TR-88-007

以字根圖像作印刷體中文字辨認的研究

TR-88-007

以字根圖像作印刷體中文字辨認的研究

中央研究院資訊研究所

鄭國揚

參與人員：羅進財

中華民國七十七年六月

誌　謝 i
摘　要 ii
第一章　緒　論 1
第二章　影像之處理 14
第一節　影像的衰減和影像處理 14
第二節　影像的前處理 16
第三節　筆畫抽取 23
第四節　三角碼 26
第三章　字根圖像之視覺結構 30
第一節　一般字根圖像之視覺結構 30
第二節　字根圖像之編碼 32
第三節　三角字根圖像的視覺結構 35
第四節　辨認三角字根圖像之自動機 37
第四章　自動推論中文字根圖像文法 41
第一節　正則語言與網脈文法 41
第二節　自動推論本文無關網脈文法 46
第五章　以學習模式解決鄰近缺失 51
第一節　學習的過程 51
第二節　樣本學習 53
第三節　三角字根學習方式 57
第四節　學習系統的流程 62
第六章　自動切除鄰近缺失 64
第七章　結　論 68
參考文獻 71
附　錄 73

摘要

中文字元辨認是圖形辨認 (pattern recognition)的難題之一，即使已有許多學者投入此項問題的研究，還是讓人有杯水車薪之感，因為中文字元辨認實在是一個極困難的問題．但中文電腦普及化的困難所在是繁瑣的中文輸入法和大量中文登錄時間，而中文字辨認正是解決這些問題的利器．所以，解決中文字元辨認的困難是我們責無旁貸的工作．

在本篇論文中，我們將中文字視為字根圖像的幾何組合，探討了其間本文無關網脈文法 (context-free web grammar) 的自動推論問題，並提出了一個有效的推論演算法 (inference algorithm).

但是中文字數繁多，包含在中文字中的字根圖像數目不是我們所能歸納得到的，而且其數目太大，即使整理出來亦可能不切實際．所以我們提出以中文輸入法中三角輸入法的三角字根圖像來作為辨認中文字的工具，因為只要從字元的角落取出三個三角字根圖像就可決定這個字元，而且三角字根圖像是經由專家審慎整理而得的有限字根集合．所以我們採用三角字根圖像來辨認中文字是可行之道．至於對三角字根圖像的比對(matching)，我們採用了 Stallings的圖形編碼方法加上三角字根圖像結構圖的檢驗，得到了很好的結果．

如同所有的圖形辨認的問題一般，中文字元辨認也會受到雜訊(noise)的嚴重影像．針對雜訊或造字方便而造成的鄰近缺失問題(near miss)，我們提出了兩個因應之法．一個是基於樣本學習(learning from example)的模式，將鄰近缺失視為負樣本(negative sample)，由使用者教導辨認機如何去切除鄰近缺失，並用人工智慧中的判斷積分(predicate calculus)來儲存知識，在學習了一段時間後，辨認機終可學到如何去辨認中文字；另一個方法是基於對三角字根抽取方法和Stallings編碼方法的啓發式(heuristic)觀察，得到了一個有效的自動切除鄰近缺失演算法．

為了方便對字根圖像的分析，我們將範圍(scope)訂在對印刷體(printed)中文字的辨認．但若我們能考慮字根的變異情況，我們的方法也可應用於手寫(handwritten)中文字的辨認．

第一章　緒　論

面臨資訊發展一日千里的時代，現代人的我們，免不了要利用更新、更快、更好的工具--電腦來幫助我們計算、文件編輯、檔案處理等．雖然電腦已經能幫我們完成許多工作，但是對中文資訊的處理，仍然有不便之處．現今電腦的發展為了包含中、英文字的同時處理，都是利用鍵盤上的特殊鍵的切換來區分中文和英文資訊的輸入，這些利用鍵盤作輸入工具的中文字輸入法有大鍵盤輸入法、注音輸入法、字根輸入法...等等，不勝枚舉．這些利用電腦基本配備的鍵盤來作中文輸入的方法，雖然對設計者有著不需增加設備的方便，但是它卻有不少潛在的缺點，就是這些方法不僅造成輸入的速度減慢，讓使用者不便，成本增高，而且容易出錯．這些原因造成了中文資訊處理的困難以及辦公室自動化(office automation)的瓶頸．

但是由於專家學者們努力之下，使得影像處理(Image processing)和圖樣辨認(pattern recognition)的技術日益純熟，而對於這些技術，我們可在Duda&Hart[1]、Gonzalez & Wihtz [2]和Rosenfeld & Kak [3]中得到了解．再加上積體電路(integrated chip)科技和硬體(hardware)技術的蓬勃發展，使得我們對中文資訊的輸入問題有了解決之道，就是利用照相機將中文字拍照形成字元的影像，再利用電腦直接來分析影像，根據所得到的影像特性，便可了解輸入的字元為何．如此利用硬體設備加上電腦的良好控制來取代人工輸入，不但使得輸入速度加快、成本降低，更使得輸入的正確性增高，可謂達到了中文資訊處理自動化的目的．這種利

用照相機配合辨認軟體用以辨認輸入字元的方法,就是所為的光學字元辨認(optical character reader)的技術.

自1950年代出期Mark Sheppard發展出第一部自動閱讀機後,專家學者們便不斷地在光學字元辨認的領域上奮鬥,直到今日已有了豐碩的成就,像是對手寫的<1.>阿拉伯數字.<2.>英文字.<3.>Fortran語言的關鍵字(keyword),和<4.>日文的Katakana字元的辨認,都得到了令人滿意的結果,在Suen [4]中可得知這些成果.雖然有了這些成果,但是對於我們所關心的中文字元辨認,卻延遲至1970年代起,由於日本人想解決日文中二仟多個漢字(Kanji)輸入電腦的問題,才開始有了初步的研究.

以中文字為對像來辨認,實在是一個很困難的問題,因為中文字在先天上對辨認有許多不利之處:

<1.> 中文字型繁複.中文字由筆劃數(stroke)為一,如"一"字,到筆劃數超過三十,如"籲",變異極大.中文字構造的變化極大,遠非阿拉伯數字、英文字母...等所能相比,所以往日能應用於阿拉伯數字和英文字母辨認的演算法(algorithm),再也不能適用於中文字元的辨認.

<2.> 中文字數繁多.在日常生活和書報雜誌中的常用字超過5000字;在一般字典中,通常包含了10000-15000字;康熙字典中更多達60000字,所以中文字的字數繁多,難以處理,遠非阿拉伯數字、英文字元所能相比.

<3.> 有許多不同的字元,卻有極相同的字型.如"已"、"己"和"巳","大"、"太"和"犬",...等等.所以我們若要作中文字元的辨認,需要更強而有力的圖形辨認(pattern recognition)技術和更精緻的辨認演算法.

所以中文字元的辨認，並沒有想像中那麼容易.

雖然中文字元的辨認問題這麼棘手，但是專家學者們相信只要我們能夠對輸入的中文字作好大分類(preclassification)的工作，那麼對中文字的辨認問題便可迎刃而解．所謂大分類的方法，就是利用一些不易變異的特性(feature)事先對所有的中文字作分類，如果分類的方法好,再加上分類正確話，那麼每一分類中的字集可能只有十幾個，甚至幾個而已．這麼一來，我們辨認的對象便不在再是數萬個中文字，而是每一分類中的幾個字，這個問題就簡化成和阿拉伯數字辨認以及英文字元辨認的問題相似，我們只要對分類中的字集取細部特徵，就可以很容易地將中文字分辨出來．例如我們可以將部分部首當成大分類的方法，那麼經過大分類後，我們辨認的對象便是同一部首內的字集，如"姊"、"妹"、"好"...為同一類,而"作"、"何"、"信"...為另一類．這樣，我們的問題就簡化成如何去分辨同部首內的字集了，至於如何去分辨它們就是圖形辨認(pattern recogition)的問題了．所以，我們在下一段中介紹了圖形辨認技術應用於字元辨認的方法.

以往專家學者在處理字元辨認的問題時，所使用的方法可分為兩大類，一類是決策理論法(decision-theoretic approach)，另一類則是文句式法(syntatic approach)．決策理論法的作法是訂立了一些特性向量(feature vector)，輸入的影像則對這特性向量作投影，只要這些特性訂得好，那麼要辨認輸入的影像，只要對這特性空間(feature space)作分割(partition)即可解決，所以決策理論又稱為區分法

(discriminant approach). <圖 1.1>列出了一個典型的決策理論法辨認字元的方塊圖.

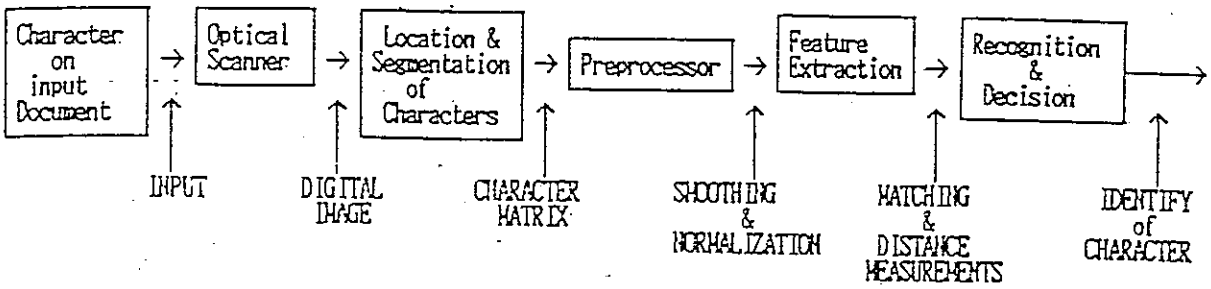
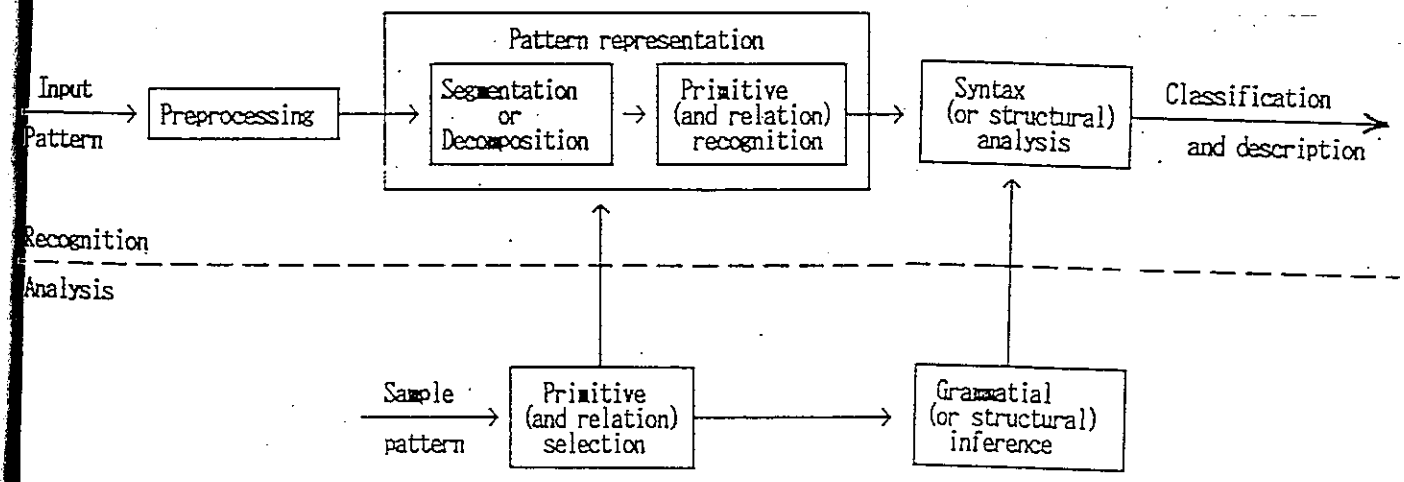


圖 1.1

在<圖 1.1>中我們可以清楚地看到一個標準決策理論法應用於OCR的系統流程. 從一個字元經由照相機(或scanner)的拍攝成為數位影像，再經由切割(segmentation)的處理，將多階灰度值(multi-gray levels)的數位影像變成二元字元矩陣(binary character matrix)後，再由前處理器(pre-processor)將之平滑化和正規化(smoothing&nomalization)以除去某些程度的隨機雜訊(random noise)，然後再依據訂立的特性向量(feature vector)使這個影像對這特性向量投影 (projection) ，最後把這投影結果和標準的特性空間(feature vector)作相似性的比較，便可比對出輸入的字元為何.

由於決策理論法的應用較為簡便，所以近二十年來，圖形辨認(pattern recognition)的研究大多集中在決策理論法的探討和應用．但是有些圖形辨認的問題，如影像分析(scene analysis)，極富結構化的資訊，適合以結構上訊息來分辨影像的類別；而且有些圖形辨認的問題極其繁複，若使用決策理論法則需要極多的特性(feature)，反而使問題繁雜而不易處理．這些問題導致了文句式法的興起，就是先從影像中取出了一些基本元(primitive)或副圖樣(sub-pattern)，然後對圖樣的辨認就是使用這些基本元或副圖樣加以結構上的組合(composition)，那麼辨認的問題就變成了利用文法(grammar)對圖形類別的描述(description)，而不只是簡單的分類而已．

<圖 1.2>列出了文句式圖形辨認法(syntatic pattern recognition)應用於字元辨認的系統方塊圖．



<圖 1.2 >

在系統開始執行辨認之前，必須先執行<圖 1.2 >下半部的分析工作(亦可稱為學習的過程)，先輸入適當的樣本影像(sample image)，根據樣本影像來取出適當且代表性的基本元，並分析出基本元間的結構關係，再依選出的基本元和結構關係我們就可推論出描述(describe)每一不同字元的文法(grammatical inference)，而對於文法我們自然可以建立與之相對應的自動機(automaton)，利用這自動機我們可以辨認(recognize)對字元結構的描述．這樣經過了分析的程序後就可以進入辨認的過程了，如<圖 1.2 >虛線上方所示：輸入的影像經由前處理(preprocessing)後，便進入了圖形表式(pattern representation)的程序中，而圖形表式程序的功能是在根據分析過程產生的基本元和結構關係，對輸入的影像切割(segmentation)出基本元並標明出基本元間的結構關係，再利用分析過程所建立的自動機來辨認輸入的字元，這樣不僅達成了辨認的目的，更可明瞭字元間的結構狀況(classification and description)．

中文字辨認是圖形辨認的領域之一，近年來專家學者在中文字辨認(包含手寫和印刷體)的工作，日益受到重視，尤其是應用決策理論法於中文字辨認上，已獲得了大量的成果[5,6,7,8,9]．以決策理論法來辨認中文字，其成敗乃在於所訂立的特性好不好，一組較佳的特性(feature)可以大大地提高辨認率(recognition ratio)，反之一組不好的特性卻可能造成辨認錯誤．

這麼看來，要是我們能夠發明一個方法和或演算法來引導我們去找到最佳的一組特性的話，中文辨認的問題便可以解決了．但可惜的是並沒有人能夠發明這個方法，也就是說：我們對特性抽取(feature extraction)並沒有原則可循

，只可依直覺的(intuitive)和啓發式的(heuristic)方法來取得．

Projection profiles	Fourier transform
Ratio of overall stroke lengths in horizontal and vertical directions to their respective spread values	Hadamar transform, rapid transform, Walsh transform
Belt patterns and density of points in four strips surrending the character	Edges and lines segment, center line of strokes of characters, contours of four corner parts of character
Distribution of points in an 8 × 8 mesh of the character	

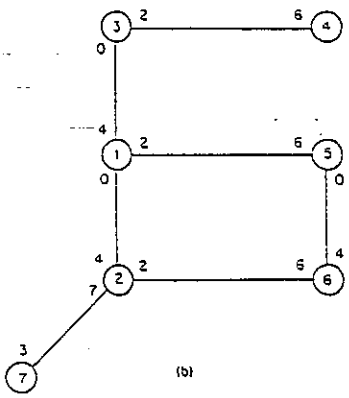
<圖 1.3>

Orthogonal expansion	K-L expanison
Stroke distribution	Line direction, crossing counts, stroke length matrix, direction continuity density
Stroke analysis	Stroke extraction, polygonal approximation
Background feature distribution	Modified Glucksman's stroke density, Glucksman's line stroke density
Background analysis	Edge propagation, line direction propagation
Combination	Crossing count and projection, stroke density and line direction propagation

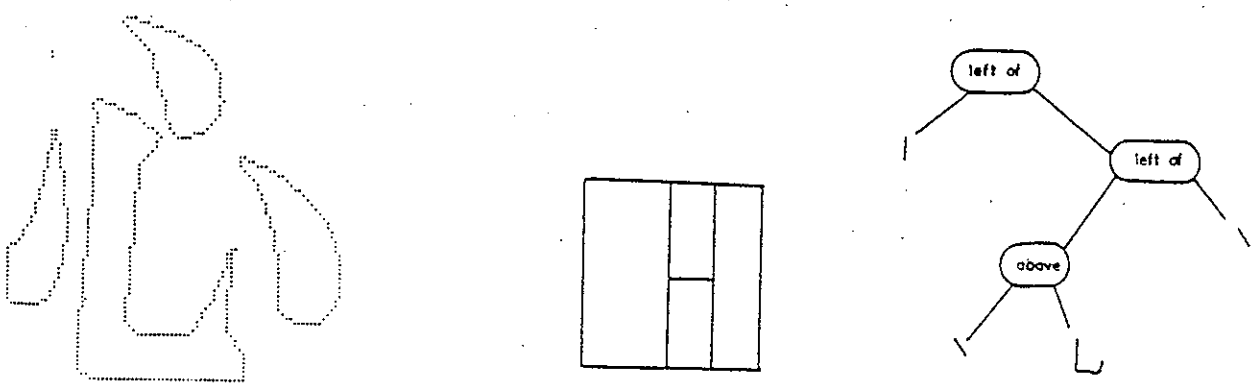
<圖1.4>

特性的抽取既然是這麼重要，專家學者們對特性抽取的研討自然是不遺餘力．<圖 1.3> 中列出了對印刷體中文字(printed Chinese character)辨認經常被採用的特性，而<圖 1.4> 中列出了手寫中文字(handwritten Chinese character)辨認經常被採用的特性．而這些特性的實際應用可在Mori[5,6,8] 和Suen [9] 論文得到瞭解．雖然我們無法得到最佳的特性(feature)，參考專家學者們所使用的特性，卻可引導我們得到一組不錯的特性．

雖然決策理論法應用於中文字辨認是如此的風行，但是我們亦不可忽視了文句式法(syntatic approach)對中文字辨認的貢獻[7,9,10]．中文字由基本筆劃加以結構上的組合，形成了部首和偏旁，而部首和偏旁再加加上結構上的組合才構成中文字．所以中文字極適合以部首、字根或偏旁當成基本元(primitive)，並對基本元之間的結構關係加以剖析(parsing)後，就如同上面所說的文句式法(syntatic approach)來辨認中文字．例如Stallings [11]就是針對印刷體(printed)中文字，以每一最多筆劃結合在一起而形成的基本單元(component)為基本元(primitive)，先透過一個基本單元分析程式將每一基本單元轉換成圖形表示法，並利用八方向碼(octal code)標明節點間探索(trace)的方向，如<圖 1.5>中所示，再利用一個副圖樣辨認的演算法對基本單元編碼，然後再經由幾何上關係(1)左-右(2)上-下(3)內-外(4)前三者的組合，用樹狀表式法(tree representation)如<圖1.6>來描述基本單元的結構．這樣子便完成了對印刷體中文字的辨認．



<圖 1.5>



<圖1.6>

至於對Stallings的基本單元分析程式，我們將會提出更穩定的方法來替代，而它的編碼方法我們將在後面章節中再詳細介紹.

中文字元的辨認一直被認為是圖形辨認(pattern recognition)領域上的難題之一，雖然專家學者們不斷地努力對這個問題加以研究和探討，但是所獲得的成就卻不是非常令人滿意. 這是因為除了前面所提及的中文字先天上不利於辨認的處理外，專家學者們所使用的方法，也值得商榷：

(1) 大分類法(preclassification)雖然被認為是中文字辨認的利器，但是若分類所使用的特性(feature)不穩定或由於影受到扭曲(distortion)和雜訊(noise)影響，將會造成分類錯誤，而無法辨認

(2) 大分類的方法常會造成分類不均，也就是說有些分類中的字集很大，那麼對這些分類的字來作辨認，亦是一個讓人頭痛的問題.

(3) 就如前面所提到的特性抽取(feature extraction)的方法沒有一定的法則，都是靠自覺的和啓發式的方法來得到，難免會失之武斷. 而且所抽取的特性，也常會受到影像雜訊的影響而變異，造成辨認的困難.

(4) 以往文句式法廣為人評擊的是無法包容影像的扭曲和雜訊. 但是自從轉換式文法(transfomational grammar)、猜測型語言(stochastic language)和錯誤修正剖析法(error-correcting parsing)的推展[10]後，使得文句式法有了一線生機. 但是這些

工具較難學習和理解，而且它們效率也被人懷疑，所以文句式法的推廣尚待學者們的努力．

(5) 為了消除雜訊所使用的前處理(preprocessing)的成效常常沒有預期中那麼理想．

所以為了解決中文辨認的問題，大分類法的技術、決策理論法和文句式法的發展以及用來消除影像扭曲和雜訊的影像處理方法，都還需要專家學者們不斷地努力以尋求突破．

在本篇文中我們探討了三個重點，第一個重點是對自動推論(automatic inference)中文字根圖像文法的研究．第二個重點是提出以三角字根圖像來當成辨認中文字的介面 (interface)．第三個重點是針對造字的方便或雜訊影響造成的鄰近缺失(near miss)，提出了兩個解決方法，一個方法是應用人工智慧 (artifical intelligence) 中樣本學習(learning from example)模式，另一個方法是觀察三角字根特性所得的自動切割(automatic segmentation)演算法(algorithm)．

我們人類以各種語言來溝通，但我們若想和電腦溝通卻必須以正則語言 (formal language)才可被電腦所了解．所以我們如果能夠以規劃過的字根圖像和結構關係來讓電腦自動推導出它之間的文法生成原則(production rules)的話，就可說是讓電腦自動去瞭解輸入的中文字，這也就是我們探討自動推論字根圖像的目的．

若想用電腦自動推論字根圖像的文法，必須先將所有中文字的字根圖像完全整理出來，但這實在是一個蠻艱巨的工作，於是我們會問，是否可由部份字根就可明瞭輸入的中文

字為何呢？我們觀察了中文輸入法中的三角輸入法，發現三角輸入法的取碼原則是在字的角落部份取出最大的三個三角字根，而每個三角字根都有其字碼，這三個取出的字根的字碼，就是這個字的三角內碼．所以我們只要能控制電腦在輸入字元的影像中取出正確的三角字根，便可了解輸入的字元為何．而且三角字根的數目有限，我們便不需要再去分析所有中文字的字根圖像了，所以我們提出了以三角字根作為辨認中文字所用的字根圖像的想法．

雖然利用三角字根來辨認中文字是可行之道，但是有些三角字根卻不獨立存在，也就是說它包含在某些不是三角字根的字根圖像之中，再加上雜訊的影響會造成影像品質不佳，甚至使原本不相連的字根連接在一起，使電腦無法瞭解這個雜訊造成的新字根為何，我們將這些無法直接處理的問題稱之為鄰近缺失(near miss)．我們當然可以利用前處理的技術濾去部分的雜訊干擾，但是對於不能被前處理濾去的鄰近缺失，仍是我們必須處理的問題．

針對這個問題，我們有兩個解決之道．一個是基於樣本學習(learning from example)模式的方法，將影像的鄰近缺失看成學習的對象，利用人工智慧中的判斷積分(predicate calculus)作為儲存知識的工具，只要我們能建立一個辨認機，而它包含了足夠多的知識去解決鄰近缺失的話，那麼這個辨認機就可用來解決中文字辨認的問題了．

另一個解決鄰近缺失的方法是我們根據三角字根取碼原則和Stallings對獨立字根的編碼特性所得的啓發式(heuristic)觀察，因而建立的一個能自動切割出三角字根的演算

法．利用這個演算法，我們不但能從字根圖像中切割出正確的三角字根．而且可以解決字根受雜訊影響而誤連的情形．

針對印刷體中文字的辨認，經過了這些探討後，我們確實地明瞭了這個問題的困難所在，並提出了我們的看法．

在下面的章節中，我們對上面所說的論點作了深入的探討：

第二章中我們介紹了對中文字元辨認的影像處理方式，並陳述了我們對影像所使用的前處理和特性抽取（feature extraction)方法．

第三章中我們對中文字的字根圖像的視覺結構加以討論．並且介紹了描述這結構所用的文法和辨識（recognize)三角字根結構的自動機(automaton)．

第四章中我們討論如何自動推論出字根圖像結構(iconic radical)的文法(grammar)．

第五章中我們探討了以學習模式來解決鄰近缺失的方法．

第六章中我們介紹了解決鄰近缺失的啓發式演算法（heuristic algorithm)．

在最後一章中，我們除了比較學習模式和啓發式演算法來解決鄰近缺失的優劣，並且提出了我們的感觸和展望．

第二章 影像之處理

2.1 影像的衰減和影像處理

當我們利用照像機(或scanner)拍攝得到影像(image)，而得到的影像並不會和原來的輸入相同，也就是說影像有衰減的現象(image degradation). Gonzalez[3]中提到了一個影像衰減的模式，<圖2.1 >就是這個模式的方塊圖. 他解釋影像衰減的過程是經由系統因素(system factor)的影響後，再加上隨機產生的雜訊(random noise)而生成. 系統因素是一個常數，而它因機器設備不同而不同，雜訊值的大小乃隨機而生，我們對的了解僅可限於統計上的猜測.

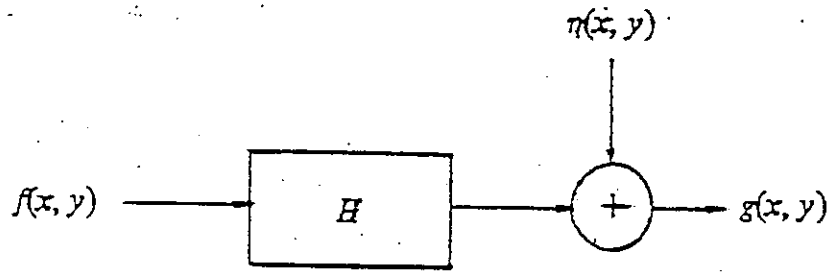
假設系統因素是常數運算元(operator)--H，而隨機雜訊為函數n(x,y)，輸入的影像若為f(x,y)，所得到的影像為g(x,y)，那麼它們之間的關係式就如同下面的式子：

$$g(x,y)=H\ f(x,y)+n(x,y)$$

就是因為如此，我們若想利用數位照像機(或scanner)來拍攝影像並利用電腦來分析它的話，首先面對的問題就是影像中隨機雜訊的現象，再加上我們拍攝時環境不佳、疏忽、設備不足...等因素影響之下，更會使影像扭曲(distortion)和雜訊的情形更加嚴重. 針對這些問題，專家學者們發展了許多方法來解決，如應用各種過濾器(filter)來強化(enhance)或平滑化(smooth)影像效果的影像增揚(image enhancement)方法；為了去除影像衰減的作用，應用最小方差(least square error)的論點，發展出各種影像重建(image restoration)的方法；影像增揚和影像重建方法所使用的工具--影像轉換(image transform). 包含了Fourier轉換、Walsh轉換、Hadamard轉換...；為了達到(1).影像資料

壓縮(image data compression). (2).影像傳輸(image transmission). (3).特性抽取(feature extraction). 目的的影像編碼方法(image encoding). ; ...[1,2,3].

而研討和應用這些方法的學問，就是所謂的影像處理 (image processing).



<圖 2.1>影像衰減的模式

2.2 影像的前處理

影像處理的重點在於得到一個品質較佳的影像，而圖形辨認(pattern recognition)則著重在辨別影像中的字元、物體...等，所以圖形辨認除了要應用影像處理的方法來得到品質較佳的影像外，還需要有一些處理技巧使得辨認能順利執行．而這些處理是在辨認之前所作的工作，所以我們稱之為前處理(preprocessing).

專家學者在處理中文字元辨認所使用的前處理步驟，以消除雜訊和方便後來的分析及處理：

(1).訂立極值(thresholding)：

數位照像機拍攝所得到的影像大多是多階的(multi-level)，訂立極值的目的就是為了得到二元影像(binary image)，使得影像的資料簡化，以方便後來的處理．數位照像機依其精密度的不同，可感受光的亮度並依光的亮度在影像上切割出或多或少的明暗度，我們稱之灰度(gray level)．影像中愈暗的地方灰度值愈大，反之影像中愈亮的地方則灰度值愈小．假如我們的數位照像機以三個位元的精密度來表式灰度值，那麼影像的解析度(resolution)最小單位--像元(pixel)的灰度值變化就由0到7來表式，最亮的地方是0而最暗的地方是7.

若是我們利用多階灰度值的影像資料來辨認中文字的話，那麼就會因為資料太大而難以處理．我們便可以取適當的灰度值當作極值(threshold)，影像中每一像元的灰度值都與這個極值作比較，若大於或等於這個極

值的話就將此像元的灰度值改成 1，反之就將灰度值改成0，這樣子我們就得到了一個灰度值不是0就是 1的二元影像(binary image)，而簡化了往後的分析.

(2).正規化(normalizing):

數位照像機拍入的影像常會大小不一，這樣子不但會使分析影像的演算法難以應用，而且會使辨認率降低，Mori[5]中告訴了我們這個現象.

為了解決這個問題，我們可用簡單的幾何轉換，將影像正規化成大小一致，以方便處理.

(3).細線化(thinning):

中文字元辨認最重要的是要知道筆畫的類別、多寡和結構，並非筆畫的粗細，而且粗的筆畫不利於筆畫的判定和結構分析，所以研究中文辨認的專家大多偏好於將影像細線化，得到筆畫的骨架(skeleton)之後再來作辨認. 細線化後的影像不但容易分析而且特別利於特性抽取 (feature extraction)並可使辨認的成效提高，所以許多專家學者們對細線化演算法(thinning algorithm)的發展不斷地在努力[12,13,19].

細線化演算法的作法大多是利用3×3的視窗(window)來檢視影像上的每一點，看它是否位於筆畫的最外側，如果是的話就把它消除，如此逐次(iteration)地將筆畫的最外層剝去，最後就得到了筆畫的骨架(skeleton)，CHEN&HSU [19]就是標準的例子. 而所謂的3×3 視窗就是每一像元和最鄰近它的8 個像元所組合而成的矩形，考慮視窗上的灰度值的變化，就可決定這個像元是否位於筆畫的最外側.

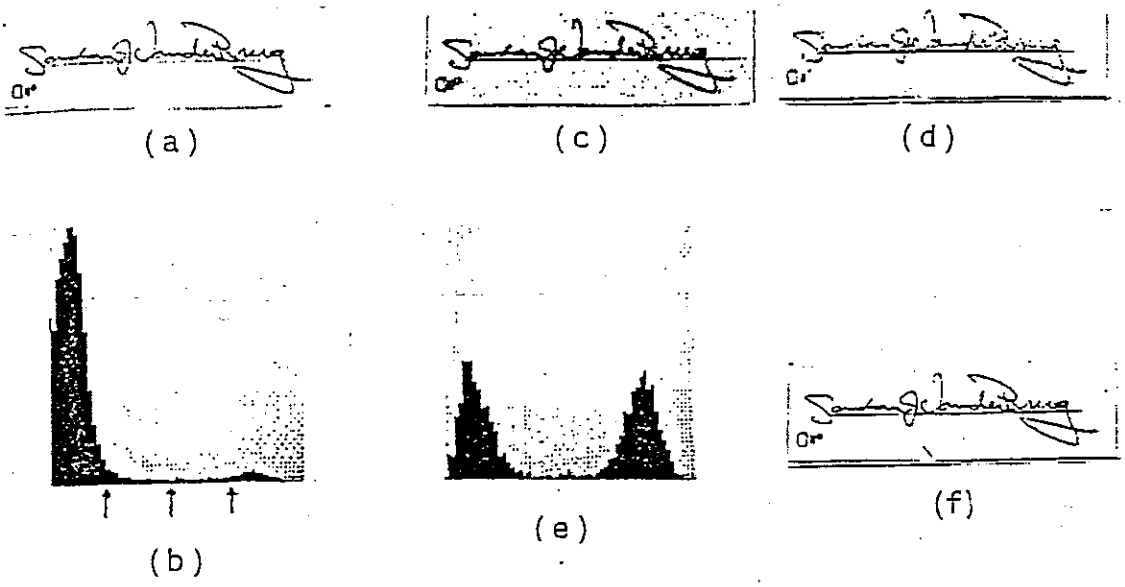
(4).平滑化(smoothing):

由於雜訊的影響，加上訂立極值和細線化的過程不當，會造成影像有缺洞(holes)、突腫(bumps)和骨架扭曲(skeleton distortion)的情形產生. 為了不影響細線化的效果以及特性抽取，我們必須對影像作平滑化的工作，就是把缺洞填平(fill)，把突腫消去(delete)並把筆畫的骨架調正(adjust). 至於我們如何知道影像中某一像元是缺洞、突腫呢？和細線化過程一樣我們可以用3×3視窗來檢視出來.

在本篇論文中，我們對影像的前處理步驟是：

(1).利用灰度值的投射圖 (histogram)來決定極值是最常使用的方法，但是這種投射圖常常是混淆而不易決定極值，如<圖2.2>(a)是一個手寫的簽名而<圖2.2>(b)是它的灰度值投射圖，其中三個箭頭所指的區間內都可能是極值所在，我們若因而取了太小的極值，那麼所得的影像就會像<圖2.2>(c)一般而包含了太多雜點，反之就會得到如<圖2.2>(d)一般失去了不少應有的點. 為了解決這個問題，我們可以先將影像經由 Laplacian運算，也就是將每一像元(pixel) 的灰度值減去它最鄰近的 8個像元灰度值平均值再取絕對值，然後我們可取 Laplacian運算後超過變化度 百分之九十以上的像元作投射圖 ，這樣就可得到如<圖2.2>(e)一般的雙態(bimodal)投射圖，我們只要選擇雙態中山谷(valley)最低部份當成極值，那麼就可得到如<圖2.2>(f)一般較佳的二元影像.

(2).針對細線化和平滑化的問題，我們結合了CHU&SUEN [13] 的交替式平滑化和剝除的演算法(alternate smoothing & stripping algorithm)和CHEN & HSU [19] 的平行細線化演算法(parallel thinning algorithm)來解決它.



<圖 2.2>

CHEN&HSU[19]的方法可分為兩個小次序(subiteration)，第一個小次序是把符合下列條件像元(pixel)的灰度值清除：

(a) $2 \leqq B(p1) \leqq 7$.

(b) A(p1)=1 ; if A(p1)=2 then goto (d).

(c) (p4=0) or (p6=0) or (p2=P8=0) ; end.

(d) (p2=p4=1 and p6=p7=p8=0) or (p4=p6=1 and p8=p9=p2=0) ; end.

而第二個小次序是把符合下列條件的像元的灰度值清除：

(a') $2 \leq B(p1) \leq 7$.

(b') A(p1)=1; if A(p1)=2 then goto (d').

(c') (p2=0) or (p8=0) or (p4=p6=0); end.

(d') (p2=p8=1 and p4=p5=p6=0) or (p6=p8=1 and p2=p3=p4=0); end.

其中像元p1,...,p8 如<圖2.3>(a)位於一個3×3視窗上；而B(p1)表示對像元p1 的鄰居p2,...,p8灰度值不是0的個數；A(p1)則代表像元p1的鄰居以順序檢查它們接連出現0-1樣本(pattern)的個數．如<圖2.3>(b)中的3×3視窗中A(p1)=2,B(p1)=5,但是對第一個次序而言，卻不合(d)的條件，所以p1像元不可以清除．

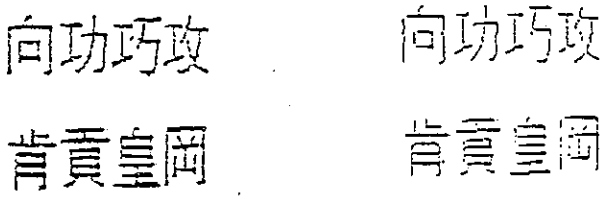
經由這兩個小次序交替著處理，最後就可以把影像細線化了，<圖2.3>(c)就是經由CHEN&HSU法細線化的結果．

P9	P2	P3
P8	P1	P4
P7	P6	P5

(a)

0	0	1
1	P1	1
1	1	0

(b)

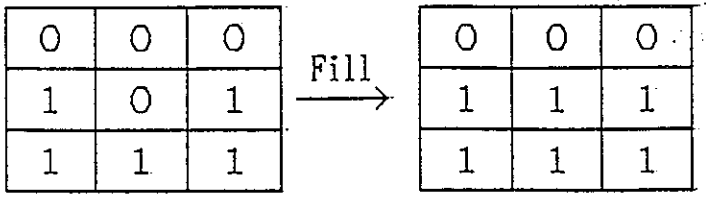


(a) Original picture (b) thinned picture

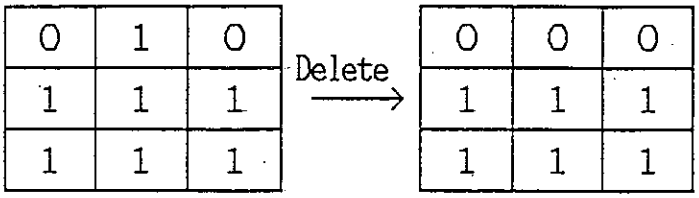
(c)

<圖2.3>

CHEN&HSU的方法效果會受缺洞(holes)和突腫(bumps)的影響造成不平均的細線化，為了改善這種現象，我們應用了CHU&SUEN[13]的平滑化 (smoothing)來消除缺洞和突腫的情形. CHU&SUEN利用了一組3×3視窗來檢視缺洞和突腫，例如<圖2.4>(a)中就是一個缺洞的情形，必須將它補平，而<圖2.4>(b)中就是一個突腫的情形，必須把它消除.



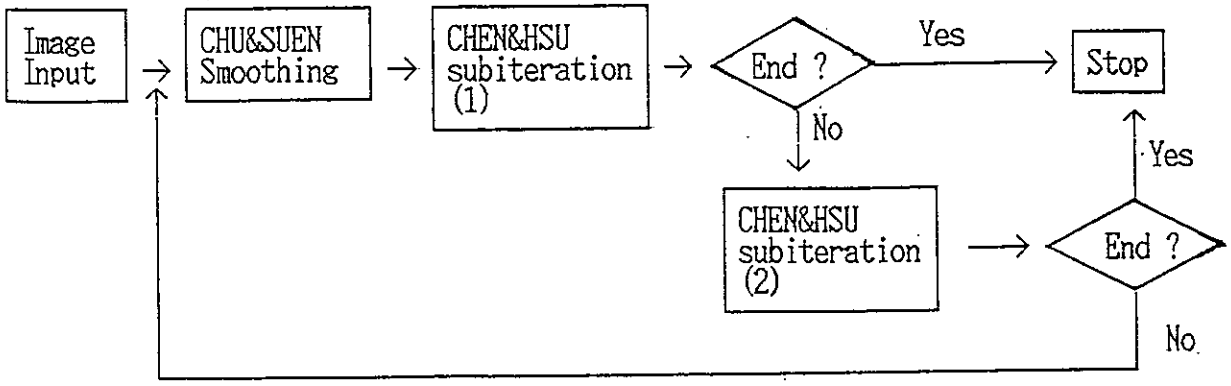
(a)



(b)

<圖2.4>

我們把CHU&SUEN的平滑化過程放在CHEN&HSU細線化過程之前，也就是說每當CHEN&HSU法的兩個小次序執行前，先利用CHU&SUEN法來消除不合理現象，這樣子所得到的細線化結果會比較好. 這樣子修正後的流程圖就如<圖2.5>中所示.



<圖2.5>

至於筆畫調正(skeleton adjust)的過程，我們在下一節的筆畫抽取(stroke extraction)中自然會解決.

(3).對於正規化的問題，我們採用了affine轉換把所有的影像正規化，成為解析度30×30的影像. 若原來的影像解析度為m×n，則正規化成解析度30×30的公式為：

$$F(x,y)=(29*(x-1)/(n-1)+1,29*(y-1/(m-1)+1)$$

我們不用擔心正規化的結果是否會影響特性的抽取，因為Grunbaum[20]告訴我們，在affine轉換後基本元(primitive)的特性是不會消失的.

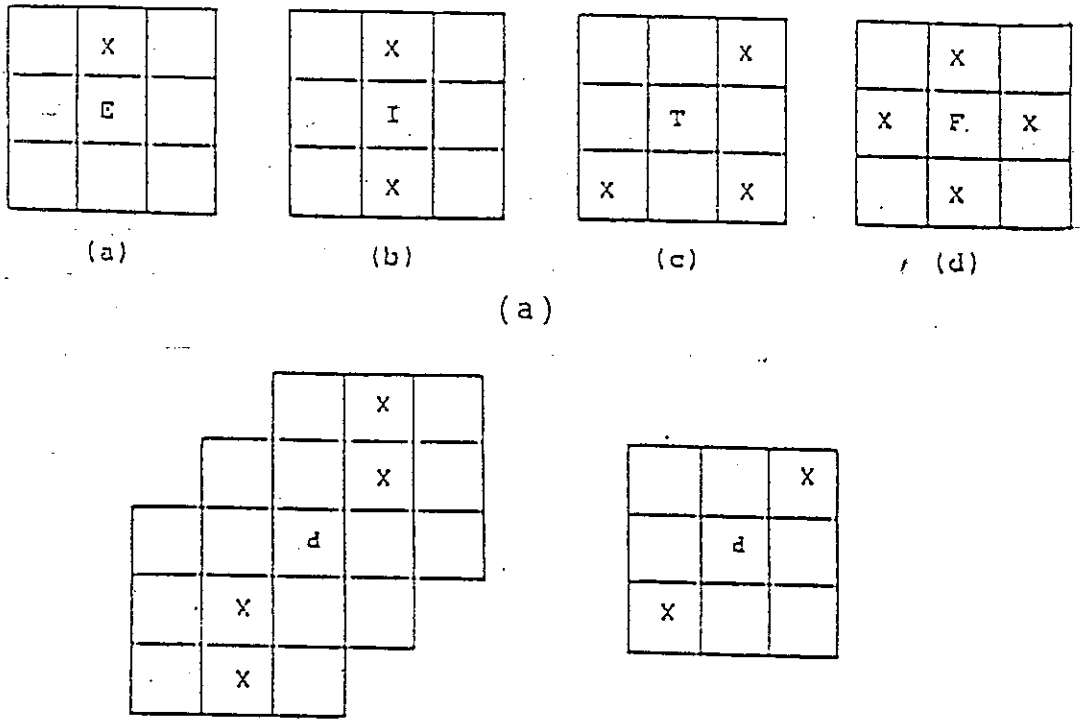
2.3 筆畫抽取(stroke extraction)

在前面的處理後，我們已得到了一個細線化後的二元影像(binary image)，接下來的工作就是特性抽取(feature extraction)的過程. 在本篇論文中我們所需的特性是基本筆畫(elementary stroke)，因為基本筆畫是中文字極重要的一個特性，正確地將基本筆畫從影像中取出，對中文字的辨認可說已成功了一大半.

基本筆畫是線段的組合，而基本筆畫間可能會相交而將筆畫互相截斷. 所以我們抽取基本筆畫的想法就是，從影像中偵測出筆畫(線段)的終點和交點，而相連線段是由這些特徵點的相連性所決定，然後真正的基本筆畫可由這些線段的組合而成.

HSU&CHENG[14]就是利用這個觀念，建立了一組3×3視窗(window)來檢視終點(end point)、角點(corner point)、三叉點(three fork point)和四叉點(four fork point)，<圖2.6>(a)列出了一組特徵點，利用這組視窗可以檢視出相對的特徵點.

除此之外，HSU&CHENG還提出了以擴展的3×3視窗(extended 3×3 window)來彌補解析度的不足，用來檢視角點，以得到較正確的結果. <圖2.6>(b)列出了一個3×3視窗和一個推展的2階(2-step) 3×3視窗.



<圖 2.6>

特徵點抽取後，就要截取出筆畫的部份線段(line segment)，我們只要從每一終點開使，以鍊結碼(chain_code)來探索(trace)線段，直到遇到另一特徵點止，就得到了一筆畫線段.

在得到了所有的筆畫線段後，我們發現這些筆畫線段並不一定是基本筆畫而是基本筆畫的部份集合，為了得到正確的基本筆畫，我們必須要有一合併(merging)的程序，而合併的程序執行時我們必須考慮筆畫線段間的方向角，筆畫長度和距離，就可將基本筆畫合併出來，最後再把不合理和長度太小的筆畫線段(雜訊)消去，就完成了對基本筆畫的抽取，於是我們得到了一組特性向量(feature vector)，包含了每一個基本筆畫的起點、終點、方向和長度，運用這些資料，我們可以很容易地得到每一最大相連的字根圖像和字根圖像的終點和交點.

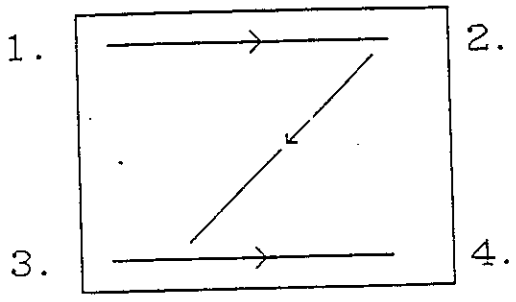
而經過了這些前處理過程，我們確信可以正確地得到每一最大相連的字根圖像和影像中所有的基本筆畫和筆畫的終點、交點和方向，而這些就是我們辨認所需的特性.

2.4 三角碼

如前章中所言，我們提議以三角字根作辨認中文字的工具，而對三角字根取碼的原則，我們在此先作一個簡介。

中文字富含結構上的資訊，在字的四個角上均有字根圖像的特性而只要指出它角落上的部份字根，我們就可以分辨出這個字是甚麼。胡立人[15]等三人細心地整理了這種特性，把中文字四角上的字根圖像加以系統化的整理，分析出300個字根(又稱基本符號)，而依照字根的形態區分成99組<附錄>，每組各有相對的字碼序。他們利用了這種字根特性，在幾何結構的角落處取出三個最大的三角字根來表示這個字，這種對中文字編碼的方法就稱之--三角碼。

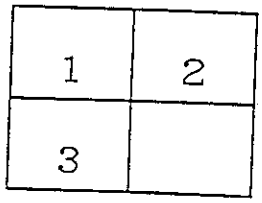
三角碼取碼的原則是由上到下、由左到右，以<圖2.7>(a)為例，取碼的順序是由1到4循環，直到取出三個三角字根為止，如<圖2.7>(b)中"矮"字的三角內碼是(87,29,42)，而"麗"字的三角內碼是(10,10,02)。



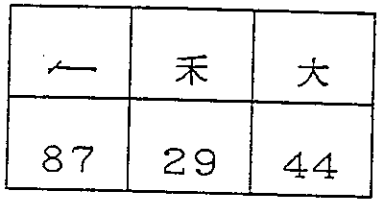
(a)

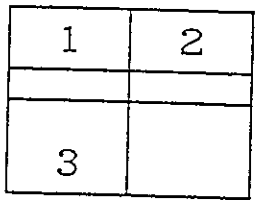
續下頁

<圖2.7>

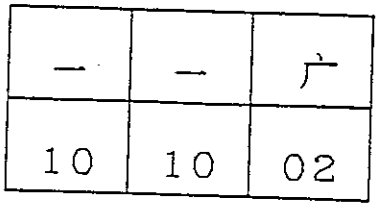


矮





麗



(b)

以三角碼來編中文字內碼有下列優點：

(1).規則簡明，適用於中文字的構造，能見字識碼，容易且正確.

(2).用數字編號，中文有字序適合資料處理.

(3).可對所有中文字加以編號，而且在常用字中極少，幾可一字一碼.

因此三角碼被當作中文資料的輸入法，也就是因為這些優點，讓我們想到了要利用三角字根來幫助我們作中文辨認，我們只要能在影像中正確地取得正確的三角字根，就可辨認出中文字來.

三角碼取碼規則簡明，所以我們可以很容易地寫出一個有效的演算法來引導我們的辨認系統，正確地取出三角字根.即使取出的字碼有少數的混淆現象(ambiguity)，我們也可以很容易地取出一些群體特性(global feature)來將這混淆現象解決.

下面就是一個SPARKS型式的演算法，輸入現在所取得的字根所佔用之角(current-corner)，輸出下一個三角字根所應抽取的角 (next-corner) ，其中變數 current-corner和next-corner都是以4位元(bit)來表示：

```
Algorithm NEXT-EXTRACT-CORNER(current-corner,
                              next-corner)
  case
    :current-corner="1000":next-corner ← "0100"
     /現在字根用去第1角，下一字根將從第2角抽取/
    :current-corner="0100":next-corner ← "0010"
     /現在字根用去第2角，下一字根將從第3角抽取/
    :current-corner="0010":next-corner ← "0001"
     /現在字根用去第3角，下一字根將從第4角抽取/
    :current-corner="0001":next-corner ← "1000"
     /現在字根用去第4角，下一字根將從第1角抽取/
    :current-corner="1100":next-corner ← "0010"
    /現在字根用去第1,2角，下一字根將從第3角抽取/
    :current-corner="1010":next-corner ← "0100"
    /現在字根用去第1,3角，下一字根將從第2角抽取/
    :current-corner="1110":next-corner ← "0001"
   /現在字根用去第1,2,3角，下一字根將從第4角抽取/
  end
end NEXT-EXTRACT-CORNER
```

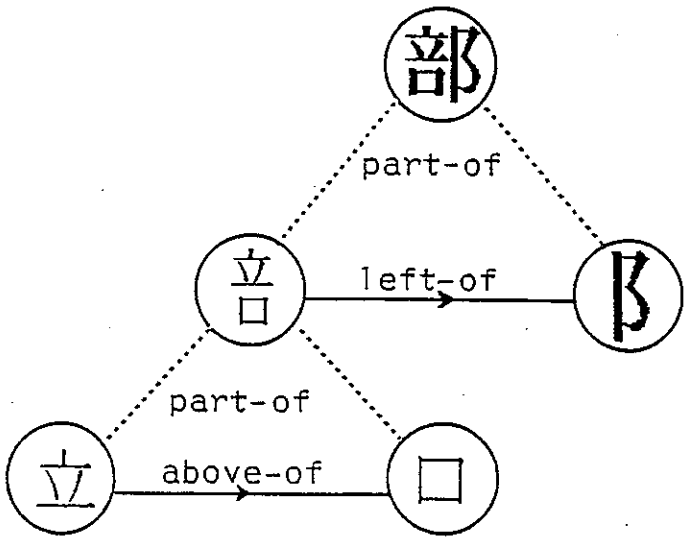
現在我們剩下的問題是如何將三角字根圖像的知識建立，也就是說在影像中抽取出的特性如何正確地比對(matching)出正確的字根？還有就是如何去解決鄰近缺失 (near miss)的問題？

這些問題，我們將在下列章節逐一討論.

第三章 字根圖像之視覺結構

3.1 一般字根圖像之視覺結構

中文字是一個或多個獨立的字根圖像(iconic radical)加上結構的組合而成，我們這裡所提及的"字根圖像"是指在字元中由相連的基本筆畫所組成最大的獨立部分，例<圖3.1>中"部"字就是由"立"、"口"和 "阝"三個字根圖像所組合而成的，而它們之間的結構關係就是 "立"在 "口"之上組成了"音"，而"音"在"阝"的右邊而組成了"部"字，我們經由這三個字根圖像和它們之間的結構關係，就可唯一地決定這個字



<圖3.1> 結構關係圖

現在我們換用資料結構(data structure)中序表示式(infix expression) 來描述中文字的結構關係. 首先，我們定義了5個字根圖像間的結構關係：

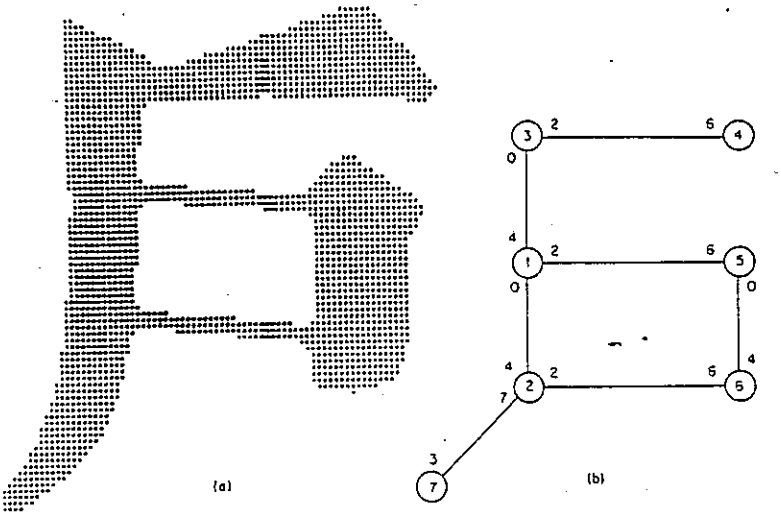
(1)Ri→Rj :字根圖像Ri在字根圖像Rj的左邊.

(2)Ri | Rj :字根圖像Ri在字根圖像Rj的上方.

(3)Ri※ Rj :字根圖像Ri包含字根圖像Rj.

3.2 字根圖像的編碼

字根圖像是由基本筆畫組合而成，它的形狀就如同資料結構中所討論的圖形 (graph)般，我們若將字根圖像的終點和交點當成節點(node)， 而節點間的筆畫線段當成邊線(edge)，那麼字根圖像的辨別就變成了不同圖形的區分，如<圖3.3>就是"戶"字和它的圖形表示法.



<圖 3.3>

利用這個觀點，Stallings [11]將字根圖像依上述原則轉換成圖形，再依據八方向碼(octal direction code)將圖中每相鄰節點間的邊線方向加以標明，如 <圖3.4>，而對字根圖像的編碼，就是依照下列原則來探索圖形(trace graph):

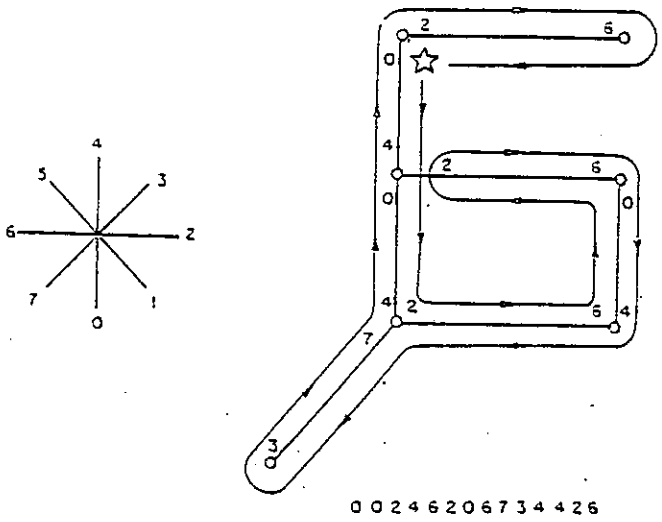
(1).從圖形中(字根圖像)最上方的節點開始向其它節探索 (trace)，每探索到一新節點就把所經過邊線的八方向碼記錄下來，每一邊線只可被來回探索一遍，最後探索回開始的節點，並將所有的邊線探索過，而所記錄的八方向碼集就是這個字根圖像的編碼.

(2).探索的優先順序是向著八方碼較小的節點來探索，並將探訪(visit)過的節點記錄下來.

(3).在每一節點向外探索時，先檢查是否有還未探索過的節點，若有的話，先向這未探索過的節點探索.

這種編碼原則是根據Ore[16]的圖像探索(graph trace)的程序而得來，如此編碼除了極少數的字根圖像因字型相似，只有差別在筆畫長短不一的情形，才會有混淆現象(ambiguity). 如"土"和"士"以及"未"和"末"等，其餘大多數字根圖像的編碼都可唯一決定.

<圖3.4>中列出了八方向碼以及"戶"字根圖像相對映的圖形和它編碼的結果.



<圖3.4>

Stallings [11]應用了這種編碼方式來比對字根圖像，並如上節一般來分析字根圖像的結構關係，對印刷體中文字辨認得到了95%的辨認率(recognition ratio).

我們分析了他的方法，發現了有下列缺失：

(1).無法處理鄰近缺失 (near miss)的問題，也就因為造字的方便或雜訊的影響，使得原本應分開的字根圖像連接在一起，以致使編出來的碼無效，無法比對 (match)出正確的字根圖像.

(2).字根圖像沒有規劃，可能會因字根圖像太多而造成對困難和浪廢記憶體.

(3).以邊界探索 (contour tracing)的方法作前處理，不易得到正確的特性.

所以，我們提出了以第二章中所言的前處理方法來取代他的前處理方法，以得到較正確的特性. 我們並且提出了以三角字根圖像來解決字根圖像沒有規劃的問. 下一節中，我們將介紹三角字根圖像的特性.

3.3 三角字根圖像的視覺結構.

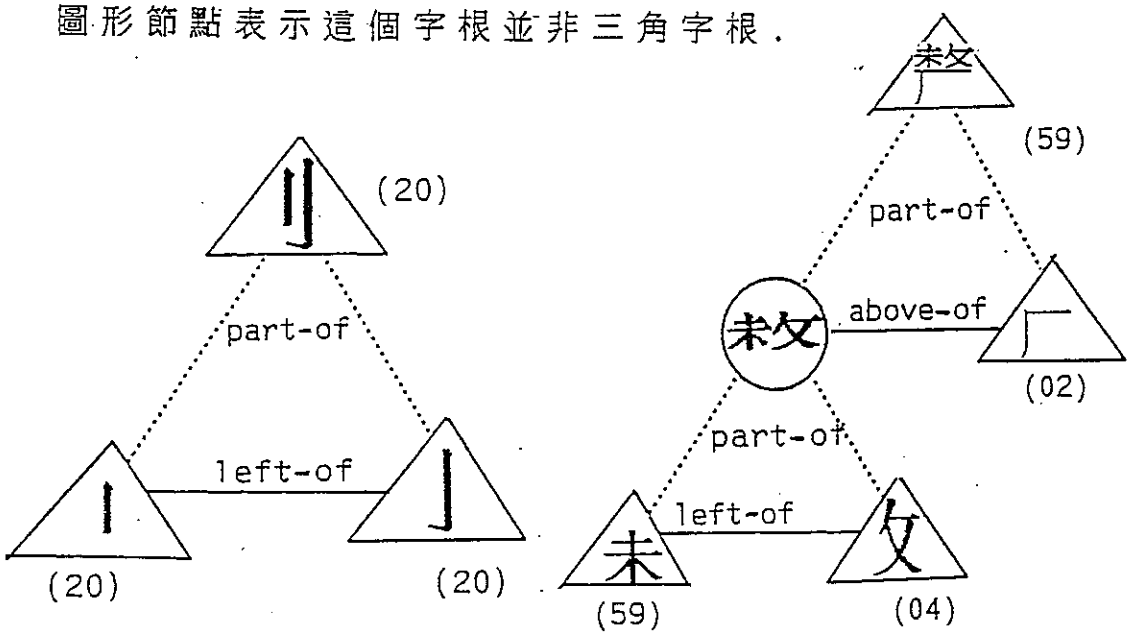
如第二章中所提到的，三角碼是胡立人等[15]為解決中文字輸入電腦的問題所提出的方法．對於一個中文字，我們只要依照他們的取碼原則在字的角落(corner)取出三角字根圖像，也就決定了這個字的內碼(三角碼)，依著他們的方法，我們所得到的結果幾乎可達一字一碼．所以，我們若能在影像中正確地抽取出正確的三角字根圖像，就可以說是解決了辨認的問題．

此外，由於三角字根圖像<附錄>是經由他們整理後的結果，字根圖像的集合是個有限集合，所以利用三角字根圖像來辨認中文字，便可解決字根圖像繁雜無規劃的問題．

而且應用三角碼來分類中文字除了少數字外幾可達一字一碼，而這些混淆現象我們可以當成特例來處理，用一些群體特性來區分它們．如"吧"和"邑"的三角內碼相同，但它們的結構關係不同，"吧"是左--右關係，而"邑"是上--下的關係；又如"指"和"揩"的三角內碼相同，但它們的總筆畫數不同，所以我們就可利用這些群體特性來區分它們，而不會造成辨認的困擾．

現在我們就來介紹三角字根圖像的視覺結構．對於三角字根圖像<附錄>的表示法，可以建立成樹狀(tree)結構關係圖，<圖3.5> 列出了其中兩個三角字根圖像關係圖．其中每一個葉節點 (leaf node)代表獨立的字根圖像，而父點代表獨立字根圖像結合 (composition)的結果，子節點的連線表示字根圖像間的空間結構關係(spatial relation)，父、子

節點間的連線表示其間的階層關係(hierarchical relation)。此外，三角形的節點表示這個字根圖像是三角字根，而圖形節點表示這個字根並非三角字根。



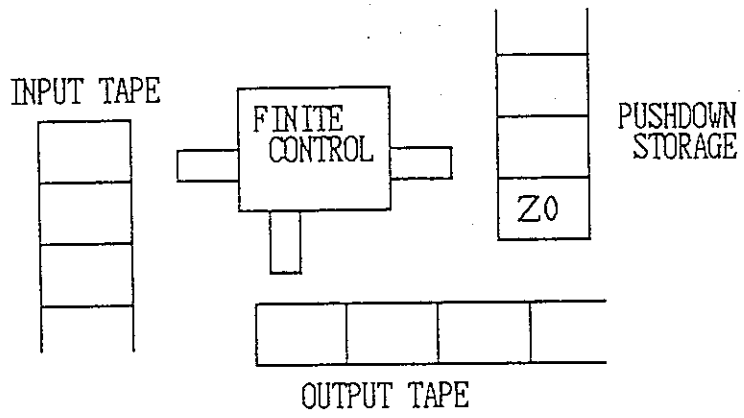
<圖 3.5>

有了三角字根圖像結構關係圖，就可以引導我們取出正確的三角字根。因為三角取碼原則是以取最大的字碼為原則，所以當影像中抽取出一獨立字根後(以Stallings的編碼來比對)，我們都必須盡力去比對結構關係圖中與這個字根有結構關係的字根，若有的話，那就可以將它們結合關係圖的父點，如此不斷地向父點發展，直到無法向上發展而停在最大的三角字根上面，那麼這個三角字根就是我們得到的正確結果。

下一節中，我們以一個自動機來說明這個字根圖像圖的辨認原理。

3.4 辨認三角字根圖像的自動機

對於前面所提到的三角字根結構關係，我們可以寫一個演算法 (algorithm)來執行它認知的功能，而這個演算法的動作原理，就可以用一個空儲存下壓式自動機(empty store pushdown automaton) 來加以說明，<圖3.6> 就是一個空儲存下壓式自動機的結構：



<圖3.6>

<圖3.6> 中上方是一個輸入的帶子，中間是控制單元，右邊是下壓儲存(pushdown storage)，而下方是輸出的帶子. 其中下壓儲存是一個先進後出 (first in/last out)的堆疊 (stack)，資料的存取都在堆疊的最上方執行. 空儲存下壓式自動機在動作時是由輸入帶子讀入輸入的字串 (input string)，而根據讀入的字串和控制單元的狀態改變來改變下壓儲存，如果輸入的字串為自動機所接受(accept)，那麼控制單元便控制讀寫頭在輸出帶子寫出正確的三角字根的字碼.

在介紹我們的空儲存下壓自動機以前，必須先把一般的下壓式自動機的定義加以闡釋：

<定義3.1>[17]一個下壓式自動機M，可定義成七維向量．

$M=(\Sigma, Q, \Gamma, \delta, q_0, Z_0, F)$，其中

(1).Σ是輸入符號的有限集合．

(2).Q 是狀態的有限集合．

(3).Γ是下壓符號的有限集合．

(4).$q_0 \varepsilon Q$是啓始狀態．

(5).$Z_0 \varepsilon \Gamma$是下壓儲存(pushdown storage)的啓始符號．

(6).$F \subseteq Q$是終止狀態的集合．

(7).δ是一個從$Q \times (\Sigma \cup \lambda) \times \Gamma$映對至有限子集(subset)$Q \times \Gamma^*$的函數．

有了這個定義，就可以介紹我們所使用的修定空儲存下壓自動機．

<定義3.2>我們修定過後的空儲存下壓式自動機M′，可以定義成一個八維的向量．

$M'=(\Sigma', Q', \Gamma', \delta', q_0, Z_0, \Phi, 0)$，其中

(1).$\Sigma'=C_1 \cup R$,其中C_1是字根結構圖中所有葉點(leaf node)字根圖像的集合，而R是結構圖中所有空間結構關係(spatial relation)的集合．

(2).Q'= q_0, q_1, q_2 ，q_0是啓始狀態.

(3). $\Gamma' = C_2 \cup Z_0$ ，C_2是結構圖中所有父點(parent node)和獨立存在的葉點(leaf node)的集合，Z_0是下壓儲存的起始符號.

(4).Φ表示終止狀態是空集合，也就是沒有終止狀態.

(5).0是輸入的三角內碼的集合.

(6). δ'是一個從$Q \times (\Sigma \cup \varepsilon) \times \Gamma$映對至有限子集$Q \times \Gamma^* \times 0$的函數.

現在我們舉一個例子來說明這個自動機作原理，<圖3.7>中列出了"㭎"三角字根的完整結構關係圖和空儲存自動機在接受(accept)這個字根所需的對映函數.現在我們只要將所得到的字根和它們之間的結構關係以後序方式 (postfix)建立在輸入的帶子(input tape)中，自動機就會自動地辨認(recognize)輸入的字串，並把正確的三角字碼輸出.

以"㭎"三角字根為例，我們輸入字串是"未""夂"L"厂"A.經過轉換函數 1,2,3,4,5,6,7，自動機便接受了這個輸入的字串，並輸出了正確的三角碼(59).

1. $\delta'(q_0,$"未",$Z_0) = \{\delta'(q_1,$"未"$Z_0, \varepsilon)\}$

2. $\delta'(q_1,$"夂","未"$Z_0) = \{\delta'(q_1,$"夂""未"$Z_0, \varepsilon)\}$

3. $\delta'(q_1,$R,"夂""未"$Z_0) = \{\delta'(q_1,$"㭎"$Z_0, \varepsilon)\}$

4. $\delta'(q_1,$"厂","㭎"$Z_0) = \delta'(q_1,$"厂""㭎"$Z_0, \varepsilon)$

5. $\delta'(q_1, A, "\text{厂}""\text{教}"Z_0) = \{\delta'(q_1, "\text{教}"Z_0, \varepsilon)\}$

6. $\delta'(q_1, \varepsilon, "\text{教}"Z_0) = \{\delta'(q_2, Z_0, 59)\}$

7. $\delta'(q_2, \varepsilon, Z_0) = \{\delta'(q_0, \varepsilon, \varepsilon)\}$

8. $\delta'(q_1, \varepsilon, "\text{教}"Z_0) = \{\delta'(q_2, "\text{未}"Z_0, \varepsilon)\}$

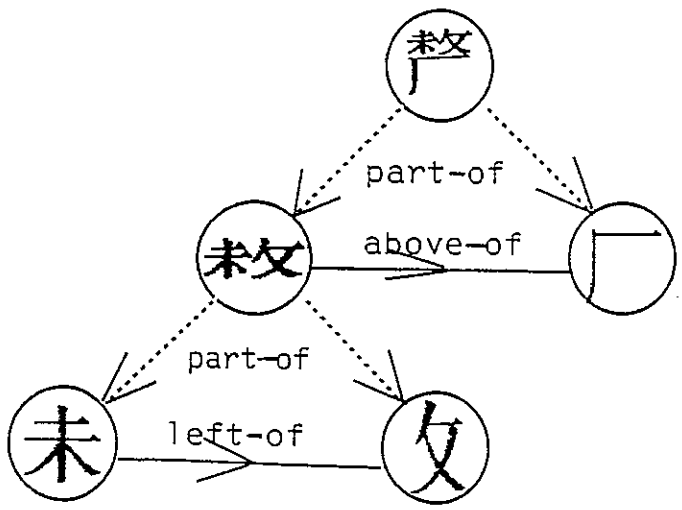
9. $\delta'(q_1, \varepsilon, "\text{攵}"Z_0) = \{\delta'(q_2, Z_0, 04)\}$

10. $\delta'(q_1, "\varepsilon, "\text{教}"Z_0) = \{\delta'(q_2, "\text{未}"Z_0, \varepsilon)\}$

11. $\delta'(q_2, \varepsilon, "\text{未}"Z_0) = \{\delta'(q_2, Z_0, 59)\}$

12. $\delta'(q_0, "\text{厂}", Z_0) = \{\delta'(q_1, "\text{厂}"Z_0, \varepsilon)\}$

13. $\delta'(q_1, \varepsilon, "\text{厂}"Z_0) = \{\delta'(q_2, Z_0, 02)\}$



第四章 自動推論中文字根圖像文法

4.1 正則語言(formal language)與網脈文法(web grammar)

電腦是以正則語言(formal language)來認知，而中文字的結構關係也可用一文法來描述，所以我們有興趣的是，在有了對中文字根和其間結構的瞭解後，是否可由電腦自動推論出描述字根結構的文法呢？要回答這個問題，我們先要對正則語言有所認識[10,17]：

<定義4.1>本文相關文法(Context-Sensitive grammar)的生成原則(production)具下列的形式：

$$q_1 A q_2 \rightarrow q_1 B q_2$$

其中$A \varepsilon V_N$，而$q_1, q_2, B \varepsilon V^*$且$B \neq \lambda$，也就是說

$$| q_1 A q_2 | \leqq | q_1 B q_2 |$$

能被本文相關文法所產出(generate)的語言被稱為第一類語言(Type 1 language).

<定義4.2>本文無關文法(Context-Free grammar)的生成原則是下列形式：

$$A \rightarrow \beta$$

其中$A \varepsilon V_N$，$\beta \varepsilon A^+$

能被本文無關文法所產出的語言被稱為第二類語言(Type 2 language).

<定義4.3>常態文法(Regular grammar)的生成原則具下列形式：

$A \rightarrow aB$ 或 $A \rightarrow b$

其中 $A, B \varepsilon V_N$ 且 $a, b \varepsilon V_T$

能被常態文法所產出的語言稱之 第三類語言(Tape 3 language).

前面所提及的文法均不足以產出字根結構的語言，必須要用網脈(web)文法才可以達到這個目的.

<定義4.4>一個網脈文法(Web grammar)G是四維的向量

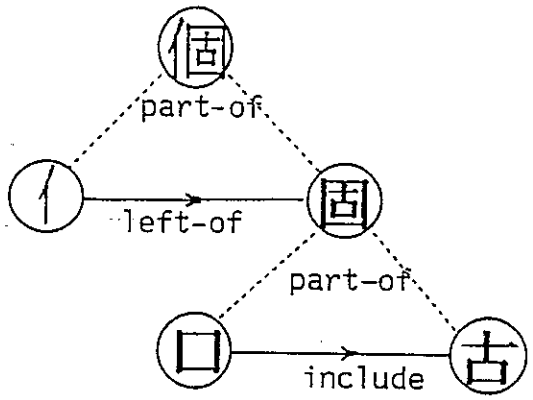
$G=(V_N, V_T, P, S)$

其中 V_N 是終結符號(nonterminals)的集合，V_T 是終點符號 (terminals)的集合 ，S是啓始網脈(initial webs)的集合，P是網脈生成原則 (web productions)的集合，而網脈生成原則定義成：

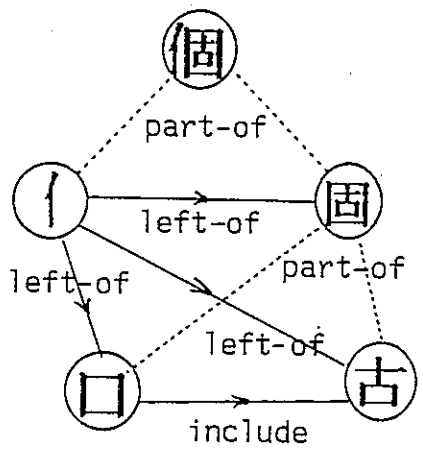
$\alpha \rightarrow \beta$, E

其中 α , β 是網脈，而E是 β 的內函式 (embedding)，它指明了一個網脈 W 的部份，網脈 α 被網脈 β 取代的條件.

有了網脈文法，我們就可以用來產出(generate)對字根圖像結構描述的語言. 以"個"字為例，<圖4.1>(a)是"個"的字根圖像結構圖，其中葉點(leaf node)上的字根圖像"亻"、"口"和"古"是基本元(primitive) . 我們先把葉點字根之間 的空間結構關係(spatial relation)完全建立起 ，這樣子得到了一個新的圖形，我們稱之推導圖(derivation diagram)，<圖4.1>(b)就是"個"字的推導圖.



(a)



(b)

<圖 4.1>

字根圖像的結構關係可以建成推導圖形式，而這推導圖可為一本無關(context-free)網脈文法所產出(generate)：

<定義 4.5>乃是一個網脈形式的推導圖，如果它符合下列條件，就可被一本文無關網脈文法 $G=(V_N, V_T, P, S)$所產出：

1. 存在一根節點(root node)S，是G的啓始符號.

2.其它的節點都只有一個父點，並且都是根節點的子代(descendant).

3.每個節點都有一個標號(lable)，而這標號都是在V中的符號.

4.如果一個節點n標號A至少有一個子代，那麼A一定在V_N中.

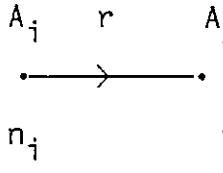
5.如果節點$n_1,n_2,\ldots,n_k$標號 $A_1,A_2,\ldots,A_k$是節點n的子節點(son node)，那麼$A\rightarrow\beta$一定是P的生成原則 (production). 其中$N_\beta=n_1,n_2,\ldots,n_k$而且$A_\lambda$是存在$\beta$中的節點$n_i$的標號(i=1,...,k).

6.n_i和n_j之間用關係r連接若且唯若：

(a)其中一個節點是另一個節點的子節點，而且r是父子關係 (direct descendant relation).

或者

(b)n_i和n_j都是A的子節點，$A\rightarrow\beta$是P中的一個法則(rule)而且



是β中的副網脈(subweb).

或者

(c) n_j和n_k用關係r連接，而n_i是經由P中法則$A \rightarrow \beta$所得n_k(標號A)的節點，且r是由A的內函應對(embedding mapping)Φ所得，在n_i和n_j間的關係。

所以，我們可以利用一本文無關網脈文法來產出字根圖像描述的語言。

以"個"為例，產出這個推導圖(<圖4.1>(b))的文法是四維向量

$$G=(V_N, V_T, P, S)$$

其中

V_N={個，固}

V_T={亻，口，古}

P: 1. 個 $\Rightarrow$ 亻 $\xrightarrow[L]{}$ 固

2. 固 $\Rightarrow$ 口 $\xrightarrow[I]{L}$ 古

L:left-of

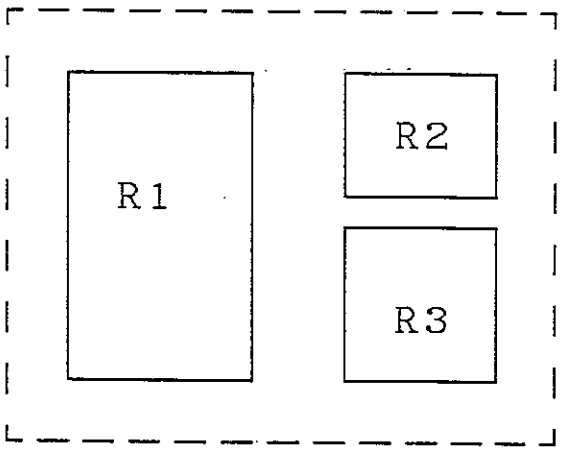
I:include

我們可以很容易地發現，以生成原則1，2的順序就可產出"個"的推導圖。

現在擺下的問題是：假使我們已瞭解了所有的字根圖型和結構關係，是否可用一演算法來自動推論出相對這字根圖像的本文無關網脈文法呢？

4.2 自動推論本文無關網脈文法(context-free web grammar)

在本節中，我們將推演出一個自動推論演算法，而這個演算法可以幫助我們自動推論描述字根圖像結構的本文無關網脈文法.



<圖4.2>

首先我們用一個例子來引導我們的思考. 以<圖4.2>字根圖像的組合情形為例，描述它結構關係的文法 $G_1 = (V_N, V_T, P, S)$

其中 $V_N = \{A_1, A_2\}$ ，$V_T = \{R_1, R_2, R_3\}$ ，$S = \{A_1\}$

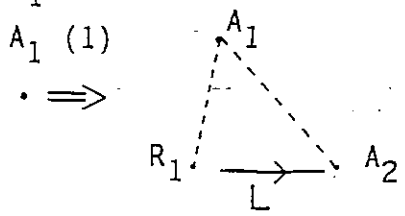
而 P : (1) $A_1 \longrightarrow R_1 \xrightarrow{L} A_2$

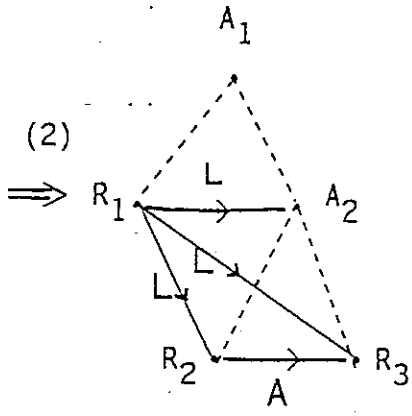
(2) $A_2 \Rightarrow R_2 \xrightarrow{L} R_3$, $R_2 \xrightarrow{A} R_3$

L : 左方 (left)

A : 上方 (above)

應用G_1產出這個字推導圖的過程是：





L: 左方(left)

A: 上方(above)

在第三章中我們提到了字根圖像的結構關係都可用中序(infix)關係來指明，對<圖4.2>這組字根圖像的中序式是：$(R_1\ L(R_2\ A\ R_3))$

比對這中序式和P的生成原則(production)，我們發現只要將中序式中對稱的括弧內結構以一個任意符號來生成，並由最內部，最左方的括弧逐一向外取代，就可得到與G中P相似的形式

$$B_1 \Longrightarrow R_2 \xrightarrow[A]{} R_3$$

$$B_2 \Longrightarrow R_1 \xrightarrow[L]{} R_1$$

所不同的地方是少了內函式(embedding).

然後我們把最後取代的符號當成啓始符號，再依反取代順序的方向逐一地把生成原則左方的結構圖形抄錄到先前之生成原則的兩方．這樣對這個例子就得到只要把B_1'，B_2用A_1，A_2替代就與P的原則完全一樣，而這樣子的替代並不會有問題．因為B_1，B_2原來就是任意取的符號，可隨便我們改名(rename)．而B_1，B_2既然可以剖析出其它的副網脈，所以它們便是文法中的非終點符號(nonterminal)．

依著這個程序，我們就可以把輸入的中序式，自動推論出字根圖像的網脈文法．而這個程序的演算法經整理後形式是：

Algorithm Automatic-Infer-Web-Grammar(I,G)

/輸入字根圖像的中序式I，得到相對應的網脈文法 G=(V_N,V_T,P,S)

STEP 1：對 I字串作剖析，找出相對應的左右括弧，並標明其深度(level)．

STEP 2：由深度多到少，左到右的次序，逐一將對應括弧內的中序式用變數A_{ij} 取代，其中 i表示深度值，j表示同一深度值由左而右遇到(occur)的次序．而A_{ij} 就是P的左方，中序式就是P的右方．

STEP 3：由A_{11}開始，依A_{ij}的以列為主(column major)次序 ，逐一將P右方中序式內變數的內函式抄錄到其它的生成原則中，這樣子就得到所到所有的生成原則．

STEP 4：所有變數 $A_{ij} \varepsilon V_N$；中序式中的所有字根圖像 $R_k \varepsilon V_T$；$\{A_1\} = S$.

END AIWG

以輸入中序式 ((R_1 include R_2) above R_3)) 為例

(* above R_3) 深度為 1

(R_1 include R_2) 深度為 2

用我們的取代原則，得到：

A_{11} => A_{21} $-\rightarrow-$ R_3

above

A_{21} => R_1 $-\rightarrow-$ R_2

include

再把內函式抄錄上去：

A_{11} => A_{21} $-\rightarrow-$ R_3

above

↗ above ↗ above ↗ above

A_{21} => R_1 $-\rightarrow-$ R_2

include

這樣就得到所有的生成原則 (production rule). 於是我們得到了一本文關文法

$G=(V_N, V_T, P, S)$

其中 $V_N=\{A_{11}, A_{21}\}$

$V_T=\{R_1, R_2, R_3\}$

$S=\{A_{11}\}$

P 列於上面

經由這個文法的推導，我們可以得到描述這組字根圖像的推導圖.

第五章 以學習模式解決鄰近缺失

5.1 學習的過程

為了方便我們的討論，我們先要有下列的定義[10]：

<定義5.1> 一個語言L的資訊序列 (information sequence)用I(L)來表示，而它是包含在 $\{+y \mid y \varepsilon L\} \cup \{-y \mid y \varepsilon V_T - L\}$ 集合的序列碼 (sequence of codes). 一個包含在語言中L的正資訊序列(positive information sequence)，我們把它表示成 $I^+(L)$ ，代表只包含在語言L中的碼. 同樣的， 一個包含在語言L中的負資訊序列 (negative information sequence)，可表式成$I^-(L)$，代表只包含V_T-L中的碼.

<定義5.2>如果合乎下列條件，便可稱語言L的一個資訊序列為完全(complete)：

1.$I^+(L)$包含所有L內的碼.

2.$I^-(L)$包含所有不在L內的碼.

如果L內每個碼都出現在$I^+(L)$，那麼這個正資訊序列$I^+(L)$稱為完全.

<定義5.3>$S_t(L)$ 表示是語言L的一個樣本(sample)，而它定義成 $\{+y_1,\ldots,+y_t\} \cup \{-y_1,\ldots,-y_t\}$ 的集合，其中$S^+ = \{+y_1,\ldots,+y_t\}$ 定義成正樣本(positive sample)而 $S^- = \{-y_1,\ldots -y_t\}$ 定義成負樣本(negative sample).

如果文法G的每一個生成元則(production)至少被用來產出(generate)包含在S^{+}中的一個碼的話，那麼語言L(G)的正樣本S^{+}稱為結構完全(structurally complete).

了解人類認知的過程是極為重要的，所以自古以來就是許多哲學家、科學家和心理學家努力研究對象，而當今世界上大力推展與倡導的人工智慧(artifical intelligence)也常以是否具有學習的能力來考量智慧的程度．人工智慧學者所研究的學習當然與哲學家和心理學家研討的方向不同，人工智慧學者著重的焦點是在於電腦不只可程式化(programming)而且使電腦有能力去學習．

對於學習的過程，我們可用下列的式子來說明：

經驗(experience)→狀態(state)→效能(performance)

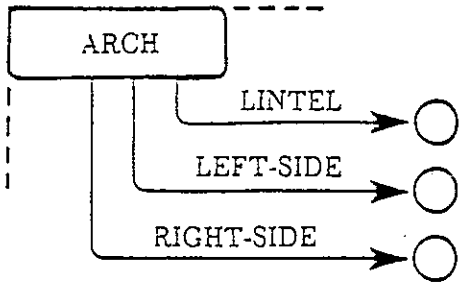
在處理一件事情時，我們會對它累積有關的知識(經驗)，而這些知識儲存在我們的腦中，也就是對處理此事的智慧(狀態)，而累積了足夠的智慧我們就有能力(效能)來對有關的事作決策、判斷和推論(inference).

5.2 樣本學習(learning from example)

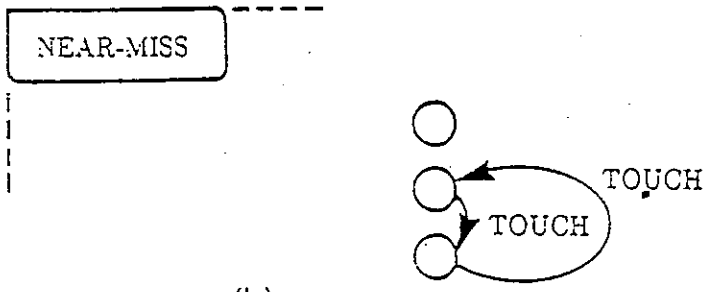
在學習模式中，樣本學習是極重要的一支．這類系統的特性在對於一特定的問題，先給予系統對此問題的概述(description)，然後再逐次對系統輸入樣本（包含正樣本和負樣本)，在經由樣本和原來的概述比對後，系統便吸收正樣品中它所欠缺的知識並排除包含在負樣本中而不該有的錯誤知識．這樣子，系統便漸漸地修正對處理這問題的知識，當我們提供了足夠的樣本後，系統終於完全的了解這個問題，進而能分辨未來輸入樣本的正確性．

在這類的學習系統中，最具代表性的是WINSTON [18]．WINSTON 系統學習的對象是針對特定影像的描述例如對拱形(arch)的結構，給予一序列的描述，而這些描述包含了正樣本和負樣本(鄰近缺失），WINSTON的系統便漸漸地吸收正樣本(generalize)而排除負樣本的知識(specialize)，而儲存這知識的文意網路(semantic nets)便不斷地被修正，最後系統在學到足夠的樣本後，終於有能力來分辨對拱形結構的描述是否正確，也就是說系統學得了對拱形的知識．

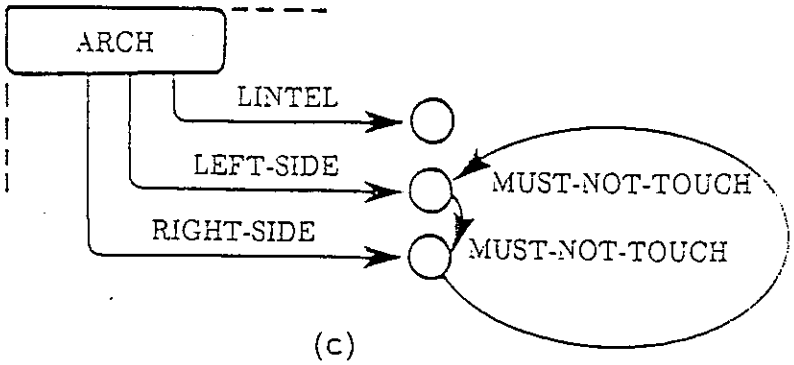
<圖5.1>(a)就是對拱形的文意網路描述，當輸入的樣本是<圖5.1>(b)後，系統便自動排除這鄰近缺失，而對拱形結構的描述便修正成<圖5.1>(c)，這種過程稱之特定化 (specialize)．


ARCH
LINTEL
LEFT-SIDE
RIGHT-SIDE

(a)


NEAR-MISS
TOUCH
TOUCH

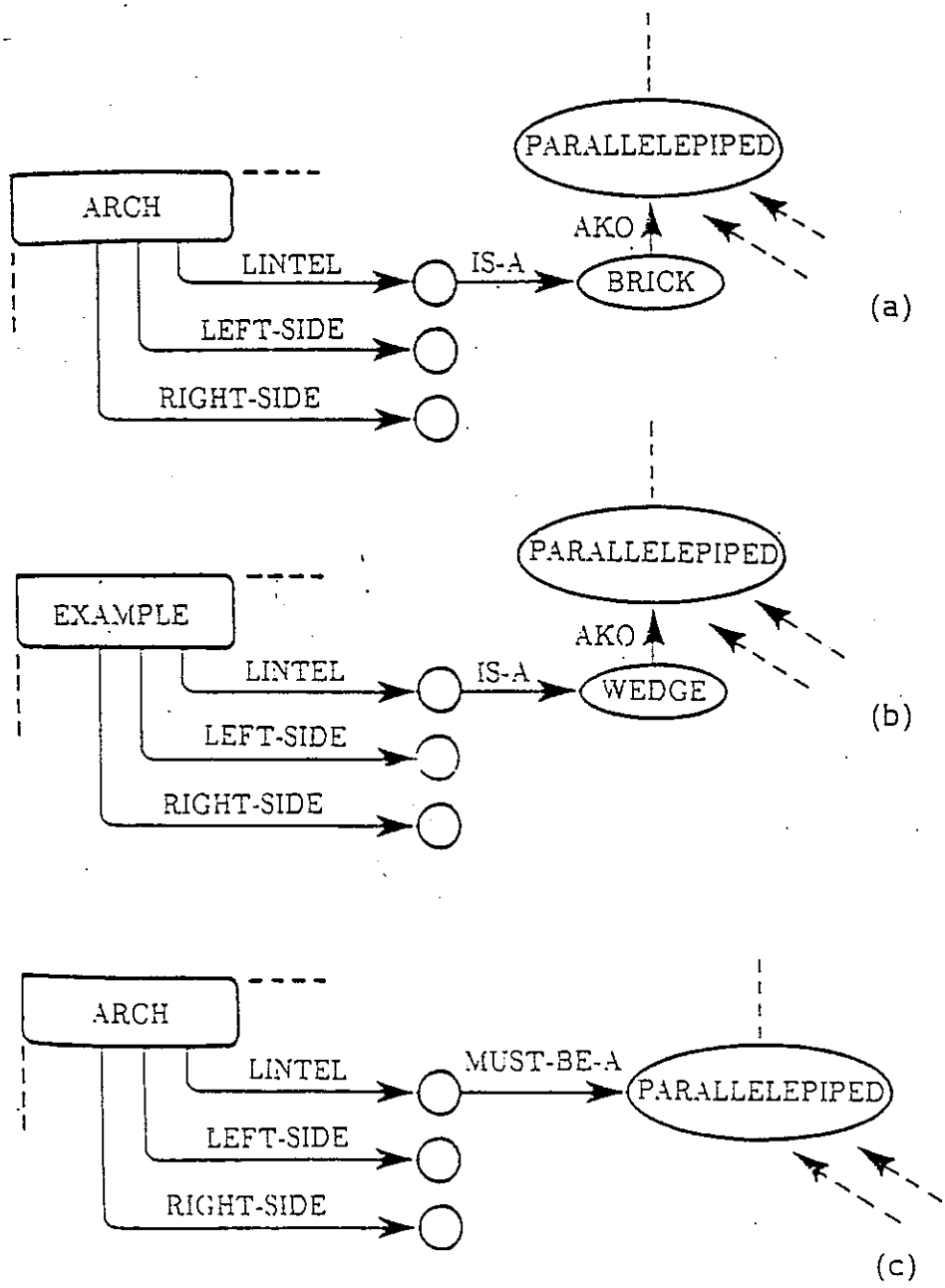
(b)


ARCH
LINTEL
LEFT-SIDE
RIGHT-SIDE
MUST-NOT-TOUCH
MUST-NOT-TOUCH

(c)

<圖5.1>

反之，當輸入為正樣本時，系統便將這知識吸收（generalize）. 若<圖5.2>(a)的拱形模式再輸入正樣本<圖5.2>(b)後，就得到了<圖5.2>(c)的較正確知識.

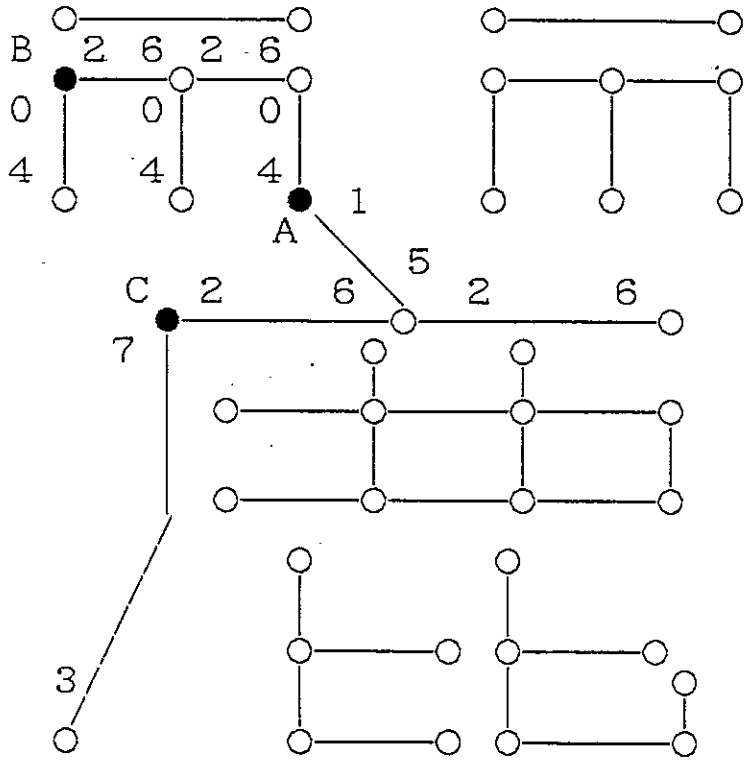


<圖5.2>

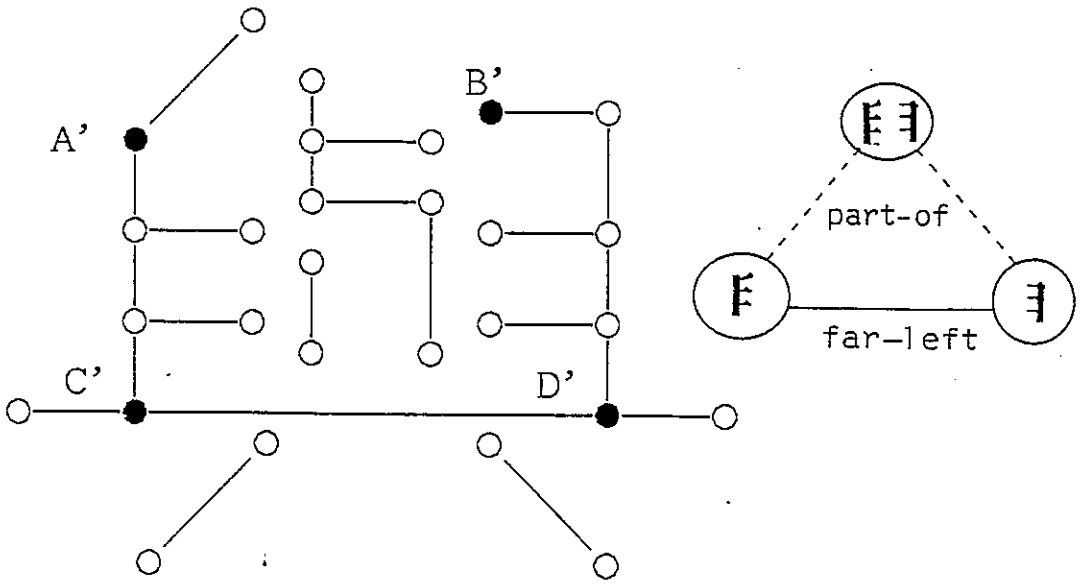
中文字鄰近缺失的問題一直是中文字辨認的困難所在，所以我們便興起了運用樣本學習模式來解決鄰近缺失的問題，也就是建造一個學習的辨認機．開始時先對辨認機輸入的字型，然後再將影像一張張的輸入讓它分析，若辨認的結果正確則表示辨認機已學得了對這張影像的了解；反之，若辨認的結果錯誤，那就是辨認機還未了解這張影像(鄰近缺失)，這時候我們便教導辨認機，讓它學得排除此鄰近缺失的形式．這樣子逐次地學習，辨認機就會像我們人類一樣慢慢地學得了辨認中文字的知識．

5.3 三角字根學習方式

如第三章所提到的，我們現在面對的是鄰近缺失問題，也就是解決字根抽取的困難.



(a)



(b)

<圖 5.3>

先讓我們來考慮一個例子，<圖5.3>(a) 是"麗" 字經由Stallings的程序轉換成的圖形，其中A節點由於雜訊影響而造成鄰近缺失，如此對這字的三角字根經由 Stallings編碼法是由B點開始探索(trace)，這樣子得到了字根的字碼，經過比對後我們會發現這個字根並非我們所定義過的字根，而無法了解這個字根是什麼； 反之，我們若有辦法將A節點打斷，而從C節點開始探索，就會得到 "广"字根的字根碼，因而得知這個字根是"广".

又<圖5.3>(b)是"與"字經由 Stallings程序轉換成的圖形，這個字的三角字根 "ㄈㄐ" 包含在一般字根中，我們若依Stallings的原則會從 A'點開始探索這個字根，由探索所得碼我們會發現它並不是我們定義過的字根，但若我們能把節點C'、D'打斷，那麼就可得到字根 "ㄈ"，"ㄐ" 字根是三角字字根圖像結構中定義過的字根，但不是一個三角字根於是我們查閱結構圖發現它必須配合第二角的字根"ㄐ"才可得到"ㄈㄐ"三角字根. 所以我們再到最接近第二角的字根來檢查，而從此字根啓始點B'開始探索就可得"ㄐ"字根，而終可合成"ㄈㄐ"三角字根.

所以只要能夠儲存下如何將鄰近缺失消去的知識再配合上三角字根圖像結構關係圖和圖中所有字根的圖形探索碼 (Stallings的編碼方法)，就可以去除鄰近缺失而正確地取出三角字根，而達成中文字辨認的目的.

至於解決鄰近缺失的知識如何儲存呢？我們選擇人工智慧中儲存知識的工具--判斷積分(predicate calculus)來儲存這個知識．

判斷積分乃是學者們為了彌補命題邏輯(propositional logic)表達能力不足所發展出來的，它的形式是一個判斷元(predicate)加上一個或數個參數 (argument)而形成，而它表達知識的類別是只可表達恆定不變的知識，不可表達可能性和猜測性等不定的知識．它除了表達單一的概念外，更可以組合起來用以表達極複雜的意念(concept)．

一個判斷元加上它的參數可稱為一個元素(atom)，其中參數可能是常數、變數或函數(function)，若結合判斷元用以表達複雜意念的形式稱為公式 (formula)，而我們對公式的定義是：

(1).一個元素是公式．

(2).如果F和G是公式，那麼 (～F)，(F∧G)，(F∨G)和(F→G)都是公式．

(3).如果F是一個公式，而 x是一個不受F所拘束的變數，那麼(∀x)F和(∃x)F都是公式．

(4).所有的公式都是應用上述原則而生成的 (generate)．

所以判斷積分是一種能表達命題 (proposition)，並且從舊有的知識中推論出新的事實(new fact)的法則(rule)，這也是我們應用判斷積分來儲存知識的原因．

正如前面所說的，我們只要標明鄰近缺失的節點就可以解決鄰近缺失的問題，所以針對<圖5.3>(a)的例子，我們止要標明A點是應被打斷的點，而這點是由筆畫"|"和"\"誤交之處，而且當這點被打斷後，在最靠近第三角的字根圖像是我們所要的三角字根"广"，於是我們可以把知識寫成：

(x,y TYPE(x,"|")∧TYPE(y,"\")∧connect(x,y))
∧FAUSE-BREAK(x,y)∧MATCH(3,"广")

→REAL-BREAK(x,y)

在這公式(formula)中，判斷積分TYPE表明某一筆畫的形態，connect表明某兩個筆畫的連接狀態，FAUSE-BREAK是個暫時將兩筆畫打斷的判斷元，MATCH是指明從字四角中最接近某一角的字根圖像對一特定字根作比對的判斷元．如果上述條件同時成立的話，我們就可啓動REAL-BREAK判斷元來將此鄰近缺失打斷．

同理，對從一般字根中抽取出所需的字根方法亦然，對<圖5.3>(b)的例子，我們只要標明C′，D′兩節點應被打斷，就可得到我們所需的字根，而它相對應的公式是：

(x,y,z TYPE(x,"|")∧TYPE(y,"|")∧TYPE(z,"一")
∧connect(x,z)∧connect(y,z))∧FAUSE-BREAK(x,z)

∧FAUSE-BREAK(y,z)∧MATCH(1,"𠂉")∧MATCH(z,"𠂇")

→REAL-BREAK(x,z)∧REAL-BREAK(y,z)

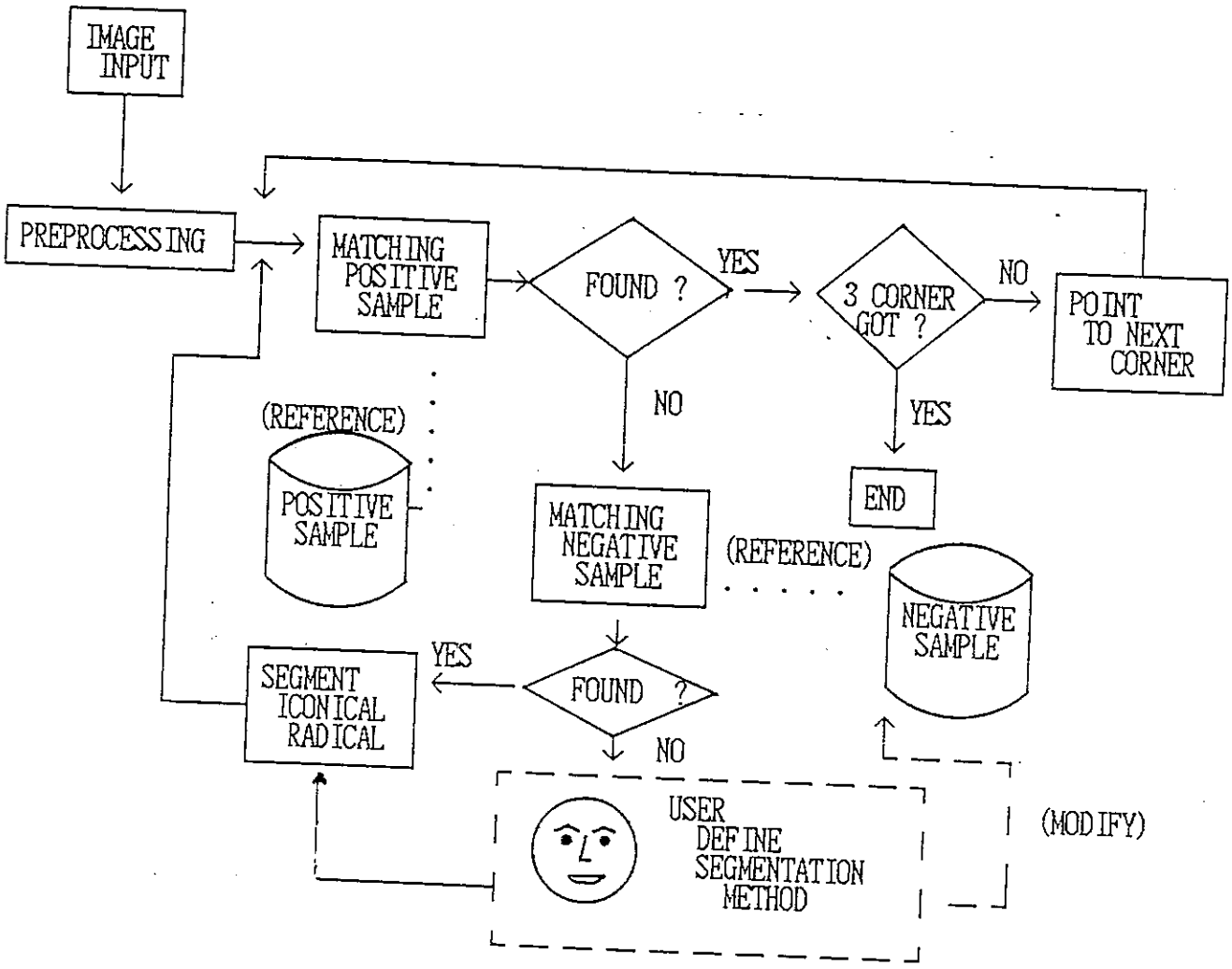
雖然這樣子可以解決我們的鄰近缺失問題，但是卻會引發另一個問題，就是當辨認機儲存大量的知識後，會嚴重地影響到系統比對的速度，因為每一個公式都需比對直至比對成功或完全比對完後才可決定是否已存有這個知識.

為了解決這個問題，我們可以應用前面所說的大分類技巧（preclassification），在每一公式的條件中加入一STROKE-TYPE 判斷元. 以指明發生鄰近缺失字根的所有筆畫狀況，這樣子就會加快我們比對知識的速度了.

下一節中，我們將介紹這個應用樣本學習的辨認機.

5.4 學習系統的流程

應用這個我們所提到的學習方式來辨認印刷中文字的系統流程列在<圖5.4>中.



<圖5.4>

系統開始動作時，先將影像經前處理後，才開始比對，而這前處理包含了第二章中述及的影像處理技巧以及Stallings的圖形處理及編碼法.

首先，我們先從第一角開始，把最接近第一角字根圖樣與正樣本(positive sample)比對，其中正樣本包含了三角字根圖像結構圖和圖中所有獨立字根的 Stallings編碼，而這個比對過程，包括了查詢三角字根圖像結構圖以得到最大三角字根獨立的動作.

如果能在正樣本中比對出三角字根，那麼系統便檢查是否已得到完整三角內碼，若得到則系統結束執行並完成辨認，若還未得到完整三角內碼，便依三角取碼原利移動取碼的角以取到下一比對的角(corner)上的字根.

反之，若系統無法於正樣本中比對得到結果，就表示發生了鄰近缺失，於是系統便比對負樣本，以解決鄰近缺失. 負樣本中就是存著我們上一節中所述的判斷積分，而這判斷積分是儲存切除鄰近缺失的方法. 如果在負樣本中能比對解決方法，那麼就依法將鄰近缺失切除，再回到與正樣本比對，而終可得到正確三角字根.

但如果在負樣本中亦無法得到解決鄰近缺失之道，那就表示發生了系統還未知類型的鄰近缺失的知識，並將知識存入負樣本中，那麼下次再發生此類鄰近缺失時，就可依法解決.

依著這樣子的流程，系統將可以三角碼原則順利地辨認印刷體中文字.

第六章 自動切除鄰近缺失

除了上一章討論的可以用學習模式來解決鄰近缺失之外，我們是否可用一演算法 (algorithm)來自動切除鄰近缺失呢？

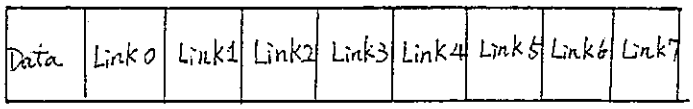
經由 Stallings[11]的編碼方法，三角字根結構關係圖中所有獨立字根的編碼均無混淆現象 (ambiguity)，因為Stallings編碼法唯一的缺失，如"未"和"末"以及"士"和"土"的編碼有所混淆--不同的字根對應到了相同的字根碼 . 但在三角字根編碼中卻不會有問題，因為三角編碼將字形相同的字根歸於同類，如"未"和"末"的三角碼都是59，而"士"和"土"的三角碼都是41.

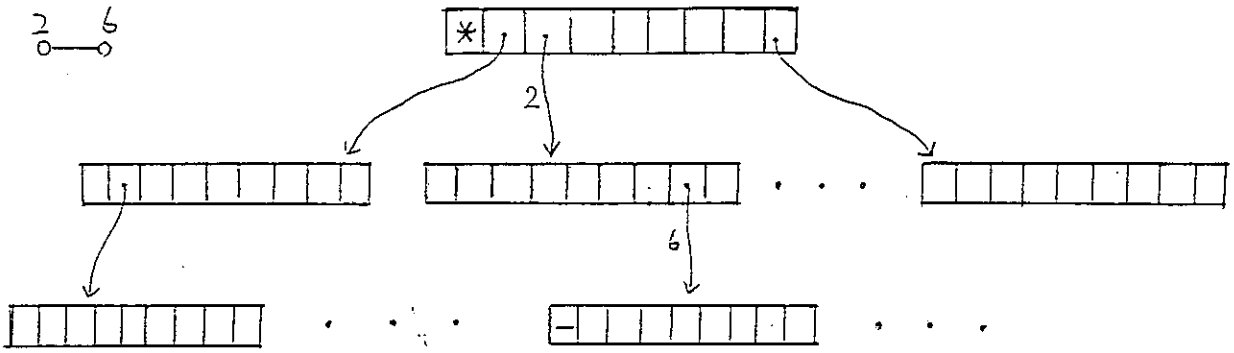
至於最後三角碼的混淆現象，如前面所說的，可以當成專案處理，用一些群體特性(global feature)來解決它們.

所以，用 Stallings圖形探索編碼法，對三角字根圖像關係結構中所有獨立字根圖像來編的碼，是我們作自動切除鄰近缺失的重要指標. 運用 Stallings的編碼法對輸入影像編碼時，我們可以同時比對關係圖中的獨立字根，如果還有字根的碼相同的話 ，表示對影像探索(trace)編碼仍有成功的希望；反之，及表示鄰近缺失發生，就把發生問題的節點標明以免下次再犯同樣錯誤，並從頭再開始編碼.

運用這個法則對影像所取出的字根，就是從此角可得到的最大的三角字根，而三角取碼法就是以最大字根為原則，所以這個所得到的字根就是正確應得的三角字根.

為了加快我們對字碼的比對，我們可將字碼建立成 <圖6.1> 的探索樹狀圖，其中每一節點包含了DATA,LINK0,...,LINK7等9個欄位，而對每個結構圖中獨立字根的內碼，都可依著其八方向碼順著樹狀圖到達一節點，並在節點的DATA欄位標記出這特定字根，所以我們就可依著樹狀結構來快速的比對出正確的結果.





<圖6.1>

此外，我們考量三角碼的取碼法時，發現三角取碼時每一基本筆畫是不可分割的單位，而每一基本筆畫可能包含數個小筆畫，如"乙"，"┐"等. 所以為免演算法在切割節點時將這種基本筆畫切開，我們必須先將影像中所有基本筆畫不可分割的節點標明. 所幸我們的前處理可得到基本筆畫，所以這項需求對我們的方法並沒有困擾.

同時，鄰近缺失可能會造成原本獨立的字根連在一起，造成抽取字根的錯誤，如 <圖5.3>(a)中第三角的字根應是"广"，但由於鄰近缺失的影響會使抽取的字根變成"𠂆". 為了避免這種誤判，我們必須在抽取字根時訂立一偵測點，如果抽取的字根不包含此偵測點，那麼所抽取的字根是錯誤，需再嚐試包含此偵測點的字根. 如 <圖5.3>(a)中，若我們訂立D為偵測點，而抽取的字根"冂"並不包含D節點， 那麼我們就必須排除"冂"字根而嚐試其下方最接近第三角的字根"广"，而終於得到正確結果.

基於上述的啓發式 (heuristic)原則，我們得到了一個有效的演算法：

Algorithm Automatic-Partition-Radical

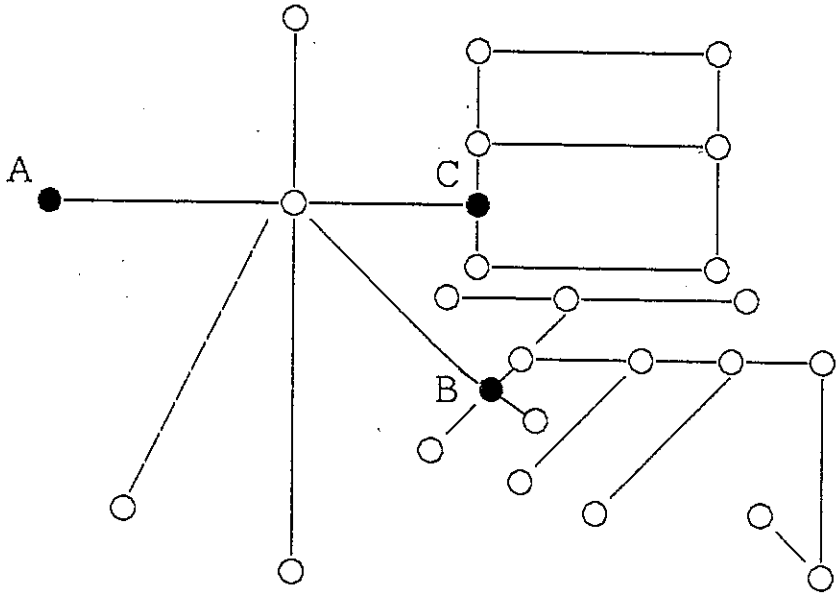
STEP 1：對最接近特定的一角上的圖像，訂立啓始點和偵測點.

STEP 2：對這圖像編碼並同時檢驗探尋樹狀圖.

STEP 3：如果檢驗樹狀圖發現錯誤，那麼就標示發生問題的節點，並回STEP 1重作.

STEP 4：如果探索結束而且偵測點在所抽取的字根內就轉向下一個抽取的角. 不然的話就嚐試下一個適合的字根，並回STEP 1重做.

以 <圖 6.2>"楊"字為例，對第一角而言，啓始點和偵測點都是A點，第一次探索到B點時，我們發現對探索樹狀圖的比對失敗，於是我們將B點標成失敗點，下次到此點便只可依原筆畫方向來探索．第二次探索直到 C點時又發生同樣錯誤，同樣地過程將 C點標為失敗點．第三次探索後，我們比對得知此字根是 "木"而且偵測點A包含在此字根中，所以就得到了正確的三角字根．



<圖 6.2>

第七章 結論

在本篇論文，除了探討中文字元辨認的困難所在和部分學者們的成就外，我們提出了以字根圖像的分析來對印刷體中文字辨認的完整解決方法：

一．中文字根圖像的結構分析可用一文法無關網脈文法(context-free web grammar)來產出(generate)．而在中文字根圖像和其間結構的關係 (structural relation)已知的條件下，我們就可以自動推論出此本文無關網脈文法，並列出此自動推論演算法(inference algorithm)．

二．解決中文字根圖像無法規範的困境，提出以中文輸入法中三角輸入法的三角字根圖像作為辨認的工具．只要從字元影像中正確地抽取出三個三角字根圖像，就達成了中文字辨認的目的．而對於字根圖像的比對是依 Stallings的編碼方法和三角字根圖像結構關係圖的檢驗得到．

三．對於雜訊和造字因素影響造成的鄰近缺失情形，我們提出了兩個解決方法：一個是基於樣本學習模式的考慮，將鄰近缺失當成負樣本來學習 ， 用人工智慧中的判斷積分(predicate calculus)來儲存切除鄰近缺失的知識，那麼這學習機在學習一段時間後就可當成辨認而實用了；另一個方法是基於對三角取碼方法和 Stallings編碼方法的啓發式(heuristic)觀察，得到了一個有效的自動切除鄰近缺失演算法．

四．以樣本學習模式來割除鄰近缺失的話，如果鄰近缺失的類別如我們想像的是一個有限集合(close set)的話那麼在一段時間後，我們的辨認將可達到不錯的辨認率；反之，如果鄰近缺失的類別不是我們想像的有限集合而是無限集合(open set)的話，那麼在學習過一段時間後，辨認機的辨認率可能只會達到一個飽和的程度(saturation)；而不易再精進．換句話說，也就是學習遇到了瓶頸，而以樣本學習方法對中文字辨認只可學到某個程度的辨認智慧，若再學習下去，只會使儲存知識的資料庫爆炸而無法精進下去．

五． 以啓發性演算法來切除鄰近缺失 ，它回應時間 (response time)的變異度 (variance) 極大．因為每次執行(iteration)都在排除一個不合理的節點然後再重做一遍，如果不合理的節點多，那麼演算法的執行次序就會多起來，而使執行時間加長．

六．為了方便對字根圖像的分析，我們將範圍訂在對印刷體中文字的辨認，但如果能考慮對字根變異的情況，我們的方法也可應用於手寫中文字的辨認．但是手寫中文字的變異度極大，如何包含了手寫中文字變異情形亦是一個值得費心的問題．

在以字根圖像分析來辨認印刷體中文字的研討中，我們發現了一些問題，並且引發了一些感觸和想法：

對於三角取碼法有極少數不合理現象，以"爾"為例，此字第一角抽取的字根竟是"不"而不是"冂"，即使以人來作三角取碼亦極易誤判，若將這種法則加在我們辨認演算法中，

不僅會造成累贅更可能會引發混淆現象 (ambiguity)。對於這種法則，我們建議不實施 (implement) 而留待後處理(postprocessing)再來解決．也就是說並不要求三角取碼一定能唯一決定中文字元，而當成細精的大分類法(preclassfication)，如此分類下來的字集一定極小，只要再加上群體特性(global feature)就可輕易地辨認出輸入的字元．

對於樣本學習模式中知識獲取的方法，我們建議以視覺語言 (visual language) 的環境來獲取．視覺語言主張以『 What you sketch is what you get. 』和『 What you see is what you get . 』的方法來和使用者溝通，讓使用者用圖像(icon)、表格(from)、功能表(menu) ...等思想的最直接的表達方式來獲取使用者的概念(concept)並表達使用者的思想．以這種方式來獲取鄰近缺失的切除方法，將是極合理且有效．

至於辨認資料的來源，建議國內建立一個國家標準之中文字影像的標準資料庫(database)，讓國內學者們很容易地去實施想法並比對結果，方可加速中文字辨認的發展．

為推展中文資訊化的理想，中文字元辨認的研究勢在必行，而它推展的步伐，尚待專家學者和有志之士的繼續努力．

參 考 文 獻

[1] R.O. Duda, and P.E. Hart, "Pattern Classification and Scene Analysis," Wieley, New York, 1973.

[2] A. Rosenfeld, and A.C. Kak, "Digital Picture Processing," Vol.1,2, 2ed Academic press., New York, 1982.

[3] R.C. Gonzalez, and P.A. Wintz, "Digital Image Processing," Addison-Wesley, Roading, Mass., 1977.

[4] C.Y Suen, and M. Berthod, and S. Mori, "Automatic Rocegonition of Handprinted Characters-The State of the Art," Proc. IEEE, Vol. 68, pp. 469-487, 1980.

[5] S. Mori, K. Yamamoto, and M. Yasuda, "Research on Machine Recognition of Handprinted Characters," IEEE Trans. on PAMI, Vol. 6, No. 4, pp. 386-405, 1984.

[6] K. Mori, and I. Masuda "Advances in Recognition of Chinese Characters," in Proc. 5th Int. Conf. Pattern Recognition, pp. 652-702, 1980.

[7] W. Stallings, "Approch to Chinese Character Recognition," Pattern Recognition, Vol. 8, pp. 87-98, 1976.

[8] S. Mori, "Research on Machine Recognition of Handprinted Characters," Computer Processing of Chinese and Oriental Languages 1, 24-39, July 1983.

[9] C.Y. Suen,"Computer Recognition of Kanji Characters," Proc. Intl. Conf. on Text Processing with a Large Character Set, pp. 429-435, 1983.

[10] K.S. Fu, "Syntatic Pattern Recognition and Applications," Prentice-Hall, Engleword Cliffs, N.J., 1981.

[11] W. Stallings, "Recognition of Printed Chinese Characters by Automatic Pattern Analysis," Comput. Graphics&Image Process. Vol. 1, pp. 47-65, 1972.

[12] H. Ma, "A Comparative Study of Thinning Algorithms in Pattern Processing," Masters' Major Rept., Concadia Univ., Nov. 1983.

[13] Y.K. Chu and C.Y. Suen, "An Alternate Smoothing and Stripping Algorithm for Thinning Digital Binary Patterns," Signal Processing, Vol. 11, pp. 207-222, 1986.

[14] W.H. Hsu and F. H. Cheng, "A Chinese OCR System," Seminar on Chinese Information Processing Recently, R.O.C., 1987.

[15] L.Z. Hu, W.W. Chang and K.D. Hwang, "Three Corner Coding--Digitied Chinese Characters," System Pub., R.O.C..

[16] O. Ore, "Theory of Graphs," American Mathematical Society, Providence, R.I., 1962.

[17] J.E. Hopcroft and J.D. Ullman, "Formal Languages and Their Relation to Automata," Addison-Wesley, Reading, Mass.,1969.

[18] P.H. Winston, "Learning Structural descriptions from examples," Ph.D. thesis, TR-76, Department of Electrical Engineering, MIT, Sept. 1970.

[19] Y.S. Chen and W.H. Hsu, "A New Parallel Thinning Algorithm for Binary Image," Procedings of NCS, pp. 295-299, 1985.

[20] B. Grunbaum,"Convex Polytopes," John Wiley, N.Y., 1967.

00	01 亠 齊	02 广 厂	03 疒	04 文 攵夊	05 亦	06 言	07 方	08 立 立辛	09 衣 衤
10 一 丶	11 工 工亞	12 丁 雨雨	13 万 万貝	14 王 壬王	15 耳 阝卩	16 石	17 乙 乚九凡儿	18 西 西覀	19 示 礻示
20 丨 亅刂巾艮	21 止	22 亻	23 彳 行攸	24 隹	25 牛 生生	26 月 月	27 厂 斤舟身	28 欠 勹	29 禾 釆
30 丿 亻川州	31 丶 户	32 冫 巜斗	33 氵 水氺	34 灬 从魚	35 馬	36 非	37 冖 光冠	38 之 辶廴	39 彡 豸豕家
40 十 七	41 土 士	42 ナ 大	43 乂 犭犬	44 艹 廿	45 革	46 女	47 力	48 走 步	49 木
50 丰 牛毛丈	51 丰 非申甲	52 車 車甫	53 戈 戊戈	54 扌 才手	55 井 非	56 中 虫	57	58 夫 夷	59 未
60 口	61 日 曰	62 目 目	63 回 母出	64 田	65 里	66 口口 皿	67 易 易	68 只 貝	69 足
70 凵 山	71 L	72 ㇇ クタタ	73 コ 弓巳艮	74 ∠ 厶幺糸系	75 厂 鳥烏	76 匚	77 ㇆ 乃刀央	78 冂 内丹	79 亅 門鬥門
80 八 ソ	81 金	82 竹 ⺮	83 人 入	84 食	85 臼 臼	86 舍 同	87 缶 气	88 又殳	89 爪 夕采
90 小 ⺌个	91 业	92 少 少	93 小 小心	94 米	95 半	96 尸 尹	97 己 巳已	98 火	99 九 尢兀