國立中正大學應用數學研究所

碩士論文

使用最佳化方法解五子棋問題

An Optimization Method for Gobang Game

指導教授：紀美秀博士

研究生：林澤榮撰

中華民國 九十八年 八月

謝辭

本篇論文能夠完成，首先要感謝我的指導教授 紀美秀老師對我的耐心指導。因為有老師仔細的教學，讓我對最佳化以及數學的領域有更深一層的認識；每當我陷入迷思，都是老師一語道破我的困惑。老師不僅在教學上給我很大的幫助，在人生上也是我的導師，這兩年下來，老師讓我對自己的人生有很多的反省與檢討，對此，我真的充滿感激。另外感謝口試委員 賴玉玲老師與 陳昇國老師抽空參加我的口試，並且在口試時給了我許多珍貴的意見讓我又有很多的想法與檢討。

我也要感謝從大學以來就一直照顧我，上研究所後依然陪我奮鬥的冠祥、俊傑、昇晏、佳成、智禾、歐陽達和耀群，有你們的幫助，我才能走到這；感謝我在研究所認識的一群好朋友彥豪、添瑀、哲安、育助、建緯、昭偉...等等，你們豐富了我這兩年的生活。另外我特別要感謝從小就與我相識，也一直在程式上給我許多指導的家瑋，沒有你我無法完成這個程式，對此我非常感激你不厭其煩的教導我寫程式，解決我的問題。

最該感謝的，還是爸媽對我的鼓勵與支持，使我在讀書這些日子沒有後顧之憂。我對你們的感激之心，溢於言表，希望我能在往後的日子，彌補我過去年少無知讓你們傷心難過的悔憾。

在中正這六年來，我經歷了許多大小事情，對於中正這塊地方的人事物，充滿了許多不捨，我也以能身為中正數學的一份子感到光榮。而我即將離開學校的保護傘，投入社會這個新環境，希望我能把在中正所學的一切，貢獻於社會。

在此，我將本篇論文獻給我的父母、師長、同學以及朋友們，你們都是這篇論文的推手，也是我人生的推手。

目錄

摘要

本篇論文旨在用賽局理論的觀念，以及使用最佳化的方法提供一個五子棋遊戲的 AI 設計。就賽局理論來看，五子棋是一個兩人的零和依序行動賽局。我們將使用一階導函數方法去找兩人零和賽局的納許均衡解；另外我們會探討不同的規則對勝負情況的影響；並且我們引用現今五子棋的 26 種開局棋型，探討各種開局棋型對雙方的優劣勢。

第一章 緒論

五子棋 (Gobang) 是一個規則簡單、複雜度極高，相當受歡迎的遊戲，但由於基本規則上的一些不公平，該愛好者不停的演進五子棋的規則。從最早的基本規則到日式規則，直到現在的國際規則等等，基本的玩法沒有太大的改變，但是進階的規則讓更多人愛不釋手。迄今亦有像西洋棋、圍棋一樣有國際性職業棋會與比賽，國外更有一句名言：“5 MINUTES TO LEARN, A LIFETIME TO MASTER.”。

在基本規則下，只要執黑或執白的某一方在任一方向（橫向、縱向、左斜、右斜）先連成五顆棋子即獲勝。在 1994 年， Allis 利用"迫著搜尋" (Threat Space Search) 提出了在 15 × 15 的棋盤上，存在先手不負的下法 [1] [2]。日規主要以禁手，來平衡先後手的差距，也許對初學者來說是一種障礙，但下久後會發現禁手使得雙方棋手必需更加精準的掌握棋子的落點，白棋還可使用逼禁的手段來取勝，增加了連珠的技術性、複雜性及趣味性。國際規則 (RIF規則) 是建立在日規之上，而後針對開局多了四個規則：

1. 先由假先手挑選 26 棋形當中其中一種。
2. 由假後手決定雙方何者執黑，何者執白。自此黑白確立。
3. 在黑方第五手，必須要下兩個非對稱的點，供白方挑其中一。
4. 此後規則同日式規則。

RIF 規則雖然有作改良，但由於棋手大量研究 RIF 規則的變化，使得可下的開局棋型越來越少，目前也在尋求新的出路。在本篇論文我們只研究日規以及開局棋型的影響，不討論 RIF 規則。

本篇論文重點在於用賽局理論的觀點去探討這個遊戲，以及模型化後利用最佳化的方法求解。在賽局的觀念中，五子棋是個兩人的零和依序行動

賽局，所謂的零和就是某方得到的報酬總值等於另一方失去的報酬總值，而五子棋是兩人輪流下子的，在賽局中我們稱這是依序行動賽局。我們把這想法轉換到最佳化的模型上，接著我們用一階導函數方法 [3] 去計算逼近的 均衡解。我們除了提供一個新的 AI 設計之外，也希望能用不同的觀點去做五子棋的問題。

在本篇論文的第二章，我們針對五子棋建構一個最佳化的模型以及日規的禁手。第三章介紹我們所要使用的方法以及此演算法一些細部的小問題跟解決辦法，之後會利用此演算法做一個賽局上的例子檢驗他的正確性。第四章我們會探討一些不同的規則之下的數據結果。第五章是結論以及未來可以改進的地方。

第二章 規則與定義

2.1 定義符號

首先我們把一般的五子棋棋盤視為一個 15×15 的矩陣，而矩陣中每個位置的值定義如下：

$$B_{i,j} = \begin{cases} 1 & \text{如果玩家一選擇矩陣中的某個位置 } (i,j) \\ -1 & \text{如果玩家二選擇矩陣中的某個位置 } (i,j) \\ 0 & \text{尚無任何玩家選擇的位置} \end{cases}, \ 1 \le i,j \le 15.$$

2.2 勝利條件

在基本規則下，每回合行動後，我們會檢查矩陣中的每個值是否滿足勝利條件。假設在某個位置 $(i^*,\ j^*)$ 符合下列情況，即滿足勝利條件。

$$\sum_{l=0}^{4} B_{i^*+l,j^*} = 5 \text{ 或 } -5,\ 1 \le i^* \le 11; \tag{i}$$

$$\sum_{l=0}^{4} B_{i^*,j^*+l} = 5 \text{ 或 } -5,\ 1 \le j^* \le 11; \tag{ii}$$

$$\sum_{l=0}^{4} B_{i^*+l,j^*+l} = 5 \text{ 或 } -5,\ 1 \le i^*,\ j^* \le 11; \tag{iii}$$

$$\sum_{l=0}^{4} B_{i^*-l,j^*+l} = 5 \text{ 或 } -5,\ 5 \le i^* \le 15,\ 1 \le j^* \le 11. \tag{iv}$$

如果符合上述條件且其值是 5，則玩家一（先手）獲勝且得到 1 的報酬，玩家二（後手）則是損失 1 的報酬。反之，若其值是 -5，則後手獲勝，先手損失 1 的報酬。在建構日規中的限制條件之前，我們先討論一些優勢情況。

2.3 優勢情況

我們先建構幾種五子棋中優勢的情況，所謂的優勢就是下棋在這個位置會有一定的機會獲勝，而這些優勢的設定可以幫助我們建構日規限制的模型。由於日規中的限制都是針對先手，所以在這邊我們只列出先手的優勢情況，後手的優勢情況跟先手相同。

(A1) 活三：我們把連三的情況分為直、橫和斜三個部分。令 $i, j, l \in \mathbb{N}$，如果 $B_{i,j} = 1$：

(i) 橫活三：

(1) 令 $E_1(i, j) = \{(i, j+l) : B_{i,j+l} = 1, l \in [0,2]\}$， $F_1(i, j) = \{(i, j+l) : B_{i,j+l} = 0, l \in [-1,3]\}$， $1 \le i \le 15$， $2 \le j \le 12$。如果我們找到某個位置 (i, j)使得 $|E_1| = 3$ 和 $|F_1| = 2$，即滿足 (A1)。

(2) 令 $E_2(i, j) = \{(i, j+l) : B_{i,j+l} = 1, l \in [0,3]\}$， $F_2(i, j) = \{(i, j+l) : B_{i,j+l} = 0, l \in [-1,4]\}$， $1 \le i \le 15$， $2 \le j \le 11$。如果我們找到某個位置 (i, j) 使得 $|E_2| = |F_2| = 3$，即滿足 (A1)。

(ii)直活三：

(1) 令 $E_3(i, j) = \{(i+l, j) : B_{i+l,j} = 1, l \in [0,2]\}$， $F_3(i, j) = \{(i+l, j) : B_{i+l,j} = 0, l \in [-1,3]\}$， $2 \le i \le 12$， $1 \le j \le 15$。如果我們找到某個位置 (i, j)使得 $|E_3| = 3$ 和 $|F_3| = 2$，即滿足 (A1)。

(2) 令 $E_4(i, j) = \{(i+l, j) : B_{i+l,j} = 1, l \in [0,3]\}$， $F_4(i, j) = \{(i+l, j) : B_{i+l,j} = 0, l \in [-1,4]\}$， $2 \le i \le 11$， $1 \le j \le 15$。如果我們找到某個位置 (i, j) 使得 $|E_4| = |F_4| = 3$，即滿足 (A1)。

(iii)斜活三：

(1) 令 $E_5(i,\, j) = \{(i+l,\, j+l) :\, B_{i+l,j+l} = 1,\, l \in [0,2]\}$， $F_5(i,\, j) = \{(i+l,\, j+l) :\, B_{i+l,j+l} = 0,\, l \in [-1,3]\}$， $2 \le i,\, j \le 12$。如果我們找到某個位置 $(i,\, j)$使得 $|E_5| = 3$ 和 $|F_5| = 2$，即滿足 (A1)。

(2) 令 $E_6(i,\, j) = \{(i+l,\, j+l) :\, B_{i+l,j+l} = 1,\, l \in [0,3]\}$， $F_6(i,\, j) = \{(i+l,\, j+l) :\, B_{i+l,j+l} = 0,\, l \in [-1,4]\}$， $2 \le i,\, j \le 11$。如果我們找到某個位置 $(i,\, j)$ 使得 $|E_6| = |F_6| = 3$ ， 即滿足 (A1)。

(3) 令 $E_7(i,\, j) = \{(i-l,\, j+l) :\, B_{i-l,j+l} = 1,\, l \in [0,2]\}$， $F_7(i,\, j) = \{(i-l,\, j+l) :\, B_{i-l,j+l} = 0,\, l \in [-1,3]\}$， $4 \le i \le 14$， $2 \le j \le 12$。如果我們找到某個位置 $(i,\, j)$ 使得 $|E_7| = 3$ 和 $|F_7| = 2$，即滿足 (A1)。

(4) 令 $E_8(i,\, j) = \{(i-l,\, j+l) :\, B_{i-l,j+l} = 1,\, l \in [0,3]\}$， $F_8(i,\, j) = \{(i-l,\, j+l) :\, B_{i-l,j+l} = 0,\, l \in [-1,4]\}$， $5 \le i \le 14$， $2 \le j \le 11$。如果我們找到某個位置 $(i,\, j)$ 使得 $|E_8| = |F_8| = 3$，即滿足 (A1)。

(A2) 連四(包括活四跟死四)：令 $i,\, j,\, l \in \mathbb{N}$，如果 $B_{i,\,j} = 1$：

(i) 橫連四：

(1) 令 $P_1(i,\, j) = \{(i,\, j+l) :\, B_{i,j+l} = 1,\, l \in [0,3]\}$， $Q_1(i,\, j) = \{(i,\, j+l) :\, B_{i,j+l} = 0,\, l \in [-1,4]\}$， $1 \le i \le 15$， $1 \le j \le 11$。如果我們找到某個位置 $(i,\, j)$ 使得 $|P_1| = 4$ 和 $|Q_1| \ge 1$，即滿足 (A2)。

舉例來說，如果先手選擇 (1,1)，那 Q_1 此集合的搜尋範圍就會包含 (1,0) 這個位置，在程式中我們把此超出矩陣範圍的值定為2。

(2) 令 $P_2(i,\, j) = \{(i,\, j+l) :\, B_{i,j+l} = 1,\, l \in [0,4]\}$， $Q_2(i,\, j) = \{(i,\, j+l) :\, B_{i,j+l} = 0,\, l \in [0,4]\}$， $1 \le i \le 15$， $1 \le j \le 11$。如果我們找到某個位置 $(i,\, j)$ 使得 $|P_2| = 4$ 和 $|Q_2| = 1$，即滿足 (A2)。

(ii)直連四：

(1) 令 $P_3(i,\, j) = \{(i+l,\, j) : B_{i+l,j} = 1, l \in [0,3]\}$， $Q_3(i,\, j) = \{(i+l,\, j) : B_{i+l,j} = 0, l \in [-1,4]\}$， $1 \le i \le 11$， $1 \le j \le 15$。如果我們找到某個位置 $(i,\, j)$ 使得 $|P_3| = 4$ 和 $|Q_3| \ge 1$，即滿足 (A2)。

(2) 令 $P_4(i,\, j) = \{(i+l,\, j) : B_{i+l,j} = 1, l \in [0,4]\}$， $Q_4(i,\, j) = \{(i+l,\, j) : B_{i+l,j} = 0, l \in [0,4]\}$， $1 \le i \le 11$， $1 \le j \le 15$。如果我們找到某個位置 $(i,\, j)$ 使得 $|P_4| = 4$ 和 $|Q_4| = 1$ ，即滿足 (A2)。

(iii)斜連四：

(1) 令 $P_5(i,\, j) = \{(i+l,\, j+l) : B_{i+l,j+l} = 1, l \in [0,3]\}$， $Q_5(i,\, j) = \{(i+l,\, j+l) : B_{i+l,j+l} = 0, l \in [-1,4]\}$， $1 \le i,\, j \le 11$。如果我們找到某個位置 $(i,\, j)$ 使得 $|P_5| = 4$ 和 $|Q_5| \ge 1$，即滿足 (A2)。

(2) 令 $P_6(i,\, j) = \{(i+l,\, j+l) : B_{i+l,j+l} = 1, l \in [0,4]\}$， $Q_6(i,\, j) = \{(i+l,\, j+l) : B_{i+l,j+l} = 0, l \in [0,4]\}$， $1 \le i,\, j \le 11$。如果我們找到某個位置 $(i,\, j)$ 使得 $|P_6| = 4$ 和 $|Q_6| = 1$ ， 即滿足 (A2)。

(3) 令 $P_7(i,\, j) = \{(i-l,\, j+l) : B_{i-l,j+l} = 1, l \in [0,3]\}$， $Q_7(i,\, j) = \{(i-l,\, j+l) : B_{i-l,j+l} = 0, l \in [-1,4]\}$， $4 \le i \le 15$， $1 \le j \le 11$。如果我們找到某個位置 $(i,\, j)$ 使得 $|P_7| = 4$ 和 $|Q_7| \ge 1$，即滿足 (A2)。

(4) 令 $P_8(i,\, j) = \{(i-l,\, j+l) : B_{i-l,j+l} = 1, l \in [0,4]\}$， $Q_8(i,\, j) = \{(i-l,\, j+l) : B_{i-l,j+l} = 0, l \in [0,4]\}$， $5 \le i \le 15$， $1 \le j \le 11$。如果我們找到某個位置 $(i,\, j)$ 使得 $|P_8| = 4$ 和 $|Q_8| = 1$，即滿足 (A2)。

2.4 日規

基本規則對先手的優勢太大，為了平衡兩邊的差距，我們採用日規來限制先手。以下是三個日規的定義：

(R1) 雙三： 先手下棋時，如果下棋點同時出現兩個活三（直、橫或斜任二種）的情況，則雙三成立，判先手輸。這邊要注意的是假設在第 t 回合(下一顆棋為一回合)出現一個活三的情況，在 第 k 回合出現另外一個活三， $t \neq k$，這樣並不違反禁雙三的規定。

(R2) 雙四： 跟 (R1) 相似，如果下棋點同時出現兩個連四的情況，則判先手輸。

(R3) 長連： 先手將棋子連線時有超過六子或以上的情況，稱之為長連，判先手輸，以下是我們檢驗是否滿足長連的設定。\

$$\sum_{l=0}^{5} B_{i^*+l,j^*} = 6,\ 1 \leq i^* \leq 10; \tag{1}$$

$$\sum_{l=0}^{5} B_{i^*,j^*+l} = 6,\ 1 \leq j^* \leq 10; \tag{2}$$

$$\sum_{l=0}^{5} B_{i^*+l,j^*+l} = 6,\ 1 \leq i^*,\ j^* \leq 11; \tag{3}$$

$$\sum_{l=0}^{5} B_{i^*-l,j^*+l} = 6,\ 6 \leq i^* \leq 15,\ 1 \leq j^* \leq 10. \tag{4}$$

假設在某個位置 $(i^*,\ j^*)$ 使得上列其中一個情況成立，即滿足 (R3)。

如果先手違反了禁手的規定，則算輸，且會損失 1 的報酬。其中有些特例情況，如果先手滿足勝利條件同時也違反日規的話，算先手贏 (長連除外)。另外，如果最後一回合先手違反日規也沒關係，比賽算平手，兩邊報酬皆為 0。

第三章 演算法設計

3.1 一階導函數方法

一開始我們給定兩玩家的機率分配 $x \in \triangle_m$ 跟 $y \in \triangle_n$， $\triangle_m := \{x \in \mathbb{R}^m : \sum_{i=1}^m x_i = 1,\, x_i \geq 0\}$ 代表先手在 m 個純粹策略中的混和機率分配所成的集合，同理 $\triangle_n := \{y \in \mathbb{R}^n : \sum_{i=1}^n y_i = 1,\, y_i \geq 0\}$ 代表後手的混和機率分配集合。給定一個報酬矩陣 A， A 中每個元素 $A_{i,j}$ 代表的是當先手選定一個策略 i， 後手也選定了一個策略 j 後，所對應到的報酬值。我們把此 min-max game problem 寫成

$$\max_{x\in\triangle_m}\min_{y\in\triangle_n} x^T A y = \min_{y\in\triangle_m}\max_{x\in\triangle_n} x^T A y, \tag{1}$$

問題 (1) 可以被重寫成一個非平滑的 primal-dual problem 如下式：

$$\max\{f(x) : x \in \triangle_m\} = \min\{\phi(y) : y \in \triangle_n\},$$

其中，

$$f(x) := \min\{x^T A v :\, v \in \triangle_n\},$$
$$\phi(y) := \max\{u^T A y :\, u \in \triangle_m\}.$$

我們整合上述式子，使它成為一個非平滑的 convex primal-dual problem 如下式：

$$\min\{F(x,y) :\, (x,y) \in \triangle_m \times \triangle_n\}, \tag{2}$$

其中，

$$F(x,y) = \max\{u^T A y - x^T A v :\, (u,v) \in \triangle_m \times \triangle_n\}. \tag{3}$$

注意到上面的函數 $F(x,y) = f(x) - \phi(y)$ 是凸函數且

$$\min\{F(x,y) : (x,y) \in \triangle_m \times \triangle_n\} = 0.$$

如果有一點 $(x,y) \in \triangle_m \times \triangle_n$是 (1) 式的逼近解，那此點若且唯若會使得 $F(x,y) \leq \epsilon$， $\epsilon > 0$。

因為(2)中的目標函數 $F(x,y)$ 是非平滑的，且又是凸函數，在這裡可以適用次梯度方法 [4]。如此，我們可以得到更好的下界。

令 $Opt := \text{argmin}\{F(x,y) : (x,y) \in \triangle_m \times \triangle_n\}$表所有最佳解的集合，且函數 dist:$\triangle_m \times \triangle_n \to \mathbb{R}$代表到最佳解的距離函數，如下式：

$$\text{dist}(x,y) := min\left\{\|(x,y) - (u,v)\| : (u,v) \in Opt\right\}.$$

令 $(\bar{u},\bar{v}) \in \triangle_m \times \triangle_n$ 和 $\mu > 0$，我們將 F 修正如下使 F_μ為一平滑函數：

$$F_\mu(x,y) = \max\left\{u^T Ay - x^T Av - \frac{\mu}{2}\|(u,v) - (\bar{u},\bar{v})\|^2 : (u,v) \in \triangle_m \times \triangle_n\right\}. \tag{4}$$

令 $(u(x,y), v(x,y)) \in \triangle_m \times \triangle_n$ 為 (4) 的最佳解，因為函數

$$u^T Ay - x^T Av - \frac{\mu}{2}\|(u,v) - (\bar{u},\bar{v})\|^2$$

對 $(u,v) \in \triangle_m \times \triangle_n$ 是個嚴格的凹函數，所以此為唯一的最佳解。

接著引用自 [4] 的定理一，因為 F_μ 是平滑的，所以它的偏導函數∇F_μ是 Lipschitz 連續，且 Lipschitz 常數為 $\frac{\|A\|^2}{\mu}$。∇F_μ表示如下：

$$\nabla F_\mu(x,y) = \begin{bmatrix} 0 & A^T \\ -A & 0 \end{bmatrix} \begin{bmatrix} v(x,y) \\ u(x,y) \end{bmatrix}.$$

令 $D := \max\left\{\frac{\|(u,v)-(\bar{u},\bar{v})\|^2}{2} : (u,v) \in \triangle_m \times \triangle_n\right\}$，我們將 Nesterov 的最佳偏導函數演算法套用在此問題

$$\min\left\{F_\mu(x,y) : (x,y) \in \triangle_m \times \triangle_n\right\}. \tag{5}$$

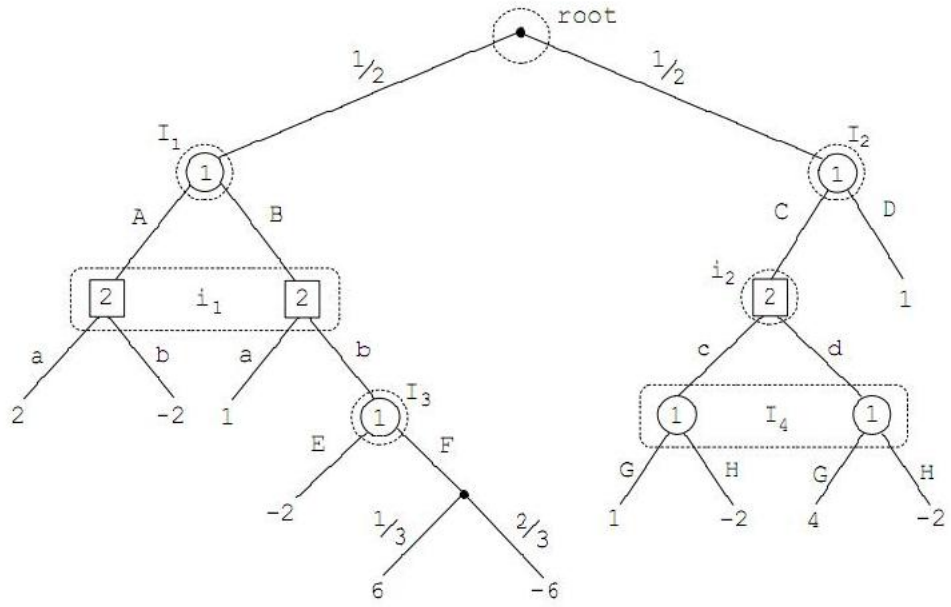
假設 $(x_0, y_0) \in \triangle_m \times \triangle_n$ 以及 $\epsilon > 0$給定，我們可以得到下面的演算法。

演算法 $(A,\ x_0, y_0, \epsilon)$

1. 令 $\mu = \frac{\epsilon}{2D}$ 和 $(w_0, z_0) := (x_0, y_0)$

2. 在 $k = 0, 1, \ldots$

- $(u_k, v_k) = \frac{2}{k+2}(w_k, z_k) + \frac{k}{k+2}(x_k, y_k)$
- $(x_{k+1}, y_{k+1}) =$argmin $\Big\{\nabla F_\mu(u_k, v_k)^T((u_k, v_k) - (x,y))$
  $+\frac{\|A\|^2}{2\mu}\|(u_k, v_k) - (x,y)\|^2 : (x,y) \in \triangle_m \times \triangle_n\Big\}$
- 如果 $F(x_{k+1}, y_{k+1}) < \epsilon$ 停止，反之，做下一步
- $(w_{k+1}, z_{k+1}) =$ argmin $\Big\{\sum_{i=0}^{k} \frac{i+1}{2}\nabla F_\mu(u_i, v_i)^T((u_i, v_i) - (w,z))$
  $+\frac{\|A\|^2}{2\mu}\|(x_0, y_0) - (w,z)\|^2 : (w,z) \in \triangle_m \times \triangle_n\Big\}.$



圖一：兩人依序行動賽局的樹狀圖。

範例 3.1. 此範例引用自 [6]，圖一在敘述一個兩人依序行動賽局。樹狀圖中的 root 代表遊戲的起始點，而末端代表結束點，剩下在內部的底點稱之為決策點。圖中的虛框代表在同一個資訊集合包含的決策點。決策點的數字代表是哪個玩家在做決策，而在邊上的記號代表玩家的選擇。另外，一個樹狀圖可能會包含一些隨機事件發生的點，我們稱之為機會點，會選擇哪條路徑就完全由路徑上標示的機率來決定，在圖一中，沒做標記的點就是機會點。在此範例中，第一個玩家的決策集合有 $I_1, \ldots, I_4$，而第二個玩家的決策集合則是 i_1 和 i_2。第一個玩家的報酬矩陣為

$$
\text{A=}\quad
\begin{array}{c}
 \\
\epsilon \\
A \\
B \\
C \\
D \\
BE \\
BF \\
CG \\
CH
\end{array}
\begin{array}{c}
\begin{array}{cccccc} \epsilon & a & b & c & d \end{array} \\
\left[\begin{array}{ccccc}
 & & & & \\
 & 1 & -1 & & \\
 & 1/2 & & & \\
 & & & & \\
1/2 & & & & \\
 & & -1 & & \\
 & & -1 & & \\
 & & & 1/2 & 2 \\
 & & & -1 & -1
\end{array}\right]
\end{array}
$$

舉例來看，在計算 矩陣中 (B,a) 的值時，我們看所有 (B,a) 之後 的路徑，因為只有一條路，所以我們把報酬值乘上在這之前走到i_1的機率。一般情況玩家一選擇路徑 A 跟 B 各有 1/2 的機率，所以 (B,a) 的值為 1/2。接著，我們看 (BF,b) 之後有兩條路徑，所以 (BF,b) 的值就是兩條路徑的報酬的總和。由於這兩條路徑都是自然機率發生，另外，我們還要考慮在這之前走到此 I_3 的機率，用一般情況來看，玩家一走 B 的機率是 1/2，玩家二走 b 的機率也是 1/2，所以 (BF,b) 的值就是 $\frac{1}{2} \times \frac{1}{3} \times 6 + \frac{1}{2} \times \frac{2}{3} \times (-6) = -1$。
我們利用之前的演算法去找這個賽局的納許均衡解，演算後的結果顯示在表3.1。

在表3.1中，我們記錄了兩個玩家各種策略的最佳機率分配，因為這範例有最佳解，所以函數F 最終的值算出來是 0。而此遊戲對玩家一的期望值則是 -0.5，這是因為起始點是交由機率決定，從圖一可以看出左邊對玩家一不利，而右邊對玩家一有利。從最佳解來算期望值的話，在左邊玩家一會得到 -2 的報酬，而右邊僅會得到 1 的報酬，所以期望值為 -0.5。

表3.1： 範例3.1的程式結果，取 $\epsilon = 10^{-7}$

玩家一的機率分配	ϵ	A	B	C	D	BE	BF	CG	CH
	[1	0	1	0	1	1	0	0	0]
玩家二的機率分配	ϵ	a	b	c	d				
	[1	0	1	1	0]				
目標值	$F(x, y) = 0.0000000$								
期望值	$x^T Ay = -0.5$								

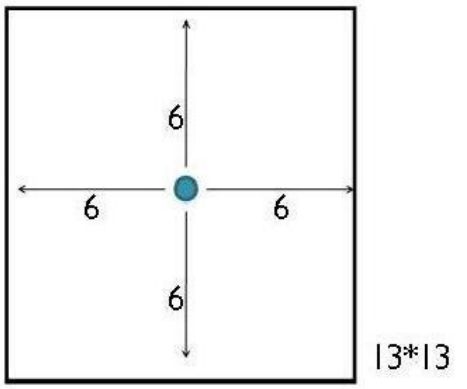
此結果與我們使用逆推法得到的納許均衡解是相同的。另外，初始值的設定跟程式的快慢也有點關係，如果假設初始值 (x_0, y_0) 跟納許均衡解完全相反的機率分配，我們的疊代次數就會變多。一般來說，像這樣的報酬矩陣大小，都是在1-2秒內就可以跑出最佳解。不過接下來要應用在我們的主題五子棋上，報酬矩陣明顯的會過大，下一節我們會提出改良的方法來適用我們的模型。

3.2 預測勝負值

在演算法中，報酬矩陣扮演一個非常重要的角色，我們利用計算每種策略產生的勝負值 v 當作我們的報酬矩陣 A，也就是說，如果先手選擇 i 策略，後手選擇 j 策略，透過計算得到的 v 值 其實就是報酬矩陣 A 裡的元素 $A_{i,j}$ 。在本篇論文中，我們預測兩步後的勝負值做為我們的報酬矩陣，另外，我們定義一個決策集合，使運算量更小，以及提高下棋速度。決策集合代表的是每回合玩家的下棋範圍，假設此回合先手的下棋點是在 $(i^*,\ j^*)$，後手的決策集集合就是

$$P_2^t = \{(i,\ j) :\ B_{i,j} = 0,\ i^* - 6 \le i \le i^* + 6,\ j^* - 6 \le j \le j^* + 6\}.$$

圖二即是我們的決策集合在棋盤上的圖形。



圖二：玩家每回合的決策集合

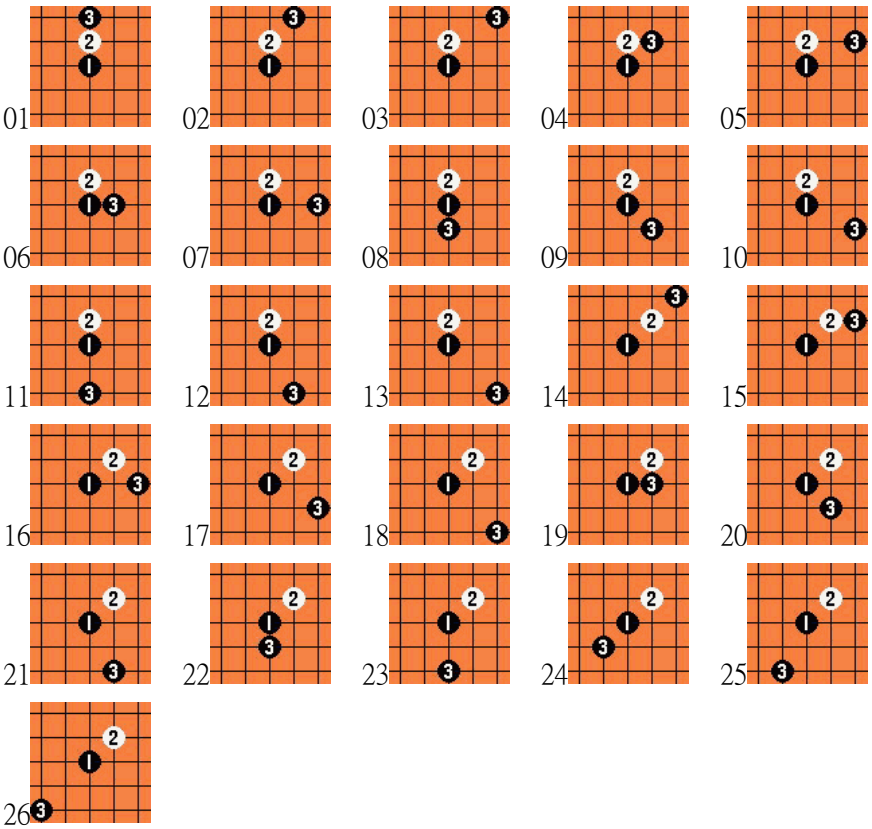
接下來我們做預測勝負值的動作。假設先手在第 t 回合下棋點在 (i^*, j^*)，後手從 P_2^t 這集合中挑選一個下棋點 $(i,\ j)$，此時先手的決策集合變成 $P_1^{t+1} = \{(p,\ q) :\ B_{p,\,q} = 0,\ i-6 \le p \le i+6,\ j-6 \le q \le j+6\}$，預測先手可能的下棋點 $(p,\ q)$，計算棋盤上的報酬值，就是我們的勝負值。

關於勝負值的詳細給分，我們在實驗結果中會提到，另外，我們將預測兩步後勝負值的程式碼部分提供在附錄一。

第四章 實驗與討論

4.1 勝負值及開局棋型

在此節中我們要針對勝負值給分，並引用五子棋的26種開局棋型 (如圖三)，另外，關於勝負值的部分，前面提到當有一方勝利時即可獲得 1 的報酬，輸家則是損失 1 的報酬。在此演算法中，我們把勝負值的部分制訂的更詳細。在我們下棋之前，先預測下在每一個點的情形，訂出給分規則如下表4.1：



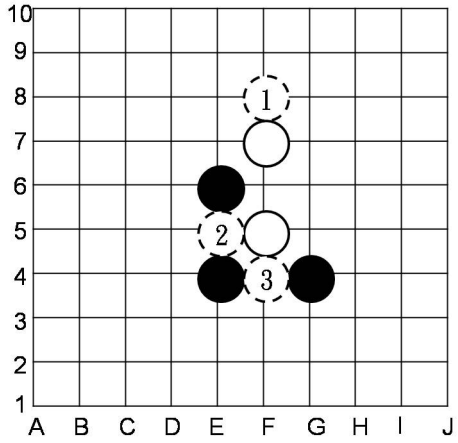
圖三：開局棋型

表4.1：給分表

進攻	我方得分	防守	我方得分
1.我方連五	+1	1.對方活四	-0.6
2.我方活四	+0.3	2.對方死四	-0.6
3.我方四三	+0.05	4.對方活三	-0.012
4.我方死四	+0.0095	5.對方連二	-0.0003
5.我方活三	+0.004		
6.我方連二	+0.0001		
7.我方禁手	-1		

在上表中，我們給分的方式是以各種條件的優勢大小去做給分，分數採用進攻得分跟防守得分的累計。並且，加減分的制訂也要考慮不阻擋對手的後果是否合理。另外，我們給分的規則多了一個連二的情況，是我們第二章未提到過的。我們定義連二的狀況是指，如果對手不做阻擋，下一步我們就有機會產生活三的情況。表中的第七點主要是應用在日規的情況。

因為雙三跟雙四都是禁手，能取勝的情況一般來說都是下出四三棋型時，為此，我們特別制訂了四三的給分，讓先手在考慮下棋時，能盡可下出四三棋型。根據不同的規則，我們會將某些給分條件拿掉，稍後補充其他規則的給分表。實際給分狀況我們以下面圖四來做個簡單的舉例。



圖四：計算分數的例圖

在圖四中，如果白子下在點 2，可以阻斷黑子的一條連二，且自己產生一條新的連二。以整個棋面來看，白子下在點 2 共有兩條連二的情況，而對手的黑子還保有一條連二，所以下在此點的分數 $0.0001 \times 2 - 0.0003 = -0.0001$。若是下在點 3，則白子形成一條活三，且阻斷黑子一條連二，此點分數為$0.004 - 0.0003 = 0.0037$。在點 1 的情況則是，白子形成一條活三，但是黑子的兩條連二都沒被阻斷，分數比點 1 略低，為 0.0034。

我們依據給分表做一個圖四中白子的報酬矩陣，因為下棋數不多，我們僅列出一部份範圍的下棋點的報酬矩陣如下 (假設下在重複的點要扣100分)：

A=

	(F,8)	(E,7)	(F,6)	(E,5)	(F,4)
(F,8)	-100	+0.0079	-0.0002	+0.0079	-0.0041
(E,7)	+0.0002	-100	-0.0001	+0.0005	-0.0037
(F,6)	-0.0002	+0.0079	-100	+0.0079	-0.0041
(E,5)	+0.0002	+0.0004	+0.0001	-100	-0.0037
(F,4)	-0.0001	+0.008	-0.0001	+0.008	-100

4.2 數據結果

接下來我們測試每種開局的勝負以及討論此開局的優劣，此外，我們還測試幾種不同規則之下的勝負狀況做比較，這樣我們可以看出哪些規則對遊戲的影響比較巨大。

在每個表格中都有各開局棋型的勝負、如何取勝的方法，以及棋子的總顆數。

表4.2 基本規則勝負情形

開局棋型	勝負(勝法)	棋子總數	開局棋型	勝負(勝法)	棋子總數
1.	黑子勝(四三)	29	14.	黑子勝(雙三)	21
2.	白子勝(雙三)	36	15.	黑子勝(四三)	27
3.	黑子勝(雙三)	31	16.	白子勝(雙三)	32
4.	白子勝(雙三)	28	17.	黑子勝(四三)	33
5.	白子勝(雙三)	42	18.	黑子勝(四三)	47
6.	白子勝(雙三)	42	19.	黑子勝(四三)	41
7.	白子勝(雙三)	48	20.	黑子勝(四三)	27
8.	黑子勝(四三)	37	21.	黑子勝(雙三)	19
9.	黑子勝(四三)	19	22.	黑子勝(四三)	37
10.	黑子勝(四三)	49	23.	白子勝(雙三)	32
11.	白子勝(雙三)	32	24.	黑子勝(四三)	29
12.	白子勝(雙三)	22	25.	黑子勝(四三)	21
13.	黑子勝(雙三)	33	26.	黑子勝(四三)	27

在表4.2中我們可以看到在基本規則底下，黑子明顯勝場數比白子多，於是我們又作了日規的測試，不過發現在日規之下白子勝場數比黑子多，於是我們嘗試把日規中的其中一些規則拿掉，看是否可以從中找到一個平衡。

表4.3 日規勝負情形

開局棋型	勝負（勝法）	棋子總數	開局棋型	勝負	棋子總數
1.	白子勝(雙三)	50	14.	黑子勝(四三)	59
2.	白子勝(雙三)	40	15.	黑子勝(四三)	25
3.	白子勝(雙三)	40	16.	白子勝(雙三)	40
4.	白子勝(雙三)	28	17.	黑子勝(四三)	47
5.	黑子勝(四三)	27	18.	白子勝(雙三)	46
6.	黑子勝(四三)	85	19.	白子勝(雙三)	44
7.	白子勝(雙三)	36	20.	黑子勝(四三)	33
8.	白子勝(四三)	18	21.	黑子勝(四三)	25
9.	黑子勝(四三)	25	22.	白子勝(四三)	18
10.	黑子勝(四三)	41	23.	白子勝(雙三)	38
11.	白子勝(雙三)	40	24.	黑子勝(四三)	23
12.	白子勝(雙三)	68	25.	黑子勝(四三)	23
13.	白子勝(雙三)	54	26.	白子勝(雙三)	30

表4.4 不禁雙三勝負情形

開局棋型	勝負	棋子總數	開局棋型	勝負	棋子總數
1.	白子勝(雙三)	40	14.	黑子勝(雙三)	21
2.	白子勝(雙三)	40	15.	黑子勝(雙三)	27
3.	白子勝(雙三)	50	16.	黑子勝(雙三)	45
4.	黑子勝(雙三)	29	17.	黑子勝(失誤)	27
5.	黑子勝(四三)	27	18.	黑子勝(雙三)	47
6.	黑子勝(四三)	81	19.	黑子勝(四三)	79
7.	白子勝(雙三)	48	20.	黑子勝(雙三)	27
8.	白子勝(四三)	18	21.	黑子勝(四三)	21
9.	黑子勝(四三)	31	22.	白子勝(四三)	18
10.	黑子勝(四三)	41	23.	白子勝(雙三)	38
11.	白子勝(雙三)	38	24.	黑子勝(四三)	23
12.	白子勝(雙三)	24	25.	黑子勝(四三)	23
13.	黑子勝(雙三)	43	26.	白子勝(雙三)	30

我們比對表4.3與4.4發現，黑子在開局棋型第4、13、16和18中，原本因為日規禁雙三而輸掉，但是在不禁雙三中獲勝，而開局棋型第14、15和20中，黑子原本在日規利用四三獲勝，在不禁雙三裡，黑子用更少的下棋數取得勝利，另外在第17個棋型，由於白子的失誤反而使黑子產生一個"不同時"的雙三取勝。

表4.5 不禁雙四勝負情形

開局棋型	勝負	棋子總數	開局棋型	勝負	棋子總數
1.	白子勝(雙三)	40	14.	黑子勝(四三)	59
2.	白子勝(雙三)	40	15.	黑子勝(四三)	25
3.	白子勝(雙四)	34	16.	白子勝(雙三)	46
4.	白子勝(雙三)	30	17.	黑子勝(失誤)	27
5.	黑子勝(四三)	27	18.	白子勝(雙三)	46
6.	黑子勝(四三)	85	19.	白子勝(雙三)	56
7.	白子勝(雙三)	34	20.	白子勝(禁手)	21
8.	白子勝(四三)	38	21.	黑子勝(四三)	19
9.	黑子勝(四三)	77	22.	白子勝(四三)	18
10.	黑子勝(四三)	41	23.	白子勝(雙三)	40
11.	白子勝(雙三)	40	24.	黑子勝(四三)	21
12.	白子勝(雙三)	30	25.	黑子勝(四三)	23
13.	黑子勝(雙四)	75	26.	白子勝(四三)	42

我們比較表4.3與4.5，黑子因為雙三禁的關係勝場數減少，其中有兩個比較特殊的情況：在開局棋型第13中，黑子利用雙四取勝，不過在第20中，白子迫使黑子下出雙三的禁手而取勝。接著我們統計了各種規則的勝負結果以及各種開局棋型對兩方的勝負情形。

表4.6：勝負結果

規則	白勝	黑勝
基本規則	9	17
不禁雙三	10	16
日式規則	15	11
不禁雙四	15	11

表4.7：開局勝敗統計

開局棋型	白勝	黑勝	開局棋型	白勝	黑勝
1.	3	1	14.	0	4
2.	4	0	15.	0	4
3.	3	1	16.	3	1
4.	3	1	17.	0	4
5.	1	3	18.	2	2
6.	1	3	19.	2	2
7.	4	0	20.	1	3
8.	3	1	21.	0	4
9.	0	4	22.	3	1
10.	0	4	23.	4	0
11.	4	0	24.	0	4
12.	4	0	25.	0	4
13.	2	2	26.	2	2

在開局棋型第 2、7、11、12 及 23 下，無論何種規則白子都能獲勝，這顯示這棋型對白子是相當有利的，另外在第 9、10、14、15、17、21、24 及 25 的開局棋型則是對黑子有利。

4.3 問題與討論

從上節的數據可以發現：在使用的演算法以及給分設計之下，日規顯得對黑子有些不利，但基本規則又對黑子有利，於是我們進一步去討論如果拿掉日規中的一些規則，勝負情況會產生怎樣的變化。在不禁雙三下，黑子勝率明顯提高很多，而不禁雙四整體來看跟日規的勝敗是一樣的。除了規則之外，開局的棋型也有影響，從表4.7來看，有些棋型對白子非常有利，有些則對黑子有利，這數據可以讓我們在下棋時去思考要怎樣開局才對自己有幫助。

從日規的實驗結果來看，跟現今一般五子棋認定的先手優勢似乎有些落差，可能是因為預測的步數不夠多，讓下棋者沒辦法在兩步內找到必贏策略，而且我們衡量到與真人對奕時的下棋速度而縮小搜尋範圍，所以會有較糟的結果。如果預測越多步數對賽局樹狀圖的全貌也會更清楚，不過相對的，因為報酬矩陣也會變大，每多預測一步，資料的大小會以約100倍增加，所以運算速度會降低。在此之前，我們曾嘗試讓預測一步的 AI 跟 預測兩步的 AI 在基本規則之下作對奕，結果顯示讓預測兩步的 AI 執黑子的話可以贏24場，如果執白子的話可以贏25場，這顯示如果預測越多步數，電腦 AI 會越強。

第五章 結論與展望

當初決定利用賽局理論的觀念，以及用最佳化的方法去解五子棋問題，主要是因為在一些相關的參考資料當中，電腦的下棋點往往是取決於給分的高低，但是在賽局中，報酬越高的策略不見得是 Nash 均衡解，這使我們思考到，如果把賽局理論跟最佳化的方法作個結合，會不會是一種新的思考邏輯。在別的參考資料的方法中也有利用到賽局的一些樹狀圖的概念。未來有幾個可以改進的地方：

1. 預測的步數不夠多，以致於程式無法做"佈局"的動作，也就是說，死四可以得到更高的分數，程式會優先考慮死四，而不是在適當的時機(有機會造成四三)再做死四，這樣對黑子是非常吃虧的，因為日規底下必須要靠四三才能贏，而不禁雙四取勝的手段也只有四三與雙四兩種。

2. 給分的方式我們僅以一定要阻擋對手或下出比對手分數更高的棋步來做給分，而分數會直接影響到此策略的機率分配大小，這部分未來希望可以找到一個較為合理的方式來給分，且除了我們原本的給分情況之外，希望可以更加定義一些有利的棋步來做更詳細的給分，這樣會讓演算法的效果更好。

3. 可嘗試引進"棋譜"建立一個知識庫使電腦有學習演進的效果，另外需制訂好知識庫的給分方式。

希望未來在程式上能預測更多步數，以及在給分機制上有突破，既能兼顧到下棋速度的效率，又可以提高黑子的勝率，並且能朝設計國際規則 AI 的目標邁進。

參考文獻

[1] Allis, L. V. .Searching for Solutions in Games and Artificial Intelligence, Ph.D. Thesis, University of Limburg,Maastricht, 1994.

[2] Allis, L. V. , Herik, H. J. van den, and Huntjens, M. P. H. . Go-Moku Solved by New Search Techniques. Computational Intelligence, Vol. 12, pp. 7 – 23,1996.

[3] Andrew Gilpin, Javier Pe$\tilde{n}a$, Tuomas Sandholm. First-Order Algorithm with $\mathcal{O}(\ln(1/\epsilon))$ Convergence for ϵ-Equilibrium in Two-Person Zero-Sum Games. Optimization Online, 2008.

[4] Yu. Nesterov. Smooth Minimization of Non-Smooth Functions. Math. Program., Ser. A 103, 127-152, 2005.

[5] Martin J. Osborne, Ariel Rubinstein. A Course in Game Theory, MIT process, 1994.

[6] Samid Hoda, Andrew Gilpin, and Javier Pea. A Gradient-Based Approach for Computing Nash Equilibria of Large Sequential Games. Optimization Online, 2007.

[7] 黃德彥. 五子棋相關棋類人工智慧之研究, 國立交通大學資訊工程學系碩士論文, 2005.

[8] 翁慈孝. 電腦五子棋程式珠連設計與製作, 國立東華大學資訊工程學系碩士論文, 2004.

附錄一：預測勝負值（預測兩步）

1. 預測下棋，假設先手下在 (i^*, j^*)

```
for (i = i* − 6; i ≤ i* + 6; i++)
{
  for (j = j* − 6; j ≤ j* + 6; j++)
  {
    if (B_{i,j} == 0)
    {
      B_{i,j} = −1;
    }
    for (p = i − 6; p ≤ i + 6; p++)
    {
      for (q = j − 6; q ≤ j + 6; q++)
      {
        if (B_{p,q} == 0)
        {
          B_{p,q} == 1;
        }
          計算勝負值 v;
          B_{p,q} == 0;
      }
    }
    B_{i,j} = 0;
  }
}
```