のリストが来るものとする。例, (sin ("x")) ==> (sin ("x")), ("a" + sin("x")) ==> (+ "a" (sin ("x"))) 単項演算子は先頭に演算子が現れるから, 演算子によって処理が二つに分けられた。算術演算子の他に関係演算子も数式中に現れるのに許される。関係演算子は, >, <, ≥, ≤ が主に使われる。又, 論理演算子 and, or, not も数式中に現れるのを許す。これは, 例えば, x の範囲を示す $m \leq x \leq n$ は $m \leq x$ and $x \leq n$ を表すから, 論理演算子が必要である。

3. 数式中使用される文字

数式で使用される変数はアルファベットの小文字で次のようなフォントのアルファベットである。

a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z

又, 行列等は, アルファベットの太文字で表現される事が多く次のようなフォントで表現される。

A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z

ギリシャ文字の小文字, 太文字は, 数理解析の分野で頻繁に現れる。ギリシャ文字の小文字の一部にアルファベットの太文字と同じ字体のものがある。それらを除いたギリシャ文字の小文字は次の通りである。

α, β, γ, δ, ε, ζ, η, θ, ι, κ, λ, μ, ν, ξ, π, ρ, σ, τ, υ, φ, χ, ψ, ω

ギリシャ文字の小文字 omicron は o と同じ字体に見えるので, 特別にそれを表すフォントは存在しない。数式には, 他の字体のギリシャ文字の小文字があり, 使用される。

ε, ϑ, ω, ρ, ς, ϕ

ギリシャ太文字は次のようなフォントが用意されている。

Γ, Δ, Θ, Λ, Ξ, Π, Σ, Υ, Φ, Ψ, Ω

他のギリシャ大文字はアルファベットの大文字のロマン体と同じである。

$A, Z, H, I, K, M, N, O, P, X$

数式には、よく下付きの添字付きの変数が使われる。それらは次のように変数の直後に下付きの添字として現われる。例えば、 x_1, y_2, z_3 数理論理学の分野では、ベクトルが頻繁に現われる。それらは次のように文字の上に右向きの矢印を置く。例えば、 $\vec{a}, \vec{b}, \vec{c}$ ベクトルを表すのにアルファベットの文字を太字にする方法があるが、ここでは、文字の上に矢印を置く方法を取る。

4. TeX による単純な数式の清書

TeX による数式の清書の例を示そう。まず、冪乗の例から始める。それは、 x^n のように書く。

$$x^2$$

$$x^a$$

$$2^x$$

$$x^2 y^2$$

分数は、plain TeX では、 $x \over y$ と書く。分数の例は次の通りである。

$$\frac{a+1}{a-1}$$

$$\frac{x}{\frac{1+x}{2}}$$

平方根は、根号を使う。 $\sqrt{x}$ のように書く。平方根の例は次のようである。

$$\sqrt{2}$$

$$\sqrt{x+2}$$

$$\sqrt{x^3 + \sqrt{\alpha}}$$

他の根号は、 $\sqrt[n]{x}$ を使う。これは、 $\sqrt[n]{x}$ のように書く。

$$\sqrt[n]{x}$$

$$\sqrt[n]{x^n + y^n}$$

$$\sqrt[n+1]{a}$$

$\sum$ は総和を示す。 $\sum$ は上限、下限を持ち、通常下付きの添字、上付きの添字の形で指定する。 $\sum_{n=1}^m x_n$ のように書き、次のように清書される。

$$\sum_{n=1}^m x_n$$

積分の記号は、 $\int$ を使用する。上付きの添字と下付きの添字は、上限、下限を与える。 $\int_m^n f(x) dx$ のように書く。次のように清書される。

$$\int_m^n f(x) dx$$

絶対値は、 $|x|$ を使い、 $|x|$ のように清書される。微分は、 $\frac{dy}{dx}$ のように書き、 $\frac{d^2 y}{dx^2}$ のように清書される。2回以上の微分は、 $\frac{d^2 y}{dx^2}$ のように書かれ、 $\frac{d^2 y}{dx^2}$ のように清書される。TeX では、種々な大きさのデミリットが用意されている。

$$(), [], \{$$

が用意されている。少し大きい、デリミッタが用意されている。

(,),[,]

もっと大きいデリミッタも用意されている。

(,),[,]

これらは、分数を含む場合に使用される。デリミッタに含まれる式によって、デリミッタの大きさを適当に決めるには、`\left(`と`\right)`等を使う。数式によく現われる関数名は、`TeX` に用意されている。

`sin cos cot exp ln arcsin arccos arctan sinh cosh tanh log`

`TeX` には種々な大きさの空白がある。

`\`, 小さい空白 (quad の 1/6)

`\>` 中間の空白 (quad の 2/9)

`\;` 大きい空白 (quad の 5/18)

`\!` 負の小さい空白 (quad の -1/6)

5. Common Lisp による文字処理

文字列を扱う主な関数は、文字列の要素である文字をアクセスする `char`, 二つの文字列が大文字と小文字を区別して、等しいかどうか調べる `string=`, 文字列中の部分文字列を取り出す `subseq`, 二つ以上の文字列を連結して一つの文字列にする `concatenate`, 文字列の長さを返す `length`, 文字列を反転させる `reverse`, 文字列中に、もし該当する文字があれば、その先頭の位置を返し、存在しなければ `NIL` を返す `position` 等がある。これらの関数の中で文字列の文字の位置を表すのに、左から最初の位置を 0 として正整数が用いられる。`TeX` では、ギリシャ文字のアルファベットのシンボル名は長いので、以下のように短くされた。

シンボル名	新しいシンボル名 (小文字)	活字 (小文字)	新しいシンボル名 (大文字)	活字 (大文字)
alpha	<code>\al</code>	α	<code>\AL</code>	A
beta	<code>\be</code>	β	<code>\BE</code>	B
gamma	<code>\ga</code>	γ	<code>\GA</code>	Γ
delta	<code>\de</code>	δ	<code>\DE</code>	Δ
epsilon	<code>\ep</code>	ϵ	<code>\EP</code>	E
zeta	<code>\ze</code>	ζ	<code>\ZE</code>	Z
eta	<code>\et</code>	η	<code>\ET</code>	H
theta	<code>\th</code>	θ	<code>\TH</code>	Θ
iota	<code>\io</code>	ι	<code>\IO</code>	I
kappa	<code>\ka</code>	κ	<code>\KA</code>	K
lambda	<code>\la</code>	λ	<code>\LA</code>	Λ
mu	<code>\mu</code>	μ	<code>\MU</code>	M
nu	<code>\nu</code>	ν	<code>\NU</code>	N
xi	<code>\xi</code>	ξ	<code>\XI</code>	X
omicron	<code>\om</code>	$\omicron$	<code>\OM</code>	O
pi	<code>\pi</code>	π	<code>\PI</code>	Π
rho	<code>\rh</code>	ρ	<code>\RH</code>	P
sigma	<code>\si</code>	σ	<code>\SI</code>	Σ
tau	<code>\ta</code>	τ	<code>\TA</code>	T
upsilon	<code>\up</code>	υ	<code>\UP</code>	Υ
phi	<code>\ph</code>	ϕ	<code>\PH</code>	Φ
chi	<code>\ch</code>	χ	<code>\CH</code>	X
psi	<code>\ps</code>	ψ	<code>\PS</code>	Ψ
omega	<code>\om</code>	ω	<code>\OM</code>	Ω
シンボル名	新しいシンボル名	活字		
varepsilon	<code>\vep</code>	ϵ		
vartheta	<code>\vth</code>	ϑ		
varpi	<code>\vpi</code>	ϖ		

varrho	\vrh	\varrho
varsigma	\vsi	\varsigma

文字列に含まれるシンボル名から、表 greek-symbols 調べて、シンボル名と一致するキーから、TeX のシンボル名を取り出す必要がある。この表は容易に修正を許すように次のような連想リストで表された。ここで、\\ は文字列中において、一つの \ を表すために使用されている。

(setf greek-symbols

```
((("\\al" "\\alpha") ("\\AL" "A") ("\\ba" "\\bata") ("\\BA" "B")
  ("\\ga" "\\gamma") ("\\GA" "\\Gamma") ("\\de" "\\delta") ("\\DE" "\\Delta")
  ("\\ep" "\\epsilon") ("\\EP" "E") ("\\ze" "\\zeta") ("\\ZE" "Z")
  ("\\et" "\\eta") ("\\ET" "H") ("\\th" "\\theta") ("\\TH" "\\Theta")
  ("\\io" "\\iota") ("\\IO" "I") ("\\ka" "\\kappa") ("\\KA" "K")
  ("\\la" "\\lambda") ("\\LA" "\\Lambda") ("\\mu" "\\mu") ("\\MU" "M")
  ("\\nu" "\\nu") ("\\NU" "N") ("\\xi" "\\xi") ("\\XI" "X")
  ("\\om" "o") ("\\OM" "O") ("\\pi" "\\pi") ("\\PI" "\\Pi")
  ("\\rh" "\\rho") ("\\RH" "P") ("\\si" "\\sigma") ("\\SI" "\\Sigma")
  ("\\ta" "\\tau") ("\\TA" "T") ("\\up" "\\upsilon") ("\\UP" "\\Upsilon")
  ("\\ph" "\\phi") ("\\PH" "\\Phi") ("\\ps" "\\psi") ("\\PS" "\\Psi")
  ("\\om" "\\omega") ("\\OM" "Omega")
  ("\\vep" "\\varepsilon") ("\\vth" "\\vartheta")
  ("\\vpi" "\\varpi") ("\\vrh" "\\varrho")
  ("\\vsi" "\\varsigma")))
```

6. 演算子の表示について

+,-,* 等の二項演算子は挿入辞記法で書かれる。例えば, $a+b, a-c, a*b$ べき乗を表す二項演算子 $^$ は二番目の項の値によって清書の仕方が異なる。それを n とすると $n=0.5$ 又は $1/2$ の時, 平方根を表す根号を書く。 n が分数で正数の時, 相当する根号を書き, $n=-1$ のとき分子が 1 の分数を書き, その他は上付きの添字で冪乗を表す。/ は分数を書くとき使用される。括弧対 () は括弧時の中の式によって大きさを換える。今までの演算子は挿入辞記法で書かれる。しかしながら, $-$ の演算子は単項演算子の場合も存在する。例えば, $-a + b$ この場合の $-a$ の $-$ は接頭辞記法で書かれている。 $-a$ は $(- ("a"))$ のように変換されて処理される。 $\sin$ や $\cos$ のような単項演算子の関数は $\sin(x)$ のように項を括弧で括る。同じように扱われる他の関数は次の通りである。 $\tan, \sinh, \cosh, \tanh, \exp, \ln$ 数学で, よく使われる微分, 積分を表す演算子がある。これらを清書する必要がある。これらを $\sin$ や $\cos$ のように書くが項の数が必ずしも一個ではない。 x について, $f(x)$ の微分を表す清書に対して, $df(f(x), x)$ のように書く。 x について, $f(x)$ を n 回微分するとき, $df(f(x), x, n)$ のように書く。 x について, 偏微分するとき, 同様に, $dl(f(x), x), dl(f(x), x, n)$ と書く。不定積分を表す時, x について, $f(x)$ を積分する時, $int(f(x), x)$ のように書く。多重積分の場合には, $int(f(x, y), x, y)$ のように書く。定積分の場合, x を n_1 から n_2 まで $f(x)$ を積分する場合には, $dint(f(x), x, n_1, n_2)$ と書く。定積分の多重積分は, $dint(f(x, y), x, n_1, n_2, y, m_1, m_2)$ のように書く。 $n=1$ から m まで総和に対して, $sum(f(x), x, 1, m)$ のように書く。又, 不等号等の関係演算子 $<, >, \leq, =, \neq$ を導入して, 関係式の処理が可能になった。又, and, or, not 等が導入され, 論理式が処理可能になった。

7. 日本語の文と数式が混在するテキストの清書

日本語のテキストの一行中に数式を挿入する時, 数式を 一対の \$ で囲み, 行間に挿入する時は, 一対の \$\$ で囲む。テキストと変換されたテキストとそれを JTeX で処理された結果は最後に例として与えられている。

8. 結論

数理論理学で現われる数式を広範囲なフォントが存在する, JTeX によって自動的に清書する事が試みられた。このプログラムは Common Lisp で作成され, 簡単な数式のみ実行されたが, 複雑な数式も自動的に清書可能な事が示された。しかし, 一般に数理論理学に現われる全ての数式について, もっと研究が必要である。

参考文献

1. Donald E. Knuth, TeX and Metafont New Direction in Typesetting 1979, Digital Press
2. Guy L. Steele Jr / 後藤英一監訳, COMMON LISP 1984 共立
3. 益子純一, 増子進, 桂重俊, 英文および和文ワードプロセッサの開発, Vol.17, No.8 p1004, 1985
4. 大野義夫, TeX 入門, 1989, 共立
5. 里山元章, 文書の論理構造を備えた日本語清書システム「浄書」の設計と実現, Vol.30, No.9, 1989, 情報誌
6. 野寺隆, 数式処理システムの現状, Vol.21 No.12 p1605p 1989, Bit

元のテキスト (数式を含む日本文)

```
%centerline{JTeX によって清書されたテキスト}
%smallskip
私たちは,$\TeX$を使って数式の清書のシステムの研究をした。例えば,$z=x^2+y^2$,$\sin(x)+1$,$y=x+1$。
文中に数式をフォトランの文中の数式のように書く事が出来る。又, 行間に数式を書く事が出来る。
$$y=\sin(x)+1$$
$$-y$$
$$y=\alpha^{0.5}*x$$
私たちは, もっと複雑な数式を清書するシステムを開発しています。
```

TEXのコマンドで書き直されたテキスト

```
%loadsfont%bigj=dm10 scaled %magstep2
%loadjfont%tendm=dm10
%uses jroma of dm10 scaled %magstep2
%def%return{%par%noindent}
%def%nextpage{%fhill%ject}
%def%space{% }
%centerline{JTeX によって清書されたテキスト}
%smallskip
私たちは,$\TeX$を使って数式の清書のシステムの研究をした。例えば,$z = x^2+y^2$,$\sin x+1$,$y = x+1$
文中に数式をフォトランの文中の数式のように書く事が出来る。又, 行間に数式を書く事が出来る。
$$y = \sin x+1$$
$$-y$$
$$y = \sqrt{\alpha} x$$
私たちは, もっと複雑な数式を清書するシステムを開発しています。
%end
```

JTeX によって清書されたテキスト

私たちは, $\TeX$ を使って数式の清書のシステムの研究をした。例えば, $z = x^2 + y^2, \sin x + 1, y = x + 1$ 。文中に数式をフォトランの文中の数式のように書く事が出来る。又, 行間に数式を書く事が出来る。

$$y = \sin x + 1$$

$$-y$$

$$y = \sqrt{\alpha} x$$

私たちは, もっと複雑な数式を清書するシステムを開発しています。