FORTRAN中文輸入輸出程式

戚　樹　紅
國立交通大學工學院
(Received 28 Nov. 1973)

摘要—本篇主要介紹的是一個以 Fortran 語言來設計中文字根合成中文字的程式，此程式是綜合參考資料(1)、(2)、(3)所得之結果。中文字根是用 relative line segment 的方法表示的。

一、簡　介

本文旨在介紹一個用 Fortran 語言寫出的中文輸入輸出程式。此程式所依據的是參考資料(1)、(2)、(3)，探討以中文字根合成中文字的概念。雖然用高階語言寫程式所佔記憶空間比用低階語言大？但用 Fortran 語言的最大好處是它能將之用於計算機 (machine independent)，使這套以中文字根合成中文字的系統得以廣泛的應用。

二、字形的組合

中文字是以字根式表示，而字根式由 496 個中文字根、括號、定位符號◬、会、ᐞᐞ及方便符號∴、⁚、∘∘，組合而成（參考(1)、(2)、(3)），字形組合時各符號組合之順序有下列三規則：

1. 與算術運算同，括號為最優先，即括號中之字根必先組合。
2. 凡遇∴、⁚、∘∘三方便符號之一則不管其後所跟為何，先將其前之字根照所遇方便符號組合。
3. 字根之定位符號優先順序為◬>会>ᐞᐞ，故在處理時先作◬之組合。

三、數據形式與結構

程式中的數據分為資料檔與運算檔兩大類，表 1 中將各數據的名稱與形式，包含的資料等一一列出。資料檔包括 Itab 1, Itab 2, Itab 3 三項，其中 Itab 1 存文字符號，是為輸入子程式所準備的，Itab 2 與 Itab 3 中則存 496個中文字根的所有資料。運算檔包括 Istk 1, Istk 2, Istk 3 三項，Istk 1 與 Istk 2 是存中文字根式中用到的中文字根資料，它是由 Itab 2 與 Itab 3 中得的， Istk 3 則存定位符號。在組合字形時數據的運算以運算檔處理，故運算檔中之資料隨字根的組合情形而改變，而資料檔則一經建檔後即不改變。 資料檔與運算檔間的關連見圖 1， 至於 Itab 2 與 Itab 3 中詳細的字根資料所代表的意義見參考資料(3)，圖 2 舉例說明「口」字的字根資料。

四、程式之結構

中文輸入輸出程式包括主程式 (main program) 和十一個大小不等的子程式 (subprogram)。主程式是用來建立資料檔，並控制從字根式的資料送入至輸出中文字形過程中所有的步驟。由圖 3 程式的結構和圖 4 主程式之流程圖即能瞭解它在整個程式中扮演的角色。圖 5 更舉例說明運算檔資料在整個組字程式的變化過程 。 主程式與各子程式間的數據連繫 (Data Link) 除了有自變數 (argument) 互通消息外（見表 2），為節省記憶空間起見， 還用了共同區域 (common block)，在共同區域中存 Itab 1, Itab 2, Itab 3, Itsk 1, Itsk 3（參考表 1）等資料。各子程式可直接自共同區域中取出這些資料，不需經由自變數而得。以下將各子程式的作用略加說明：

1. 輸入子程式 (INP Subprogram)：它用來將字根式讀入，並將之轉變為數字串 (number string)，例如 CD+AC⟶56 602 3，在 Itab 1 中存有A至Z 26個文字及（，+，－，等符號，這些文字符號所存的位置即代表其本身的數值，如C即為 3，每一個字根是由二個英文字母組合而成，由字根轉變成數字時，將第一個英文字母所代表的數值減 1 乘26加第二個英文字母所代表的數值，如 CD=(3−1)×26+4=56 。符號則以公式：600+（符號−26）來計算，如「+」在第28

表1. 中文輸入輸出程式中數據之形式

類別	名稱	所存資料	形式	備註
資料檔	I TAB1	ABC....Z(+−*#&'),	I TAB1(1)=A, I TAB1(2)=B ..... I TAB1(35)=,	+——△△ #——○○ −——△ &——○○ *——△ '——○
	I TAB2	class, WX, WY, L1, L2, L3, L4 pointer 1	I TAB2(I,1)=Class WX WY I TAB2(I,2)=L1 L2 I TAB2(I,3)=L3 L4 I TAB2(I,4)=pointer 1	Class, L1, L2, L3, L4 are expressed by two digits WX, WY are expressed by one digit
	I TAB3	relative line segment data IX, IY, IB	I TAB3(J)=IB IND AX AY IB: "+" positive means the line segment data is bright "−" negative means the line IND: "0" means IX>0, IY>0 "1" means IX>0, IY<0 "2" means IX<0, IY>0 "3" means IX<0, IY<0 AX: the absolute value of IX AY: the absolute value of IY	IND is expressed by one digit AX, AY are expressed by two digits Note: Between two radical data there exists I TAB3(J)=99
運算檔	I STK1	class, WX, WY, L1, L2, L3, L4, pointer 2	ISTK1(J,1)=pointer 2' ISTK1(J,2)=class WX WY ISTK1(J,3)=L1 L2 ISTK1(J,4)=L3 L4	pointer 2 is the position in which the begin line segment Date of the Jth radical set.
	2 ISTK2	IX, IY, IB	ISTK2(NO,1)=IB ISTK2(NO,2)=IX ISTK2(NO,3)=IY	Note: Between two radical data there exists ISTK2(NO,1)=2 ISTK2(NO,2)=0 ISTK2(NO,3)=0
	I STK3	operator	ISTK3(KOP)=operator	

Class: the class of the radical
WX: Weight in X direction
WY: Weight in Y direction
L1, L2, L3, L4
represent the efficient location or contained position of the radical

Class= 0 一般根
10 上 根
20 左 根
30 下 根
40 右 根

Class= 1 口日白土
11 大八天
50 門勹勹…
60 走是尤…
61 ナ戶尸…
62 截 气

包含根

Note: 資料檔存字根數據爲16×16字形，但於運算檔則存64×64字形之數據

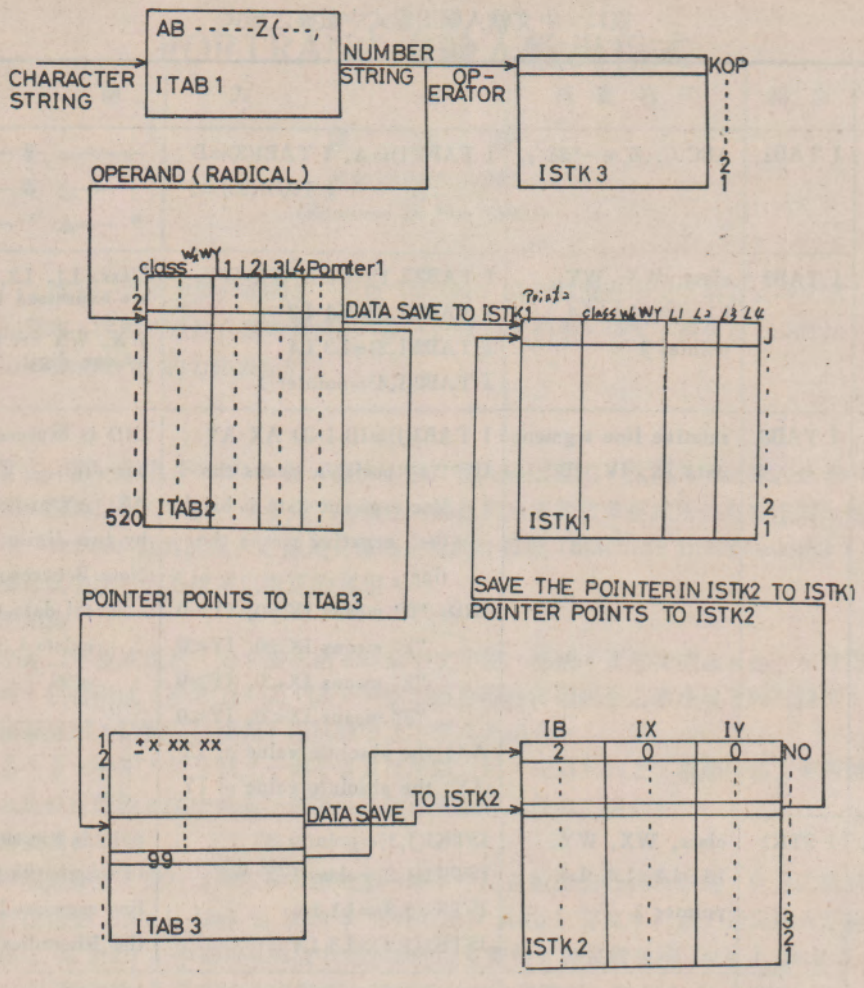
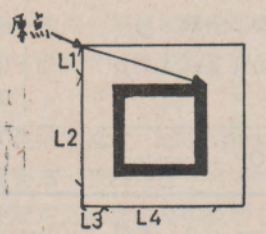


圖 1. 數 據 之 結 構



J	IX	IY	IB	ITAB3(J)
1	8	4	0	-01104
2	-8	0	1	20800
3	0	8	1	8
4	8	0	1	800
5	0	-8	1	10008
6	7	-4	0	10704

class = 0 WX=3 WY=3

L1 = 2 L2=13

L3 = 2 L4=13

Pointer1 = 1

ITAB2(1,1) = 33

ITAB2(1,2) = 213

ITAB2(1,3) = 15

ITAB2(1,4) = 1

□ : AA⟶1

圖2. 字根□之資料擋

表2. 中文輸入輸出程式中之各

類 別	名 稱	自 變 數 (Argument)	函數 (Function Algorithm)	備 註
輸 入	INP 輸入 子程式	IN, ICH, IND IN:the length of the number string ICH: the number string IND: the indicator of the error	ICH=f (character string) IN =g (character string) IND=h (character string)	
控 制	HVSUB 橫直連 子程式	ISTAR, IEND, KOP, J, ISTK2 ISTAR:the starting radical position IEND: the end radical position KOP: the pointer of the operator J: the pointer of the radical ISTK2: the relative line segment data of the radical	ISTK2=f (ISTAR, IEND, ISTK2) KOP =KOP−(IEND−ISTAR) J =J−(IEND−ISTAR)	CALL SUBROUTINE VERTC HORIZ
組 字	CNTAN 包含根 處理 子程式	J, KOP, ISTK2	$ISTK2=ISTK2_{J-1} \blacktriangle ISTK2_J$ J =J−1 KOP=KOP−1	
	VERTC 直連 子程式	IV, IEND, J, KOP, ISTK2 IV: the number of vertical operators	$ISTK2=\overset{\triangle}{\triangle}{}_{i=IEND-IV}^{IEND} ISTK2_i$ KOP=KOP−IV J=J−IV	$\overset{\triangle}{\triangle}{}_{i=n}^{m} ISTK2_i$ $=ISTK2_n \overset{\triangle}{\triangle} ISTK2_{n+1} \overset{\triangle}{\triangle} \cdots \overset{\triangle}{\triangle} ISTK2_m$ where m>n
	HORIZ 橫連 子程式	IH, IEND, J, KOP, ISTK2 IH: the number of horizon operators	$ISTK= \triangle\triangle_{i=IEND-IH}^{IEND} ISTK2_i$ KOP=KOP−IH J=J−IH	$\triangle\triangle_{i=n}^{m} ISTK2_i$ $=ISTK2_n \triangle\triangle ISTN2_{n+1} \triangle\triangle \cdots \triangle\triangle ISTK2_m$ where m>n
	THRE $\overset{\circ}{\circ\circ}$子程式	J, NUM, ISTK2 NUM: the pointer of the relative line segment data	$ISTK2=ISTK2_{J\overset{\circ}{\circ\circ}}$ NUM=f(ISTK2, J)	
	TWOV $\overset{\circ}{\circ}$子程式	J, NUM, ISTK2	$ISTK2=ISTK2_{J\overset{\circ}{\circ}}$ NUM=f(ISTK2, J)	
	TWOH $\circ\circ$子程式	J, NUM, ISTK2	$ISTK2=ISTK2_{J\circ\circ}$ NUM=f(ISTK2, J)	
輸 出	PRT 輸出1 子程式	NO, ISTK2, INOX, INOY NO: the pointer of the relative line segment data INOX: the length of the output from in the X direction INOY: the length of the output from in the Y direction	$\sum_{i=1}^{NO} ISTK2_x=\sum_{i=1}^{NO} ISTK2 *INOX/64$ $\sum_{i=1}^{NO} ISTK2_y=\sum_{i=1}^{NO} ISTK2 *INOY/64$	
	OUT 輸出2 子程式	NO, ISTK2, INOX, INOY	MOUT=f(NO, ISTK2, IX, IY)	MOUT is the dot matrix from of the Chinese character
其 他	SEYXP	N, IX, IY, IB N: the number to be seperated IX: the number seperated from N in X direction IY: the number seperated from N in Y direction IB: index of N whether it is positive or not	IX=f(N) IY=g(N) IB=h(N)	

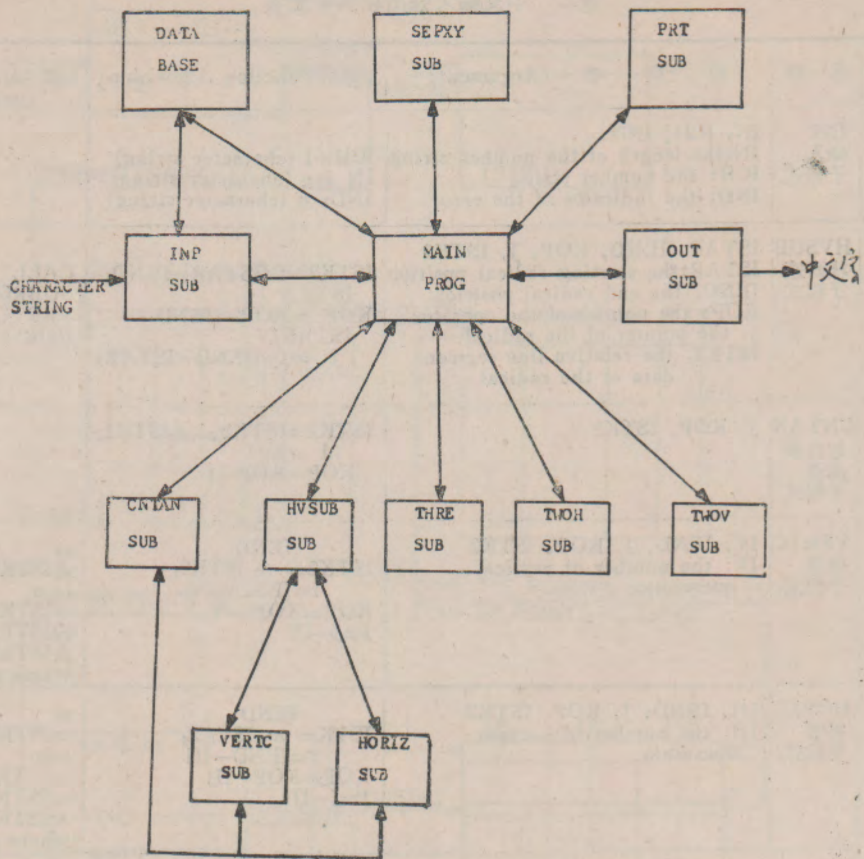


圖3. 程式之結構圖

位置即爲600+(28−26)=602。

2 橫直連控制子程式 (HVSUB Subprogram)：此子程式主要處理一連串的橫直連的字根組合。由於在運算中直連定位符號較橫連定位符號爲優先，所以直連組合應先行處理。處理橫直連組合時用直連子程式與橫連子程式，所以它是一個控制子程式。假若有一連串的直連組合，則須一起處理，而非兩個，兩個的處理，如「樹」的字根式爲木⩍土㑹口㑹⩍⩍寸，則「土」，「口」，「⩍」三者直連而非「土」與「口」直連成「吉」再與「⩍」直連成「壴」。

3 包含根處理子程式 (CNTAN Subprogram)：它是用來處理包含定位符號的字根組合（參考(3)），處理方法分爲下列幾項：

(A) 類別 (class) 爲50者將被包含字根縮小至其所應佔位置如圖 6 (A)，合成字形的比重則與包含根同。

(B) 類別爲60時，按比重將兩字根橫連組合，必須考慮字根字形的最後一筆，數據中最後一筆爲歸向原點，故字形的最後一筆實際爲數據中的最後二筆。所以若包含根最後第二筆爲鈎，如「九」即X方向變化爲〇，僅Y方向有變化，故將包含根最後第三筆延長，否則將最後第二筆延長（見圖 6 (B)。合成字形的比重按橫連處理。

※在此類別爲60之字根在寫數據時卽已規定其最後第二筆爲鈎或捺。

(C) 類別爲61時則按比重以直連組合，直連組合後將包含根的第二筆延長（見圖 6 (C)）。合成的字形比重按直連組合處理。

※在此類別爲61之字根在寫數據時已規定第二筆爲需延長的撇。

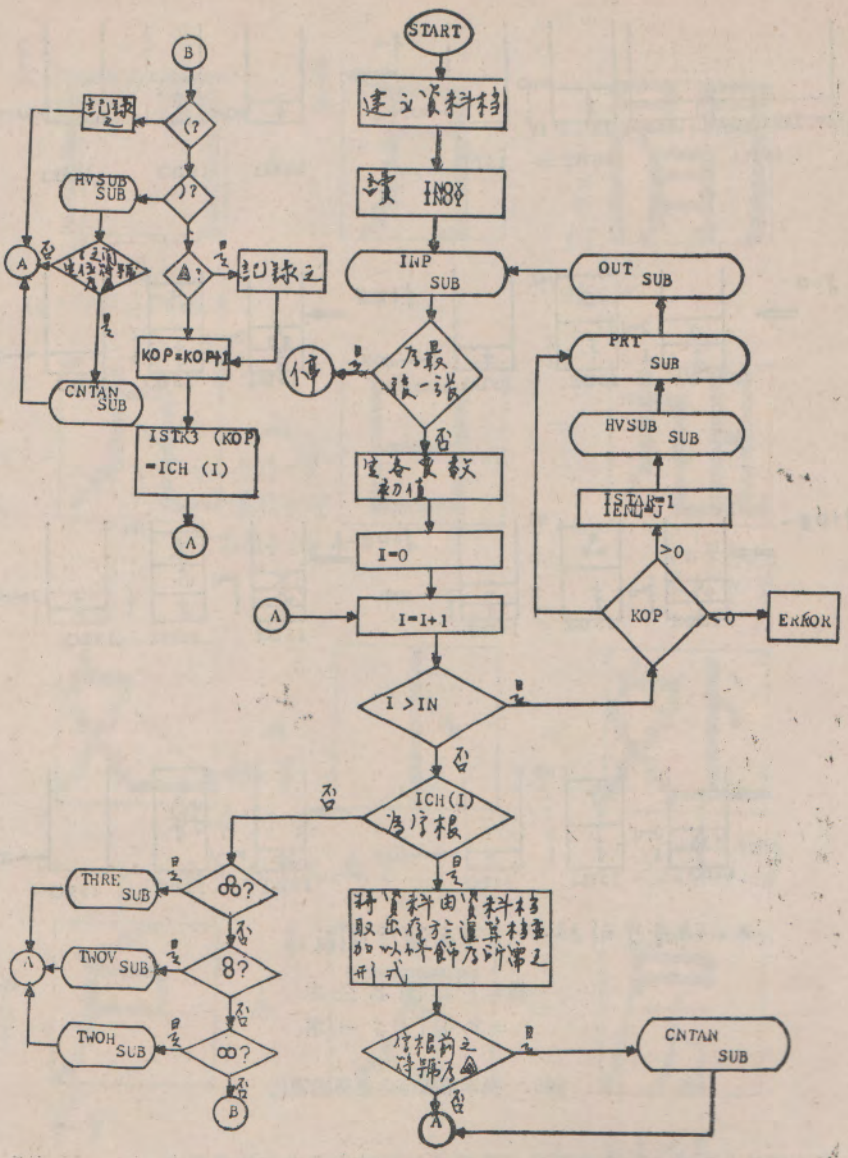


圖4. 主程式之流程圖

(D) 類別爲62時組合與類別61同，但是將最後第四筆延長而非第二筆延長。如「气」與「戈」，此兩字根在寫數據時亦應注意。

(E) 當字根之類別非屬包含根而定操作符號却以包含處理，如「奈」的字根式爲：大会＝会小，則以直連處理之。

若被包含字根不爲包含根則令合成字形後的類別爲60，此乃配合「鼸」「鼹」等所定。並令 L1, L3=0 且 L2, L4=64 。否則令其類別爲被包含者之類別，而被包含位置之大小 L1, L2, L3, L4 隨被壓縮字根改變見圖7。

4. 直連子程式 (VERTC Subprogram)：此子程式專門處理直連字根的問題。處理方式爲按比重的大小與有效位置之關係，將直向應壓縮的字根壓縮組合，（參考(3)）。但需考慮是否屬於下列幾種情形：

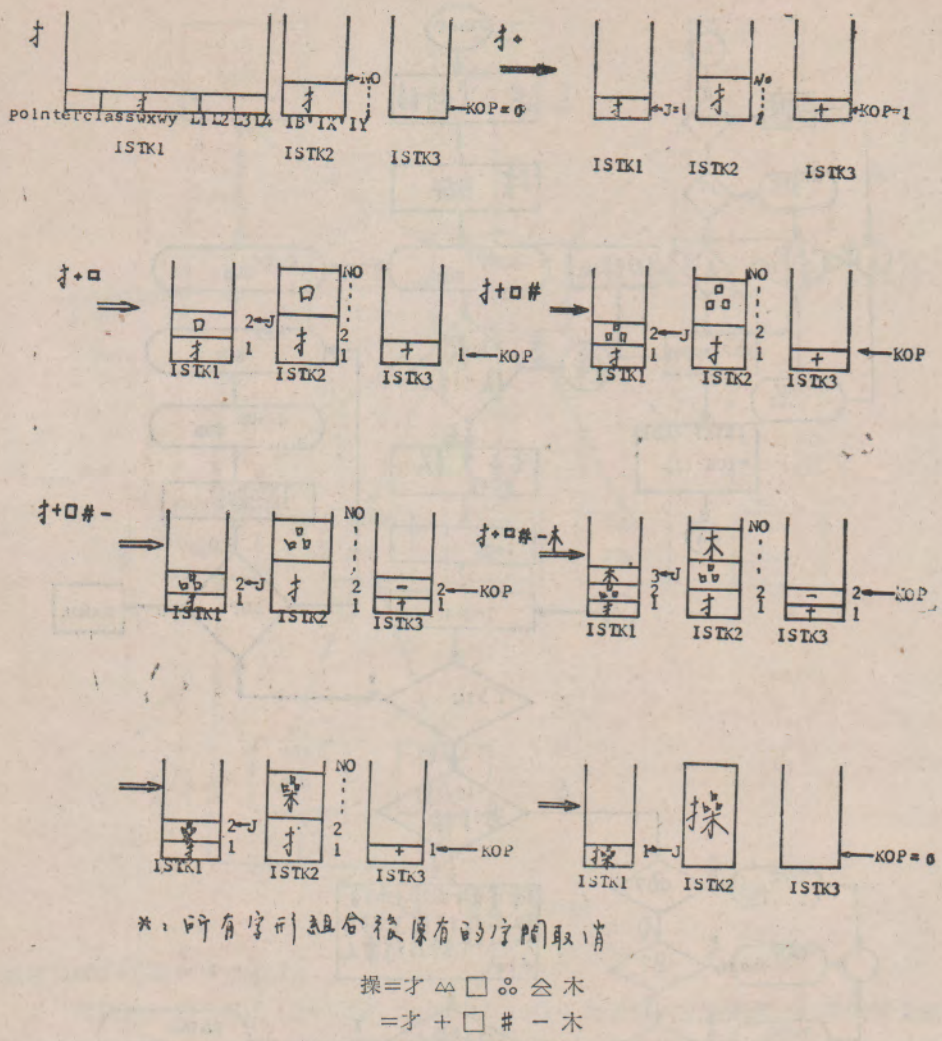


圖5. 操字形組合之運算擋變化

(A) 是否屬於 3 (C)成 3 (D)之情形？若是則按其所該處理之方法處理之。

(B) 是否屬於「大」，「天」，「八」等類字根？此類字根定爲類別11，依此類字根定有效位置之大小，以便有些字有較多的重疊部份，而使字形較爲美觀如「奈」，「添」，但如「奘」字，「大」在最下面，直連時若以有效位置直連則會超出位置（圖 8），故若此類字根爲出現在直連組合中之最下面的位置時，此等字在直連最後時，則有效位置應更換成L1, L3=0, L2, L4=64。

(C) 是否屬於「口」，「日」，「白」等字根？因爲此等字根在直連組合時，橫連必按一定比例縮小，如「呆」字，此類字根定類別爲 1 。

將字根直連組合後字形的類別定爲 0 ，且直向比重爲各直連根直向比重之和，而橫連比重則取各橫連比重之最大值。

5. 橫連子程式 (HORIZ Subprogram)：此程式專門處理橫連組合，組合時按字根橫連比重的大小比例壓縮，如爲下列兩種情形時，則應予特別處理。

(A) 是否屬於「口」，「日」，「白」，等字根類的字根？此類字根橫連時直向亦需壓縮，如「喝」字之「口」。

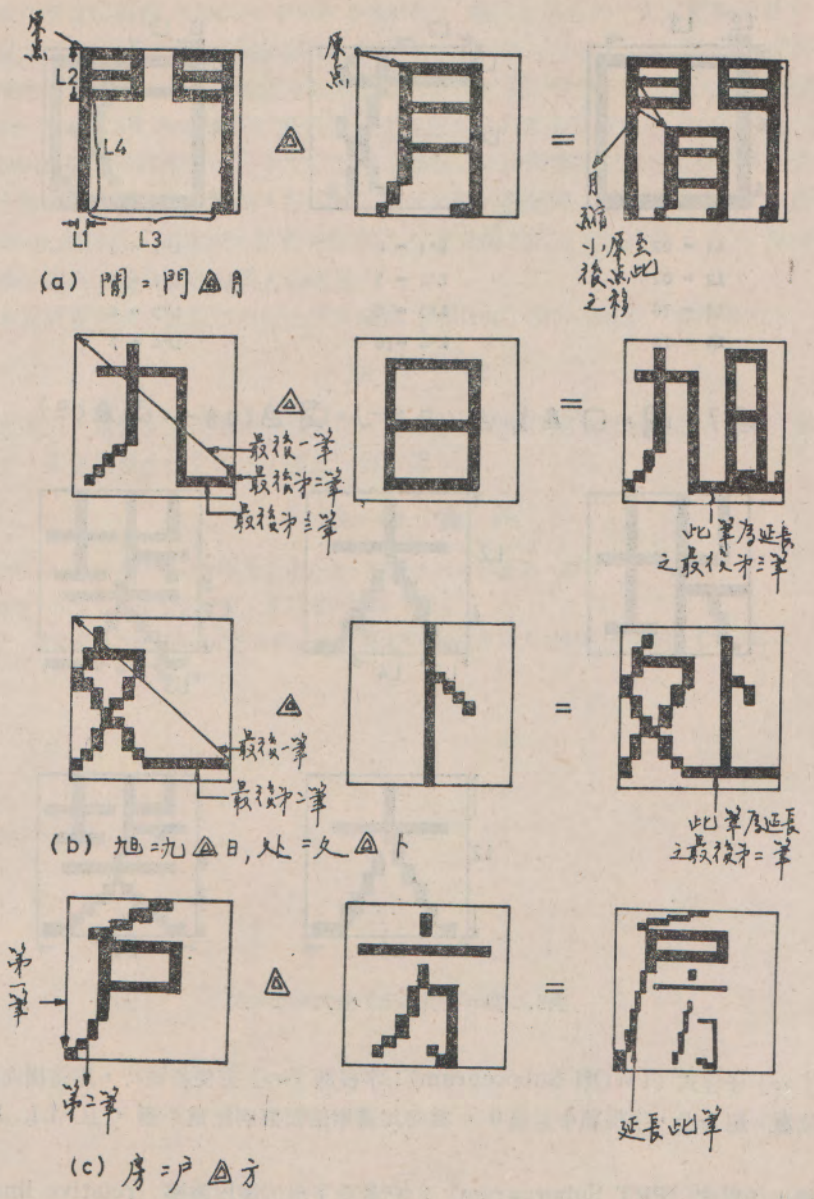


圖6. 包含根的組合

(B) 是否屬於 3 (B)中之情形？若是則按其法處理之。

橫連字根組合後，橫向之比重爲各橫向組合字根之橫向比重和，而直向比重爲其中各字根直向比重之最大值，類別則變爲 0 。

6. ⁂子程式 (THRE Subprogram)：字根遇「⁂」方便符號時，則將字根橫直各壓爲一半，放在適當位置。組合後的字根類令其爲 0 ，橫向比重加倍，直向比重加倍，且 L1, L3=0, L2, L4=64。

7. 「⁘」子程式 (TWOV Subprogram)：字根遇「⁘」方便符號時，則將直向壓縮一半，放在適當位置。組合後字根類令其爲 0 。橫向比重不變，直向比重則變爲兩倍，且 L1, L3=0, L2, L4=64。

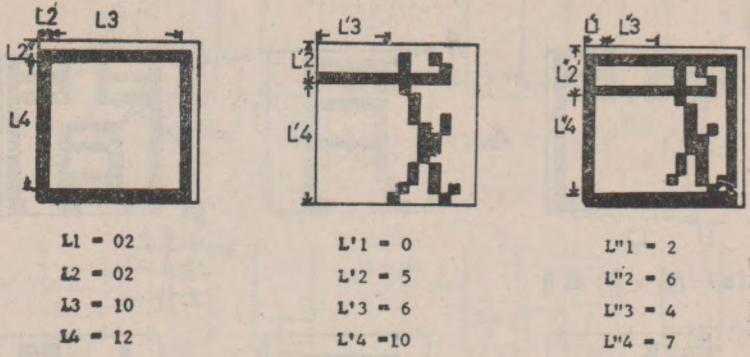


圖7 國＝□ △戈 △（口合一）＝国 △（口合一）＝国 △（口）

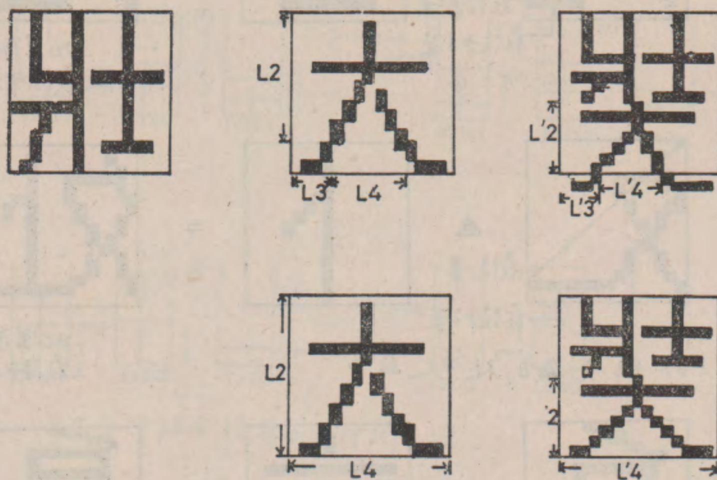


圖8. 奘＝（爿亼士）亼大＝壯亼大

8. 「◦◦」子程式 (TWOH Subprogram)：字根遇「◦◦」方便符號時，則將橫向壓縮一半，放在適當位置。組合後，字根類令其為 0 ，橫向比重增倍而直向比重不變 ，且 L1, L3=0, L2, L4=64。

9. 輸出子程式[1] (PRT Subprogram) ：它是將字形的線段數據 (relative line segment data) 縮小或放大為所需輸出字形的大小。

10. 輸出子程式[2] (OUT Subprogram) ：它是負責將線段數據轉換成點矩陣 (dot matrix) 將之輸出。

11. SEPXY 子程式：資料由資料檔 Itab 3 取出放於運算檔 Istk 2 時（參考圖 1 ），由於兩者之間的數據形式不一致，必須將原有以濃縮方式存於 Itab 3 中的 IX, IY, IB 分析出來（參考表 1 ），然後才能存於 Istk 2 中，此即SEPXY 子程式之作用。

五、結　　論

1. 這個以 Fortran 寫成的中文輸入輸出程式可供給任何具有處理 Fortran 能力的計算機來應用。僅需將輸出的中文字形大小資料，中文字根式打成卡片，放在建檔的資料卡後面即可。輸出字形的大小可任意變化。

2. 程式中包含主程式和十一個大小不等的子程式，最大的好處是若程式中的部份處理方法改

變時，將處理此部份的子程式加以更改即可。例如輸出子程式現在是在本校 IBM 360 終端機上輸出，若改用靜電印刷機 (electro static printer) 輸出，則將此子程式更換即可。

3. 程式所佔的記憶單位爲54千排 (K bytes)，共同區域 (common block) 需24千排，故總共需78千排的記憶單位。用高階語言寫程式的最大缺點就是記憶單位的浪費，通常一般計算機都能用 Fortran 語言 Call Assembly 語言做，故有些有關的數據安排可用 Assembly 語言寫。

4. 以64×64點矩陣形式輸出中文字形時，僅輸出部份所需的記憶單位就要 8 千排。事實上有些字形組合複雜，會超出預留範圍，故必須保留68×68的點矩陣空間，即約需 9 千排所以輸出程式爲用 Fortran 語言寫，則比較，浪費記憶單位，而且處理的速度也嫌慢，若用 Assembly 語言改寫輸出部份則對上述缺點會有很大的改進。

5. 本程式僅完成中文組字部份，至於編輯 (editing) 部份還沒做，有待努力。

謝　　誌

本文之完成，承蒙謝清俊教授、蔡新民、鄭武皇等講師提供寶貴意見及多方的指導，復承袁正春、李正芳、張慧雲諸小姐之協助，謹此一併致謝。

參 考 資 料

1. 謝清俊等，中文字根的貯存和中文字的合成，交大學刊第六卷第一期。
2. 謝清俊等，中文字根之分析，交大學刊第六卷第一期。
3. 朱素琴、戚樹紅，"中文字根之貯存與字形之美化"，國立交通大學科技研究報告 CS-005。