文書処理とヒューマンインタフェース　14－1
（１９８７．９．７）

PostScriptにおける漢字出力の一方法

河村豊実、古川善吾、牛島和夫
九州大学工学部情報工学科

漢字フォントを持たないPostScriptプリンタで漢字を含む日本語文書を印刷する漢字出力プログラムを実現した。PostScriptは、印刷形式の文書を記述するためのインタプリタ型の言語である。PostScriptインタプリタで制御されるレーザプリンタが実用に供されており、これをPostScriptプリンタと呼ぶ。このプリンタで日本語文書を印刷するには、従来、漢字をビットマップデータに変換してプリンタに送信し、これをプリンタが出力している。この方法では、送信するPostScriptプログラムの量が多く、印刷に時間がかかる。

PostScriptは、利用者が文字フォントを定義する機能を持っているので、漢字のフォントを定義できる。しかし、現在のPostScriptプリンタは、全漢字のフォントを格納するだけの記憶領域を持っていない。我々が開発した漢字出力プログラムは、対象文書中に出現する文字のフォントのみを定義するPostScriptプログラムを作成する。漢字全体を1つのフォント辞書とするのではなく、複数のフォント辞書として登録し、印刷する時に、漢字コードに合わせてフォント辞書を切り替える。これによってテキストデータのみで文書の印刷形式を記述できるというPostScriptの利点を失うことなく印刷を高速化できる。この方法により、従来の方法に比較して約7倍高速に印刷できた。

A method for KANJI printing in PostScript

Toyomi KAWAMURA, Zengo FURUKAWA and Kazuo USHIJIMA
Department of Computer Science and Communication Engineering,
Kyushu University,
Hakozaki 6-10-1, Higashiku, Fukuoka 812, Japan

We developed a KANJI printing program which prints Japanese documents including KANJIs (Chinese characters, Japanese characters, etc.) on a PostScript printer which has no KANJI fonts. PostScript is an interpretive programming language for page printing description. The interpreter for the language is embedded in a controller for a raster printer. All KANJIs in a document to be printed are translated to image data and printed by the PostScript printer. Using this method, PostScript programs become so large that it takes a long time to print.

The PostScript language includes a feature for user-defined fonts. It is possible to define KANJI fonts on the PostScript printer. However, it does not have enough memory for all KANJI fonts definitions. The KANJI printing program generates a PostScript program which defines fonts for KANJIs occurring in a document only once and prints the document. This improves the printing speed of the KANJI printing program by about 7 times.

このページは、J Starで出力した。

1．はじめに

ワードプロッセッサの出現以来、文書の計算機処理の普及、およびレーザプリンタの印字品質の向上と、文書の標準的な記述方式の開発などによって、パソコンやワークステーションを利用した卓上出版が注目を集めている [OOY87]。そのなかで、英文の文書記述方式として有名なTeX[KNU84]やPostScript[ADO85a,ADO85B]の日本語化が検討されている [YAM87,OOY87]。記述方式の日本語化を実現するには、

- 漢字を表す２バイトコードの処理、
- 漢字フォントの作成、
- 禁則処理など日本語独自の組版処理

が必要である。我々は、ページ記述言語PostScriptを用いて、漢字フォントを持たないプリンタで日本語文書を印刷する方式を開発した。

ページ記述言語PostScriptは、インタプリタ型の言語である。この言語で記述されたプログラム（以下ＰＳプログラムと呼ぶ）を直接解釈・実行して文書を印刷するプリンタ（以下ＰＳプリンタと呼ぶ）が実用に供されている [APP86]。これまでのプリンタでは、印刷したい文字列（プリンタの持つ文字フォントに対応したコード列）とプリンタの制御コード列とを応用プログラムがプリンタに送信する。一方、ＰＳプリンタでは、文字列のみで記述されたＰＳプログラムを応用プログラムが送信する。

ＰＳプリンタの利用によって、

- 印刷形式の文書が文字コードのみから構成されているためにネットワークを利用した送信が容易である
- PostScriptの持つ高度なグラフィック機能を用いた文書が作成できる
- PostScriptのアウトラインフォントの使用によって美しい文書が得られる

などの効果がある。現在のＰＳプリンタは、英数字フォントのみを持っており、漢字フォントは持っていない。このＰＳプリンタの日本語化として、漢字フォントの開発やPostScriptインタプリタ（以下ＰＳインタプリタと呼ぶ）の拡張、などが検討されている。

Macintoshの出力装置の１つに、LaserWriter Plus（以下ＬＷＰと呼ぶ）を接続することができ、これで日本語文書を印刷できる。Macintosh で英数字をＬＷＰに出力するには、ＬＷＰがもつフォントを利用するための文字コードを送信する。一方、漢字の出力では、Macintosh 上で、漢字をイメージデータとしてPostScriptのビットマップデータに変換してから送信する。この方法では、文書のイメージデータが大きくなり印刷に時間がかかる。

また、漢字フォント全てをＬＷＰに送り、そのフォントに対するコード列として文書を送信する方法が考えられる。この方法は、ＬＷＰが受け取ったフォントを格納するのに十分なメモリを持たないために実現できない。

ＬＷＰでの漢字ファイルの印刷を高速化するために、TeX におけるＰＳプリンタへの出力方式を参考にして、ＬＷＰで日本語文書を印刷する漢字出力プログラムを作成した。漢字出力プログラムは、

(1) 日本データゼネラル社のECLIPSE MV/10000（以下ＭＶと呼ぶ）上で文書処理に使用されている文字フォントをPostScriptのビットマップデータに変換する、
(2) PostScriptの利用者フォント定義機能を利用して文字の種類毎にフォントを辞書に登録する、
(3) 登録されたフォント辞書を切り替えながら文字列を印刷する、

という手順で漢字ファイルの印刷を行なう。

本報告では、漢字出力プログラムの概要と、使用経験を述べる。以下、２章では、PostScriptの概要とＭＶについて説明し、３章では、PostScriptにおける漢字出力プログラムを説明し、４章で使用経験を述べる。

2．背　景

2．1　PostScriptの概要

PostScriptは、１章で述べたように、ページ記述用の言語である。PostScriptを用いた文書の印刷は、一般に、図2.1 に示す手順で行なわれる。応用プログラムを用いる利用者からは、図の右側の処理は見えないために、従来の印刷方法との違いはわからない。ところが、PostScriptでは、テキスト形式のプログラムに基づいているために、(a) ＰＳプログラムの編集やインタプリタとの会話によって文書印刷の微調整が可能になる、(b) 印刷形式の文書の保存や通信回線を用いた文書情報の交換が容易になる、などの効果がある。次にPostScript言語とＰＳプリンタについて説明する。

2．1．1　PostScript言語

PostScript言語は、インタプリタ方式のプログラミング言語である。文字列 'PostScript Example' のフォントを変えて、１５ポイント・サイズで印字するＰＳプログラム (a) とその実行結果 (b) を図 2

.2に示す。

一般にＰＳプログラムは、ページの印刷に必要な変数や手続きを定義する準備(prologue)部とページ毎に印刷する印刷(script)部から成っている。各々の命令は、オペランドとオペレータから成り、逆ポーランド記法で書く。図 2.2では、準備部で変数word、fntt、fnth、fntcと手続きnewfを定義している。

wordは

/word (PostScript Example) def

により文字列（'('と')'で囲まれた部分）を値としてもつ変数名として定義される。

newfは、

/newf {findfont 15 scalefont} def

により{..}部分を値とする手続きとして定義される。ＰＳプログラム中に名前newfが現れたら{..}の中が実行される。実際には、オペランドで指定された名前のフォントを取出し(findfont)、それを１５ポイント・サイズにする(scalefont)。

変数fntt、fnth、fntcには各々、newfで取り出したTimes-Roman、Helvetica、Courier のフォントが入る。

印刷部では、まず、movetoオペレータで文字列の印刷起点をページ上に設定する。setfont オペレータで使いたい文字フォントを現在のフォント辞書として設定する。showオペレータは、変数wordの値である文字列'PostScript Example'をそのフォントで印字する。即ち、文字列の各要素の文字コードをキーとして、現在のフォント辞書から文字記述を選択・実行し文字を書く。最後に、showpageオペレータによって、印字された内容を用紙に印刷する。

```
%---- prologue section
/word (PostScript Example) def

/newf {findfont 15 scalefont} def

/fntt /Times-Roman newf def
/fnth /Helvetica   newf def
/fntc /Courier     newf def

%---- script section
72 720 moveto
fntt setfont word show
72 705 moveto
fnth setfont word show
72 690 moveto
fntc setfont word show
showpage
```

(a)ＰＳプログラム

PostScript Example
PostScript Example
PostScript Example

(b)出力結果

図2.2　プログラムと出力例

オペレータには多様なグラフィック命令があり、図形や画像が組合わさった文書もプログラムとして記述することが出来る。また、文書内のテキスト／図形の配置、回転、拡大／縮小の記述も可能である。

２．１．２　ＰＳプリンタ

ＰＳプリンタは、ＰＳインタプリタとプリンタとを組合わせたものである。

ＰＳインタプリタの操作対象（オブジェクトと呼ぶ）には、数、配列、文字列、辞書等がある。辞書

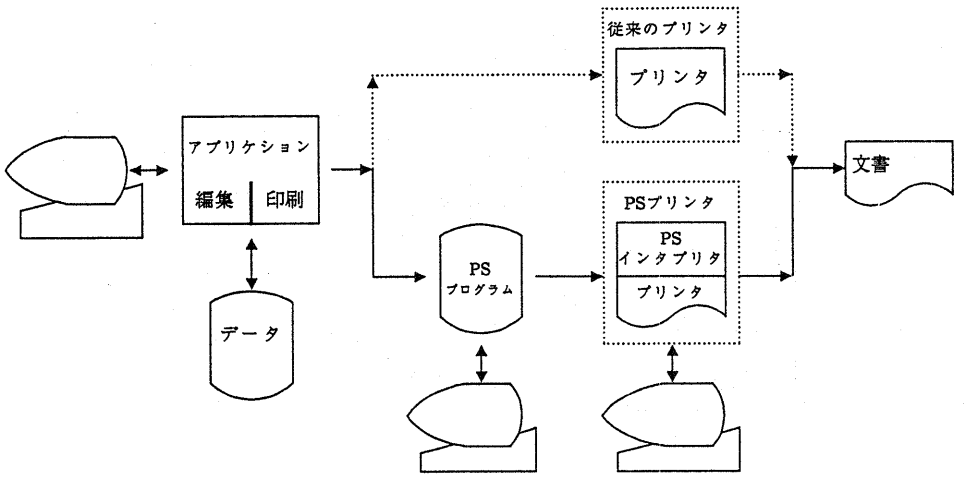

図2.1 PostScriptの位置づけ

とはキーと値の結合表であり、変数名と数値、名前と手続き等を結合している。フォントもキーとしてのフォント名と値としてのフォント辞書とによって定義される。図2.3 にフォント定義のデータ構造を示す。フォント辞書は、フォントの区毎に作成され、フォントディレクトリで辞書名と結び付けられる。実際のフォントに関すデータは、フォントデータに格納されている。フォントデータとフォント辞書は、エンコーディングベクトルを介して結合されている。ＰＳプログラム中に現れる文字コードは、エンコーディングベクトルのインデックスとして利用される。

ＰＳインタプリタは、オブジェクト操作のためにスタックを用いる。数、文字列、配列等を、オペランドスタックに積み、名前(オペレータ名等)を読み込むと、その名前をキーとして辞書を検索し、値として登録されている操作（組込み動作、手続き呼出し等）をオブジェクトに対して行なう。

例えば、図 2.2で、インタプリタは、最初に、word('/' は単にリテラルであることを示す）を、次に、文字列'PostScript Example'をオペランドスタックに積む。次のdef がオペランドスタックから２つのオブジェクトを取出し、それらを辞書に登録する。wordがキーであり、文字列'PostScript Example'が対応する値である。

PostScriptでは、文字フォントは一つの図形、即ち、線引きオペレータを使った文字のアウトライン（図形記述）フォントとして定義されている。このことが、文字を拡大縮小しても美しい印字のできる理由である。ＰＳプリンタにフォントを登録する時、文字フォントはアウトラインフォントだけでなく、画像オペレータを使ったビットマップフォントとしても定義できる。

ＰＳインタプリタは、スタックの他に、ＶＭ(Virtual Memory)と呼ばれる記憶領域を持っており、利用者が定義したビットマップフォント等の文字列データや配列、辞書等のオブジェクト本体を格納する。ＶＭの利用状況を調べたり、中間状態を保持・復元したりするオペレータが用意されている。

２．２　日本語情報処理システムＩＫＩＳ

漢字出力プログラムで出力の対象となる漢字ファイル、漢字コード系について概説する。IKISは、ＭＶのオペレーティングシステムAOS/VS下で動作する日本語情報処理システムである。IKISが使用するIKIS漢字コード系はJIS C 6226に準拠し、漢字を次の２バイトで表現する。

第１バイト＝（区番号＋２０Ｈ）＋８０Ｈ
第２バイト＝（点番号＋２０Ｈ）＋８０Ｈ
（ｎｎＨは１６進数を表す）

例えば、'漢字'という文字列をIKIS漢字コード系で表したものを図2.4 に示す。このコード系では、エスケープ・シーケンスなしで１バイトの英数データと２バイトの漢字データとの混在が可能である。

漢字ファイルは、一般に、漢字コード・英数コードが混在し、さらに、書式情報としてセンタリングや文字の大きさ選択などの制御コード等を含むことがある。ここでは、書式情報を取り除き、文書情報

```
 漢　字
B4C1 BBFA
 | |
 +-+- 第１バイト＝区番号＋Ａ０Ｈ＝２０＋Ａ０Ｈ
   +- 第２バイト＝点番号＋Ａ０Ｈ＝３３＋Ａ０Ｈ
```

図2.4　IKIS漢字コードの例

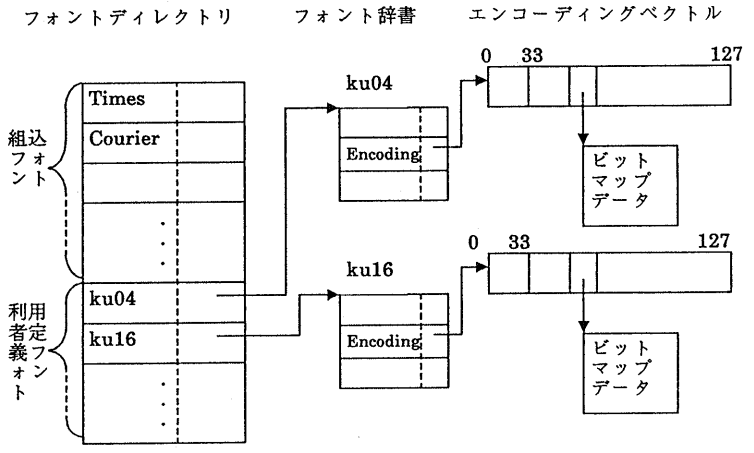

図2.3 フォント定義のデータ構造

のみを抜き出しIKIS漢字コード・英数コードからなるIKISファイルを漢字出力プログラムで使用する。

3．漢字出力プログラム

3．1　方針

ＰＳプリンタに漢字を出力するプログラム(iks2ps)を作成するに当たって、次のような方針を立てた。

(1) PostScriptの機能の活用

ＰＳプリンタには、漢字フォントがない。漢字出力のために、漢字コード、漢字フォントが扱えるように、インタプリタを拡張することが考えられる。しかし、ＬＷＰのＰＳインタプリタはＲＯＭ化されており、インタプリタ本体の機能拡張は出来ない。そこで、PostScriptの利用者フォント定義機能を使い、漢字（２バイトコード文字）の、フォント登録・エンコーディングの方法を考える。

(2) 実用的な処理スピード

ＭＶの利用状況からみて漢字出力プログラムの入力となるのは、ワープロ作成の文書、パソコン等からアップロードした文書などである。その出力に際して長時間の待ち時間には耐えられないので、数枚／分程度の実用的な処理速度が必要である。

3．2　構成

ＰＳプリンタへの漢字出力プログラムの構成を図3.1に示す。ＭＶ上で行なう仕事を点線で囲んだ。図 3.2は、文字列 "ＰｏｓｔＳｃｒｉｐｔにおける

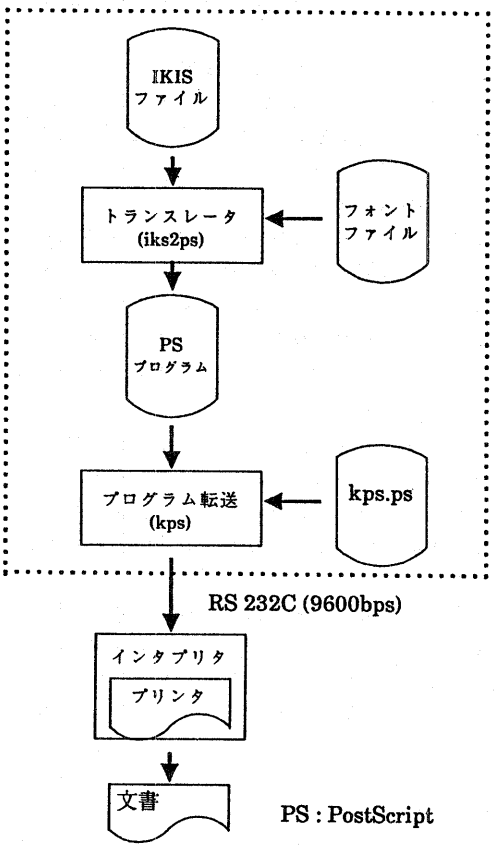

図3.1 漢字出力プログラムの構成

```
  0   A3D0   A3EF   A3F3   A3F4   A3D3   A3E3   A3F2   A3E9
  8   A3F0   A3F4   A4CB   A4AA   A4B1   A4EB   B4C1   BBFA
 10   BDD0   CECF   A4CE   B0EC   CAFD   CBA1   0A00
```

(a) IKISファイル

```
  1: @bop0
  2: /ku03 @newfont
  3: ku03 @sf
  4: <0000000000000003FFF80003FFFE0000E0070000E0038000E0038000E001C0
  5:  00E001C000E001C000E001C000E001C000E0038000E0038000E0070000FFFE00

121:  7FFFFFFC3FFFFFFC0007003800070038000700380007003800070038000700038
122:  0007003800070038000E0038000E0038000E0038001C0038001C003800380038
123:  0038003800700070000E0007001C000F003800FE0070003E00C0001C000000000>
124:  79 @dc
125: @bop1
126: ku03 @sf (PostScript) s ku04 @sf (K*1k) s ku20 @sf (A) s ku27 @sf
127:  (z) s ku29 @sf (P) s ku46 @sf (O) s ku04 @sf (N) s ku16 @sf (l) s
128: ku42 @sf (}) s ku43 @sf (!) s @nl
129: @eop
```

(b) ＰＳプログラム

ＰｏｓｔＳｃｒｉｐｔにおける漢字出力の一方法

(c) 出力結果

図3.2 漢字出力プログラムのファイル例

漢字出力の一方法" を処理した時の各ファイルである。

(1) トランスレータ（C 言語で実現）

トランスレータは、IKISファイルを印刷用のＰＳプログラムに変換する。IKISファイルは、ワープロ等の応用プログラムが作り出したファイルである。（図3.2(a)参照。）

ＰＳプリンタで漢字を印刷するためにIKISシステムが持っている漢字フォントを利用する。３２×３２ドットのビットマップフォントを、PostScriptのビットマップデータに変換し、ＰＳプログラムに利用者定義フォント用のデータとして設定する。（図3.2(b) 参照。）

(2) プログラム転送（コマンド・マクロで実現）

トランスレータで作成したＰＳプログラムと、これが参照する手続き定義(kps.ps)とを、ＰＳプリンタへ転送する。転送に使うコマンド・マクロを図3.3 に示す。

１、２、３行目で、転送に用いるファイルの準備を行なう。４行目のcopyコマンドで手続き定義(kps.ps)を、５行目のcopyコマンドでＰＳプログラム(%1%.ps:%1% は仮引数を示す)を一つのファイルにする。６行目のqprintコマンドでスプールキュー(lwq) に作成したファイルを送る。

(3)ＰＳプリンタによる印刷

ＰＳプログラムを解釈・実行し、プリンタに出力する。（図3.2(c)参照。）

３．３　処理方式

本節では、トランスレータ及び手続き定義(kps.ps)について述べる。

３．３．１　トランスレータ

トランスレータはテキスト入力部とフォント取出し部、文字列変換部から成っている。

テキスト入力部では、以下のことを行なう。

(1) IKISファイルを読み込む。ここでは漢字コード・英数コードからなるIKISファイルを入力対象とする。

(2) 制御コードの中で改ページ、改行、タブのみを処理する。タブは、英数８字単位で設定されているものとした。

(3) 読み込んだテキストをページの書式（４０行／ページ、３８文字／行）に合わせてテキスト領域に格納する。

(4) 英数コードは、漢字コードの０区に割り当て、英数コードと漢字コードの全てを２バイトコードに変換する。これによって、以下での文字列作成が簡単になり、ＰＳプログラムの記述が短くなる。

(5) 初めて出現した漢字コードに対する利用状況表を作る。

```
1: string  ?[!pid].kps
2: del/2=ig [!str]
3: create   [!str]
4: copy/a/b [!str] ps:kps.ps
5: copy/a/b [!str] %1%.ps eof
6: qprint/del/bin/que=lwq [!str]
```

図3.3 転送マクロ

フォント定義部では、上記の利用状況表を参照しながら、各ページで初めて使用される文字のフォントを読み込み、１６進文字列に変換し、フォント定義のＰＳプログラムを作成する（図 3.2(b) の２行目から１２４行目に対応）。ＰＳインタプリタのフォント定義は、図2.3 のデータ構造を持っている。そのために、フォント定義部で作成されるＰＳプログラムは、図 3.2(b) ２行目に示すように、区の中の文字が初めて使用されるページで、対応区のフォント辞書を定義（@newfont）する。フォントを定義する文字の対応区の辞書を操作対象として宣言（３行目の @sf）する。その後で、文字のフォントデータ（'<','>' で囲まれた部分はビットマップデータである）と文字コード（１２４行目の７９）とを定義する（ @dc）。３２×３２ドットフォントを用いるので、漢字１文字を表すのに１０２４ビット必要である。これを、４ビット毎に１６進数文字列に変換する。そのため、ＰＳプログラムで１つの漢字フォントを表すのに、２５６バイトを必要とする。

文字列変換部では、同じ区で連続した文字（２バイトコード）の２バイト目を用いて文字列印字のＰＳプログラムを作成する（図3.2(b)の１２６行目から１２８行目に対応）。このＰＳプログラムの部分では、文字列が使用するフォント辞書（kuxx：xxが区番号を示す）に随時切り換え（@sf） 、文字列をエンコーディングベクトルのインデックスとして印字（s）する。

３．３．２　手続き定義

手続き定義(kps.ps)は、ＰＳプログラムが参照する手続きの定義で、フォント辞書の定義・参照のための手続き等からなる。使用している手続き定義は、TeX のＰＳプリンタ出力で使用されている手続きを参考にした。TeX 手続きと比べてkps.psは、以下の特徴を持つ。

(1) 英数字を漢字の０区のフォントとして扱うためのフォント辞書名(ku00)を定義した。
(2) 漢字の文字幅を一定としたのでフォントデータの定義が簡単である。

4. 漢字出力プログラムの評価

漢字出力プログラムの性能について次に述べる。性能としては、出力量と出力時間について調べた。比較対象としてMacintosh 本来の日本語の印刷処理を用いた。

4．1 出力量

ＬＷＰは、ＲＡＭを持っており、インタプリタの作業用領域（スタック領域等）とＶＭ領域として利用している。この中で、ＶＭ領域に利用できるのは、約２４０ｋバイトである。計測に用いた文書は以下の通りである。

(A) 大部分が漢字の文書（Ａ４サイズでの本報告書）
(B) 漢字と英文が混じった文書（JUNET の手紙のバックログであり、宛先や差出人、経路など半分以上が英文である）
(C) 英文のみの文書（文書Ｃ）

図4.1 は、登録されたフォントのページ毎の累積値である。図4.2 は、漢字出力プログラムで文書を印刷したときのＶＭの使用状況である。このデータは、各ページを印刷（showpage）する前にPostScriptのvmstatusオペレータでＶＭの使用量を調べることによって得た。

前章で述べたように、１文字の漢字フォントを表すのに２５６バイトを必要とし、フォントデータは累積するので、図4.1 と図4.2 から分かるように、ＶＭの使用量は、漢字の累積フォント数にほぼ比例する。但し、文書の文字列がＶＭ上に作成されるために、字数の少ないページではＶＭの使用量が減少することもある。ＬＷＰのＲＡＭは有限であるために、フォントの登録が可能な文字数に限界がある。現在のＬＷＰでは３２ドットのフォントを利用した時には、約７５０文字のフォントを登録できる。そのため、文書の出力可能枚数は使用する文字の種類数によって決まる。

Macintosh では、漢字データはすべてビットマップデータに変換してからＬＷＰに送信するために、ページ毎にＶＭに登録した文字列がクリアされるので出力枚数に制限はない。

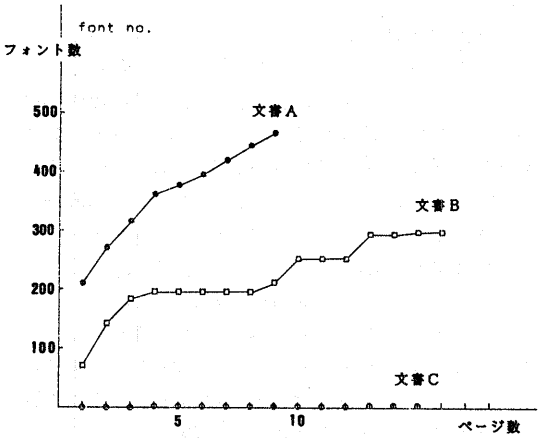

図4.1 フォント数の増加

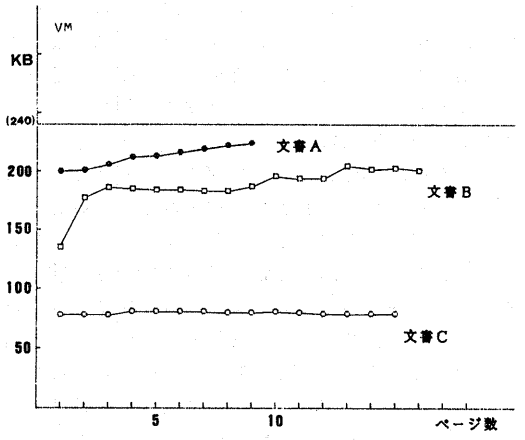

図4.2 ＶＭの使用状況

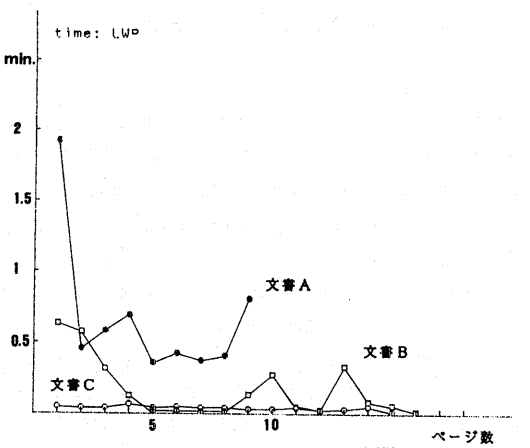

図4.3 各ページの印刷時間

４．２　出力時間

漢字出力プログラムは、図3.1 に示すように、ＭＶ上で漢字ファイルをＰＳプログラムに変換してから、ＬＷＰに送信する。現在ＭＶとＬＷＰとは9600 bpsのRS232C で接続している。上記の３つの文書を漢字出力プログラムで各ページを印刷するのに要した時間を図4.3 に示す。ＰＳプログラムでPostScriptの時間読み出しオペレータ(usertime)でページ毎の開始時刻を読み出し、ページ印刷の直前の時刻から差し引くことで、所要時間を測定した。

漢字出力プログラムで日本語文書を印刷するためには、漢字ファイルからＰＳプログラムへの変換が必要であり、その時間をＭＶ上のＣＰＵタイマで測定したところ、最も長くかかった場合でも約３０秒であり、ＰＳプログラムの転送と実行に要した時間の１／１０程度である。しかし、文書全体の変換が終了した後にＰＳプログラムを転送・実行しているために、最初の１枚の出力開始までの時間に変換時間が加わる。そのために、できるだけ短くすることが必要である。本報告の本文全体（最初のページは除く）の印刷時間は、約６．５分である。

Macintosh では、ＬＷＰで印刷するために、文書をＰＳプログラムに変換しながら送信している。文書Ａの各ページの印刷に３－９分要しており、全体で約４６分かかった。印刷に要した時間は、漢字出力プログラムの約７倍である。一方、文書Ａについて Macintosh 上で作成されたＰＳプログラムは、約２．２ＭＢあり、元々の文書（約１７ＫＢ）の約１３０倍、漢字出力プログラムで作成したＰＳプログラム（約２１５ＫＢ）の約１０倍である。このことから、ＰＳプログラムの実行では、フォントの登録や、辞書の切り替えが必要なために、漢字出力プログラムの方が長くかかっていることが分かる。

５．おわりに

漢字ファイルをページ記述言語PostScriptの機能を利用して、漢字フォントを持たないＰＳプリンタで印刷するための漢字出力プログラムを作成した。このプログラムは、ファイル内で使用されている漢字のフォントを１回だけＰＳプリンタに送信し、PostScriptの辞書切り替え機能を用いて漢字を印刷する。この方式は、日本語の文字列をすべてビットパターンに変換して送信する Macintoshの日本語出力プログラムの方法に比較して約７倍高速に印刷できる。しかし、漢字フォントのビットマップデータをＬＷＰの利用者メモリに登録して利用するので一度に出力できる文書量に限界がある。

現在、漢字出力プログラムは、主に、コンピュータネットワークJUNET 上のニュース記事や手紙の印刷に利用されている。JUNET 上では、通信に用いる文書の漢字のエスケープシーケンスが何度か議論になっている。また、図形情報の交換についても話題となっている。漢字出力プログラムが作り出すＰＳプログラムは、漢字コードがフォント辞書名と辞書内での対応番号を表す文字コードに展開されるために、漢字コード変換を考慮する必要がなく、漢字フォントを意識する必要もない。但し、ＰＳプログラムがフォント情報も含むために大きくなってしまう欠点がある。一方、漢字出力プログラムは、ＰＳプリンタの存在を前提としている。しかし、ＰＳインタプリタとデバイスドライバがあれば、漢字フォントを持たないプリンタやディスプレイでも日本語文書を表示できる。すでに石田等がＰＳインタプリタをプログラム化している[ISH87]。

（本報告の本文は、ここで述べた漢字出力プログラムで出力した。）

[参考文献]

[ADO85a] Adobe Systems Inc. : PostScript Language Reference Manual, Addison-Wesley Pub., Co., 1985.

[ADO85b] Adobe Systems Inc. : PostScript Language Tutorial and Cookbook, Addison-Wesley Pub., Co., 1985.

[APP86] Apple Computer, Inc. : LaserWriter and LaserWriter plus, 1986.

[ISH87] 石田晴久、本岡茂哲、杉山武信：PostScript事始め　(1)-(3)、日経バイト、No.31-33,4-6月 1987.

[KNU84] Knuth,Donald E.; The TeX book, Addison Wesley Pub., Co., 1984

[NDG78] 日本データゼネラル株式会社：AOS,AOS/VS 解説書　CLI編、1978.

[OGA86] 小方一郎：レイアウト言語PostScript、bit, Vol.18, No.12, pp.4-11, 11月 1986.

[OOY87] 大用昌之、手島透：日本語デスクトップ・パブリシング実現の可能性を探る、日経バイト、No.35, pp.90-100, 7月 1987.

[YAM87] 山内長承、来住伸子：TeX 文書清書システム、コンピュータソフトウェア、Vol.4, No.1, pp.44-52, 1月 1987.