On a Technique for Compact Generation of Symbols

RYUICHI SUZUKI*, KATSUO IKEDA* AND TAKESHI KIYONO*

1. *Introduction*

Several methods for software generation of symbols have been reported in some papers. In these papers lattice points arranged regularly are used to construct characters (1) by precise co-ordinate designation, or (2) by assigning codes to the points. In the former, considerably many points are necessary, figures of symbols are legible, but the amount of data bits required for a symbol becomes very large. In the latter, points can be selected as few as possible, probably the figures of symbols become ilegible, but the amount of data for symbols can be greatly reduced. This paper discusses a technique which combines the merits of each method and we set our goal to generate symbols as much as possible similar to those used in ordinary print using a comparatively rather small amount of data (32 bits) for each symbol.

The symbols we consider are all characters and marks adopted in ISO-code, including Japanese Kanamoji (Katakana).

2. *Distribution of the Characteristic Points*

Symbols are formed by the following two modes of lines.

1) unblank line
2) blank line

where these lines are connected in the drawing sequences.

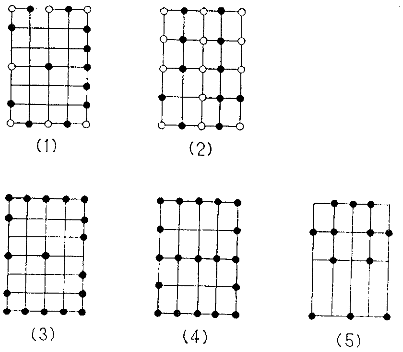
Now, we call the terminal points of the line segments the characteristic points. Symbols are constructed by connecting these points according to proper sequences and modes. Considering the shape (symmetrical, non symmetrical) and classification (alphanumerics, kanamoji, etc.) of symbols, we assort symbols into the following five groups:

1) Alphanumerics without symmetry
2) Kanamoji without symmetry
3) Alphanumerics with horizontal axis of symmetry
4) Alphanumerics with vertical axis of symmetry
5) Kanamoji with vertical axis of symmetry

By investigating the distribution of each group, we get the result shown in Fig. 1. Where ○ and ● denote centers around which many and a few charac-

This paper first appeared in Japanese in Joho Shori (the Journal of the Information Processing Society of Japan), Vol. 11, No. 9 (1970), pp. 517-522.

* Department of Information Sience, Kyoto University.



(1) asymmetric alphanmmerics (2) asymmetric kanamoji
(3) alphanumerics with horizontal axis of symmetry
(4) alphanumerics with vertical axis of symmetry
(5) kanamoji with vertical axis of symmetry

Fig. 1. The distribution of the characteristic points.

teristic points cluster, respectively.

3. *Expression of Symbols*

From the facts mentioned in the above section, each symbol can be constructed by properly selecting basic constructional points from the groups of characteristic points, and connecting some of them (constructional points) with two kinds of lines. The more constructional points chosen, the more legible symbols become. But the memory space required for symbols also increases, therefore it is necessary to find a compromise. Hereinafter, the data requirements for expressing the shape of symbols and the processing for their generation are considered.

The symbols treated here have the following properties:

i) there are many kinds of symbols,

ii) the distributions of the characteristic points are biased,

iii) in many cases unblank lines follow unblank lines.

Taking the above into consideration, one code not assigned to the position code is used for mode-designator, and in order to designate the other mode (either unblank or blank), some rules are established in the arrangement of codes.

Symmetric symbols generally require more constructional points than those without symmetry, but by a repeating process, the data required for a symbol are reduced by half. Therefore by classifying symbols as follows, and in each case choosing proper groups of characteristic points, symbols can be expressed more compactly.

1) Alphanumerics without symmetry
2) Kanamoji without symmetry
3) Alphanumerics with horizontal axis of symmetry
4) Alphanumerics and kanamoji with axis of symmetry

4. *Selection of Basic Constructional Points and Frames*

An important problem is to determine the suitable number of basic constructional points (p) and constructional points (N) from the distribution of the characteristic points shown in Fig. 1. We obtained $p=15$, and $N=8$ as the point of compromise between the legibility and data length of symbols so that each symbol can be generated rather legibly requiring 32 bits per symbol.

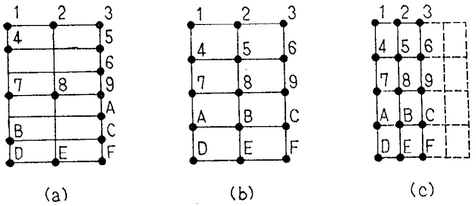
The 15 points shown in Fig. 2 are selected as basic constructional points so that as many symbols as possible may be constructed similar to those used in ordinary print, and are called frames for each group.

(a) is the frame for alphanumerics without symmetry and with horizontal axis of symmetry,

(b) is the frame for kanamoji without symmetry,

(c) is the frame for alphanumerics and kanomoji with vertical axis of symmetry.

The hexadecimal number in Fig. 2 indicates position code assigned to each basic constructional point.



(a) alphanumerics with horizontal axis of symmetry and asymmetric alphanumerics
(b) asymmetric kanamoji
(c) alphanumerics and kanamoji with zertical axis of symmetry

Fig. 2. Frames consisting of fifteen basic constructional points.

The code "0" is allocated to the mode-designator.

If the first data for a symbol is a "0" or "4", the frame for horizontal or for vertical axis of symmetry is selected, respectively; otherwise, frames for asymmetry are choosen and the code (other than "0" or "4") is interpreted as the position of the constructional point. Thus the symbols without axis of symmetry should start from a position other than the one coded "4". This, however, does not pose any question for the drawing sequence can be changed properly.

Frame for kanamoji and alphanumerics without symmetry is selected by shift-out (E of hexadecimal) and shift-in (F of hexadecimal), respectively.

5. *Generation of Symbols*

The code "0" is allocated to mode-designator of lines constructing symbols.

Lines to the points corresponding to codes following "0" are blank lines, while a continuation of codes not containing "0" is interpreted as unblank line in all other cases.

The co-ordinates of the points assigned by codes are computed refering to the tables storing co-ordinates of basic constructional points for each frame. A symmetrical symbol is divided into two parts, a fundamental part and a symmetric part. The co-ordinates of fundamental part are found in the previous manner while those of the symmetric part are calculated concerning symmetry.

The end of data for asymmetric symbols is (1) when eight constructional points are processed, or (2) when "0" appears contiguously. After the end of data, the processing of the symmetric part starts. About twice processing time is necessary for symbols with symmetry, it is reasonable to consider symbols as non symmetrical as much as possible. Finally, the following symbols are treated as symbols of symmetry.

1) symbols with horizontal axis of symmetry *B* 3
2) symbols with vertical axis of symmetry *A I W* 8 ハ ホ ニ

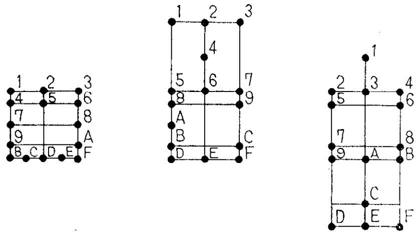
6. *Extention to Other Symbols*

6.1 *Marks*

Marks are contained in both alphanumerics and kanamoji. Examining whether marks in alphanumerics and in kanamoji can be expressed by frames for alphanumerics without symmetry and with symmetry, and by frames for kanamoji without symmetry and with axis of symmetry, respectively, all marks except "ⓐ" can be constructed satisfactorily, and even "ⓐ" can be expressed by "□" keeping the intuitive image of the marks.

6.2 *Lower Case Alphabet*

To lower case alphabet, the four kinds of frames mentioned above can not be applied. Taking notice that in ISO-code the left 4 bits, excluding the parity bit, have the same bit-patterns for the lower case alphabet, new frames can be set up. The following three frames (see Fig. 3) are necessary according to the domain occpied by each symbol.



(i) (ii) (iii)

Fig. 3. Frames for lower case letters.

Table 1. Generated patterns and data based on ISO-code. add {ア} denotes the address of ア

B_0~B_3 / B_4~B_7	0010		0011		0100		0101		0110		0111		0010		0011		0100		0101	
0000	SP	0-0	0	5-2-4-B E-C-5-0	@	E-D-1-3 F-E-8-5	P	D-1-2-5 6-8-7-0			p	4-D-2-3 6-8-A-9	SP	0-0	ー	7-9-0-0	タ	8-7-2-3 D-0-0	ミ	1-9-0-C 4-0-7-F
0001	!	4-3-2-C 0-F-0-0	1	F-D-E-2 4-0-0	A	4-D-3-0 8-9-0-0	Q	C-5-2-4 B-E-C-8	a	F-3-6-2 4-9-D-A	q	4-F-4-3 5-7-A-B			ア	1-3-8-0 5-B-D-0	チ	3-7-9-0 5-B-D-0	ム	2-D-C-0 9-F-0-0
0010	"	2-8-0-0	2	C-F-D-B 6-5-2-4	B	0-1-2-5 6-8-7-1	R	D-1-2-5 6-8-7-F	b	0-1-D-E C-9-6-5	r	1-B-0-7 2-3-0-0	「	3-2-E-0 0	イ	3-7-0-5 E-0-0	ツ	1-4-0-2 5-0-3-D	メ	3-D-0-4 C-0-0
0011	#	2-B-C-0 E-5-4-0	3	0-7-8-6 5-2-4-0	C	5-2-4-B E-C-0-0	S	5-2-4-7 9-C-E-B	c	6-2-4-9 D-A-0-0	s	6-2-4-7 8-A-D-9	」	2-E-D-0 0	ウ	2-5-0-7 4-6-9-E	テ	1-3-0-4 6-5-8-D	モ	7-9-0-1 3-2-E-F
0100	$	B-C-4-5 0-2-E-0	4	C-B-2-E 0-0	D	1-D-E-A 6-2-1-0	T	1-3-0-2 E-0-0	d	0-3-F-E B-8-6-7	t	0-5-7-0 4-E-F-0	、	B-E-0-0	エ	8-3-1-0 2-E-D-F	ト	1-D-0-7 9-0-0	ヤ	8-6-4-0 2-E-0-0
0101	%	D-3-8-2 4-C-E-8	5	3-1-7-8 A-C-E-B	E	3-1-D-F 0-9-7-0	U	1-B-E-C 3-0-0	e	7-8-6-2 4-9-D-A	u	1-9-D-A 3-F-0-0	・	E-0-0	オ	6-4-0-2 E-0-D-5	ナ	6-4-0-2 8-D-0-0	ユ	1-2-E-0 D-F-0-0
0110	&	F-1-2-D C-0-0	6	5-2-4-B E-C-9-7	F	3-1-D-0 7-9-0-0	V	1-E-3-0 0	f	0-3-2-E 0-5-7-0	v	1-D-3-0 0	ヲ	1-3-9-D 0-7-9-0	カ	E-6-4-0 2-D-0-0	ニ	4-5-6-0 A-C-0-0	ヨ	1-3-F-D 0-7-9-0
0111			7	E-3-1-4 0-0	G	5-2-4-B E-C-9-8	W	4-1-E-3 0-0	g	4-D-F-4 3-5-9-B	w	1-C-2-E 3-0-0	ァ	add {ア}	キ	6-7-0-A 9-0-2-E	ヌ	1-3-D-0 4-C-0-0	ラ	1-3-0-4 6-D-0-0
1000	[	3-2-E-F 0-0	8	4-3-4-8 9-8-A-F	H	1-D-0-7 9-0-3-F	X	1-F-0-3 D-0-0	h	6-1-D-0 5-7-F-0	x	1-F-0-3 B-0-0	ィ	add {イ}	ク	7-2-3-D 0-0	ネ	2-4-6-A 0-E-8-F	リ	1-7-0-3 9-E-0-0
1001	]	1-2-E-D 0-0	9	B-E-C-5 2-4-7-9	I	4-2-3-F E-0-0	Y	1-8-3-0 8-E-0-0	i	0-4-0-6 E-0-0	y	4-2-A-0 4-D-0-0	ゥ	add {ウ}	ケ	2-4-6-0 5-B-D-0	ノ	3-6-D-0 0	ル	1-D-0-2 E-9-0-0
1010	*	2-E-0-5 B-0-4-C	:	4-6-0-C 0-0	J	2-3-C-E B-0-0	Z	C-F-D-3 1-4-0-0	j	4-1-0-3 C-D-0-0	z	A-F-B-3 1-4-0-0	ェ	add {エ}	コ	4-6-F-D 0-0	ハ	4-2-D-0 0	レ	1-D-9-0 0
1011	+	2-E-0-7 9-0-0	;	4-6-0-C F-0-0	K	1-D-0-3 7-F-0-0			k	0-1-D-0 7-A-F-0			ォ	add {オ}	サ	D-9-3-6 4-0-2-8	ヒ	1-D-F-0 9-7-0-0	ロ	1-D-F-3 1-0-0
1100	,	8-E-0-0	<	5-7-C-0 0	L	1-D-F-0 0			l	0-2-E-0 0			ャ	add {ヤ}	シ	1-5-0-8 4-0-D-9	フ	1-3-D-0 0	ワ	7-1-3-9 D-0-0
1101	-	7-9-0-0	=	4-A-C-0 6-4-0-0	M	D-1-E-3 F-0-0			m	B-1-4-2 D-5-3-F			ュ	add {ユ}	ス	1-3-D-0 8-F-0-0	ヘ	7-5-C-0 0	ン	1-5-0-6 F-0-0
1110	.	E-0-0	>	B-9-4-0 0	N	D-1-F-3 0-0			n	B-1-4-3 F-0-0			ョ	add {ヨ}	セ	9-6-4-0 2-E-F-0	ホ	4-F-3-6 4-0-8-D		
1111	/	D-3-0-0	?	E-8-6-5 2-4-0-0	Ø	5-2-4-B E-C-5-8	_	D-F-0-0	o	6-2-4-9 D-A-6-0			ッ	add {ツ}	ソ	1-5-0-3 D-0-0	マ	1-3-6-B 0-7-F-0		

i) a c e m n o r s u v w x z

ii) b d h i k l t

iii) g j p q y

The method of selection of each frame is similar to that of upper case alphabet, if the first code of data is "0", "4", or some other code then the frame for (ii), (iii), or (i) is selected, respectively.

6.3 *Yo-on (Contracted Sounds of Kanamoji)*

Yo-on are similar to ordinary Kanamoji, but are not equal in size and position.

Reflecting upon actual processing routine, the method only using data for kanamoji and setting up a frame for Yo-on is considered to be most suitable.

The frame for Yo-on is selected like that of the lower case alphabet, by the left 4 bits except for parity bit and right four bits.

7. *Conclusion*

This paper discusses a technique for the compact generation of all symbols adopted in ISO-code. Symbols similar to those used in ordinary print are generated by using frames which consist of 15 basic constructional points. These points are determined by both the symmetry of symbols and the biased distribution of characteristic points in each set of symbols such as alphanumerics, kanamoji, etc.

Symbols are generated, by selecting an approximate frame according to shift-codes, bit-patterns, and special codes, and by connecting less than eight constructional points.

By adopting this technique, data for each symbol can be stored within 32 bits, and all symbols in ISO-code can be generated legibly. So, this technique seems to be suitable for a standard way of generation of symbols in ISO-code.

Reference

[1] Togawa, H.: A Method to Pack Data for Generation of Symbols by Strokes within a Memory Word. Information Processing Society of Japan, Preprints for Lecture of the Tenth Conference in 1970.

[2] Hitt, D. C., G. H. Ottaway and R. W. Shirk: The Mini-Computer—A New Approach to Computer Design. *FJCC*, *33*, pp. 655 (1968).

[3] Suzuki, R., K. Ikeda and T. Kiyono: A Technique for Compact Generation of Symbols. *J. Information Processing Society of Japan*, *11*, 517–522 (1970).