電腦圍棋佈局系統之設計與製作

Design and Implementation of an Opening Game System for Computer Go Programs

顏士淨
Yan Shi-Jim
國立台灣大學資訊工程研究所
Department of Computer Science and Information Engineering of National Taiwan University
d2506016@cc.ntu.edu.tw

許舜欽
Hsu Shun-Chin
國立台灣大學資訊工程研究所
Department of Computer Science and Information Engineering of National Taiwan University
schsu@csie.ntu.edu.tw

摘 要

本文詳細描述一個包含佔角、定石、拆邊和模樣處理等四個部分的電腦圍棋佈局系統。這個佈局系統已運用在電腦圍棋程式Jimmy 4.0，經過棋力鑑定以及高段棋士的測試之後，我們估計這個系統在佈局方面的表現約有六級左右的棋力。

關鍵字：電腦圍棋、佈局、電腦圍棋程式

Abstract

This paper describes on a computer Go opening game system. This system has been used in a computer Go program-Jimmy 4.0. Having been tested by professional Go players, this system is estimated at about 6 kyu in terms of its opening game performance.

Keywords: Go, computer go, opening game.

一、引言

電腦對局是人工智慧領域之中相當引人注目的一門學問，在電腦西洋棋方面，自從 1950 年 Shannon 提出基本理論之後，歷經四十多年的研究，如今棋力已經達到國際特級大師(international grand master) 的水準。在1997年，電腦西洋棋程式深藍在擊敗世界西洋棋冠軍棋手之後，更將電腦西洋棋的棋力提升到令人難以想像的境界。而在研究過程中，也開發出許多知名的搜尋(search)或切捨(cut-off)方面的演算法[Frey, 1980] [Allis *et al.*, 1991]。

相對於西洋棋，電腦圍棋的進展卻是相當緩慢，1962年H. Remus首先利用圍棋來研究機器學習[Remus, 1962]。此後，威斯康辛大學的Albert L. Zobrist、史丹佛大學的Jonathan Ryder以及密西根大學的Walter Reitman等人也分別開始研究電腦圍棋[Zobrist, 1970][Ryder, 1972] [Reitman and Wilcox, 1978]。

目前世界上有許多公開的電腦圍棋比賽，如應氏杯、FOST杯和奧林匹亞杯等。這些比賽都直接或間接的推廣了電腦圍棋的研究。在這些比賽中，表現最好的圍棋程式是中國廣東中山大學陳志行教授的程式，此程式約有八、九級棋力，是目前為止公認世界最強的電腦圍棋程式[顏、許 1997] [Fotland, 1996]。

二、電腦圍棋的基本原理

運用電腦來下圍棋，似乎是一個很直接的想法，因為圍棋的規則很簡單，勝負定義也很明確，這些都和電腦本身的特性相符合。另一方面由於它已被古今中外的許多專家研究了數千年，許許多多的戰術觀念及思考方法已被開發出來，這些幾乎顛撲不破的真理，都是可以在發展電腦圍棋時去應用或參考的。

但是電腦圍棋的發展過程，卻沒有想像中順利，雖然圍棋規則很簡單，但是由於盤面廣大（一般的對局棋盤是 19 × 19 ），對局時的變化卻比其它的棋戲複雜得多。例如西洋棋或象棋，已能藉由一些簡單的推理與深度的搜尋思考而達到相當高的棋力，但這種方法卻不太適合應用在圍棋這種高複雜度的棋戲中。A.Samuel 估計 checker 的複雜度大約是 10 的 40 次方[Samuel, 1959]，而 A.Newll 估計西洋棋的複雜度大約是 10 的 120 次方[Newell *et al.*, 1958]。這兩種棋戲的複雜度雖然已是天文數字，但比起圍棋的複雜度則要小得多了，Brown 及 Dowsey 估計圍棋所有可能的變化大約是 10 的 700 次方[Brown and Dowsey, 1981]。

由於圍棋的複雜度太高，如果僅用窮舉搜尋的技巧，並不能得到我們要的結果，因此必需發展其它策略來製作電腦圍棋程式。直觀上來說，最直接的方式，就是讓電腦去模擬人類下棋的思考方式。而就人們下棋思考方向而言，選擇著點時大都根據該點是否利於佔地、是否利於攻防、是否有關死活等，這也是現今的電腦圍棋程式最常用的方法[Hsu and Liu, 1991][Hsu *et al.*, 1994][Hwang and Hsu, 1994]。

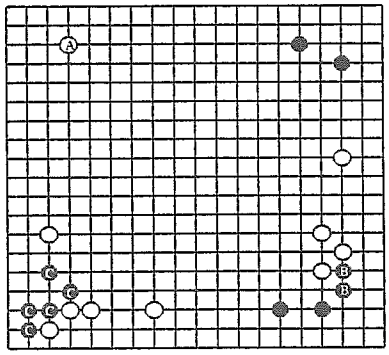
三、基本資料結構

在圍棋中，盤面上的狀況爲目前雙方落子的情形。而棋盤上可有棋子、棋串、棋塊、影響力評估值四種結構，以下依序介紹。

(1) *棋子*：棋子是最基本的結構。圖二左上角標示爲A的黑子就是一個例子，其特質包含顏色及位置。棋子並不必使用單獨變數來表示，而是存在於圍棋的陣列結構中，利用陣列元素表示此點上是否有黑子或白子。

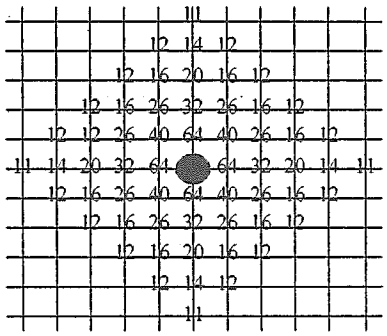
(2) *棋串*：棋串由同色且互相連接的棋子所構成。圖二右下角標示爲B而互相連接的黑子就是一個棋串，其特質包含氣數及氣點等資訊。當兩個以上的棋串互相連接時，即合併爲一個棋串。每個棋串可用一個結構變數來表示。內含此棋串的顏色、氣數、第一個棋子的位置，以及棋塊狀態等資料。較複雜的結構還包含氣點及相鄰接的棋串等資訊。

(3) *棋塊*：同顏色的棋串以某種關係位置互相接近但未連接，而且不易被對方切斷時，這些棋串便構成棋塊。圖二左下角標示爲C的五顆黑子就形成一個棋塊。棋塊是圍棋戰術運用上最基本的結構。大部分的圍棋戰術就建立在棋塊的攻防上。在佈局時，尤其是拆邊要點的選擇，必須考慮如何拆邊圍地、防守自己的棋塊，或是考慮如何打入對方的地，予以破壞，並攻擊對方的棋塊。因此我們的佈局系統必須有正確的棋塊資訊，方可正確運作。每個棋塊可用一個結構變數來表示。因爲棋塊的資訊直接被戰術系統所使用，所以結構較爲複雜且不易計算。其基本的結構包括此棋塊的顏色，內含棋串數，棋塊圍取的地域點數及位置、附近的其它棋塊、勢力的影響、棋塊安危程度、棋塊本身的價值以及棋塊其它的狀態等資訊。



圖一 棋子、棋串與棋塊

(4) *影響力評估值*：棋盤上的每一顆棋子，對其周圍的空點及棋子，都會產生相當程度的影響力。距離越近，影響力越大。距離越遠，影響力越小。爲求程式可以方便運用起見，我們將影響力量化爲數字(影響力評估值)。圖三爲一顆黑子對其周圍的影響力評估值。棋盤上所有棋子所成的影響力評估值的做法大致爲所有黑子所成的影響力減去所有白子所成的影響力。但最正確的影響力評估值還需考慮到棋形、棋塊的死活等。詳細情形可參考[Hwang and Hsu, 1994]。



圖二 一顆黑子對其周圍的影響力評估值

以上的資料結構中的資訊是在佈局系統運作之前，先由形勢判斷系統計算而得。我們的形勢判斷系統的輸入爲盤面上所有棋子的位置，輸出則爲上述之棋子、棋串、棋塊和影響力評估值資料結構中的資訊。在我們的佈局系統中最常用到的資訊爲棋塊圍取的地域點數及位置、棋塊安危程度、棋塊本身的價值以及影響力評估值。形勢判斷系統所提供的資訊，對佈局系統的好壞有著絕對的影響，詳細的形勢判斷系統運作情形可參考"電腦圍棋的形勢判斷系統"一文[Hwang and Hsu, 1994]。

四、佈局系統的設計

一局圍棋的對局可分爲三個階段：佈局、中盤、終盤。佈局主要是架構一盤棋的骨幹，所以佈局的能力可說是電腦圍棋程式棋力強弱的關鍵之一。爲了設計一個棋力更強的電腦圍棋佈局系統，我們根據專家棋士的理論再將佈局階段細分爲四個部分：佔角、定石、拆邊和模樣[大竹 1992][武宮 1984]。一般棋局的進行大約都是先佔角、再進行角上的定石、再拆邊、最後考慮模樣的著點。當然此順序並不是絕對的，必須視實際棋局狀況而定。

以往一般電腦圍棋程式佈局的方法大約都是以下列兩種方式完成，首先在開局的前幾手時，因爲此時棋盤上只有幾顆棋子，變化並不大，故大都是以棋形(定石)直接比對的方式來完成。此方法類似於人類棋手依據定石來下棋的方法。但因爲圍棋的複雜度很高，通常資料庫中的資料很快就會用完。此時，有些程式的做法是直接進入中盤，並無特定的模組處理佈局。另外一些較好的程式的做法如下：首先在棋盤上辨識出邊上較大的空白段落，然後根據兩端棋子的高低位置及顏色，決定下在此空白段落中間的第三線或第四線。以上兩種方式雖然略嫌粗糙，但相對於當時圍棋程式的棋力，在對

局時還勉強可以應付[Hsu *et al.*, 1994] [Hsu and Liu, 1991]。

近幾年來由於電腦圍棋的中盤攻殺、形勢判斷和棋形辨認方面的技術已有所進步[Chen, 1989] [Hwang and Hsu, 1994] [Lorentz, 1995]。相對以往較粗糙的佈局系統已不敷所需。另一方面，由於這些技術的進步，也使得我們得以藉此開發更新、棋力更高的佈局系統。

我們的做法是首先找出有效的合法候選棋步，然後一一評估(量化)這些候選棋步的著手價值，最後再根據量化所得的值來挑選所要的棋步。這種做法與以前圍棋程式不同之處約有下列三點：

(1)以前程式選擇著手的主要依據是程序導向，各個程序的優先順序是固定的，當某順位大的程序(例如定石程序)算出某個值時，則以此值爲所求。而我們的做法則是先執行所有的程序，找出有效的合法候選棋步，並根據各個候選棋步的特性去量化各候選棋步的價值之後，再從中選擇最好的著手。

(2) 更有效率地找出有效的合法候選棋步。由於我們已發展出良好的形勢判斷和棋形辨認程序，這些程序可以幫助系統更有效率的找出有效的合法候選棋步。

(3) 更正確地量化候選棋步的價值。除了良好的形勢判斷程序，我們還利用搜尋技巧以及擷取專家知識的方式來幫助量化候選棋步。

根據電腦圍棋的需求，我們將佈局系統細分爲佔角、定石、拆邊、及有關模樣的處理等四個部分。在以下的篇幅中，將詳述如何根據各部分的特有的手法以及棋型、戰略等等細節，一一分別去研究及設計程式。在每一部份中我們除了研究如何發現好的候選棋步之外，另外我們並根據專家棋士的理論，設法量化每一個棋步的著手價值。再將每一個棋步的量化值，加以整合，使得我們的電腦圍棋程式可以正確地運用。以下將詳細描述我們的系統運作情形。

4.1 棋步產生的基本原理

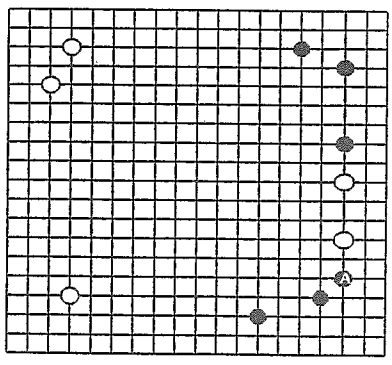
要找出合法的候選棋步很容易，但是爲了能夠有效評估候選棋步的好壞，我們希望在此階段所產生的候選棋步數量盡量減少，但是在候選棋步中至少必須包括正確的棋步。也就是說，希望無效的候選棋步越少越好。如此，一方面可以避免錯誤的候選棋步干擾系統評估棋步，另一方面，也可讓系統快速而有效的評估候選棋步。一般來說，我們的候選棋步數目大約是十五到二十五個左右。

4.2 量化棋步價值的基本原理

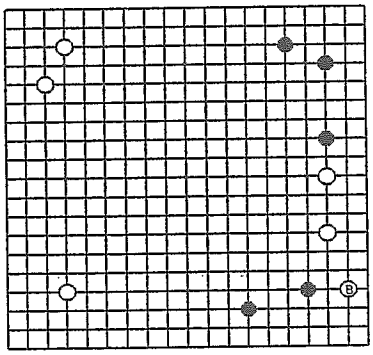
量化棋步價值系統的主要構想是將以往處理終盤的觀念應用在佈局方面。在棋局進入終盤時，由於合法步變得很少，形勢判斷也比較容易，故電腦圍棋程式可以運用組合對局理論(combinatorial game theory)來判斷最佳的著手，而且可以達到相當好的效果[Müller, 1995]。但事實上，我們發現，數學家們所開發出來的組合對局理論思考方式和人類棋手某些處理終盤(官子)的思考方式竟然非常類似。近年來許多圍棋專家棋士如大竹英雄及依田記基等[1992 大竹]，也都嘗試將這些思考方式(官子理論)更進一步運用在處理佈局方面，而且有一些不錯的成果。而由於我們在電腦圍棋的棋塊分析、形勢判斷和棋形辨認方面的技術已有所進步，因此我們可以參考這些思考方式，開發新的佈局系統。

圍棋佈局時期量化棋步的價值，可說是一局棋中最困難的部分。在此我們提出兩個辦法來解決此問題，首先必須將一些常見的棋形預先存在資料庫中。再將高段棋士對於此棋形的評估結果量化爲一整數值，此整數值代表的意義就是下在此位置可得的目數。我們用目數來評估著手的大小，其優點是高段棋士容易理解，而易於將他們的專家知識輸入資料庫，達到擷取專家知識的目的。這也是一般電腦圍棋程式所少見的做法。

但當遇到資料庫中沒有的棋形時，我們就必須採用搜尋的方式去量化棋步的價值。此方法主要是將官子理論運用在佈局階段。由於佈局階段著手的選擇的特點是從大而小，故通常只要搜尋一個節點就可得到相當正確的評估值。我們的做法如下：當要評估某一著手m時，首先假設己方在此下一手，由形勢判斷系統可得己方下此一著手後，己方目數增加a。再回到原來盤面，假設對方在著手m周圍對方的最佳著手下一手，由形勢判斷系統可得對方下此一著手後，對方目數增加b，則$a-b$即爲著手m的評估值。近年來已有一些專家棋士也用此方法來計算一手棋的大小[大竹 1992]，證明此方法顯然相當有效而正確。圖三和圖四爲一個量化棋步價值的例子，在圖三中，黑A尖後，黑棋右邊一帶有40目的優勢(黑棋約44目，白棋4目)。圖四中，白B小馬步飛後，黑棋右邊一帶有20目的優勢(黑棋約30目，白棋10目)。所以我們可根據官子理論判斷出黑A尖及白B小馬步飛各有20目的價值。



圖三 計算量化值的例子(一)



圖四 計算量化值的例子(二)

4.3 佔角棋步的產生及量化

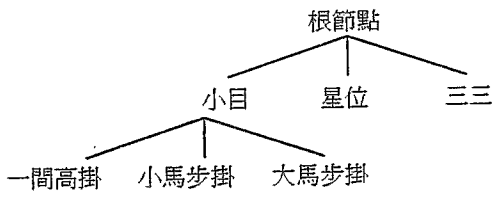
由於開局佔角時的前幾手，棋盤上的變化有限，我們在此階段採用棋形比對的方式產生候選棋步。佔角的棋形大致可分爲星位、三三、小目、高目、目外等等[林 1984] 。

由於佔角前後，角上的狀況只由沒有棋子到多了一顆棋子，變化並不多。所以我們採用棋形比對的方式去量化此棋步。我們將佔角棋步的量化值儲存在棋形上，在產生候選棋步時，系統也同時知道此棋步的大小。而一個佔角棋形的量化評估值早已被許多專家棋士討論過，只要去參考專家棋士的討論結果，就可以得到所有佔角棋步的量化評估值，一般佔角的量化評估值約爲20目[大竹1992]。

4.4 定石棋步的產生及量化

定石是角上，或有時及於邊上變化的一連串的模範下法[石田 1982]。由於定石可表示爲一連串的下子順序(手順)，故我們可將定石之著手順序建成樹狀的資料結構(定石樹)，如圖五所示。在實戰對局剛開局時，每一個空角皆有其相對應的指標指向這顆定石樹的根節點(root node)。當有一顆棋子下在一個空角上時，其相對應的指標亦隨之而改變指向的節點。我們可由搜尋相對應的指標指向的節點的子節點知道下一棋步的位置，這些子節點的位置就可成爲我們的定石候選棋步。

由於定石的手順幾乎是固定的，手順中每一著手的大小，在一些有關定石的棋書中，也多有討論[林 1984][石田 1982]。所以我們也可參考現有的資料，在輸入定石的同時，也輸入每個定石棋步的大小目數。如此，在搜尋到此節點的同時，也就得到了相關棋步的量化值。



圖五 一棵定石樹

4.5 拆邊棋步的產生及量化

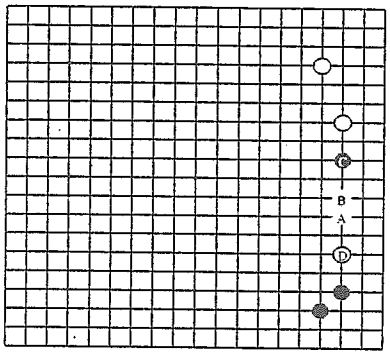
當雙方棋子佔據各個角落，定石告一段落時，接著就要進行拆邊的工作。在拆邊棋步的產生方面，我們有兩種做法。對於盤面已存在的棋子周圍附近的空點，我們以棋形比對的方式產生候選棋步。例如拆邊基本觀念中的立二拆三，立三拆四等[藤澤 1981]。而當某一邊上只有兩邊角落有棋子時，因爲此邊上的可著點很多，且範圍很大，棋形比對的方式顯然不太適用。此時可根據圍棋佈局觀念中的"割分"法來產生候選棋步。"割分"法的基本概念如下:當某一邊上只有兩邊角落有棋子時，則與兩端角落的棋子距離相等的邊上空點通常就是好點[藤澤 1981]。我們的系統根據此方法，首先找出此一邊上割分點，再根據兩端角落棋子的位置和顏色決定要下在第三線或第四線。

在拆邊棋步的價值量化方面，由於一般來說，棋局到了要考慮拆邊時，通常棋盤上已有相當多的棋子，各棋塊亦已有了強弱的分別。所以我們在考慮每一個拆邊的棋步的大小時，除了要考慮雙方佔地目數的出入之外，也要考慮此一棋步對雙方棋塊的影響[大竹 1992]。實際量化的值就是考慮上述兩項所得之利益的值的總和。以下我們就分別敘述如何計算這兩個值。

在計算佔地目數的出入方面，我們根據前述的官子理論的方法去搜尋候選棋步，再配合一個相當精準的形勢判斷系統，就可計算出佔地目數的出入。在搜尋時，有兩點是我們所必須注意的。首先是如何找出對方在此候選棋步周圍的對方最佳著手。我們的做法是，將這個資訊預先儲存在棋形資料庫中，當系統以棋形比對的方式找出某一候選棋步時，同時也可從這個棋形資料中知道對方在此候選棋步周圍的對方最佳著手。第二點則是所謂佔地目數，其實也必須包括對周圍空點的影響(圍棋術語稱爲厚勢或薄味)，在我們的系統中，可以根據影響力評估值的增減，來計算此部份的值。圖六是一個量化拆邊棋步的例子；當黑棋下在A點時，黑棋可圍到4目的空並增加黑棋約2目的影響力。而白棋下在B點時，白棋也可圍到4目的空並增加白棋約2目的影響力。所以在這裡計算佔地目數的出入方面，黑棋下在A點的價值爲12目(4+2+4+2) 。

在對雙方棋塊的影響方面，我們首先根據形勢判斷系統找出雙方的不安全的棋塊，因爲拆邊時，所佔的目數一定會增加，所以拆邊棋步對於支援或攻擊雙方的不安全的棋塊必定會有一定的效果。所以當某一候選棋步落在雙方的不安全的棋塊附近時，我們就再將其對周圍的不安全的棋塊的影響程度加入此候選棋步的量化值。例如在圖六中，棋塊C的價值爲10目左右，棋塊D的價值也在10目左右。在此例子中，由於棋塊C或棋塊D在被對方攻擊時，對方都必須在其周圍連下兩手，才可吃掉這棋塊。所以在考慮對雙方棋塊的影響方面，黑棋下在A點的價值大約是10目((10+10)/2)的價值，加

上先前算出的佔地目數的出入方面的12目，此拆邊著手的量化值爲22目。



圖六 量化拆邊棋步的例子

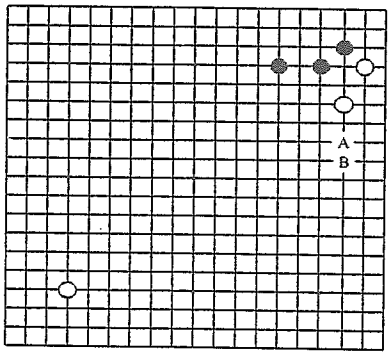
4.6 模樣棋步的產生及量化

在模樣棋步的產生方面，我們是以棋形比對的方式產生候選棋步。主要的模樣棋形有一間跳、二間跳、小馬步飛、大馬步飛等等。模樣的棋步大多是產生在棋盤的中央，且通常最重要的模樣棋步是介於黑棋及白棋的交界(即圍棋中俗稱的"天王山")。我們可根據以上幾項圍棋的棋理，去找出模樣棋步的候選棋步。

而在模樣棋步的量化方面，與量化拆邊棋步的不同點是我們只考慮計算佔地目數的出入。至於在對雙方棋塊的影響方面，由於模樣棋步對於雙方棋塊的安危通常並沒有絕對的影響，因此我們並不考慮。而佔地目數的出入的計算方式與拆邊棋步的計算方式大致相同，都是要考慮雙方棋塊本身地域的增減以及影響力評估值的增減來計算量化的值。

4.7 子系統的整合

我們的佈局系統首先根據以上四個子系統找出有效的合法候選棋步，再一一分別評估(量化)這些候選棋步，最後再根據量化所得的值來挑選所要的棋步。在此要注意的一點是我們在挑選所要的棋步時，只有考慮各候選棋步量化所得的值，至於此候選棋步是佔角、定石、拆邊或模樣等等，在挑選的過程中並不考慮。例如圖七中，白棋未完成右上定石，就手拔轉佔左下角，此時黑棋雖然還有角可佔，但因爲A點可威脅白棋的棋塊(白棋棋塊的價值爲32)，且對白棋而言，在此處拆邊(下在B點)也有10目的利益(佔地7目，增加勢力3目)。而黑棋下A點時，在在此處圍到的地和勢力則爲3目(佔地0目，增加勢力3目)，綜合以上三點，A點的量化值爲29目((32/2)+10+3)，高於佔角的值(20目)，故我們選擇A點。由於不是以往圍棋程式死板的佔角、定石、拆邊的進行順序，所以我們的佈局系統較能靈活地選擇著點，因而擁有較高的棋力。



圖七　　一個選擇著點的例子

五、結果與評估

我們在半年前實作出以上所敘述的理論，並將這佈局系統實際的應用在電腦圍棋程式Jimmy4.0上。爲求正確得知此程式佈局系統的效能，我們採用一般棋書常見的做測驗題的方式來測驗棋力[張1984]。答題方式類似一般考試的單選題，通常每一題有五個左右的答案，而每個答案的分數在0到10之間，做完所有的題目之後，計算出每題所選答案的分數總和，最後再到總分-棋力對照表中即可查出對應的棋力。我們參考一些棋書[林 1985][張 1984][陳 1992]，挑選出約五十題左右的佈局階段的題目，再一一讓本系統計算。測試結果本系統所得的分數約在四級左右。

相對於世界目前最強的程式八、九級的棋力，以上的結果相當令人滿意。但我們發現在棋局進入佈局後段與中盤之初時，此程式的棋力卻明顯下降。經過探討之後，發現其原因之一是因爲程式是以量化值的大小爲選擇著手的依歸，而此佈局系統的量化值目前並不夠精確，導致整個程式在無法正確決定此時要下佈局要點或其他中盤要點。因此，除了以上的測驗題式的分析之外，還需考慮此佈局系統對整個程式的配合度。也就是說，必須有更精準的量化值以供程式判斷。因此，我們也進行評估著手的量化值是否精準的工作，以求改進此佈局系統。我們評估的方式爲比較高段棋士與此佈局系統對一些相同棋局的看法：先由高段棋士計算出某些著手的大小，再將系統的對這些著手的量化值與之作比較，計算其誤差率(誤差/正確值)。各個子系統量化值平均誤差率結果如表八。

表八 佈局系統的量化值平均誤差率

	佔角	定石	拆邊	模樣
平均誤差率	8%	15%	32%	47%

由上表可以看出佔角及定石的量化值的誤差較小。這主要是因爲其著手的產生及量化皆是直接由棋形資料庫而來，而棋形資料庫正是由高段棋士

所建立的。因此，如果棋形資料庫愈完備，則其效果愈佳，棋力也越接近高段棋士。而拆邊及模樣的誤差相對的就大多了。其原因主要是因爲這兩者的著手產生及量化都必須用到形勢判斷系統所提供的棋塊和影響力評估值資訊。也就是說，其發展瓶頸是在形勢判斷系統。而我們系統的形勢判斷部分目前的棋力約在六級左右。經由以上的討論並根據表九的結果，我們可以大致判斷此佈局系統的棋力約在六級左右。

六、結論及未來展望

在本篇文章中，我們針對佈局階段，提出了一個有效解決這個問題的方法。我們首先將佈局系統細分爲佔角、定石、拆邊、及有關模樣的處理等四個部分，再根據各部分的特有的手筋以及棋型、戰略等等細節，一一分別研究及設計程式，在每一部份中我們除了研究如何發現好的棋步之外，並根據專家棋士的理論，設法量化每一個棋步。從每一個棋步的量化值加以整合，使得我們的電腦圍棋程式可以正確地整合及運用此系統。這個電腦圍棋佈局系統已經被成功的運用在電腦圍棋程式Jimmy 4.0，經過高段棋士的測試之後，我們估計這個佈局系統的棋力約有六級左右。

從第五章的結論中，可以知道雖然此佈局系統在佈局方面的表現尚稱滿意。但若要搭配其他中盤的系統來運作，卻仍有不足之處。其原因是在於每一棋步的量化值仍不夠精確的關係，這也是我們未來努力的方向。

參考文獻

1. [林 1984] 林海峰九段，定石、佈局實用入門，大孚書局，1984。
2. [林 1985] 林海峰九段，初段棋力的測試，大孚書局，1985。
3. [大竹 1992] 大竹英雄九段，大竹英雄圍棋系列4－圍棋大局觀，6至17頁，理藝出版社，1992。
4. [石田 1982]石田芳夫九段，定石辭典，世界文物出版社，1982。
5. [武宮 1984] 武宮正樹九段，初級佈局，世界文物出版社，1984。
6. [許 1989] 許舜欽，電腦圍棋在台灣的回顧與前瞻，中國工程師學會，日本分會，*1989年學術研討會論文集*，1989。
7. [張 1984] 張續儒，圍棋段級測驗第一集，圍棋段級測驗第二集，王家出版社，1984。
8. [陳 1992] 陳憲輝，圍棋實力測驗系列，理藝出版社，1992。
9. [顏、許 1997] 顏士淨、許舜欽，電腦圍棋的發展概況，*Communications of IICM*，Vol. 1, NO. 2, April, 1997, pages 23 -- 30。
10. [藤澤 1981] 藤澤秀行，序盤下法，世界文物出版社，1981。
11. [Allis *et al.*, 1991] L.V. Allis, Van Den Herik, and H.J. Herschberg. *Heuristic Programming in Artificial Intelligence 2*, Ellis Horwood 1991.
12. [Brown and Dowsey, 1981] D.J. H. Brown and Dowsey, S. The challenge of Go. New Scientist, 1981, pages 303--305.
13. [Chen, 1989] Ken Chen. Group identification in Computer Go. *Heuristic Programming in Artificial Intelligence*, Levy & Beal (Eds.), Ellis Horwood 1989, pages 195--210.
14. [Fotland, 1996] David Fotland. World Computer Go Championships, WWW. page, http://www.mth.kcl.ac.uk/~mreiss/ bill/comp/.
15. [Hsu *et al.*, 1994] S.C. Hsu, J.C. Yan, and H. Chang. Design and implementation of a computer Go program Archimage 1.1. *Journal of Information Science and Engineering* 10, pages 239--258, 1994.
16. [Hsu and Liu, 1991] S.C. Hsu and D.Y. Liu. The design and construction of the computer Go program Dragon 2. *Computer Go*, No. 16, pages 3--14, 1991.
17. [Hwang and Hsu, 1994] Y.J. Hwang and S.C. Hsu. Design and implementation of a position judgment system for computer Go programs. *Bulletin of the College of Engineering, N.T.U.*, No. 62, pages 21--33, Oct. 1994.
18. [Lorentz, 1995] Richard J. Lorentz. Pattern matching in a Go Playing Program. *Game programming workshop in Japan*, pages 167--174, 1995.
19. [Newell *et al.*, 1958] A. Newell A., J.C. Shaw, and H.A. Simon. Chess playing programs and the problem of complexity. *IBM Journal of Research and Development*, Vol. 4, No. 2. Pages 320--335, 1958.
20. [Remus, 1962] Remus, H., "Simulation of a Learning Machine for Playing Go", Proc. IFIP Congress 62, Munich, North Holland Publishing Company, Amsterdam, 1962, pages 192—194.
21. [Reitman and Wilcox, 1978] Walter Reitman and Bruce Wilcox. Pattern recognition and pattern-directed inference in a program for playing Go. *Pattern-Directed Inference Systems*, pages 503--523, 1978.
22. [Ryder, 1972] Ryder, J. L.,"Heuristic Analysis of large Tree as Generated in the Game of Go", AIM-271, Stanford Univ., 1972.
23. [Zobrist, 1970] Zobrist, A. L. Feature Extraction and Representation for Pattern Recognition and the Game of Go, Ph.D. Dissertation, University of Wisconsin, 1970.