UNIX上でのBASE形式の
トランスレータの作成

平成 18 年 1 月 31 日

情報電子工学科 竹野研究室
西片 雅和

目 次

1 はじめに **1**

2 点字エディターについて **1**

2.1 点字エディタの種類 1

2.2 自動点訳ソフトについて 2

2.3 点訳ソフトの考察について 3

3 BASE について **3**

3.1 BASE の使用環境 3

3.2 BASE の入力方法 4

3.3 BASE の 6 点入力について 4

4 点字とは **4**

4.1 点字使用の意義 4

4.2 点字の特徴 5

4.2.1 濁音・半濁音 5

4.2.2 拗音 6

4.2.3 数字・アルファベット 7

4.2.4 わかち書き 7

5 トランスレータの考察 **8**

5.1 BASE 形式の文字の解析について 8

5.1.1 解析方法 8

5.2 解析結果 8

5.3 BASE のヘッダ部分の解析 10

5.3.1 ヘッダ部分の解析方法 10
5.3.2 ヘッダ部分の解析結果 11
5.4 BASE の形式の確認について 12
5.4.1 確認手順 13
5.4.2 NABCC(北米点字) コード 13
5.5 解析したヘッダの確認について 15
5.5.1 確認手順 15
5.6 確認した結果 16
5.7 トランスレータのアルゴリズムについて 16
5.8 トランスレータのプログラムについて 17

6 まとめ **19**

参考文献 **20**

概要

2003 年の松本 賢一氏の卒論で課題となっていた漢字かな混じり文の自動点訳を基に研究を進める。点字エディタ、自動点訳ソフトについて、BASE について、点字の構造について調べた。BASE とはキーボードから文字を打ち込めば点字データにしてくれる点字エディタの一つである。BASE の形式は MS–DOS、MS–Windows で広く使われていて、その形式は点字データのファイル形式としては最も標準的である。そこで UNIX 上で BASE の形式に変換できないか考えた。BASE での詳しい形式の解析を行った結果 BASE は NABCC という形式が使われていた。これらの解析したデータを基に UNIX 上で日本語 (ひらがな) 入力されたテキスト形式のファイルを NABCC 形式のファイルに変換するトランスレータを作成した。

1　はじめに

点字で文字を書くには点筆と点字板というものを使って一点一点手作業で打っていかなければいけない。点字の熟練者は、点字タイプライターや点字板を用いて、晴眼者がボールペンや鉛筆で普通の文章を書くのよりも早く、点字の文章を書くことができる。しかし、点字の構成を知らない人やまだ点字に慣れていない人、さらに中途失明者には点字の文章を書くのに非常に時間がかかってしまう。また、物理的に突起をつくるわけなので間違ってうってしまった場合は修正するのが大変困難である。しかし、BASE という点字エディタを使えば点字を間違って打った場合や改行するといったことが容易にできる。また、入力方式としてローマ字入力やかな入力に対応しているので一点一点の点字の構成を覚えていない人でも容易に点字文をつくることができる。

BASE という点字エディタは MS–DOS や MS–Windows で広く普及しているエディタである。しかし、UNIX 上ではこのような点字エディタはほとんど見当たらない。そこで本研究では UNIX 上でも容易に点訳ができる点字エディタの作成を目標とする。まず点字の特徴について調べ、BASE のデータ形式がどのような形式であるか解析する。解析したデータを基に最終的には UNIX 上でテキスト形式のファイルを BASE で読み込み可能な形式に変換するトランスレータを作成する。テキスト形式のファイルをその形式に変換できれば UNIX 上で点訳ができるといえる。

2　点字エディターについて

点字エディタとは、コンピュータに点字を入力したり、編集したりするためのソフトのことである。点字本を作るには、印刷物とは違って点筆と呼ばれる専用の針や点字タイプライターで紙を突いて作る。物理的に突起を作る作業であるので、やり直しも困難であるし、もしも原文の 1 行を飛ばしてしまったような場合、最初からやり直さなければいけない。そこで考えられるのは、キーボードから文章を打ち込んでデータを作り、そのデータを専用プリンタから点字にして打ち出せば、飛躍的に効率が上がるのではないかということである。そのような考えで様々な点字エディタが作られた。

2.1　点字エディタの種類

点字エディターの種類としては、「Win-BES99」、「ういんびー」、「T・エディタ」などがある。これらの点字エディタについてまとめると以下のようになる。

	対応 OS	入力方式
Win-BES99	MS-Windows	6 点入力、カナ入力、英字入力
ういんびー	MS-Windows	6 点入力、カナ入力、英字入力
T・エディタ	MS-Windows	6 点入力、カナ入力、英字入力

- 主な特徴

Win-BES99

IBM 社が開発したソフトで、同時に複数のファイルを開きます。マルチタスクで点訳作業ができる。点訳者の作業を効率化するために音声出力機能を追加することができる。ただし、音声読み上げは IBM 社の ProTALKER97 が導入されることが必要である。

ういんびー

@nifty の点訳グループのゴン氏が開発した点訳ソフトで、BASE の操作性をそのまま盛り込んでいる。点字表示のために「ういんびー点字フォント」が必要である。

T・エディタ

BASE 形式ファイル (.BSE) の編集を行う。BASE の使用者にも違和感なく使えるよう、できるだけ BASE の機能・操作方法を継承するよう作成されている。ただし、キー操作についてはむしろ一般的なエディタやワープロのほうに近いといえる。

2.2 自動点訳ソフトについて

自動点訳ソフトとは日本語の文章を自動的に点字の文章に変換してくれるものである。自動点訳ソフトの種類としては、「お点ちゃん」、「IBUKI-TEN」、「EXTRA」などがある。これらの自動点訳ソフトについてまとめると以下のようになる。

	対応 OS	入力データ形式	出力データ形式
お点ちゃん	MS-Windows	テキスト	BSE 形式
IBUKI-TEN	MS-Windows	テキスト	BSE 形式,NABCC 形式
EXTRA	MS-Windows,UNIX	テキスト, ワード, エクセル, 一太郎、PDF, マイワード,HTML,XML	EXF 形式

- 主な特徴

お点ちゃん

お点ちゃんは、漢字仮名交じり文の入ったテキストファイルを入力し、これを変換して分かち書きの仮名テキストファイルと、点字プリンタ用の BASE 形式ファイルを出力するソフトウェアである。

IBUKI-TEN

岐阜大学工学部応用情報学科池田研究室で開発された自動点字翻訳システムで、漢字かな混じりで書かれたテキストファイルを対象として自動的に点字変換する。日本語解析システム IBUKI をベースにしていて、高精度の分かち書きを実現している。IBUKI-TEN には校正・編集機能を含まない IBUKI-TEN というタイプと校正・編集機能を含む IBUKI-TEN Edit タイプの二つのタイプがある。

EXTRA

XTRA は、テキストファイル、ワードや一太郎などで書かれた文章のほか、HTML ファイル、PDF ファイルのテキスト部分などを自動的に点字のデータに変換し、即座に点字としての編集作業を行うことのできる自動点訳ソフトである。

2.3 点訳ソフトの考察について

テキスト形式で入力された文章を BASE 形式に変換し、それを BASE という点字エディタに読み込ませることにより、点字文に変換するというものを考察する。そのためには BASE がどのような点字エディタであるか、BASE の文字の形式がどのようなものなのかを調べなければいけない。また、点字の特徴についても調べる。

3 BASE について

BASE とは無量寺点字出版所が開発した MS-DOS のソフトで点字エディタのひとつである。点字の形を知らなくても、キーボードから文字を打ち込めば点字データに変換される。画面にはかな文字及び点字が表示される。

3.1 BASE の使用環境

- 動作可能なパソコンについて

BASE は NEC の PC-9801/9821 および EPSON 製 NEC PC-98 互換機上の MS-DOS Ver 5.0 以降のもの、富士通の FM シリーズで動作する。

- キーボードについて

  BASE では 6 点入力という入力方式があり、点字一マスを入力するのに決められたキーを同時押す場合がある。キーボードを 6 点同時に押したときにそのキーイベントを認識するキーボードとそうでないキーボードがある NEC 製 PC-9801/9821 シリーズに付属のキーボードは 6 点同時押しを認識する。

3.2 BASE の入力方法

入力方法はローマ字仮名入力で文字 (墨字) を入力することができ、ファンクションキーの F10 で入力方式を変えることができる。 F10 を押すごとにローマ字仮名入力、半角英数字入力、 6 点入力に切替えることができる。文字 (墨字) を入力して、ファンクションキーの F2 を押すと入力された文字 (墨字) が瞬時に点字表示に切り替わる。

3.3 BASE の 6 点入力について

6 点入力とは点字の 1,2,3 の点をキーボードの F,D,S に 4,5,6 の点をキーボードの J,K,L に置き換えた入力方式でこの方式をパーキンス方式という。他の入力方式としては 1,2,3 の点を O,K,M に 4,5,6 の点を E,F,V に置き換えたカニタイプ方式や 1,2,3 の点をテンキーの 1,2,3 に 4,5,6 の点をテンキーの 4,5,6 に置き換えたテンキー 1 方式などがある。

4 点字とは

点字とは一マス 6 点 (縦 3 点× 横 2 点) の突起で文字を表し、視覚障害者がその突起を指でなぞることで読みとる事が出来るものである。これを 6 点点字と呼び、1825 年フランスの Louis Braille(1809 ~ 1852) によって考案され、1854 年にフランスで採用されたのを皮切りに、次第に各国で採用された。日本では 1890 年に石川倉次 (1859 ~ 1944) によって翻案された。また、紙に書かれた普通の文字のことを点字に対して墨字と呼んでいる。

4.1 点字使用の意義

点字を書くのにコンピュータは非常に便利である。いろいろな音声化ソフトが開発されたり、視覚障害者用のパソコンも開発されて、視覚障害者にとって今やコンピュータはペ

ンであり、目でもある。

4.2 点字の特徴

点字は横書きで左から右に読む。6 点点字は 6 つの点で構成されていて 1 の点、2 の点、4 の点が母音を表していて 3 の点、5 の点、6 の点が子音を表している。点字はローマ字と似ていて母音の「アイウエオ」に子音をたして他の清音を構成していて、それぞれカ行は 6 の点、サ行は 5、6 の点、タ行は 3、5 の点、ナ行は 3 の点、ハ行は 3、6 の点、マ行は 3、5、6 の点、ラ行は 5 の点を母音にたしたものになっている。しかし、ヤ行とワ行は不規則である。ヤ行は「ヤ、ユ、ヨ」の 3 つであるが、それぞれの母音「ア、ウ、オ」の点字を下にずらして、それに 4 の点を加えた形で構成している。ワ行は「ワ、ヲ」の 2 つであるが、それぞれの母音「ア、オ」を一番下に下げた形で構成している。「ん」は子音を表す 3 の点 5 の点 6 の点で構成されている。また、「ッ」のような促音は 2 の点だけで表す。

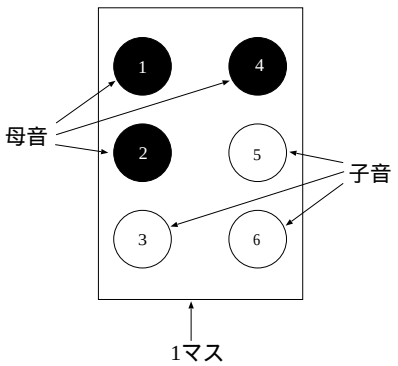

Fig. 1 点字の構成図

4.2.1 濁音・半濁音

墨字の場合は清音の右上に濁点・半濁点をつければいいが、点字の場合は指で触ったときに濁音・半濁音の点字であるとわかるように清音の前に濁音符・半濁音符をつける。

- 濁音
  濁音は清音の点字の一マス前に 5 の点だけで構成された濁音符をつける。
- 半濁音
  半濁音は清音の点字の一マス前に 6 の点だけで構成された半濁音符をつける。

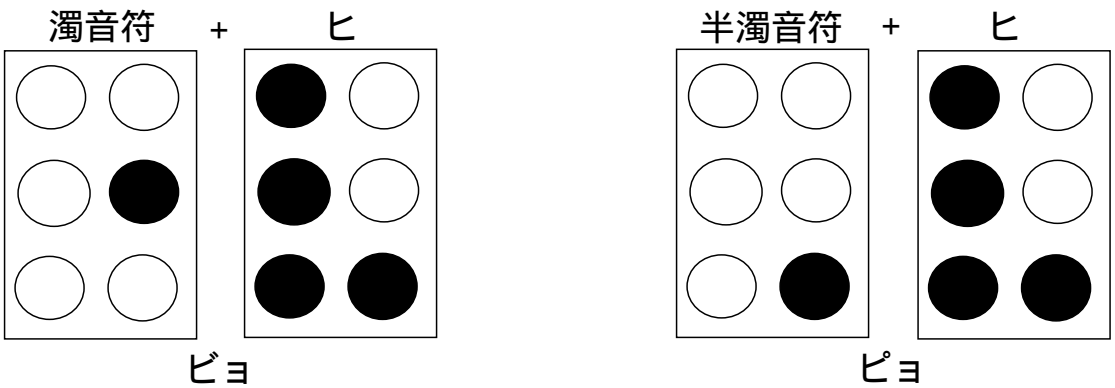

Fig. 2 濁音・半濁音の構成

4.2.2 拗音

拗音を点字で表す場合拗音符をつける。拗音符とは 6 点点字の 4 の点のことである。また、拗音の前の文字が濁音、半濁音の場合は濁音符、半濁音符をそれぞれ拗音符に加える。濁音符は 5 の点、半濁音符は 6 の点である。また、墨字で「ビ」から「ビャ」に変えるときは「ビ」に拗音の「ャ」をつければいいが、点字の場合は「ビ」の前の濁音符に拗音符を追加し、「ヒ」の点字を「ハ」に変えなければいけない。「ビュ」の場合は「ヒ」の点字を「フ」に「ビョ」の場合は「ヒ」の点字を「ホ」に同じように変えなければいけない。

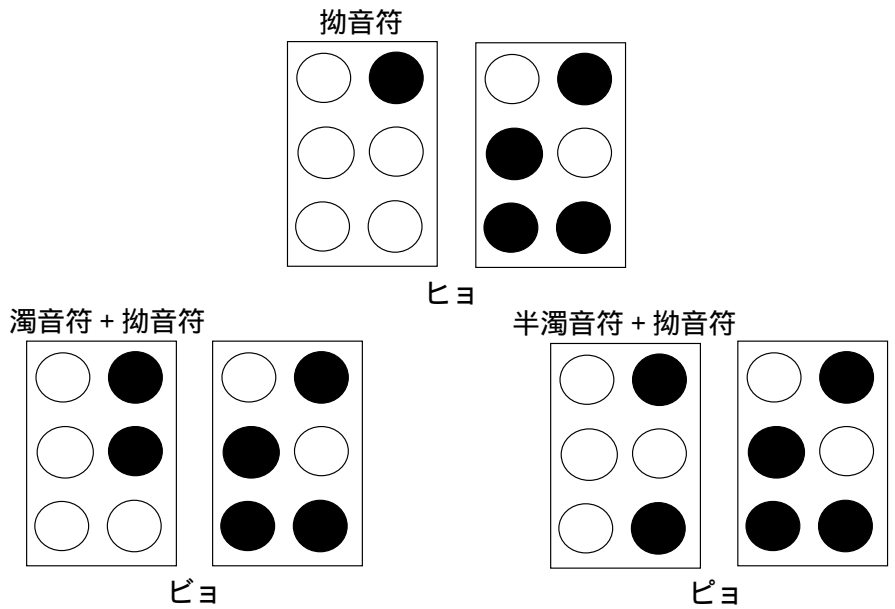

Fig. 3 拗音符に濁音符、半濁音符を加えた時の構成図

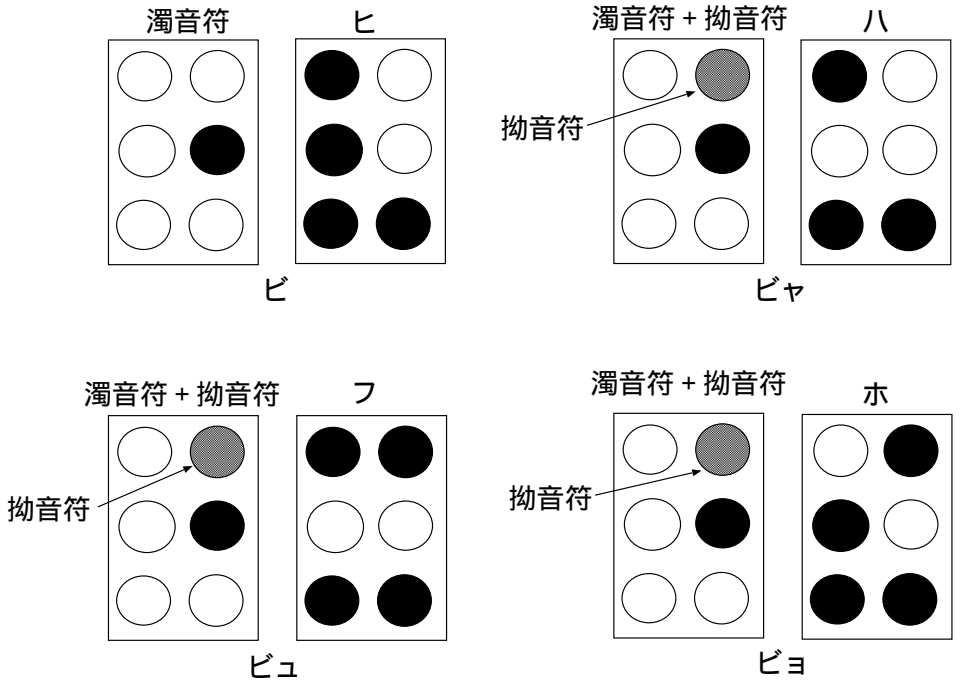

Fig. 4 拗音の母音の変化の構造図

4.2.3 数字・アルファベット

- 数字
  数字記号の前に 3 の点、4 の点、5 の点、6 の点で構成された数符をつける。
- アルファベット
  それぞれのアルファベットの前に 5 の点、6 の点で構成された外字符をつける。大文字が使われている場合は外字符のあとに 6 の点で構成された大文字符をつける。2 文字以上大文字が続いている場合は外字符のあとに大文字符を 2 つつける。

4.2.4 わかち書き

点字をなにも区切らずにそのまま書くと、墨字の文章を全部ひらがなで書いたものと同じでとても読みにくい。点字では文節ごとに 1 マスあけるのが原則である。このことをわかち書きという。

5　トランスレータの考察

わかち書きされたテキスト形式の文章を BASE 形式に変換し、それを BASE で読み込ませることにより点字文に変換するトランスレータを考察する。

5.1　BASE 形式の文字の解析について

トランスレータを考える上で BASE 形式の文字がテキスト形式ではそれぞれどの文字に対応しているかを解析する必要がある。

5.1.1　解析方法

1. BASE でひらがなの清音、濁音、半濁音、数字、アルファベットの大文字、アルファベットの小文字を入力し、保管する。
2. MS-DOS のプロンプトの後に「fd」と入力し、BASE の中の 1 で作ったファイルを選択する。
3. 1 で入力した内容を確認する。

5.2　解析結果

5.1 の手順で解析すると以下の表にまとめた結果のようになった。

	清音					濁音					半濁音				
あ行	あ	い	う	え	お										
	A	B	C	F	I										
か行	か	き	く	け	こ	が	ぎ	ぐ	げ	ご					
	*	<	%	$	[	”*	”<	”%	”$	”[					
さ行	さ	し	す	せ	そ	ざ	じ	ず	ぜ	ぞ					
	:	¥	?	]	W	”:	”¥	”?	”]	”W					
た行	た	ち	つ	て	と	だ	ぢ	づ	で	ど					
	O	R	N	Q	T	”O	”R	”N	”Q	”T					
な行	な	に	ぬ	ね	の										
	K	L	M	P	S										
は行	は	ひ	ふ	へ	ほ	ば	び	ぶ	べ	ぼ	ぱ	ぴ	ぷ	ぺ	ぽ
	U	V	X	&	!	”U	”V	”X	”&	”!	,U	,V	,X	,&	,!
ま行	ま	み	む	め	も										
	Z	(	Y	=	)										
や行	や		ゆ		よ										
	/		+		>										
ら行	ら	り	る	れ	ろ										
	E	H	D	G	J										
わ行	わ		を		ん										
	’		9		0										

拗音			拗濁音			拗半濁音		
きゃ	きゅ	きょ	ぎゃ	ぎゅ	ぎょ			
@*	@%	@[	^*	^%	^[			
しゃ	しゅ	しょ	じゃ	じゅ	じょ			
@:	@?	@W	^:	^?	^W			
ちゃ	ちゅ	ちょ	ぢゃ	ぢゅ	ぢょ			
@O	@N	@T	^O	^N	^T			
にゃ	にゅ	にょ						
@K	@M	@S						
ひゃ	ひゅ	ひょ	びゃ	びゅ	びょ	ぴゃ	ぴゅ	ぴょ
@U	@X	@!	^U	^X	^!	.U	.X	.!
みゃ	みゅ	みょ						
@Z	@Y	@)						
りゃ	りゅ	りょ						
@E	@D	@J						

数字									
1	2	3	4	5	6	7	8	9	0
#A	#B	#C	#D	#E	#F	#G	#H	#I	#J

アルファベット大文字												
A	B	C	D	E	F	G	H	I	J	K	L	M
;,,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
;,,N	O	P	Q	R	S	T	U	V	W	X	Y	Z

アルファベット小文字												
a	b	c	d	e	f	g	h	i	j	k	l	m
;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
;N	O	P	Q	R	S	T	U	V	W	X	Y	Z

解析した結果このようになった。それぞれの気がついた点をまとめると以下のようになった。

- 濁音は清音に「”」をつけた形だった。
- 半濁音は清音に「,」をつけた形だった。
- 拗濁音は「@」を拗半濁音は「.」をつけた形だった。
- 数字はアルファベットに「#」をつけた形だった。

5.3 BASE のヘッダ部分の解析

BASE 形式のファイルをテキスト形式で表示させると本文の上に 6 行のなにも入力されていない行が表示され、その下の行に 8 桁の数字が表示される。このなにも入力されていないヘッダのようなものの解析する。またこのヘッダの部分には BASE で目録というものを入力することができる。この目録についても解析する。

5.3.1 ヘッダ部分の解析方法

1. BASE を開いて新しくファイルを作る。

2. 「桁行数をいれて下さい [3222]」と表示されるので、桁行数を [1020] に変更して RETURN キーを押す。
3. 横が 10 マスのファイルが作られるので、何も入力しないで保管する。
4. BASE を閉じて MS-DOS のコマンドメニューでテキストファイルの内容表示を選択、固定ディスク内の BASE を選んで 3 で作ったファイルを選択する。

5.3.2 ヘッダ部分の解析結果

- 8 桁の数字の解析結果について

  同様の手順で 3 の左上のページ数が 12 になるまで改行をしてファイルを保管してテキストファイルで内容を表示させるとヘッダの部分には「00121020」と表示された。

  このことからヘッダ部分の数字の左から数えて 4 桁目まではページ数を意味していて、5、6 桁目は BASE の画面上の横マス数を表していて、7、8 桁目は縦の行数を表していることがわかった。

- ヘッダ部分の解析結果について

  ヘッダ部分の桁数は 512 桁であった。1 桁目から 504 桁目までは何も入っておらず 505 桁目から 8 桁の数字になっていた。視覚的には 7 桁のように見えるが、実際には行毎に改行はされておらず 1 桁目から 512 桁目まで一まとまりであった。改行されていたのは 8 桁の数字の後だけであった。

- 目録

  目録とはヘッダ部分に入力できる書名などのことである。目録の中身としては書名、サブタイトル、著者、出版社、発行日、図書コード、点訳、校正、作成日、備考の順で 10 個の項目で分けられていた。それぞれの桁数をまとめると以下のようになっていた。

目録の内容	桁数
書名	50 桁
サブタイトル	50 桁
著名	50 桁
出版社	50 桁
発行日	20 桁
図書コード	36 桁
点訳	36 桁
校正	36 桁
作成日	20 桁
備考	100 桁

目録を入力してテキストファイルの内容編集で確認した結果、一番初めの書名の前に 10 桁のスペースが入っていた。また、備考の後に 46 桁のスペースが入っていた。

目録は点字ではなく半角のカタカナ、英数字であった。目録の部分を BASE で入力して、テキストファイルの内容編集で確認した結果、入力された文字がそのまま表示された。

Fig. 5 BASE のヘッダ部分の構成図

5.4 BASE の形式の確認について

NABCC 形式は BASE 形式のヘッダ部分を無くしたものか確認する。

5.4.1 確認手順

1. BASE でひらがなの清音、濁音、半濁音、数字、アルファベットの大文字、アルファベットの小文字を分けて入力する。
2. CTRL + ^ + L + E ををを押して NABCC 形式を表示する。
3. 5.1 で解析した結果と見比べる。

5.4.2 NABCC(北米点字) コード

NABCC (北米点字) コードは、ASCII コードの 20h ~ 7Fh までの部分に 6 点点字のドットパターンを割り当てた最も標準的な点字コードである。但し、数字は下がり数字である。

NABCC コード					
16 進	ASCII	ドットパターン	16 進	ASCII	ドットパターン
20h	スペース	(——)	44h	D	⠙(1–45-)
21h	!	(-234-6)	45h	E	⠑(1—5-)
22h	“	(—-5-)	46h	F	⠋(12-4–)
23h	#	(–3456)	47h	G	⠛(12-45-)
24h	$	(12-4-6)	48h	H	⠓(12–5-)
25h	%	(1–4-6)	49h	I	⠊(-2-4–)
26h	&	(1234-6)	4Ah	J	⠚(-2-45-)
27h	’	(–3—)	4Bh	K	⠅ (1-3—)
28h	(	⠶(123-56)	4Ch	L	⠇ (123—)
29h	)	⠶(-23456)	4Dh	M	⠍(1-34–)
2Ah	*	(1—-6)	4Eh	N	⠝(1-345-)
2Bh	+	(–34-6)	4Fh	O	⠕(1-3-5-)
2Ch	,	(——6)	50h	P	⠏(1234–)
2Dh	-	⠤(–3–6)	51h	Q	⠟(12345-)
2Eh	.	(—-10)	52h	R	⠗(123-5-)
2Fh	/	(–34–)	53h	S	⠎(-234–)
30h	0	(–3–5)	54h	T	⠞(-2345-)
31h	1	(-2—-)	55h	U	⠥(1-3–6)
32h	2	(-23—)	56h	V	⠧(123–6)
33h	3	(-2–5-)	57h	W	⠺(-2-45-)
34h	4	(-2–56)	58h	X	⠭(1-34-6)
35h	5	(-2—6)	59h	Y	⠽(1-3456)
36h	6	(-23-5-)	5Ah	Z	⠵(1-3-56)
37h	7	(-23-56)	5Bh	[	⠪(-2-4-6)
38h	8	(-23–6)	5Ch	\	(12–56)
39h	9	(–3-5-)	5Dh	]	⠾ (12-456)
3Ah	:	(1—56)	5Eh	^	(–45-)
3Bh	;	(—-56)	5Fh	_	(—456)
3Ch	<	(12—6)	60h	‘	(—4–)
3Dh	=	(123456)	61h	a	⠁ (1——)
3Eh	>	(–345-)	62h	b	⠃ (12—-)
3Fh	?	⠦(1–456)	63h	c	⠉(1–4–)
40h	@	⠈(—4–)	64h	d	⠙(1–45-)
41h	A	⠁ (1——)	65h	e	⠑(1—5-)
42h	B	⠃ (12—-)	66h	f	⠋(12-4–)
43h	C	⠉(1–4–)	67h	g	⠛(12-45-)

NABCC コード		
16 進	ASCII	ドットパターン
68h	h	(12–5-)
69h	i	(-2-4–)
6Ah	j	(-2-45-)
6Bh	k	(1-3—)
6Ch	l	(123—)
6Dh	m	(1-34–)
6Eh	n	(1-345-)
6Fh	o	(1-3-5-)
70h	p	(1234–)
71h	q	(12345-)
72h	r	(123-5-)
73h	s	(-234–)
74h	t	(-2345-)
75h	u	(1-3–6)
76h	v	(123–6)
77h	w	(-2-456)
78h	x	(1-34-6)
79h	y	(1-3456)
7Ah	z	(1-3-56)
7Bh	{	(-2-4-6)
7Ch	—	(12–56)
7Dh	}	(12-456)
7Eh	~	(—45-)
7Fh	DEL(削除)	(—456)

5.5 解析したヘッダの確認について

今で解析してきたヘッダの桁数などを基にを UNIX でつくったテキスト形式のファイルをフロッピーディスクにコピーし、そのファイルを BASE で読み込ませせる。

5.5.1 確認手順

1. UNIX でファイル名「fd1.txt」というテキストファイルをつくる。

2. ファイルの中には 1 桁目から 504 桁目までスペースを入力し、その後に「00013222」と半角で入力し、改行をする。
3. 8 桁の数字の下の行からスペースを 1 桁入力し、その後に「ABCFI」と半角で入力し、改行する。
4. スペースを 2 桁入力して改行するという作業を 20 回繰り返して、ファイルを保存する。
5. kterm 上で nkf -Lw fd1.txt | cat - ~foo/lib/0x1a > fd2.txt と入力して、4 で保存したファイルを MS-DOS 用のファイルに変換する。
6. 変換した「fd2.txt」というファイルをフロッピーディスクにコピーする。
7. フロッピーディスクを MS-DOS にセットし、MS-DOS のコマンドメニューのファイルのコピー (ファイル単位) でフロッピーディスクに入っている「fd2.txt」というファイルを BASE のディレクトリの中にコピーする。
8. MS-DOS のコマンドメニューのファイル名の変更でコピーしたファイルのファイル名を「fd2.BSE」に変更する。
9. BASE で「fd2.BSE」というファイルを開く。

5.6 確認した結果

- 正しく入力した場合

  BASE で「fd2.BSE」を開くと 1 行目にスペースが 1 桁あいた後に「あいうえお」と表示され、2 行目から 21 行目まで何も入っていない行が表示された。

- 誤って入力した場合

  変換手順の 2 でスペースを 504 桁ではなく、501 桁にして手順通りに変換していき、ファイルを BASE で開こうとすると「ヘッダが異常です」と表示され、ファイルを開くことができなかった。

5.7 トランスレータのアルゴリズムについて

ここまで解析してきた BASE 形式の文字や BASE 形式のヘッダを基にトランスレータのアルゴリズムを考える。(図 6 参照)

1. わかち書きされた文章 (墨字) を左から右に一文字づつ点字に変換していく。
2. 今読んでいる文字が濁音・半濁音であるか？
   - 濁音・半濁音の場合
     一文字前に濁音符・半濁音符をつける。
   - 濁音・半濁音でない場合
     次の文字 (一文字右) にいく。
3. 今読んでいる文字が拗音であるか？
   - 拗音の場合 (拗音の「ャ」であるか「ュ」である場合)
     (a) 1 文字前に濁音符・半濁音符はある場合
     「ャ」なら一文字前の点字の母音を「ア」にかえ、「ュ」なら 1 文字前の点字の母音を「ウ」に変える。 1 文字前に濁音符・半濁音符があった場合、濁音符・半濁音符に拗音符を追加する。
     (b) 1 文字前に濁音符・半濁音符がない場合
     「ャ」なら 1 文字前の点字の母音を「ア」にかえ、「ュ」なら一文字前の点字の母音を「ウ」に変える。
   - そうでない (「ョ」) 場合
     (a) 1 文字前に濁音符・半濁音符はある場合 1 文字前の点字の母音を「オ」に変える。1 文字前に濁音符・半濁音符があった場合、濁音符・半濁音符に拗音符を追加する。
     (b) 1 文字前に濁音符・半濁音符がない場合
     1 文字前の点字の母音を「オ」に変える。
   - 拗音でない場合
     次の文字 (1 文字右) にいく。

5.8 トランスレータのプログラムについて

前節のフローチャートを基に Perl でトランスレータのプログラムを作成した。このプログラムはあらかじめ日本語が入力された perl.data をプログラムにかませて、perl.data を NABCC コードに変換していくものである。その実行結果を以下に記す。

確認手順

1. perl.data に以下のような日本語を入力し, それぞれの行毎に改行し、保存する。

   あいうえお かきくけこ さしすせそ
   たちつてと なにぬねの はひふへほ
   まみむめも や ゆ よ らりるれろ
   わ を ん

   がぎぐげご ざじずぜぞ
   だぢづでど ばびぶべぼ
   ぱぴぷぺぽ
   きゃきゅきょ しゃしゅしょ ちゃちゅちょ
   にゃにゅにょ ひゃひゅひょ みゃみゅみょ
   りゃりゅりょ
   ぎゃぎゅぎょ じゃじゅじょ
   ぢゃぢゅぢょ びゃびゅびょ
   ぴゃぴゅぴょ

2. プログラムを実行すると kterm に NABCC コードに変換されたものが表示されるので、それをコピーして tr.txt に貼り付ける。
3. 2 で作った tr.txt というファイルに 504 桁のスペースを入力し、半角で 00013222 と入力し、その数字の後を改行して保存する。
4. 5.5 で確認した同様の手順でフロッピーディスクに保存し、MS-DOS でファイル名の拡張子を「.txt」「.bse」に変更して BASE で読み込ませる。

確認結果

プログラムによって変換された NABCC コードを BASE で読み込ませると tr.txt に手順の 1 で入力した日本語がそのまま表示された。

6　まとめ

今回の研究では点字エディター、自動点訳ソフトについて、BASE について、点字の構造について調べた。

その結果、点字エディタにはいくつもの種類があり、視覚障害者が使えるように音声出力機能があるもの、点字を知らない晴眼者でも使えるようにかな入力できるものなど、それぞれ特徴が違っていた。同じように自動点訳ソフトにもそれぞれ違った特徴があり、日本語の文章をわかち書き文に変換するための、日本語を解析するソフトがそれぞれ違っていた。

BASE という点字エディターは視覚障害者にも点字を知らない晴眼者にも入力方式を変えることにより点字の文章を容易に書ける事がわかった。しかし、BASE の 6 点入力という入力方式ではキーボードの使用環境によっては使えないキーボードもあるという欠点も見つかった。

点字の構造について調べた結果、点字は視覚障害者が使うものなので、濁音や半濁音、拗音の場合に普通の文字 (墨字) とは違い、文字 (点字) を一つ一つ順番に読むことによって理解できるように作られていることがわかった。

さらに BASE 形式でのヘッダ部分の解析、また BASE 形式の文字が ASCII コードのどの文字に対応しているかについての解析を行い確認作業も行った。その結果 BASE では NABCC という形式が用いられていることがわかった。それら解析したデータを基に、わかち書きされた文章を NABCC 形式に変換する Perl によるトランスレータを作成した。そのプログラムに実際に拗音、拗濁音、拗半濁音の文章を入力した結果、NABCC 形式で正しいコードが出力された。しかし、そのプログラムだけでは漢字かな混じり文の変換はできないので、日本語解析システムを作成し、それを基に点字に変換するプログラムの作成が今後の課題である。

参考文献

[1] 松本賢一：コンピュータによる点字の学習と入出力に関する考察
新潟工科大学卒業論文 (2003 年)

[2] 初めての点字
(全国視覚障害者情報提供施設協会)

[3] ビジネス点字検定 3 級公式テキスト
(日本ビジネス点字検定協会 ビジネス点字検定委員会 [編著])

[4] パソコン点字工房
http://www2d.biglabel.ne.jp/~tenyaku/

[5] 障害のある子供の教育の広場
http:www.nise.go.jp/portal/elean/tenzi_com.html

[6] 石さんのホームページ
http://kiransou.hp.infoseek.co.jp/index.htm

[7] ケージーエス株式会社
http://www.kgs-jpn.co.jp/index.html

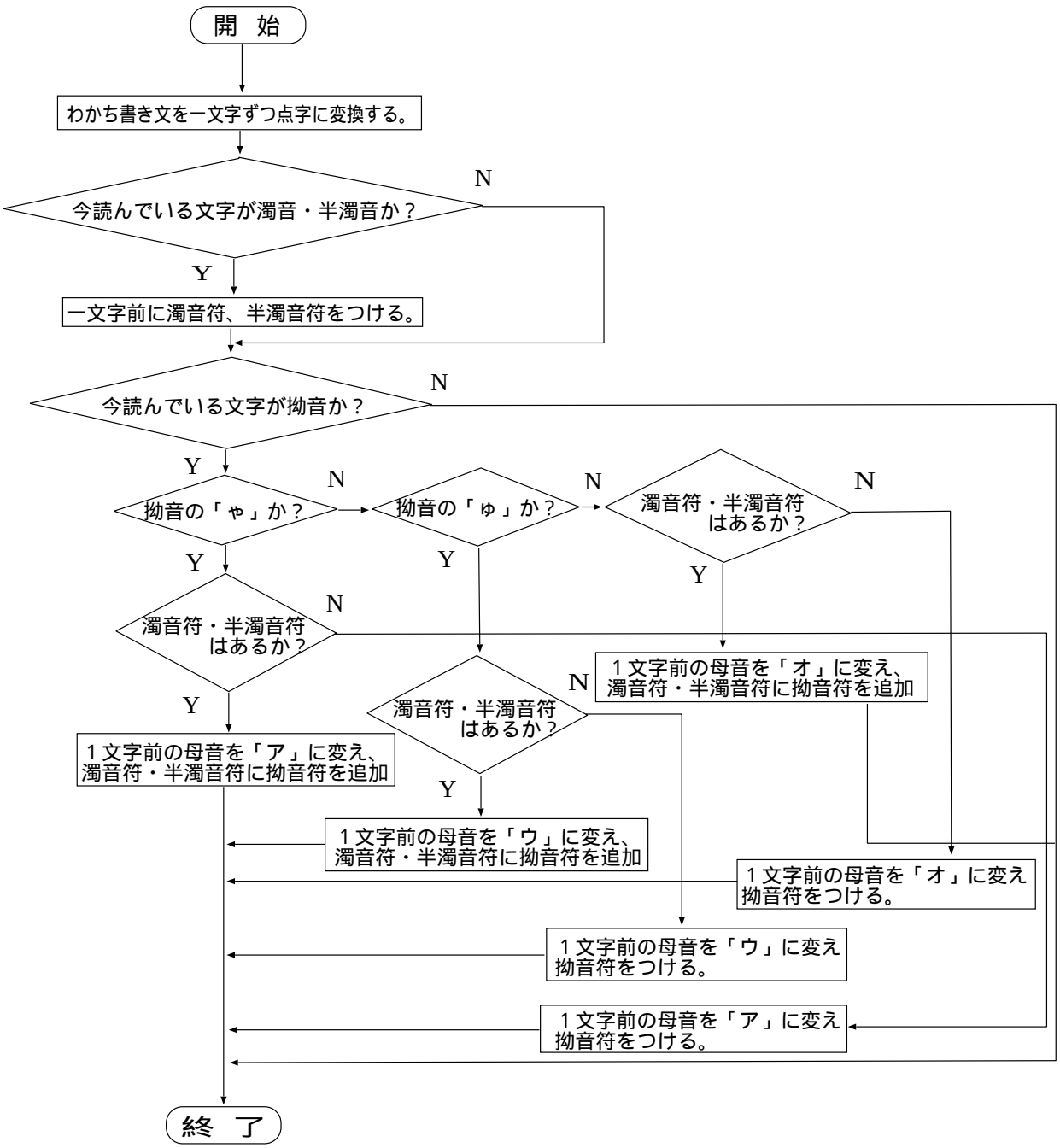

Fig. 6　拗音におけるトランスレータのアルゴリズム