國 立 交 通 大 學

電信工程學系

碩士論文

以有限狀態機辨識大詞彙連續中文語音

Large Vocabulary Continuous Mandarin

Speech Recognition Using Finite-State Machine

研 究 生：姜翔耀

指導教授：陳信宏　博士

中華民國九十七年十二月

國 立 交 通 大 學

電信工程學系

碩士論文

以有限狀態機辨識大詞彙連續中文語音

Large Vocabulary Continuous Mandarin

Speech Recognition Using Finite-State Machine

研 究 生：姜翔耀

指導教授：陳信宏　博士

中華民國九十七年十二月

以有限狀態機辨識大詞彙中文語音

研究生：姜翔耀　　指導教授：陳信宏　博士

國立交通大學電信工程學系碩士班

摘要

近年來，有限狀態機已廣泛使用於語音辨認、語音合成及對話管理。本論文正是以有限狀態機辨識大詞彙中文語音系統之實作，以有限狀態機發展之語音辨識器，可以將聲學模型、發音辭典及語言模型分別表示為有限狀態機，並進一步將這些有限狀態機，整合為單一有限狀態機，便可得到完整的語音辨認搜尋空間。本論文可分為四部份：第一部份是有限狀態機的基本定義。而第二部份，我們將傳統語音辨識所使用的聲學模型、發音詞典和語言模型建立成有限狀態機。第三部份，我們發展一個「合併演算法」來減少各有限狀態機的狀態及轉移數目，並探討如何以「組合演算法」將搜尋空間整合起來，再將整合好的搜尋空間以維持比光束搜尋，得到辨認結果。最後，我們以 TCC300 作為此辨認系統的測試語料，實驗結果得到之詞辨認率、字元辨認率和音節辨認率分別為 52.1、72.4 及 83.5，大致與 HTK 辨識結果相當，以此證明系統的正確性。

Large Vocabulary Continuous Mandarin Speech Recognition Using Finite-State Machine

Student：Shang-Yao Chang **Advisor：Dr. Sin-Horing Chen**

Department of Communication Engineering

National Chiao Tung University

Abstract

In recent years, Finite-state Machine has widely used in speech recognition, the speech synthesis and the dialogical management. In the thesis, I used the finite-state machine to recognize large vocabulary continuous Mandarin speech .The speech recognition which developed by finite state machine could express the acoustic model, the lexicon and the language model as the finite state machine. Furthermore, I had merged these finite-state machine to the unitary finite-state machine which made the searching space of speech recognition completely. I had divided this thesis into four parts. The first part is the basic definition of the finite state machine. Second, the models which used by the traditional speech recognition had been established as the finite-state machine. Third, to simplify the numbers of state and transition in the finite-state machine I made a algorithm “Marge”. Then, we discuss how to use the algorithm “Compose” to combine the search space, and do beam search in this search space to obtain results. Finally, I use TCC300 as testing and training data. After the experiment, the accuracy of word is 52.1, the accuracy of character is 72.4, and the accuracy of syllable is 83.5. Because of the results approximately with HTK recognizing results, it can proof our system.

誌謝

終於，在漫長兩年多的時間裡，將碩士研究給完成了。在這段時間裡，首先，要感謝的是指導教授陳信宏教授，總是不厭其煩的訂正實驗上的目標，將研究導向正確的方向，更為了使我能早一步畢業，特別犧牲自己的休息時間，不斷的為我的論文提出指導。接下來，該感謝的是王逸如教授，由於教授提供了許多實驗上的經驗，並對於系統設計上有許多成果，因此，使我的研究能夠少走許多的冤枉路，並且，也為我的研究提出許多建設性的目標。

在這悠悠的學習路上，很高興能有實驗室的伙伴們相伴。志豪總是給予我許多研究上的鼓勵，讓我不至於半途而廢；庚達則是提供他研究上的資料，讓我的實驗能夠順利進行；小廣的熱心，讓課業和研究中少了許多的事務；和阿宅一起修課時，則提供功課上的提醒。真的很感謝各位同伴的相助，能讓碩士修業能一切順利。

最後，是我的家人和小伊，在我面對瓶頸時，給了我後盾以及擁抱，讓我能一步一步往成果邁進，這段時間，真的給了我許許多多的支持，如果沒有你們，大概也撐不到這一步，現在，終於完成了，真的很感謝你們。

目錄

表目錄

圖目錄

第一章 導論

語音辨識系統，已是開發不下數十年的語音應用系統；隨時代演進，諸多表示法、演算法日新月異，使得語音辨識系統的開發，衍生了更多元的組合。近年來，在詞典的大量擴充之下，為了尋求一套高效率且簡單的表示方式，使得「有限狀態機」（Finite State Machine, FSM）成為語音辨識領域的新興議題。學習專業電子領域的基礎課程中，有一門課程名為「數位電路設計」；其中一主要章節，便是以有限狀態機為表示方式的專屬設計流程。

有限狀態機在語音辨識上的應用，統合了許多語音處理不同層級的表示法，非但保有原本龐雜的語音處理系統，更大幅增添其便利性與整合能力。處理自然語音的過程中，有限狀態機不僅可以使用於模擬訊號模型，更可以進一步模擬自然語言中諸多重要且繁複的文法結構與文法特性，是為語音研究的重大功臣。因而在運用語音辨認系統的同時，除了單方面處理聲音訊號的特性之外，為了得到連續語音的高辨識結果，必得先行將文法結構或者是研究中所構築的文法邏輯加諸系統。

為進行更有效的語音辨識，首要之務為建立目前已知的語音模型系統，其中包含：聲學模型（Acoustic Model）、語言模型（Language Model）、詞典（Lexicon），並透過有限狀態機將傳統上各司其職的此三部分整合（Compose）在一起。本篇論文的核心議題在敘述有限狀態機的基本定義、建構有限狀態機流程、探討有限狀態機在語音辨認系統應用之結果與改進，以期完成更有效率且辨識率高的語音辨認系統。

1.1 語音系統中的有限狀態機

在眾多科學文獻與論文發表之中可見，近年來語音系統的研究領域開始以有限狀態機為新興的核心議題。有限狀態機的應用，除了專究電信方面的語音辨

識系統之外，亦大為協助了各領域的研究開發。此皆導因於有限狀態機的簡明特性，能夠很簡易的將語言的特性表示出來，例如：音韻學上的規則[1]、文法結構的遞迴轉移網路 [2] 等。

而有限狀態機在各領域的應用中，也依循個別需要產生了不同的設計；僅於語言領域的研究方面，前述音韻學所需之有限狀態機的規則，和本文中所建構的語言辨識系統，兩者原為看似相近且同為語言方面的研究，然而輸出字元的有無與否，就需要兩種截然不同的有限狀態機以切合其需求。是故，最簡明實用的FSM，有了包羅萬象的分身，亦成就了最廣泛的應用。

在本論文中，設計有限狀態機的同時，為了讓有限狀態機得以不佔用過多可用空間且能有效運算時間，除了多番驗證狀態機的穩定性之外，盡可能運用了一些演算法來整合等價的路徑，以期壓縮並藉此得到最小儲存空間的有限狀態機，且不損及任何應有的功能。在本文的第四章，將會提到我們如何以一個合併的演算法來減少有限狀態機的狀態。

1.2 成果

本篇論文的成果，主要是在探討如何以有限狀態機來建購一套大詞彙連續中文語音辨識系統。此系統包含了由聲學模型到語言模型的基本模型架構，再以整合方式，將完整的語音辨識搜尋空間延展出來，並以實驗驗證系統的正確性。

1.3 章節安排

- **第二章**

本章介紹關於有限狀態機的基本知識，含括了表示法、加權值運算和基本定義，以及一部份會使用到的相關基礎數學，包含聯集運算等。

- **第三章**

本章說明有限狀態機用於語音辨識時，如何將模型轉換成有限狀態機，包含

聲學模型、發音詞典、和語言模型等。

■ **第四章**

本章說明大中文辭彙進行語音辨識的系統流程。除了有限狀態機的整合演算法，來整合語音辨識時所需用到的一完整有限狀態機，以及介紹合併(merge)演算法來減少等價的路徑外。

■ **第五章**

本章將第三章和第四章的內容做結合，詳加描述如何在中文大字彙語音辨識上，實現各有限狀態機，和加以整合的結果。

■ **第六章**

本章說明實驗結果及結論，更包含了維持比光束搜尋來提供辨識結果。。

■ **第七章**

提供未來相關研究的方向。

第二章　基本知識

本章介紹有限狀態機的基本定義以及在語言辨識中所會遇到的名詞。關於有限狀態機的基本設定，可以參考莫氏的論文[9]和余氏的論文[5]。

2.1　有限狀態機

有限狀態機又稱有限狀態自動機，或簡稱狀態機，實為簡單且高效能的數學模型。它能夠反映有限的狀態中，輸入之初始動作至當下時刻的所有行為及轉換，使得給定時刻的動作皆能根據轉移函數而轉移到下一狀態，讓所有屬於此有限狀態機的函數，都對應不同的輸入關係而產生殊異的結果。

由於簡單實用，有限狀態機被廣泛運用於包括語言學、電子工程學、生物學、數學等各領域中，無論是電路系統設計、軟硬體工程、網路組合技術乃至生物醫學，只要有組合邏輯可循，它都能成為資訊動作處理的極大助力。

有限狀態機中，往往使用一個具方向性的圖來表示其動作環境;而有限狀態機的所有圖形，均以 Node、Arc、和 Input-Output 來表示。

在本文的有限狀態自動機中，每個有限狀態機基本皆由六個元素所構成：Q 是所有狀態的集合，i 是初始狀態（必定屬於 Q），F 是終止狀態的集合，Σ 是輸入字元集（所有可接受的輸入字元），Δ 是輸出字元集，δ 是對應到狀態和輸入字元再對應到下個狀態的轉移函式。

2.1.1 游標

在有限狀態機中，游標會出現在當下時間的輸入字元位置，亦即有限狀態機目前正在處理的字元。在起始時，有限狀態機的游標多半位於輸入字串的第一個字元位置；每對應一個轉移（轉移定義在 2.2.3 節），游標就向後指向後一個字元，直到字串結束為止（必須預先設定字串結束後有一個代表結束的特殊字元）。在有限狀態機得到結束字元時，我們就藉由檢查最後的狀態，來判斷路徑是否可被全然接受，以及路徑所對應到的字串是否皆已輸出。

2.1.2 狀態(state)

在有限狀態機中，Node 即稱為狀態。而在有限狀態機的理論中，狀態即是某個時間點所對應的空間位置，同時代表了有限狀態機處理至當下的背景過程。在使用有限狀態機時，系統必須是以有限的狀態所組成，並且要有一個起始狀態和零個以上的終止狀態來表示，如圖 2.1 所示，起始狀態以粗線圈來表示，一般狀態（非起始狀態或終止狀態）則以單細線圈表示，終止狀態則以雙線圈示之。在我們所使用的系統之中，除了以上三種狀態外，另外有狀態可能同時為起始又為終止的特性，此時我們將以雙粗線圈來表示。

語音辨識系統在操作時，必定由起始狀態進入，經由輸入字元進行一連串的狀態轉移；而當最後一個狀態轉移完成後，藉由檢查此轉移後的狀態，來判定此一路徑可否為接受路徑。倘若此狀態為終止狀態，便是輸出接受；若此狀態不為終止狀態，便是輸出拒絕。

2.1.3 轉移（Transition）

有限狀態機中，除了狀態，另外尚具有表示狀態間關係的圖形，就是有方向性的轉移。轉移，即用來表示狀態間的轉換關係，亦可視為一種狀態和輸入字元的函式。轉移 *t*，是一個有限狀態機裡面的函式庫，*s*[*t*]表示此一轉移 *t* 的來源狀態（source state），*d*[*t*]表示此轉移 *t* 的目的狀態（destination state），*i*[*t*]則表示此轉移 *t* 所對應的輸入字元。預先將這些轉移 *t*，集合成轉移集合 *x*；當游標鍵入一輸入字元時，會自動由集合 *x*，查表找出對應 *i*[*t*]和 *s*[*t*]的 *d*[*t*]，來使有限狀態機轉移到下一個狀態。而當 *i*[*t*]無法和 *s*[*t*]找到符合的轉移函數時，有限狀態機將否定這一路徑，並直接輸出拒絕。

2.1.4 輸入和輸出

一個有限狀態機，必須定義其所能接受的語言，若一語言可被有限狀態自動機接受，我們就稱之為正規語言。我們將輸入和輸出，放在轉移上以文字表示，並且以冒號分開，倘若一轉移的輸入為 a，輸出為 b，我們就以 $a:b$ 來表示。

2.1.5 路徑（Path）

路徑是由有限狀態機中一連串的狀態和轉移所組成的。假設，目前有一條路徑稱為 $P=P_1 \cdots P_i \cdots P_n$，其中 P_i 是代表此路徑上的第 i 個轉移，$i=1,\cdots,n$；由於 Pi 是第 i 個轉移，所以 $s[P_i]=d[P_{i-1}]$；此設定成立後，可以推導整段路徑的起始狀態為 $s[P] = s[P_1]$，而整段路徑的目的狀態 $d[P]=d[P_i]$，對應到這段路徑的輸入字串為 $i[P]=i[P_1]*i[P_2]*\ldots*i[P_n]$。

2.1.6 空轉移

在有限狀態機中，我們允許輸入和輸出字元可以為 ε，當一轉移輸入字元為 ε 時，即表示此轉移，不需要輸入也可進入下一個狀態；而若當一轉移輸出字元為 ε 時，即表示此轉移，在轉移後並不會具有任何的輸出，只要具有其中一種空字元 ε 的轉移，我們都稱為空轉移。在我們設計有限狀態機時，有許多藉由空轉移來表達或整合有限狀態機的特性

2.1.7 確定性與非確定性

在有限狀態自動機的設計中，我們對於來源狀態相同的所有轉移，其輸入字元都不重覆，我們就定義這個有限狀態自動機是具確定性的（deterministic）。在確定性的有限狀態機中，不論何時，我們必然只會有一個狀態的可能性；反之，非確定性的有限狀態機，則會有多於一種的可能性狀態產生。

在我們所建立的系統之中，會用到非確定性的有限狀態機。在建立的時候我們允許輸入字元為 ε 的空轉移存在，這表示我們的有限狀態機遇到這種空轉移

時，就算沒有輸入，也可以進入下一個狀態。因此我們在同一個時間點，就有一種以上的可能存在狀態產生。

又由於我們建造的有限狀態機是非確定性的，所以我們會儘量考慮所有可能到達的狀態，在輸入一字串後，只要有一個以上的狀態可以為終止狀態，有限狀態機就產生接受的訊號，並把相應的字串輸出。反之，當沒有任何一個狀態可能為終止狀態時，有限狀態機才會輸出拒絕。

2.1.8 等價性

如果兩個有限狀態自動機所接受的語言相同，我們就稱它們為**等價**（equivalent）。由於，此文的等價只定義在接受語言相同即可，因此等價的有限狀態機，不一定包含相同的狀態和轉移。

2.1.9 加權值

在我們所建立有限狀態機的系統時，需要更多的參數來描述有限狀態機所具有的特性。加權值是一項很重要的參數設定。我們藉由加權值，來賦予不同的轉移不同的權重，我們用 $w[t]$ 來表示轉移 t 的加權值，並且在圖上以斜線後的數值來表示。

在 2.1 節一開始時，我們以 $(Q\times\Sigma\rightarrow Q\times\Delta)$ 來表示，再加上了加權值後，我們更可進一步的把轉移函式進一步改寫為 $(Q\times\Sigma\rightarrow Q\times\Delta\times K)$，其中 K 表示的是加權值的集合。由於我們的加權值是用來表示機率的，所以當路徑上的加權值要運算的話，我們必須相乘才可以得到。

2.1.10 聯集運算

聯集運算在建構本論文中的有限狀態機時，是一種非常重要的運算；我們在建立有限狀態機時，皆由小而大，由簡入繁，因此我們總是先個別建立小的有限狀態機，再藉由聯集運算的方式，建構成一整個有限狀態機。當 A 和 B 要聯

集成一個有限狀態機時，其 A 和 B 的正規語言就以聯集的方式結合。

當我們要聯集 A 和 B 時，我們都會運用空轉移來達到不改變原本有限狀態機的內容。一開始，我們都會先產生一個新的起始狀態（一個有限狀態機只能有一個起始狀態），然後再產生兩個空轉移（輸入輸出皆為 ε），將這兩個空轉移的目的狀態連到兩個的起始狀態，然後再將原本兩個有限狀態機的起始狀態，改為一般狀態。這樣子就得到 A 和 B 聯集結果了。

2.2 字元、字串、語言

在語言辨識中，有些基本的名詞必須介紹。我們分為字元(symbol)、字串(string)和語言(language)。我們一般稱所有字元的集合為字元集（alphabet），而字串是由字元所構成的有意義詞所串接而成，最後語言是由字串所集合而成。在字串中，我們會以連接(concatenate)來將兩個字串接在一起。$a \cdot b$ 我們就看成是 b 字串接在 a 字串後面。

第三章 語音辨識中所建立的有限狀態機

本章介紹的是用有限狀態機來描述傳統大詞彙連續語音辨認中的數學模型。在語音辨認中，所用到的數學模型主要有三種：隱藏式馬可夫模型（Hidden Markov Models, HMMs）、發音詞典（Lexicon）、以及語言模型（Language Model）。

在傳統大詞彙連續語音辨認的演算法之中，本論文運用了三層不同的架構來分別處理。首先，在聲學模型的研究上，先利用隱藏式馬可夫模型來描述每個不用單元音的發音過程，接著將每一個單元音藉由發音詞典串成不同詞彙的隱藏式馬可夫模型序列。繼而使詞條的首尾相接，形成一連續的隱藏式馬可夫模型的搜尋空間。由於在搜尋過程之前，將詞彙首尾相接需要大量的語言模型，故加入能預測詞與詞相接情況的機率模型，便於進行語音辨認工作。

而在搜尋答案的演算法中，我們選擇利用維特比光束搜尋演算法來尋求最接近的答案，結合運用前述之隱藏式馬可夫模型的搜尋空間，來得到一條最佳路徑的解。由於本論文的語音辨認系統擬運用於大詞彙連續語音的實作上，倘若搜尋空間擴大，有可能會降低系統於實作上的可行性，因此使用光束搜尋演算法，期望藉動態規劃的方式增益系統的效能與實用性，並能將搜尋對象聚焦於可能性較高的族群，盡可能提高辨識效率，同時有效降低硬體使用率。

基於本論文之核心立論，乃是將發音詞典的隱藏式馬可夫模型和語言模型同時用在有限狀態機的語音辨認上，因而必須採用多元化的演算法，進一步的將所有的數學模型整合在同一層級，使其成為此有限狀態機之建構特色。

經由諸多統整之步驟後，只要在最後整合完成的有限狀態機上找一條最佳路徑，即是最佳的解；因而使用此有限狀態機，只需要將隱藏式馬可夫模型和語言模型在同一維空間中搜尋。有限狀態機既已整合為一個層級，同理而言，有限狀態機上的所有分數（包含語言模型所帶著的分數）也可藉由有限狀態機的特性，使語言模型的資料得以提早 look-ahead，而能提高辨識結果。

3.1 聲學模型

在語音辨識的搜尋空間中，聲音模型是辨識上所需要的第一個層級模型，目前語音辨識應用最廣的隱藏式馬可夫模型自然是不可或缺，詳細的探討可參考雷氏著作[11]。隱藏式馬可夫模型是一種方便且清楚代表語音訊號的模型，這個模型的架構，主要以音長、頻譜與時間軸之狀態轉換構成，藉由當前與過去時間點的每個狀態來描述聲音的特性、再藉由一連串的狀態轉換辨識其前後排序出的聲音，此一狀態序就是某一聲音訊號的模型。圖 3.1，即代表某一語音訊號的典型隱藏式馬可夫模型。

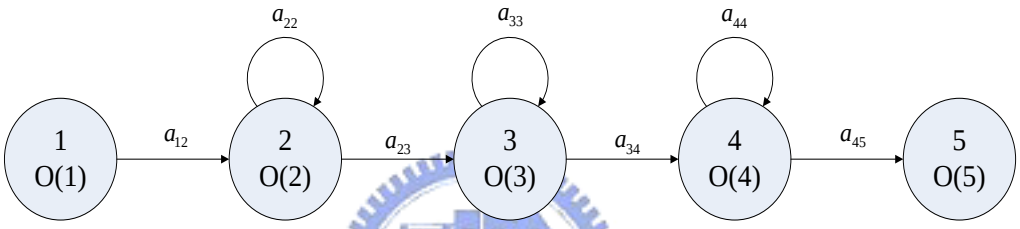

圖 3.1:五個狀態的單一聲學模型(隱藏式馬可夫模型)

由於本論文所欲建立的有限狀態機必須結合並描述隱藏式馬可夫模型，語言處理上必須一致以提高效率，因此建立系統時，必須為隱藏式馬可夫模型中的「狀態機率分佈」建立訊號路徑，使有限狀態機得以辨識。是故建構的第一步，必須先將隱藏式馬可夫模型中的狀態機率分佈從原始的表示方式，轉換到有限狀態機的轉移路徑上，並藉由正規語言表示成（i 我們定義成隱藏式馬可夫有限模型的狀態編號，又定義 $b_i = O(i)$，所以 b_i 為原 model 第 i 個 state 的 observation probability），使此語言在有限狀態機中也視為正規的輸入。另外藉由路徑的轉移，將原本隱藏式馬可夫模型中可承認的訊號路徑，藉由有限狀態機的表示方式，而轉換後的有限狀態機的轉移輸出訊號，即為此一模型的輸出音碼或音節碼，而轉移上的輸入訊號，即為前述的 b_i。以此規則，我們即將原本的隱藏式馬

可夫模型，以有限狀態機的方式表示，即表示成圖 3.2。此一隱藏式馬可夫模型的有限狀態機，在搜尋時，所展開的路徑空間為 $b_2^+ b_3^+ b_4^+$，即代表輸入為所有可接受的字串。而輸出的位置除了放在狀態 0 到狀態 1 之間，亦可放在狀態 1 到狀態 2 之間，甚至可放置在任何狀態間的轉移（必須放置在不同狀態的轉移路徑之上，此種路徑才是一成功路徑的必經路徑）。在本論文的系統之中，我們選擇將輸出放置在第一個狀態間的轉移位置上。

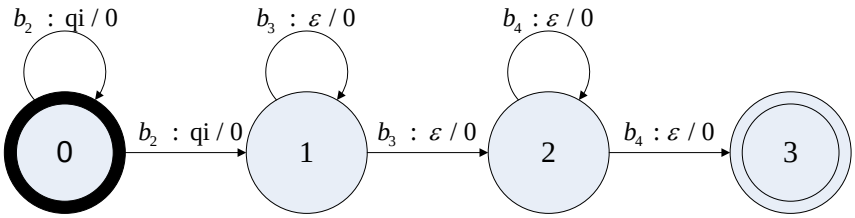

圖 3.2 有限狀態機(單次音節聲學模型)

建立好個別的隱藏式馬可夫模型的有限狀態機後，為了方便搜尋，進一步開發出單一的有限狀態機。先將每一個獨立的有限狀態機建立好，運用第二章中所提到的**聯集演算法**，將所有的有限狀態機聯集成同一起始狀態的單一有限狀態機，再將所有的終止狀態加一條路徑，回到起始狀態，以便供語音辨識時所需展開的連續語音空間使用（此路徑的輸入和輸出字元皆為空字元）。如此一來，便可得到表示連續語音空間的有限狀態機。其經由聯集演算法所完成的聲學模型有限狀態機圖例，如圖 3.3。

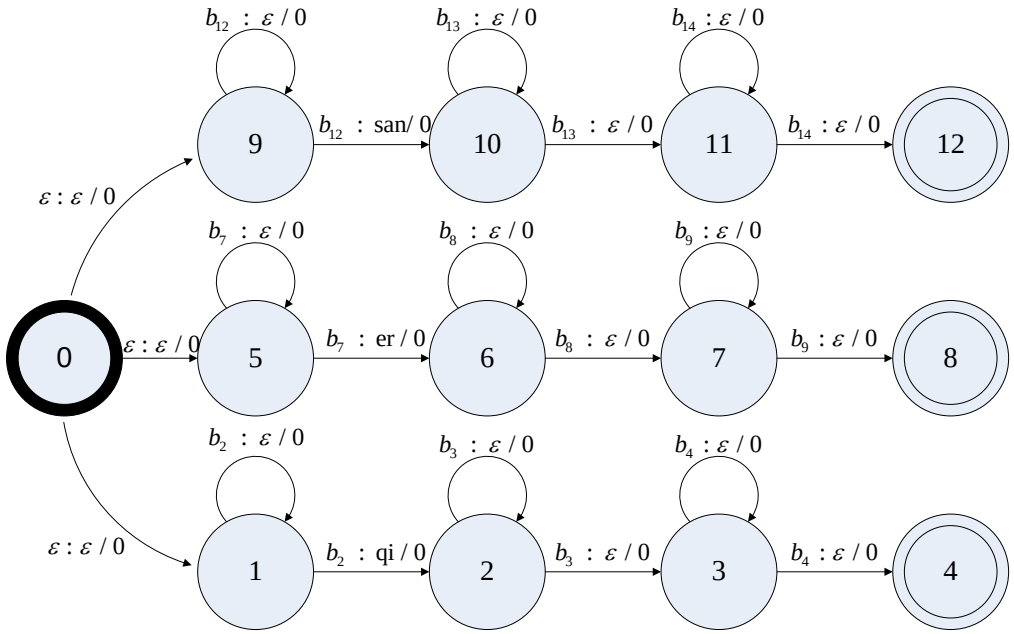

圖 3.3 聯集後的有限狀態機(多次音節聲學模型)

3.2 發音詞典

中文語音辨識在得到一連串的音碼後，必須經由發音詞典來對應到有意義的詞，才能表達出令人通曉的詞，在建立有限狀態機時，可參考莫氏的著作[4]。而本論文欲將原本的發音詞典建構為有限狀態機，期望得以在未來便於整合，使發音詞典的有限狀態機亦成為有意義的搜尋空間。

建構發音詞典的有限狀態機，首先，是將每一個詞用相同的方法建立成有限狀態機，如同圖 3.4，是發音詞典中「老鼠」這個二字詞轉換成有限狀態機表示出來的結果。由於「老鼠」這個詞，是由二個音節碼所組成的，所以有限狀態機由 state 0 -> state 1 -> state 2 的路徑中，每段轉移的輸入便編上一個音節碼，由於這個有限狀態機是對應到二字詞「老鼠」，所以便在其中一個輸出編上這個二字詞，其餘的輸出以空字串表示。

與聲學模型的有限狀態機相同，將各個詞建立有限狀態機後，由於起始狀態並不為任何轉移的目的狀態，因此可直接將所有詞條的有限狀態機聯集起來，將所有的起始狀態全部設為空轉移的初始狀態，聯集成一個有限狀態機，並且在

所有的終止狀態上，各加上一條空轉移回到起始狀態，這就完成了一個發音詞典的有限狀態機。

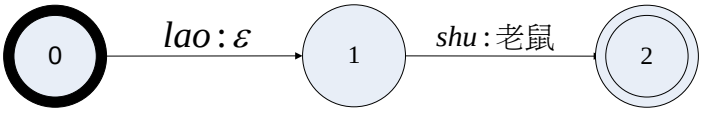

圖 3.4 線性詞典有限狀態機

上述為線性詞典所建立的有限狀態機，經共用字首結構後，有限狀態機也可表現較複雜的樹狀詞典。樹狀詞典是以線性詞典基礎，進而用樹狀結構來描述所有的詞。由於建造詞語時，我們將每個完整的詞對應在其中一個輸出之上，因此樹狀詞典在相同字首的部份，每遇相同的次音節字首，就自動共用同一字首結構，例如：「老鼠」和「老少」。如此一來，就可建構成如圖 3.5 的有限狀態機，共用同樣的狀態以及轉移路徑，大幅減省詞典空間。

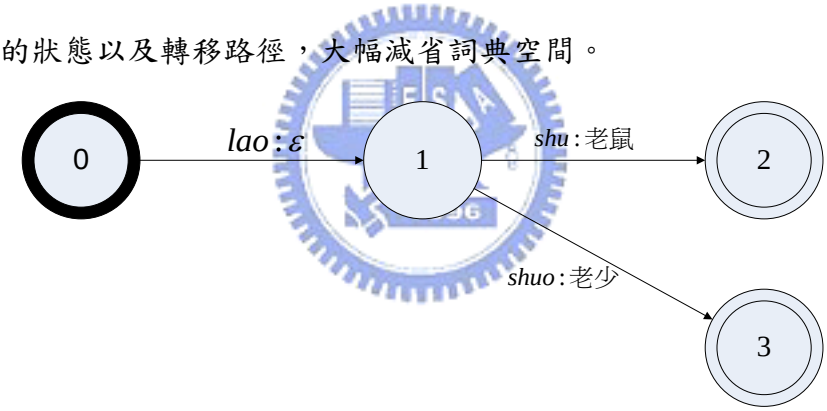

圖 3.5 樹狀詞典有限狀態機

3.3 語言模型

在大字彙連續語音辨識中，語言模型和聲學模型對於協助語音辨識具有相同的地位：聲學模型是拿來做語音訊號上的相似性比較，而語言模型則是提供了語言架構的相關資訊。在本論文中，我們使用了運用最廣泛的語言模型：統計式 n－gram 語言模型。這個模型主要是以統計的結果，來描述詞與詞之間相連接的機率模型，細節可參考朱氏（Daniel Jurafsky）[12]的著作。

在連續語音辨識中，語言模型主要是用來計算一個詞串出現的機率；

n-gram 語言模型。假設任一個詞在詞串中，只會受到前 n-1 個詞的影響。假設 $W = w_1 w_2 \cdots w_i \cdots w_m$ 是一個具有 m 個詞的詞串，由於前述 n-gram 的假設，所以 w_i 只會受到 $w_{i-n+1} \cdots w_{i-1}$ 的影響，所以 w_i 在詞串中出現的機率用 n-gram 語言模型來表示即為 $P(w_i \mid w_{i-n+1} \cdots w_{i-1})$，而我們所得到 W 詞串的機率為 $P(W) = P(w_1) P(w_2 \mid w_1) \cdots P(w_i \mid w_{i-1} \cdots w_1) \cdots P(w_m \mid w_{m-1} \cdots w_1)$。

除了上述基本的語言模型之外，如果有未出現的詞條，我們將無法得到機率 $P(w_i \mid w_{i-n+1} \cdots w_{i-1})$；若在辨識時遇到此詞串，我們將會因語言模型的機率為 0，而不可能得到此結果輸出。**後撤平滑化（back-off smoothing）**就是用來解決出現無法計算機率的問題，調整整個語言模型的機率分佈。當我們在語料中沒有 $w_{i-n+1} \cdots w_{i-1}$ 詞串時，我們便改用低一階 $w_{i-n+2} \cdots w_{i-1}$ 的機率 $P(w_i \mid w_{i-n+2} \cdots w_{i-1})$ 並乘上後撤加權值。以這個乘積來表示原本所沒有得出現的 $P(w_i \mid w_{i-n+1} \cdots w_{i-1})$。以此推類，假如沒有 $P(w_i \mid w_{i-n+2} \cdots w_{i-1})$，我們就再繼續後撤然後再逐一乘上後撤加權值，直到有可計算的語言模型機率；而此加權值的表示法，我們則根據不同的狀態所代表的歷史詞串 $w_1 w_2 \ldots w_{N-2} w_{N-1}$，把每個狀態的後徹值表示為 $b(w_1 w_2 \ldots w_{N-2} w_{N-1})$。

由於我們所得到的語言模型，是藉由 HTK 所訓練好的語言模型，所以在建立語言模型的有限狀態機之前，我們必須先確定 HTK 所輸出的語言模型。如表 3.1，第二行 N 表示狀態數，L 表示轉移數，而第三行起表示語言模型的各個狀態和轉移的細部資料，I 是顯示狀態，W 是顯示每一個狀態所表示的詞；J 代表轉移亦代表詞接詞的機率，S 代表轉移的起始狀態，E 代表轉移的目的狀態，l 代表這轉移的機率（語言模型的機率分數）。由於 HTK 本身的語言模型，和我們有限狀態機的架構方式很類似，故只要將 HTK 語言模型和有限狀態機不同的部份做轉換即可；HTK 語言模型和有限狀態機最大的差異，在於 HTK 語言模型的輸出字串放在狀態上，而有限狀態機的輸出字串是放在轉移之上；首先，我們先產生一個起始狀態，然後開始讀取 HTK 語言模型中的轉移參數，每讀取一個轉移，我們就在有限狀態機上生成一個轉移，和一個狀態，並參考 HTK 語言模

型中的轉移參數，將目的狀態所代表詞串讀取出來，放在轉移的輸入詞條以及輸出詞條（由於之後語言模型要拿來整合成一個完整的有限狀態機，所以設計上需要其輸出入詞條皆相同），然後複製原本轉移上的語言模型分數。

表 3.1 HTK 的語言模型格式

```
VERSION=1.0
N=60004 L=7577177
I=0       W=!NULL
I=1       W=!!UNK
I=2       W=!EXIT
I=3       W=!ENTER
I=4       W=A376_ME1
    ……
J=2552262 S=25141 E=16719 l=-10.08
J=2552263 S=25176 E=16719 l=-9.81
J=2552264 S=25584 E=16719 l=-9.77
J=2552265 S=25668 E=16719 l=-9.43
J=2552266 S=25679 E=16719 l=-9.28
J=2552267 S=25699 E=16719 l=-9.31
```

圖 3.6 是一個運用上面的演算法，所建出來的二連語言模型的有限狀態機。為了增加有限狀態機的可讀性，我們在狀態上除了編號外，另外加上到達這個狀態時，所需的路徑過程。例子是以詞串 $w_1w_2w_3w_4$ 來建立有限狀態機，狀態 0 是起始狀態，此狀態沒有前面的路徑轉移，所以此狀態上寫上 ε（Null），表示無前詞，其餘的狀態配合轉移路徑，將由狀態 0 到達各狀態間所經過的路徑，作為每一個狀態上的歷史狀態。

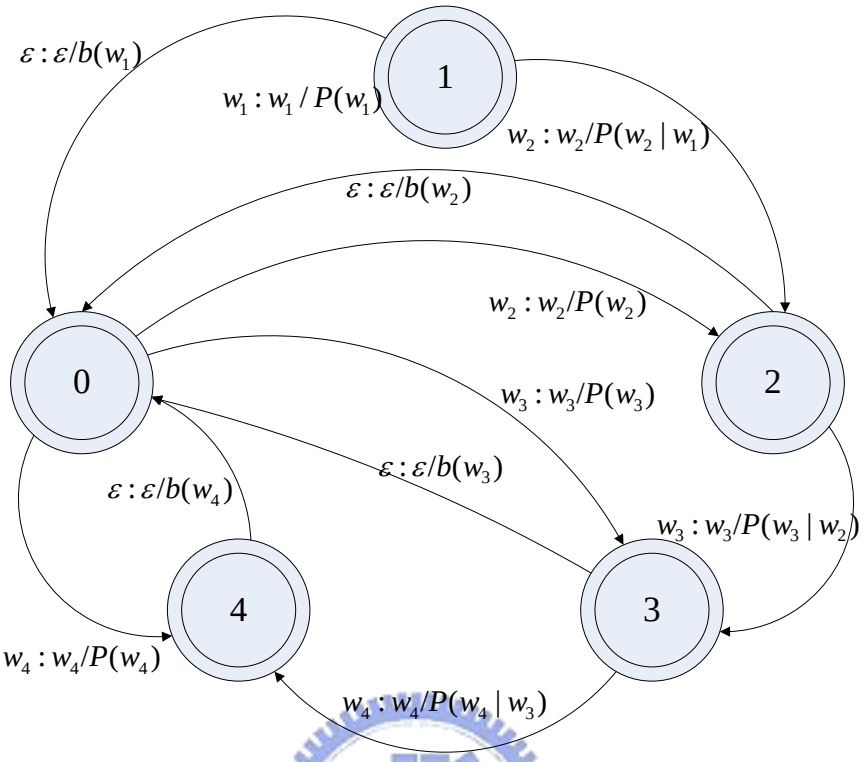

圖 3.6　語言模型的有限狀態機

除了狀態 0 之外，其餘每個狀態，皆有二種轉移，一種是輸入為有效詞的轉移，另一種是空轉移；有效輸入即為一般詞的輸入，狀態即轉移到原本的詞條加上輸入詞的目的狀態，而空轉移，代表此狀態沒有符合輸入詞的轉移，因此我們藉由空轉移，表示後撤，例如：在狀態 2 時，若下一個輸入為 w_3，我們就可進入狀態 3，並且得到加權值 $P(w_3 \mid w_2)$，若下一個輸入不是 w_3，是 w_4，我們會找不到相對應的轉移，這時，就先經由空轉移，讓狀態 2 轉移到狀態 0，然後再依照輸入 w_4，再轉移到狀態 4，這段路徑可得到加權值 $b(w_2)P(w_3)$。

在前述語言模型的有限狀態機中，我們有可能以相同詞串，對應到非唯一路徑；以語音辨識的觀點來說，相同的詞串，應只能對應到相同的語言模型路徑，然而因為在訓練有語言模型時，所有的詞幾乎都不可能有與所有其他詞相接的完整語料；因此，所建立的有限狀態機中，除了起始狀態外，我們每個狀態必然含有一個空轉換來做後撤，這些後撤會使我們的詞串對應到非唯一路徑。例如前述

的有限狀態機，在輸入 w_3 w_4 時，我們將得一條路徑為 0→3→4（狀態），和此路徑加權值 $p(w_3)p(w_4 \mid w_3)$ ，和另一條路徑 0→3→0→4(狀態)，此路徑加權值為 $p(w_3)b(w_3)p(w_4)$ 。在辨識中，我們假設經過無後撤路徑的機率，一定比有經過後撤的路徑來得高，因此，在搜尋上，只會留下經過無後撤的路徑。

第四章 有限狀態機的整合

在第三章中，我們介紹了各階的有限狀態機的建立，在本章中，我們將介紹如何使用組合演算法，來為我們的語音辨識，提供單一搜尋空間的有限狀態機，此演算法可參考莫氏的著作[7]。並藉由合併演算法，來減少有限狀態機的狀態個數，以及轉移個數，來有效減少有限狀態機搜尋空間大小。

4.1 合併有限狀態機的狀態

在語音辨識的搜尋中，為了搜尋效率與有效的儲存空間，我們儘可能的將有限狀態機中的狀態和轉移變少，在減少狀態和減少轉移的前提是，必須為一等價的有限狀態機，意思為含有原本有限狀態機所有的資料。這一節將說明以一簡單的合併演算法來完成減少有限狀態機的狀態及轉移。

合併演算法是一個很簡單的演算法，我們將其演算法寫成程式的虛擬碼如演算法 4.1。在此合併演算法中，我們將倆倆狀態拿來做檢查，若所有出自於兩狀態的轉移皆相同，我們就將其合併；出自於兩狀態的轉移相同，意謂著其狀態後的所有轉移皆相同，所以可視為由這兩狀態後可接受的輸入字串或成功路徑皆相同，因此可合併這兩個狀態。此合併演算法是初階的演算法，倘若可結合加權值的搬移，相信合併演算法將可大大減少有限狀態機的狀態數和轉移數。

圖 4.1 是我們一般看到的有限狀態機，我們可以發現狀態 4 和狀態 8 擁有相同的轉移，經過合併演算法，將所有目的狀態為狀態 8 的轉移，把這些轉移的目的狀態，全改成狀態 4，之後再將所有起始狀態為狀態 8 的轉移連同狀態 8 一起刪除，即完成我們的合併演算法，圖 4.2 就是我們合併演算法完成的結果。

MARGE(A)

0 : Let $A = \{Q, i, F, \Sigma, \Delta, \delta\}$

1: Let $T_a \Leftarrow \{\}$

2 : Let $T_b \Leftarrow \{\}$

3 : $w[t]$

4 : for all $\{q_a | q_a \in Q\}$ do

5: for all $\{q_b | q_b \in Q$ and $q_b \neq q_a\}$ do

6: for all $\{t_a | t_a \in \delta$ and $s[t_a]=q_a\}$ do

7 : $T_a \Leftarrow T_a \cup t_a$

8 : end if

9: for all $\{t_b | t_b \in \delta$ and $s[t_b]=q_b\}$ do

10 : $T_b \Leftarrow T_b \cup t_b$

11: end if

12 : if $d[T_a] = d[T_b]$

13 : for all $\{t_a | t_a \in T\}$ do

14: $s[t_a] \Leftarrow q_b$

15 : end for

16: end if

17: $\delta \Leftarrow \delta \setminus T_a$

18 : $Q \Leftarrow Q \setminus q_a$

19: end if

20:end if

演算法 4.1 合併演算法

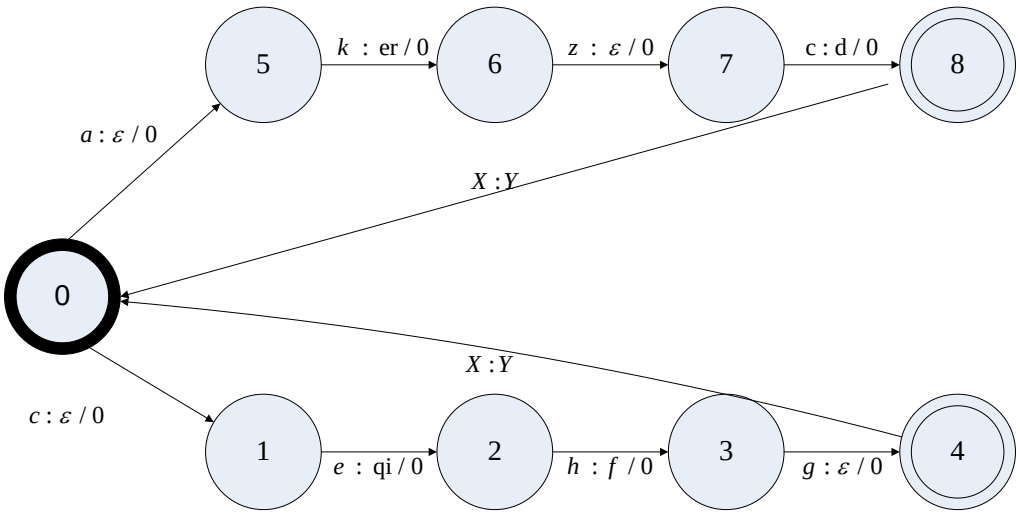

圖 4.1 未合併的有限狀態機

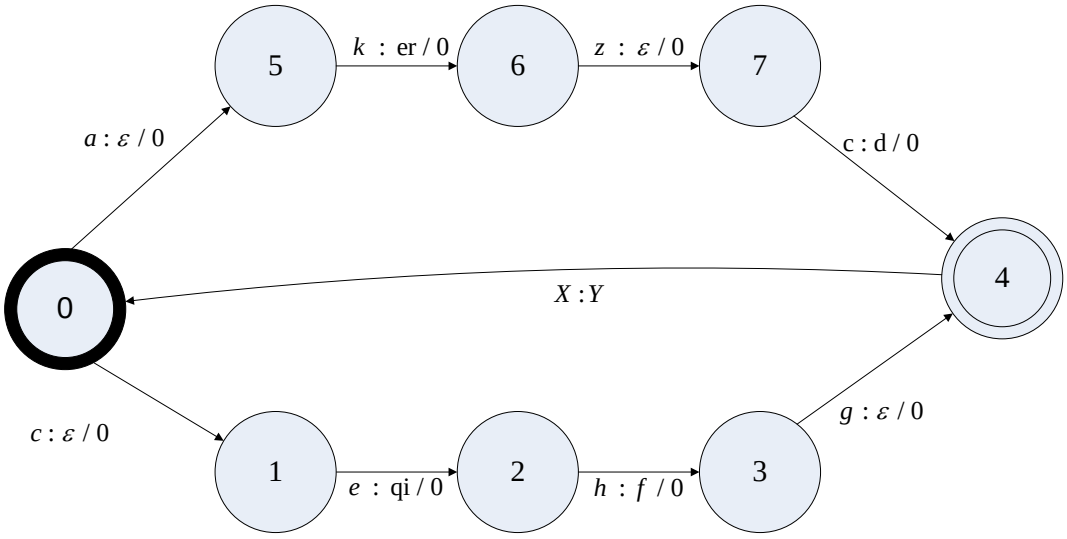

圖 4.2 合併後的有限狀態機

4.2 整合有限狀態機

上一章，我們將聲學模型、發音詞典、和語言模型逐一建立成有限狀態機，本章是為了要將上一章所各別建立的有限狀態機，藉由組合演算法，將全部的有限狀態機整合成單一搜尋空間。

4.2.1 組合運算

所謂的組合運算，即是為了將兩個有限狀態機 A 和 B 整合成一個全新的有限狀態機 C，C 包含 A 和 B 共有的有限狀態機特性以及搜尋空間，我們用 $C = A \circ \mathrm{F} \circ B$ 來表示。在有限狀態機 C 中，我們每一個狀態，皆由 A 的狀態和 B 的狀態所組成。C 的每一個成功路徑 p，都是由一個 A 的成功路徑 p_a 和一個 B 的成功路徑所組成的，在組合運算中路徑 p 輸入 $i[p] = i[p_a]$，輸出 $o[p] = o[p_b]$，並且此路徑的加權值為 $w[p] = w[p_a] \otimes w[p_b]$。

在有限狀態機中，我們每一層皆具有空轉移，來方便描述有限狀態機的特性，然而，在組合運算時，空轉移會造成不同的組合結果，因此，我們將分別討

論。

首先，我們先介紹**無空轉**移的有限狀態機的組合運算。此運算我們分別解釋演算法，並藉由圖解方式和二個假設的有限狀態機來表示。在演算法 4.3 中，我們是將組合演算法表示成虛擬語言。在此演算法中，演算法 4.2 是組合運算中最為重要的函式，這是我們運算兩個有限狀態機中的轉移是否能夠組合的組合轉移演算法。

$Trans(q_a, q_b, \delta_a, \delta_b)$
$0: T <= 0$
1: if all $d[t_a] = d[t_b]$
2:
3:
4:
5: endif
6: endfor
7:endfor
8:return T

演算法 4.2:組合轉移演算法

在 4.1 的演算法中，輸入為兩個狀態，以及兩個有限狀態機的轉移，輸出為配對成功的轉移集合。經由所有自 q_a 起始的轉移和所有自 q_b 起始的轉移對，來檢查這些轉移對是否符合 $o[t_b] = i[t_a]$，若符合條件的轉移對，我們將儲存進 T 轉移集合，當檢查完所有轉移對後，就將此轉移集合輸出。此演算法的時間複雜度為 $O(|\delta_a| \times |\delta_b|)$。在實現此演算法時，由於檢查的配度數很多，所以我們將 δ_b 的輸入字元和 δ_a 的輸出字元排序，來使時間複雜度降到 $O(|\delta_a| + |\delta_b|)$，大大提升演算法效率。

COMPOSE(A, B)

0 : Let $A = \{Q_a, i_a, F_a, \Sigma_a, \Delta_a, \delta_a\}$

1 : Let $B = \{Q_b, i_b, F_b, \Sigma_b, \Delta_b, \delta_b\}$

2 : Let $C = \{Q, i, F, \sum, \Delta, \delta\}$

3 : $i \Leftarrow \langle i_a, i_b \rangle$

4 : $Q \Leftarrow \{i\}$

5 : $F \Leftarrow \{\}$

6 : $\sum \Leftarrow \Sigma_a$

7 : $\Delta \Leftarrow \Delta_b$

8 : $\delta \Leftarrow \{\}$

9 :

10 : $S \Leftarrow \{\}$

11 : while $S \neq 0$ do

12 : $\langle q_a, q_b \rangle \Leftarrow$ an element in S

13 : $S \Leftarrow S \setminus \langle q_a, q_b \rangle$

14 : $Q \Leftarrow Q \cup \langle q_a, q_b \rangle$

15 : if $q_a \in F_a$ and $q_b \in F_b$ then

16 : $F \Leftarrow F \cup \langle q_a, q_b \rangle$

17 : endif

18 : $T \Leftarrow$ TRANS($q_a, q_b, \delta_a, \delta_b$)

19 : $\delta \Leftarrow \delta \cup T$

20 : for all $\in T$ do

21 : if $\langle q'_a, q'_b \rangle \notin Q$ then

22 : $S \Leftarrow S \cup \langle q'_a, q'_b \rangle$

23 : endif

24 : end for

25:endwhile

26:return C

演算法 4.3：組合運算

在演算法 4.3 中，我們可以看到組合運算的演算法。首先我們先假設有三個有限狀態機，並且我們欲將有限狀態機 A 整合有限狀態機 B，來產生 C 有限狀態機。我們演算法中，一開始先定義 C 為空的有限狀態機，由於我們知道有限狀態機 C 的成功路徑，是由 A 和 B 的成功路徑，所對應產生的，根據 A 和 B 所有的成功路徑集合，我們可以確定 C 的輸入和輸出字元必定根據 A 和 B，我們可以在產生 C 的轉移之前，先將 A 的輸入字元和 B 的輸出字元定義給 C。然後

再將 A、B 的起始狀態對，給 C 做為起始狀態，再根據前述的演算法，不斷的產生新的狀態對和轉移，來產生新的狀態，直到所有的狀態對皆檢查結束，有限狀態機 C 即可輸出。

在前述的組合運算中，我們優先假設了做為組合運算輸入的兩個有限狀態機皆不包含任何的空轉換；但具有空轉移的有限狀態機，亦可使用相同的組合運算，不同的地方是，有限狀態機 C 中，因為有空轉換，所以產生的路徑也會因空轉換，而產生不唯一的路徑。在圖 4.3 中，我們將以兩個具有空轉換的有限狀態機來做例子，並以圖 4.4 表示 $C = A \circ B$。

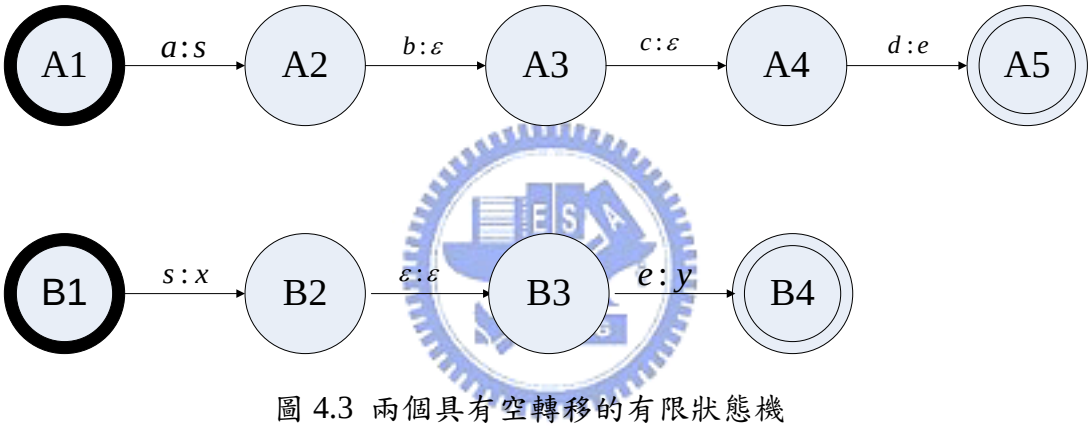

圖 4.3 兩個具有空轉移的有限狀態機

在有限狀態機做組合運算時，單空轉移在配對時，並不會造成不唯一的路徑，只會造成多餘轉移和狀態。然而，倘若轉移對中，兩個轉移同時都是空轉移時，就會造成多餘的狀態配對結果，以致於有不唯一的路徑：如 4.4 中的有限狀態機，假如 C 有限狀態機已產生 $\langle 2,2\rangle$ 的狀態後，若先以 A 有限狀態機的轉移下一個輸出為 ε，我們可以組合出轉移 $(\langle 1,1\rangle, \mathrm{b}) \rightarrow (\langle 2,1\rangle, \varepsilon, 0)$；但若先按照 B 的下一個轉移其輸入為 ε，我們又可得到轉移 $(\langle 1,1\rangle, \mathrm{b}) \rightarrow (\langle 1,2\rangle, e, 0)$。甚至，假若同時轉移又可以得到 $(\langle 1,1\rangle, \mathrm{b}) \rightarrow (\langle 2,2\rangle, e, 0)$，假若直接使用組合運算來整合這兩個有限狀態機，會產生圖 4.4。

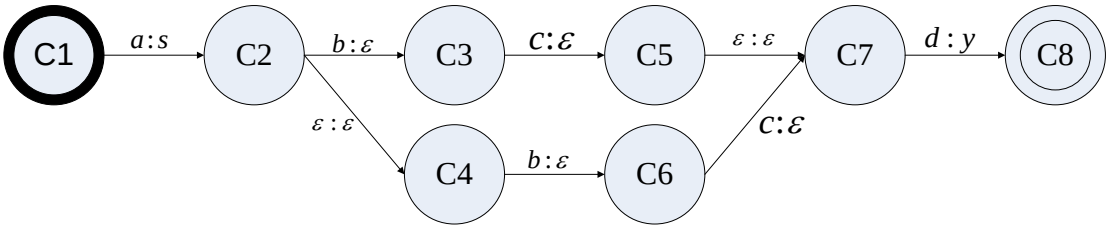

圖 4.4 $C = A \circ B$

在C_4、C_5、C_6、C_7中，由於對應到的是不同的狀態對，所以會產生兩條以相同輸入字串和相同輸出字串的轉移路徑。在定義上，這兩條路徑，都是正確，甚至是存在的。但是，在組合運算時，倘若都運算並儲存下來，會造成運算複雜度增加，並浪費記憶空間。

在這個問題之下，我們以決定唯一路徑為前提，來設計解決方法。首先，我們先將所有 A 有限狀態機中，都加上一個輸入字元為ε，輸出字元為ε_1的空轉移（由於ε的輸入並不需要佔用一個輸入就可轉移，而ε的輸出也不會影響輸出結果，所以加上此空轉移不會有任何問題），為了區別原本輸出為ε的轉移，和我們後來加上的空轉移的不同，我們特別把原本的改成ε_2。對應 A 的空轉移，我們也在 B 的有限狀態機上，增加一個轉移到自己的空狀態轉移，而這些空轉移的輸入定義為ε_2，輸出為ε，原本有限狀態機就有的空轉移輸入則另外定義為ε_1。我們會這樣定義兩個有限狀態機中的空轉移，是為了方便安排兩有限狀態機的空轉移的出現順序，藉由安排順序的規則來使轉移路徑唯一。

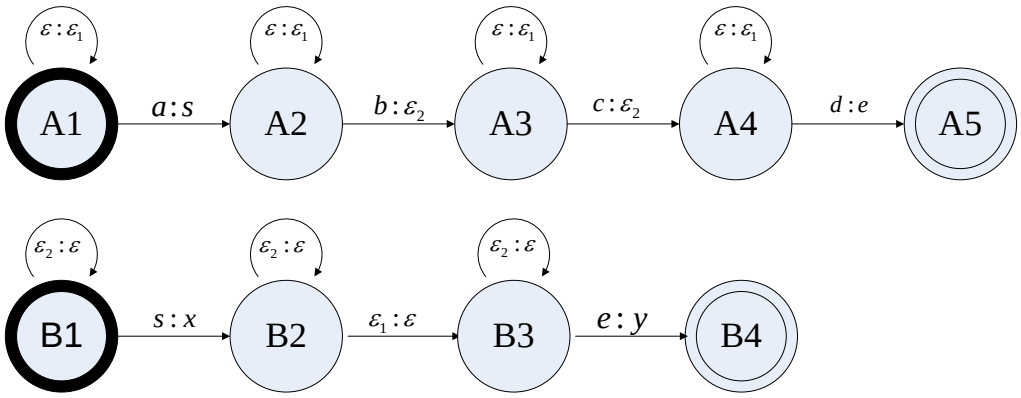

圖 4.5 增加空轉移的有限狀態機

按照這樣的想法，我們可以利用兩個有限狀態機的空轉移出現順序，來使我們的路徑唯一化。由於有限狀態機，有方便整合的特性，我們可設計一個有限狀態轉換過濾器（Filter），來完成我們的規則。

在我們所建立的系統中，希望能夠將所有 A 有限狀態機的空轉移都先走過，再考慮 B 有限狀態機的空轉移。因此，我們使用圖 4.6 中的有限狀態過濾器，並用組合演算法將 $A \circ F \circ B$ 來代替原本的 $A \circ B$。

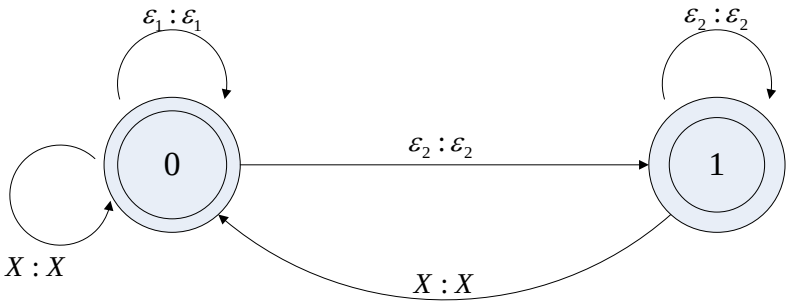

圖 4.6 有限狀態過濾器

由於有限狀態過濾器，主要是拿來做空轉移的排程，因此，我們將所有不是空轉移的輸入和輸出皆全部以 X 為表示，又過濾器是為了將空轉移做一個先後順序的排程，因此，我們不可以將全部有限狀態機有的空轉移拿掉。在我們設計的有限狀態過濾器中，我們可以看到圖 4.6 中，起始狀態為狀態 0，ε_1 是原本空

轉移的輸出，我們可以由這個過濾器來解讀，我們可以發現經由過濾器後，所有 ε_1 和 ε_2 可同時出現的字串中，ε_1 只允許出現在 ε_2 之前，所以 $A \circ F$ 後，我們可以得到圖 4.7。

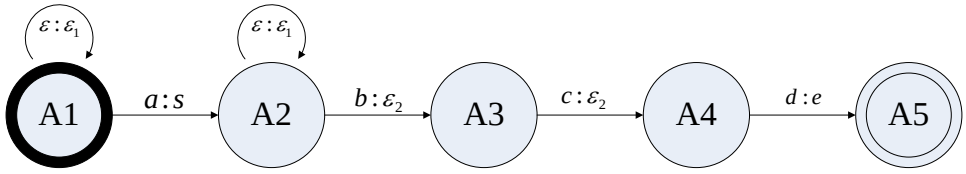

圖 4.7 $A \circ F$ 的有限狀態機

然後，我們再將這個有限狀態機再和 B 做組合運算，又可以進一步圖 4.8 的有限狀態機 C。在圖 4.8 中，我們將圖 4.7 的有限狀態機 A 和有限狀態機 B 組合運算時，我們有一些路徑，無法藉由轉移，前往終止狀態，那些狀態必然不可能成為成功路徑，因此，在加入有限狀態過濾器後，可以很明顯的發現，我們得到想要的結果，拿掉了多餘的路徑。

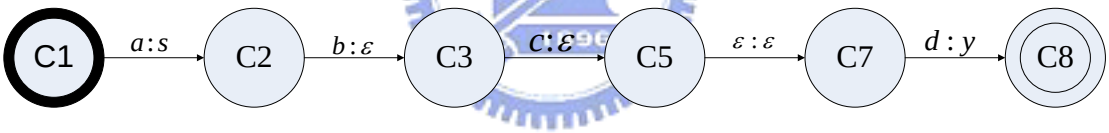

圖 4.8 的有限狀態機 C

在介紹完整合演算法後，我們便將前一節所建立好的三個有限狀態機透過組合運算，來整合成一完整的**語音辨識搜尋空間**。在圖 4.9 中，我們將三個有限狀態機的輸入和輸出定義好，在組合運算完後的搜尋空間，我們即可視為一簡單搜尋空間，輸入以最初的有限狀態機輸入，輸出為最終的有限狀態機輸出。

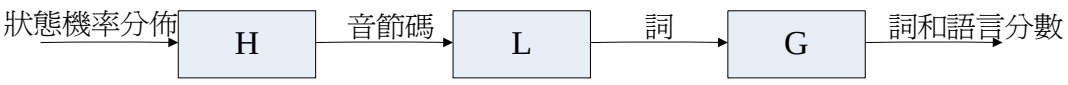

圖 4.9　有限狀態間的串聯

在圖 4.9 中，我們以 H 表示聲學模型所建立的有限狀態機，又以 L 表示發音詞典所建立的有限狀態機，G 為語言模型所建立的有限狀態機。我們將三者串聯，做組合運算，即可藉由狀態機率分佈對應到詞串以及語串所代表的語言分數。我們做語言辨識時，即是用語音訊號和聲學模型的狀態機率分佈做比對，然後再對應到詞和語言分數。在語言辨識時，就是以最後的詞的分數和語言模型的分數來搜尋一條最佳的成功路徑，此成功路徑所對應到的輸出字串即是語音辨識的結果。

第五章 中文大詞彙有限狀態機

在前兩章中，我們分別詳述了如何建立各階有限狀態機，以及如何將有限狀態機，加以整合成為一個搜尋空間，也為了整合的運算速度，提出了讓各階有限狀態機縮小的方式。

在本章中，我們將會詳述結合前兩章的內容，以圖解的方式，來呈現中文大詞彙有限狀態機。此有限狀態機將供給辨識系統做辨識之用，至於辨識的驗證部份，我們將到下一章再加以詳述。

5.1 各階有限狀態機 v.s 合併演算法

5.1.1 聲學模型

在第三章時，我們已將 generic 的作法，詳加描述了。這個作法，所產生的結果，如圖 5.1 所示，此圖中包含了三個單音節的隱藏式馬可夫模型，其中分別為 zhi、shung、sp（靜音模型），而每一個模型分別先以十個狀態表示（包含了起始狀態和終止狀態，這兩個狀態在隱藏式馬可夫模型中，是沒有 observation probability 的），所以我們可以看到有限狀態機中，只有十七種輸入（不包含ε輸入），輸入的表示法是將每一個代表的音節碼加上底線，再加上原本隱藏式馬可夫模型中，所對應的狀態編號。則靜音模型必須和原本隱藏式馬可夫模型的轉移設定相同，包含一條沒有輸入的轉移（倘若失去了這條轉移，在組合運算後所構成的中文語音搜尋空間，會遺失字和字中間沒有靜音的搜尋路徑）。

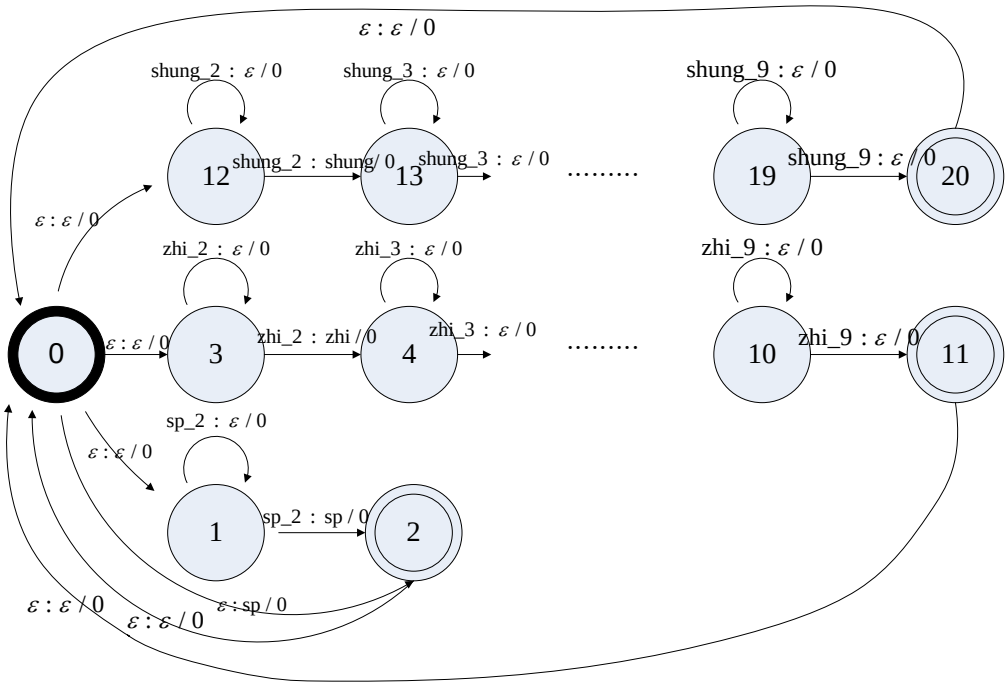

圖 5.1 直接產生出來的 HMM model FSM

在經過合併演算法之前，我們可以發現，其每個音節最後都有一個回到起始狀態的空轉移。因此，我們將聲學模型的有限狀態機，以合併演算法運算後，我們可以將所有聲學模型的最後一個狀態合併起來，得到圖 5.2：

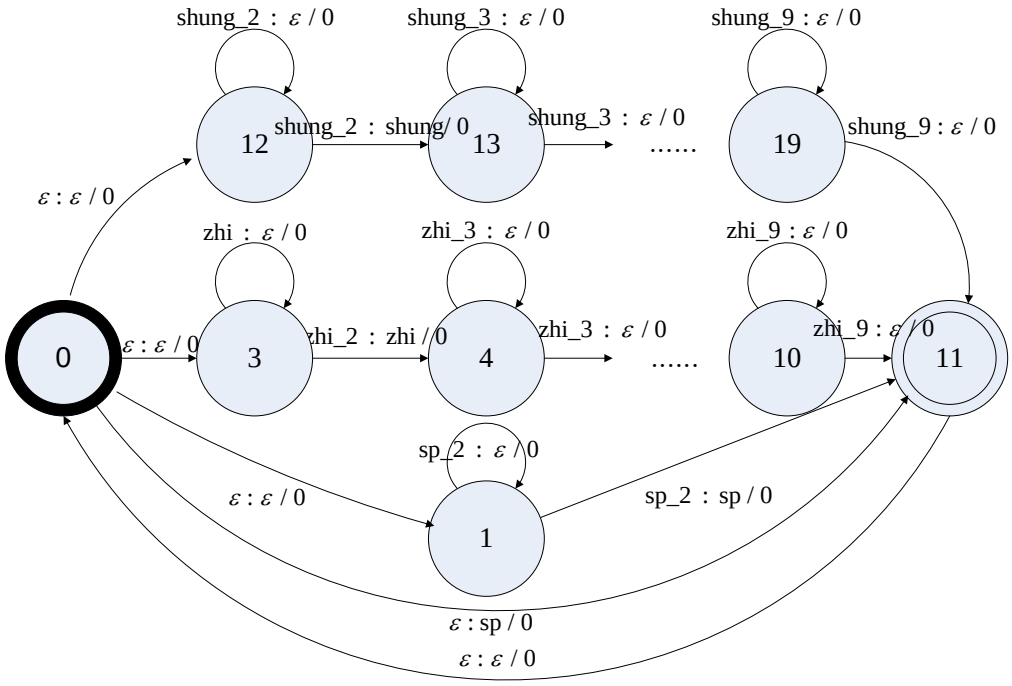

圖 5.2 合併演算法後的 HMM model FSM

由圖 5.2 的圖示中，我們可以發現，假如我們有 n 個單音節隱藏式馬可夫模型，經由整合演算法之後，我們可以減少 n-1 個狀態和 n-1 個空轉移。

5.1.2 發音辭典

第三章中，我們所提到建立的發音辭典，有兩種方式，一種是建立線狀辭典，一種是建立樹狀辭典。然而，我們在做組合運算的演算法實驗中，我們發現到，倘若使用樹狀辭典，則由起始狀態的第一個轉移大部份都會包含空輸出，此種轉移，會使我們在做組合運算時，容易展出額外、多餘且不合理的路徑（一條合理的路徑是由起始狀態開始，經由轉移至終止狀態結束）；因此，此文中所建立的有限狀態機，採用線狀辭典，並將所有詞條的輸出，放在第一個轉移上，如圖 5.3 所示；在圖中，我們以 A457A8AD_SHANG4SHEN1、CB54_ZHI3、A544AFAB_ZHU3SHEN2 三個詞做為例子，建成有限狀態機。在我們的發音辭典中，我們將每個詞的最後一個 sp 用 sp1 來取代(sp1 模型是用我們原本的 sil 模型加上一些相似於 sp 的路徑來做成，使得詞和詞相接時，可有較精準的聲學

模型），所以以 A457A8AD_SHANG4SHEN1 為例，我們所建構出來的路徑即為 shang→sp→shen→sp1。

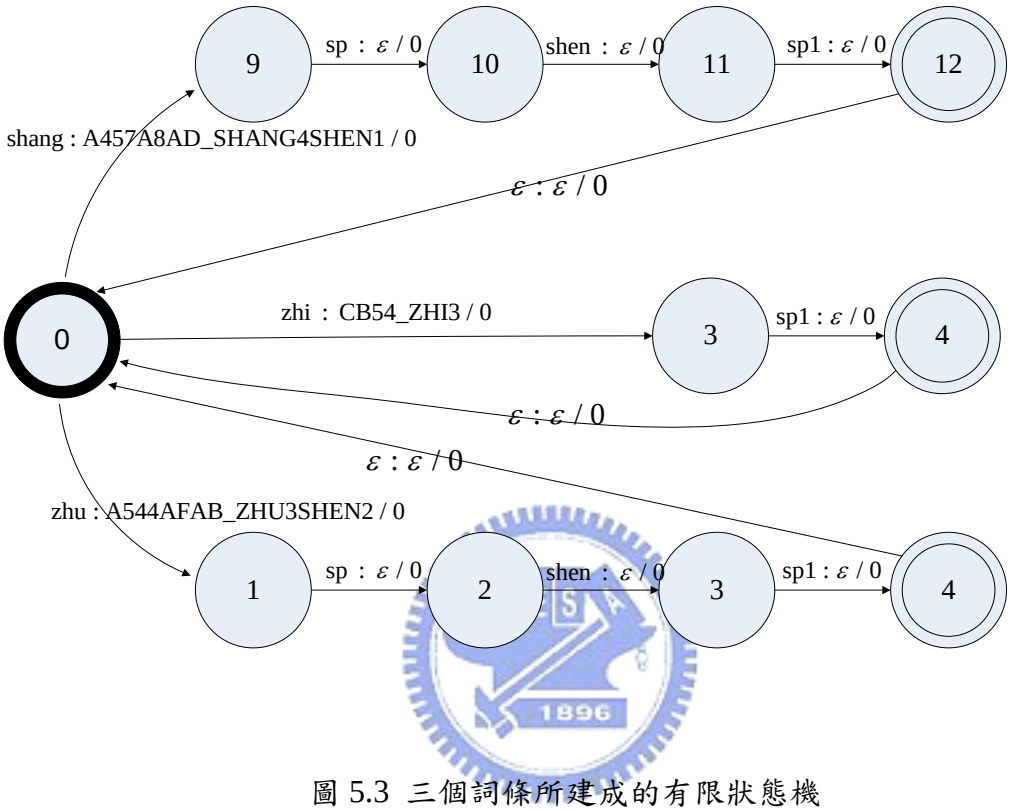

圖 5.3 三個詞條所建成的有限狀態機

我們可以發現，辭典所建的有限狀態機中，除了各個終止狀態具有相同的轉移外，部份的有限狀態機也具有相同對應的輸入輸出路徑，因此我們可以預期合併演算法後會有不錯的結果，如圖 5.4 所示：

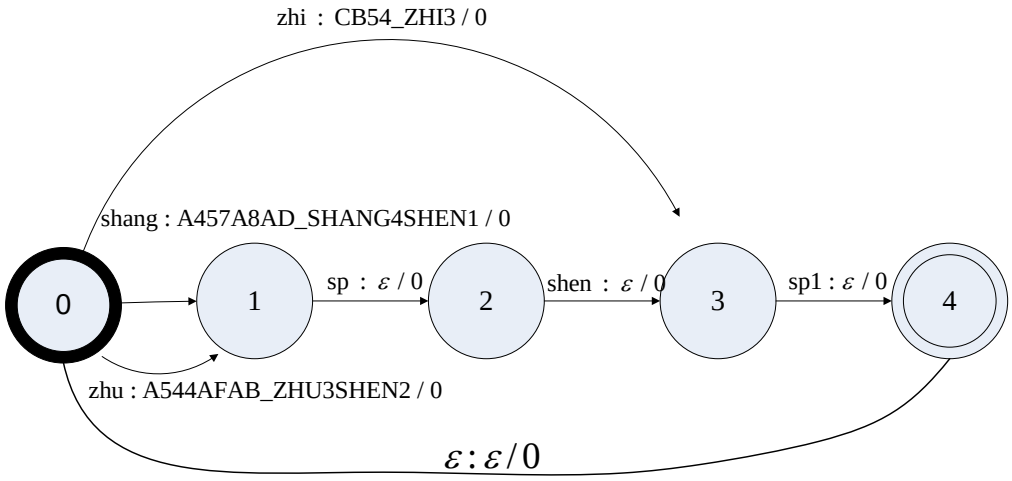

圖 5.4　經由合併演算法的三個詞條有限狀態機

經由圖例，可以看出合併演算法，對於辭典有限狀態機有不錯的表現，因此我們可以在下一章系統所建出來的狀態數和轉移數，看出有十分可觀的效果。

在發音辭典中，我們額外加上 sil 這個詞條，其只具有一個狀態，這個詞條，是為了方便我們在組合運算時，不會將句首和包尾所會用到的長靜音模型給遺忘。

5.1.3 語言模型

由於語言模型的資料來源，是由 HTK 所訓練出來的模型，因此，在建立時，本系統直接跟隨其原有的架構，建立出來的模型架構大致上如原有的相同，如圖 5.5 所示；我們將以兩個詞做為例子，分別為 A457A8AD_SHANG4SHEN1、CB54_ZHI3，又由於圖中的空間有限，所以以 α_1、α_2 表示，α_1 代表 CB54_ZHI3，α_2 代表 A457A8AD_SHANG4SHEN1，語音模型的構造為，由起始狀態→進入狀態→詞…詞→結束狀態→終止狀態，特別要注意的是狀態的轉移方向，進入狀態只能由起始狀態以 sil 輸入進入，而結束狀態也同樣的只能以 sil 輸入進入。

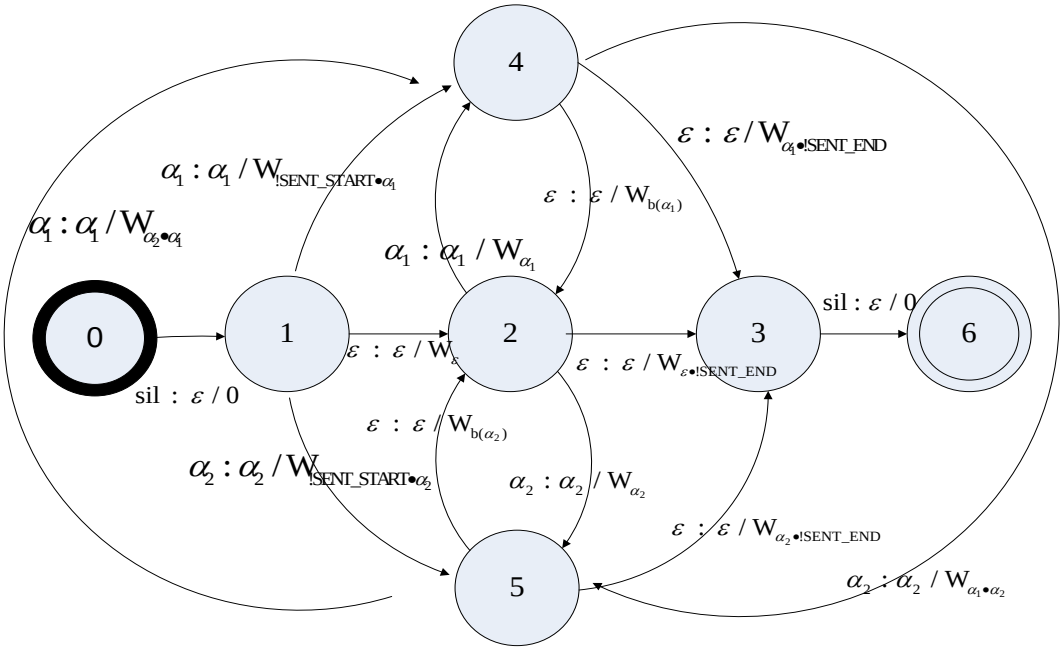

圖 5.5 兩個字的語言模型有限狀態機

圖 5.5 中，狀態 0 為起始狀態，狀態 1 為進入狀態，狀態 3 為結束狀態，狀態 6 為終止狀態，而狀態 4、5 為詞所使用的狀態。狀態 2 為一特例狀態，是提供所有詞做 back-off 時所使用的，當一個詞做 back-off 後，其歷史中已沒有任何詞時，則進入狀態 2（由於本系統目前使用的語言模型只用於 bi-gram 模型，所以所有詞使用的狀態經由 back-off 後，皆回到狀態 2）。

由於語言模型使用合併演算法後，並沒有任何改變，所以在此就不予討論。

5.2 各層有限狀態機 v.s 組合運算

各層有限狀態機在如前一節建立好，並且經由合併演算法整理之後，必須經由組合運算，才可得到同一維度的搜尋空間。

組合運算，是分別將各層的有限狀態機，以上對下關係，兩兩做運算，得到一個新的有限狀態機，由於，我們較常將我們的聲學模型重新訓練；因此，我們在做組合運算時，可先作辭典和語言模型的組合運算，再將此結果和聲學模型做組合運算；倘若之後，重新訓練聲學模型，且改變了聲學模型的架構，我們只需

要將聲學模型再和辭典與語言模型所做的結果重新運算，就可以省下一次運算的時間了。

5.2.1 組合運算（發音辭典，語言模型）

以第四章中，所提到的演算法，將發音辭典和語言模型整合起來，以發音辭典組合語言模型，發音辭典為上層有限狀態機，語言模型為下層有限狀態機。由於，組合運算，是將上層輸出對應到下層輸入，將有對應的轉移，組合成新的轉移並且輸出。

所以在發音辭典和語言模型做為組合運算後，我們會得到和語言模型類似圖型的有限狀態機，但其中的每一個轉移，皆由發音辭典中的片段路徑所取代，而取代的路徑必須具備原來語言模型轉移上的權重。倘若語言模型上的轉移輸入，在發音辭典中，找不到對應路徑，則輸出的有限狀態機，將不包含其轉移路徑。

以 5.1.3 的部份語言模型為例，來表示組合後的有限狀態機包含哪些資訊，如圖 5.6 所示，圖例中，包含了一個詞α_1代表 CB54_ZHI3 和句首包尾。

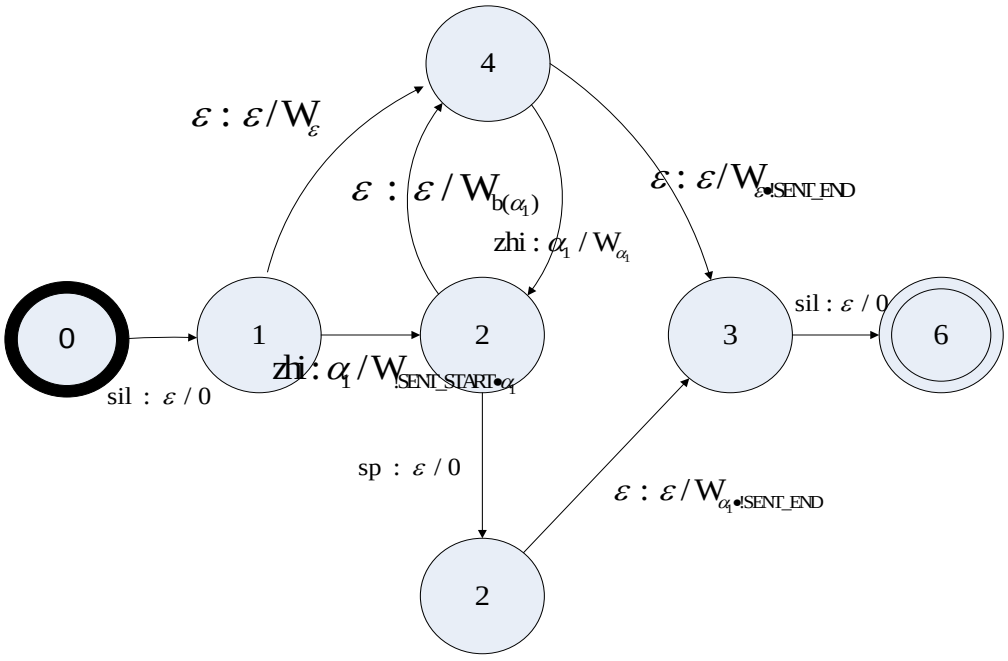

圖 5.6 單一詞和語言模型組合運算後的有限狀態機

在這兩層組合運算完的有限狀態機，即為具有發音結構，又同時具有每一個詞連接權重的有限狀態機。

5.2.2 組合運算（聲學模型，(發音辭典，語言模型)）

在完成前一節的有限狀態機後，我們要將聲學模型也整合進有限狀態機中，將聲學模型設為上層有限狀態機，而發音辭典和語言模型的整合結果，視為下層有限狀態機；和前一節相似，我們將下層有限狀態機所輸入的單音節轉移，套上聲學模型中的路徑，依照此例，即可得到具有聲學模型對應以及詞接詞權重的有限狀態機。

最後的有限狀態機，輸入為每一音節狀態的 observation probability 對應表，輸出為詞，路徑上的權重為語言模型的權重，因此，就將三層有限狀態機成功整合出一個完整的有限狀態機，具有聲學模型、發音辭典和語言模型資料，可供辨識系統做辨識了。

由於三層組合運算不易於用圖表示，所以在此不以圖示之。

第六章 辨識結果和結論

在整合完成的有限狀態機後，我們以有限狀態機的大小和辨識時間以及辨識率，來驗證我們的有限狀態機。

6.1 語音辨認

在傳統的語音辨識下，我們常使用隱藏式馬可夫模型的一階段式語音辨認，這個方法，是將聲學模型按照發音詞典互相連接，然後再將每個詞互相連接，長成一個搜尋空間。在這搜尋時，我們只允許往下一個隱藏式馬可夫模型的狀態或者是原本停留的狀態，而假如我們現在已在隱藏式馬可夫模型的最後一個狀態時，我們下一個時候便轉移進下一個隱藏式馬可夫模型的第一個狀態。然後，根據每條路徑的相似機率，以及加上語言模型的分數，來判斷我們的辨認結果。

在有限狀態轉換機中，由於已將聲學模型的機率分佈對應到輸入字元上，並且也將語言模型的機率整合到了有限狀態機的轉移加權值上。因此，我們不需要在搜尋時，同時搜尋隱藏式馬可夫模型的空間和計算語言模型的機率，這可以使我們寫搜尋程式時的複雜度大幅降低。

6.1.1 維特比光束搜尋演算法

在本文中，我們在有限狀態機中搜尋最佳路徑的演算法，和傳統大詞彙連續語音辨識一樣是採用與時間同步（time-synchronous）的維特比光束搜尋演算法。基本的維特比搜尋是計算在時間點 t 時，所有路徑結束在每個狀態的最佳分數。而時間 t+1 各狀態的最佳分數又是以時間 t 的最佳分數而求得。在最後一個時間時，我們以最高分數的狀態做往回追溯，即得到最佳路徑。

我們如果需要定義每個狀態的最佳分數，首先必須先定義一有限狀態機 $\{Q,i,F,\Sigma,\Delta,\delta\}$，狀態 q 和 $q'\in Q$，輸入字元 $\sigma\in\Sigma$，B_σ 為 σ 對應的狀態機率分

佈，o_t 為時間點 t 的語音向量，$S(q,t)$ 表示是狀態 q 在時間點 t 的最佳分數，此分數表示為：

$$S(q,t) = \min_{\delta(q',\sigma)=(q,x,w)} S(q',t-1) \otimes B_\sigma(o_t) \otimes w$$

但在有限狀態機的開始時，我們必須給予搜尋空間中的狀態，一個起始的分數：如果 $q \in i$ 時，$S(q,0) = \bar{1}$，我們以機率為 1 來表示，其餘的狀態 $S(q,t) = \bar{0}$，我們以機率為 0 表示不會存在。狀態分數不為 0 的狀態，我們稱為活躍狀態（active state）。在隨著時間陸續增加時，活躍狀態也會著有限狀態轉換機不斷增加。

由於整合後的有限狀態機，狀態數非常的大，因此，倘若時間一增加後，我們計算所有的狀態最佳分數時間複雜度為 $O(|Q|^2)$，必然十分可觀。為了可以有效率的計算狀態分數，我們可以假設最後所得到的最佳路徑中，每一個時間點的狀態分數必然為該時間點的前幾名。因此，我們引用光束搜尋，在搜尋時設定一個光束寬度（beam width），在我們確立每個時間的最高分數後，若排名超過光束寬度的狀態，則我們就使該狀態分數 $S(q,t) = \bar{0}$。在本系統中，我們使用了另一種方式來限制狀態數，我們設定了每個時間點所存活的狀態數(window size)，藉此，來使有限狀態機的活躍狀態個數得以控制。

6.1.2 實驗環境

以連續中文語音辨認的環境和結果。測試平台使用的是 Intel(R) Core(TM)2 Extreme CPU X9650 @ 3.00GHz，記憶體為 8Gbytes。

聲學模型

使用的聲學模型是隱藏式馬可夫模型，由 415 個音節所構成，其中含三個靜音模型（一個長 sil 具有十個狀態構成、一個長 sp 為十個狀態構成、一個短 sp

具有三個狀態構成)，一個 unk 模型（具有三個狀態構成)。其餘非靜音模型中，每個音節皆為十個狀態構成 ，每一個狀態機率分佈為六十四個高斯分佈的模型。

發音詞典

以六萬詞做為發音詞典。發音詞典由一到八字詞所組成。

語言模型

將 TCC300 中的 845 個檔案做為訓練語料，此語言模型為 bi-gram 語言模型，具有 7457177 個 bi-gram 機率，和 60000 個 uni-gram 機率，以及其 60000 個詞後撤機率，共有 7577177 個機率分數。

測試語料

我們將測試分為三個部份。

第一部份－我們將 NCKU 中 83 個音檔，共 9739 個音節，做完整的測試，來得到測試結果。此結果，將和 HTK 做比較以證明系統正確性。

第二部份－我們取 NCKU 的前 25 個音檔，來做為測試語料，來分析辨識系統中所需要的參數。(Window size 和 Language model weight)

第三部份－TCC300 語料中的 NCKU 和 NTU 和 NCTU 三個部份，各取 25 個音檔來做為測試語料，並分別得到測試結果。其中 NCTU 具有 2448 個 syllable，NTU 具有 894 個 syllable，NCKU 具有 3274 個 syllable。

有限狀態機

我們以上述各模型來整合成有限狀態轉換機，我們可以得到各層有限狀態機；我們已知有限狀態機組合的演算法，十分耗費記憶體資源和時間，故我們先在組合運算前用合併演算法，來降低各有限狀態機的狀態和轉移數。表 6.1 和 6.2 為使用合併演算法前後所得到不同的轉移個數。

我們可以發現合併演算法所獲得的效果，只有發音詞典才有獲得部份縮小的

效益，這是由於只有發音辭典中，才具有能夠合併的路徑，但由於本論文中的合併演算法，並沒有結合其餘有限狀態機的分數轉移技術，因此，相信合併演算法尚有發展的空間。

表 6.1 各別轉成有限狀態機後的資料

合併前	總狀態數	終止狀態數	總轉移數
隱藏式馬可夫模型(*H*)	3722	414	7447
發音詞典(*L*)	233449	60000	353451
三連音模型 (*G*)	60004	1	7577177

表 6.2 合併演算法後各有限狀態機資料

合併後	總狀態數	終止狀態數	總轉移數
隱藏式馬可夫模型(H)	3307	1	7032
發音詞典(L)	218380	1	278382
三連音模型 (G)	60004	1	7577177

在經過合併演算法所產生新的有限狀態機後，我們將新的有限狀態機拿來做組合運算，將三層的語音資料，整合成一個具有所有語音資料的大型有限狀態機，以此有限狀態機來做為語音辨識時所使用的搜尋空間。整合後的有限狀態機如表 6.3 所示。

表 6.3　整合完成後的有限狀態機

	總狀態數	終止狀態數	總轉移數
$H \circ L \circ G$	1951087	1	11434003

6.1.3 辨識結果

我們將辨識時所需的有限狀態機準備好後，先進行 411 音的測試，我們先將測試的音檔，以 HMM model 的有限狀態機進行辨識，由於我們是以 411 音作 free-run 的 syllable 辨識，來證明我們的 HMM model 和 Viterbi 演算法的正確性，故作此辨識，得到的辨識結果如下：

411 音 free run : syllable_acc(%) = 66.5, syllable_corr(%) = 72.3

由於我們是作 411 音 free run 的辨識結果，因此辨識率會因 insertion 過高而使辨識率降低，因此此結果是正確的。

第一部份：

由於我們的實驗對照組，是以 HTK 做為驗證，故先以 HTK 做相同的語音辨識，HTK 辨識相同資料，ｐ值設為-10，Ｓ值（Language Model Weight）設為 9，得辨識結果如表 6.4。

表 6.4 HTK 的辨識結果

HTK 結果	Syllable	Character
正確率(%)	83.18	74.80

我們將以上面 HTK 所辨識的結果做為依據，將我們的系統也以相同的參數做 NCKU 部份的完整辨識，由於我們的參數設定和 HTK 並非完全相同，因此我們儘量以最好的參數對來做辨識。其辨識結果如表 6.5：

表 6.5 本系統在相同語音資料所辨識的結果

	Syllable	Character	Word
正確率(%)	83.48	72.35	52.09

第二部份：

我們在這個部份做兩個不同的分析。首先，我們先以 Window Size 設為實驗項目，根據 HTK 的設定，我們也將 S 值先以 9 做實驗。可得到表 6.6：

表 6.6 Window_Size 大小對辨識率的測試

weight_lm	Window_size	word_acc	char_acc	syl_acc
9	500	11.5	49.6	68.3
	1000	14.0	63.3	72.2
	2000	38.1	73.2	78.7
	5000	68.6	82.2	90.1
	6000	68.6	82.4	90.2

我們可發現 window size 假如設得愈大，辨識的結果也就愈好，但是，相對的所使用的辨識時間，也就跟著愈長，我們可發現上列 window 設為 5000 時，我們能得到適當的辨識結果（和 HTK 的結果相差在 2%以內）。雖然，window 再開更大，能得到更好的辨識結果，但辨識率成長的速度很慢，而且所耗費的運算時間約為 5000 時的 1.5 倍。故我們的辨識系統，以 5000 做為 default 值。

我們以上表做圖來確定辨識長成率。如圖 6.1。

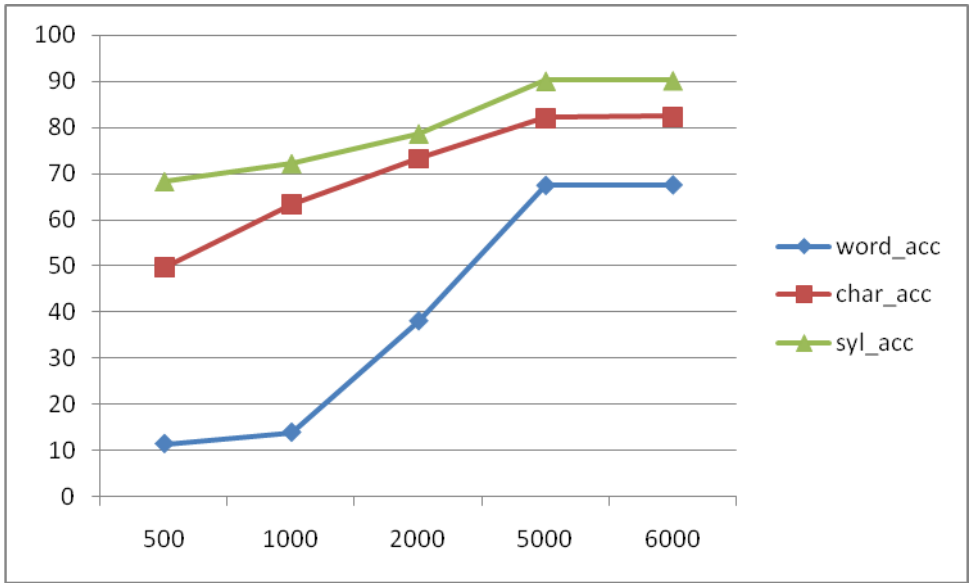

圖 6.1 Window size 對辨識率的分析圖

我們做了 window size 大小對辨識率的分析後，我們再針對 language model weight 對辨識率大小的分析做下列的實驗，我們期待得到 language model weight 對辨識率影響的極致，由於 Language Model 和 HMM Model 兩者對於辨識結果都會有所影響，但相對地，若一直增加其中一方的權重，則辨識結果，必然會傾向那一方，因此，我們必須找到最恰當的權重比，來使我們的系統得到最佳的辨識結果。而 window size 方面，由於我們已做了前列的實驗，因此在此實驗中，我們將固定 window 的大小，將大小選定為 5000。其實驗辨識結果如表 6.7 所示：

表 6.7 Language model weight V.S. 辨識率

Window_s	weight_lm(比值)	word_acc	char_acc	syl_acc
5000	0.5	36.08	61.88	72.34
	0.8	37.41	62.05	73.66
	1	37.88	62.35	75.93
	4	56.38	72.43	87.36
	5	67.31	79.47	88.25
	6	66.52	79.51	89.13
	7	67.01	80.42	89.52
	8	67.51	82.11	90.11
	9	68.57	82.15	90.14
	10	68.46	82.03	90.08

由上表，我們可以發現，此系統最佳的辨識率，在於語言模型和聲學模型兩者分數，所給予的權重比值，在 9 左右時，我們可以得到最佳的辨識結果。為了方便分析，我們可以用上表作圖，來求取比值增加和辨識率之間影響的變化，我們可以發現在比值 5 以上辨識率上升的速度就非常緩慢了；然而將此比值調大，並不會影響辨識時所需要的資源和時間，因此，我們儘量將最好的比值求出，來獲得我們最好的辨識結果。作圖結果如圖 6.2

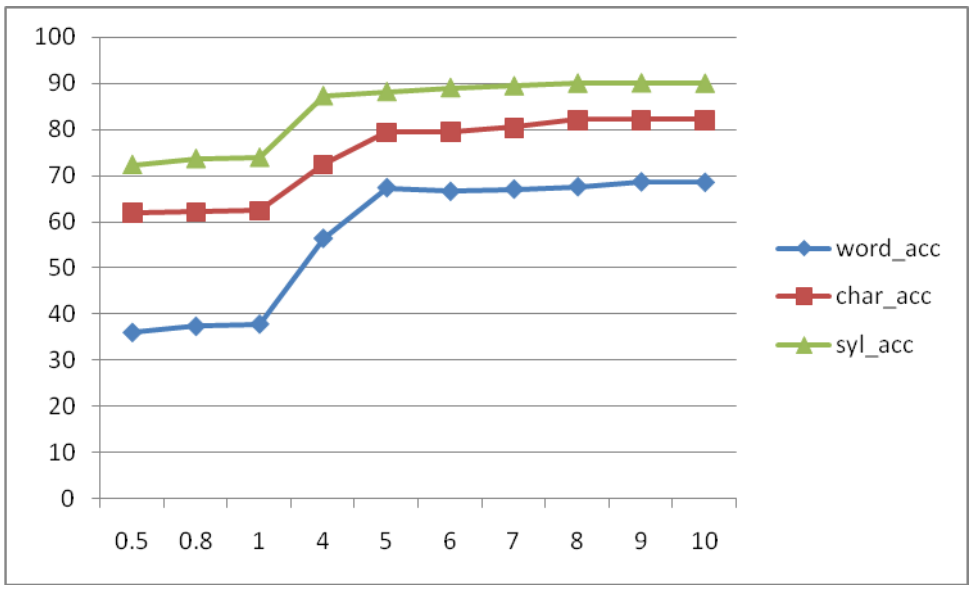

圖 6.2 LM-weight 對辨識率的分析圖

第三部份：

由於 TCC300 是由三個不同校區所錄製的語音檔案所組成，因此我們分別將三個部份的音檔，各取二十五個來做測試，實驗環境則是以前述最好的實驗結果來加以設定，window_s = 5000、LM_weight = 9，測試結果如表 6.8 所示：

表 6.8 TCC300 三部份的辨識率

正確率(%)	Word_acc	char_acc	syl_acc
NCKU 部份	65.27	72.35	83.48
NCTU 部份	47.01	67.26	82.96
NTU 部份	91.25	93.26	95.72

我們可以發現，三個部份的音檔由於音長不同以及錄製環境不同，而造成辨識率的差異，此辨識度和本系統的設立無直接關係，若要調整辨識度，須由測試音檔的錄製環境和錄音實效來改進，或許可以平均出更好的結果。

第七章 未來展望

在本篇論文的中，我們雖以實作大詞彙中文語音辨識系統為主，然而，此文所撰寫系統主要的功能，是以提供一個可以簡單整合各搜尋空間的平台，以及實現這個辨識的系統。

在實作中，可以輕易的發現系統在操作時，所會遇到的問題和瓶頸。而莫氏在其有關有限狀態機的著作方面，尚有許許多多的改良方式。未來系統若有機會加以發展，則將以優化有限狀態機為目標，不論是加權值的推移、或是最小化有限狀態機等，甚至於增加不同層級的語音資料加以整合，都是可以提高有限狀態機效果的方向。以這些為改良目標之後，相信一定能為系統的辨識速度和辨識率提供不少的進步空間。

參考文獻

[1] Ronald M. Kaplan and Martin Kay. Regular models of phonological rule systems. *Computational Linguistics*, 20(3):331-378, 1994.

[2] Mehryar Mohri. Compact representations by finite-state transducers. *In Proceedings of the 32nd conference on Association for computational linguistics*, pages 204-209. Association for Computational Linguistics, 1994.

[3] Mehryar Mohri. *Weighted Automata in Text and Speech Processing*. AT&T Labs, 1996.

[4] Mehryar Mohri. *Full Expansion of Context-dependent Networks in Large Vocabulary speech Recognition.* AT&T Labs ,1998

[5] 余家興, *以有限狀態機辨認大字彙連續中文語音*，國立台灣大學，碩士論文，2004

[6] Mehryar Mohri. *Integrated Context-Dependent Networks in Very Large Vocabulary Speech recognition*, AT&T Labs, 1999

[7] Mehryar Mohri. *Weighted Finite State Transducers in Speech Recognition*. AT&T Labs, 2002.

[8] Cyril Allauzen and Mehryar Mohri. *A GENERALIZED CONSTRUCTION OF INTEGRATED SPEECH RECOGNITION TRANSDUCERS.* AT&T Labs. 2004

[9] Mehryar Mohri. *Finite-State Transducers in Language and Speech Processing, AT&T Labs, 2003*

[10] AT&T Labs Research. http://www.research.att.com/index.cfm

[11] Lawrence Rabiner and Biing hwang Juang. *Fundamentals of Speech Recongition. Prentice Hall.* 1933.

[12] Daniel Jurafsky and James H. Martin. *SPEECH and LANGUAGE PROCESSING,* 2008.