中華民國第 58 屆中小學科學展覽會

作品說明書

國中組　生活與應用科學(一)科

032811

智慧翻轉，創造「棋機」

~人工智慧黑白棋 App 研發

學校名稱：高雄市立明華國民中學

作者：	指導老師：
國二 胡芯綾	陳晏閔
國二 陳芃安	王天佑
國二 林奕睿	

關鍵詞：人工智慧、黑白棋、積木程式

摘要

本研究以 MIT App Inventer 2 積木程式實作黑白棋 App。其中 AI 棋手程式的估值函數參數有位價值、翻轉數和行動力。位價值是由預編的位置價值表查找，翻轉數是核算該棋步的翻子數量，行動力是前瞻一步時我方與敵方的合法棋步數差值。App 下棋操作介面的設計，具有人與人、人與 AI 程式及 AI 與 AI 對弈等三種執行模式。其提供完整的棋步紀錄、合法棋步的提示及回溯上一步等功能，方便學習者使用。以內建的 AI 棋手程式對戰作實驗，用勝率來評估人工智慧程度及優劣，並可藉以不斷的調校權值以達最佳智能。結果可知：具有棋局遊戲開局、盤中、終局三階段走法有不同加權者的 AI 智能較高。未來還可將遊戲樹等相關 AI 走法搜尋演算法加入實作。

壹、 研究動機

近幾年來人工智慧已然變成顯學，而最近圍棋程式 AlphaGo 戰勝職業冠軍棋士的消息更是令人感到驚艷。AlphaGo 是使用了蒙地卡羅樹搜尋，並藉助估值網路與走棋網路這兩種深度神經網路，它透過估值網路來評估大量選點，並透過走棋網路選擇落點。AlphaGo 最初透過模仿人類棋手，以機器學習的方式嘗試符合職業棋士的棋局，其資料庫中有著眾多的棋譜可供它學習。後來當它達到了一定的熟練程度，便開始和自己對弈大量棋局，使用深度學習進一步改善它本身。在這種設計下，電腦可以結合遊戲樹做不同深度及廣度的探索，又可像人類的大腦一樣自發學習進行訓練與嘗試，以提高下棋實力。AlphaGo 的崛起，已經引發了 AI 人工智慧的學習熱潮。於是我們也想試試，藉由實作另一種棋類程式黑白棋 App 來初步探索 AI 人工智慧的奧秘。

貳、 文獻探討

一、文獻一：黑白棋簡介

黑白棋，又叫奧賽羅棋(Othello)、反棋(Reversi)，是一個經典的策略性遊戲。遊戲使用 8x8 的棋盤，由兩人執黑子和白子輪流下棋，通過相互翻轉對方的棋子，最後以棋盤上誰的棋子多來判斷勝負。開局時，棋盤正中央的 4 格先置放黑白相隔交錯的 4 枚棋子。通常黑子先行，雙方輪流落子。只要落子和棋盤上任一枚己方的棋子在一條線上（橫、直、斜線皆可）夾著對方棋子，就能將對方的這些棋子翻面，黑白變色轉變為己方。如果在任一位置落子都不能夾住對手的任一顆棋子，就要讓對手下子。當雙方皆不能下子時，遊戲就結束，盤面子多的一方勝。

隨著網路的普及，黑白棋逐漸流行起來。在 1994 年，黑白棋程式的編寫方法有了突破性的發展，首先是有了一個稱為 IOS 的網站，讓不同的程式同時連上去互相對局，在同期 Michael Buro 做出了能由程式自我學習的 Logistello，許多程式用類似的方法來編寫。程式設計師不再把人工的策略和下棋方法等固定地寫在程式裡，而是由程式進行機器學習，並記錄好、壞的形式，根據實戰的結果自動調整策略，又會把不同的開局棋譜根據實戰的過程來評分、保存，有些程式能保存幾十萬步的開局棋譜。

因為 Logistello 在根本方法的改進，其運用先進的演算法、提升程式編寫的效率和準確性等等，一直在黑白棋程式競賽保持世界冠軍。1997 年 8 月，Logistello 擊敗了 1996 年的世界冠軍，從此黑白棋程式把人類棋手遠遠甩在後面。直到在 1998 年 1 月 Logistello 宣布退休為止。現在已經有數個程式在棋力上超越已退休的 Logistello，包括 Edax, Cyrano, Saio，主要突破在於它們把演算法改善為多執行緒。 與上述齊名的是 Gunnar Andersson 的 Wzebra 黑白棋自由軟體，它是 Zebra(斑馬)的 Windows 版本，除了可以下棋外，還提供了打譜、復盤、棋局分析、自我學習等功能，也可以載入 Thor 格式棋譜文件，進行針對性訓練。

二、文獻二：Wzebra 黑白棋用詞及策略

1. 角(Corner)：角就是位於 a1、a8、h1 和 h8 的位置。它們通常是好位置，要盡可能佔取。
2. C 位、星位(C-squares and X-squares)：C 位就是位於 a2、a7、b1、b8、g1、g8、h2 和 h7 的位置，星位就是位於 b2、b7、g2 和 g7 的位置。會致使失去角位，務必要小心避開。
3. 中心(Center)：局面的中心就是內部子的集合。
4. 控制中心策略(Control of the center)：一種策略，它試圖在局面中心擁有盡可能多的棋子，使邊界擁有盡可能少的棋子，以獲得最大可能的行動力。
5. 爬邊策略(Edge creeping)：一種以弱邊(不平衡邊……)為代價，在一條或兩條邊上獲得最大數量棋步的策略。爬邊者試圖通過將全部邊界都留給對手來快速耗盡他的棋步，但是如果爬邊不能湊效，壞邊產生的效應將使他的局面迅速變弱。
6. 邊界(Frontier)：邊緣子的集合，也就是說那些與空位相鄰的棋子。邊界越小越好。
7. 獲得餘裕手(Gain a tempo)：在棋盤的某個區域內比對手多下一步棋，以迫使對手在其他地方先開始下棋(從而延長對手的邊界)。
8. 效應(Influence)：當棋手所下的子使他同時在多個方向上翻轉棋子時，我們就說這些棋子產生了好的效應。
9. 內部子 (Internal discs)：內部子就是不與空位相鄰的棋子。沒有內部子是不好的。
10. 邊緣子 (External discs)：邊緣子就是與空位相鄰的棋子。太多邊緣子是不好的。
11. 自由度(Liberty)：安全的棋步數。但“缺少自由”意味在不久的將來不得不奉送角位。

12. 多子策略(Maximum disc strategy)：許多初學者所用的錯誤策略，他們每步棋都試圖翻轉最大數量的棋子。
13. 行動力(Mobility)：棋手合法的可下手棋步數量。亦即，當棋手擁有大量的可能棋步時，他就擁有好的行動力。
14. 奇偶性(Parity)：一種在對手佔據的每個區域內都留下偶數個空位的策略。
15. 安靜步(Quiet move)：不翻轉邊緣子的一步棋，通常這是步好棋。
16. 穩定子(Stable discs)：絕對不會被翻轉的棋子。角就是一個穩定子的實例。
17. 斯通納陷阱(Stoner Trap)：一種針對弱邊局面強迫進行角交換的攻擊。
18. 不平衡邊(Unbalanced edge)：邊上 5 個同色棋子彼此相鄰組成的結構，但都不在角位上。意即邊上的一端留有角位，另一端留有角位和 C 位的一排棋子。

三、文獻三：演算法的樂趣

本書介紹了關於賽局樹與各類遊戲的演算法內容，並簡介了常見的棋類遊戲運作的數學模型，估值函數及估值演算法(如：行動力的估值模型、位值表的估值模型…)，走法搜尋的演算法(如：α-β剪枝搜尋法、Zobrist 雜湊置換表…)。並給出常見演算法的實作範例，可供學習參考。

參、 研究目的

一、研究黑白棋規則與策略之實作方法

二、以 MIT App Inventer 2 實作黑白棋 App 之研究開發

(一)、 分析探討棋盤與棋子之數學模型及資料結構：實作 Warren Smith 模型

(二)、 估值函數與估值演算法:：實作三種估值模型

1. 位值表的估值模型：位價值參數
2. 總得分的估值模型：翻轉數參數
3. 行動力的估值模型：行動力參數

(三)、 整合控制介面：建立三種下棋操作介面執行模式

1. 人與人對弈模式
2. 人與 AI 程式對弈模式
3. AI 程式與 AI 程式對弈模式

(四)、 AI 程式對戰實驗：調校參數權值比例

名詞解釋

一、黑白棋

黑白棋，又叫奧賽羅棋(Othello)、反棋(Reversi)，是一個經典的策略性遊戲。遊戲使用 8x8 的棋盤，由兩人執黑子和白子輪流下棋，通過相互翻轉對方的棋子，最後以棋盤上誰的棋子多來判斷勝負。

二、MIT App Inventer 2 積木式 Android 程式線上開發環境：

App Inventor 讓我們可在網路瀏覽器上來開發 Android 手機應用程式，開發完成的程式可下載到實體手機或在模擬器上執行。App Inventor 伺服器會儲存工作進度還會協助程式版本管理。其程式的特色是：設計概念類似 Scratch 以程式積木塊堆疊而成，適合無 Java 程式碼編寫基礎的初學者，而且開發作業都是透過瀏覽器在雲端上完成。

肆、 研究器材與設備

一、 Windows 10 作業系統、Google Chrome 網路瀏覽器

二、 iOS 作業系統、Safari 網路瀏覽器

三、 MIT App Inventer 2 積木式 Android 程式線上開發環境、Internet

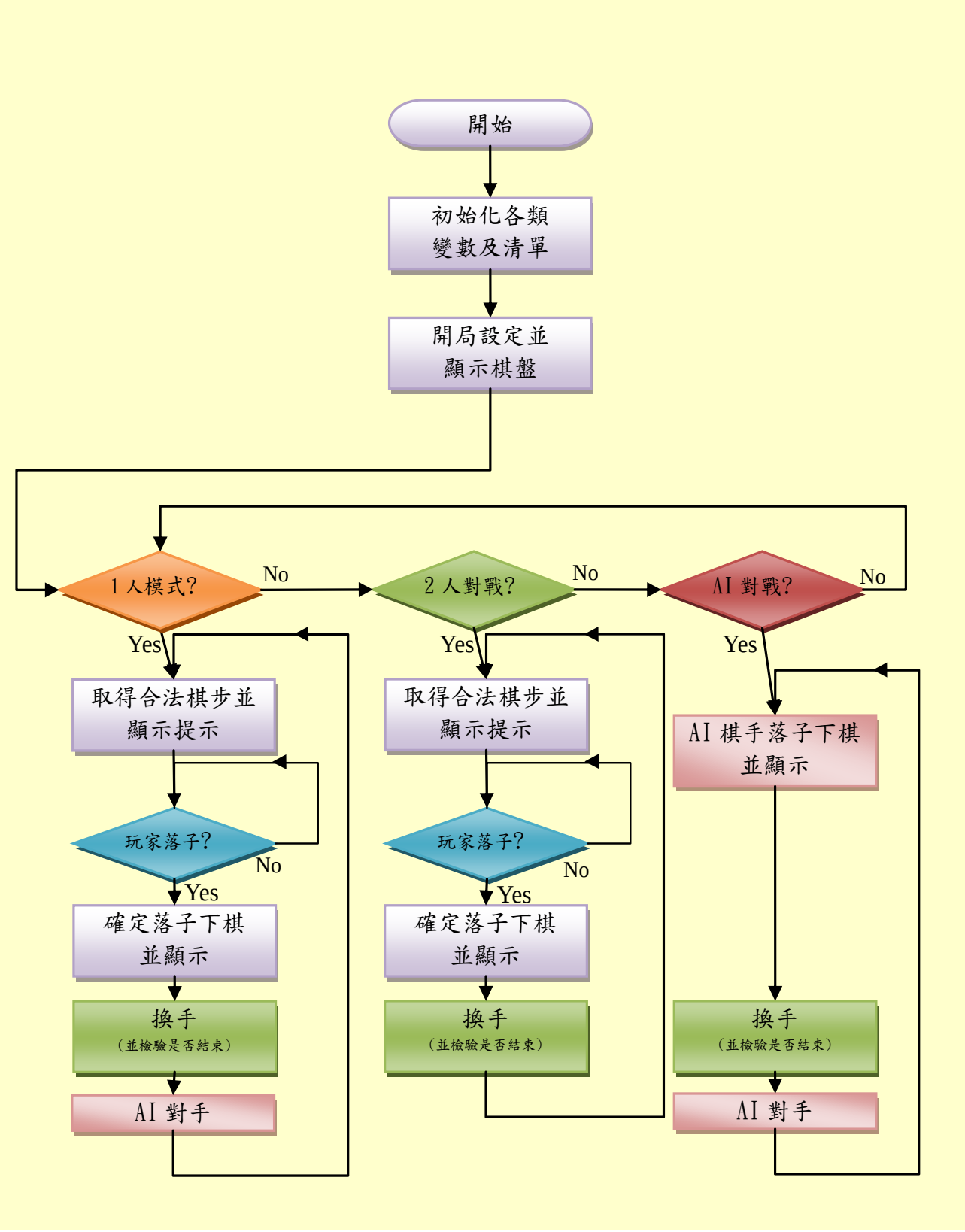
四、 Android 手機

五、 MS-Word(文書軟體)、MS-Excel(試算表軟體)

伍、 研究實作及方法

一、系統流程及操作界面:

(一)、系統流程圖：



(二)、主要介面配置：

智慧翻轉─黑白棋 App 的遊戲頁面，如圖 1。主要的輸入介面為 8x8 棋盤格之按鈕，而棋盤框的配置也是按鈕但是設定為未啟用。另外棋盤右下角有「退一步」按鈕，左下角有「結束此局」按鈕，以通用的按鍵圖示表示。下方顯示有「已下棋步」的列表及編號。畫面可看到有提示給執黑棋的合法棋步「黃色 x 記號」。圖 2 為遊戲模式選單頁面，圖 3 為二人對戰模式，提示給執白棋的合法棋步「黃色 o 記號」。圖 4 為終局畫面判定最後比分及輸贏，並呈現整局的棋步。

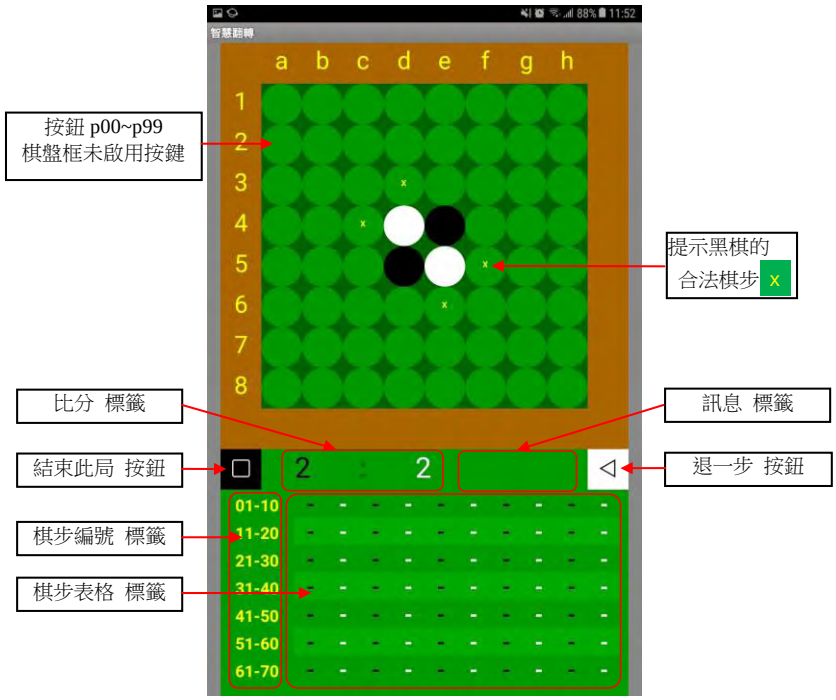


圖 1-1

圖 1-2 遊戲模式選單頁面
三種執行模式

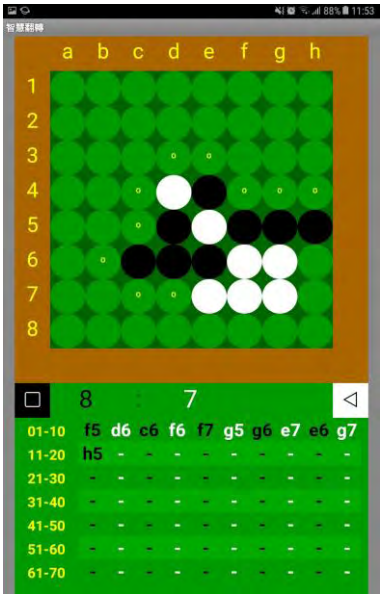


圖 1-3 二人對戰模式
提示白棋的合法棋步 o

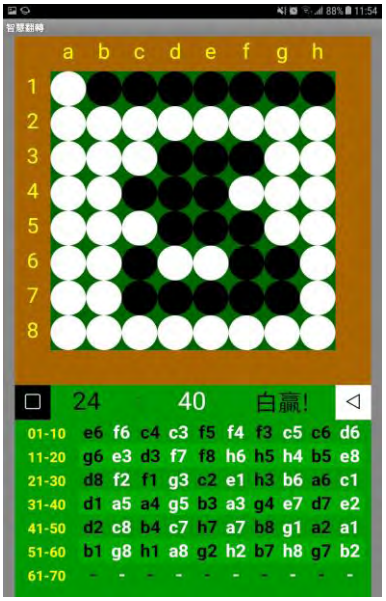


圖 1-4 終局畫面
判定最後比分及輸贏

二、程式積木塊分析及功能說明：

A. 定義全域變數：

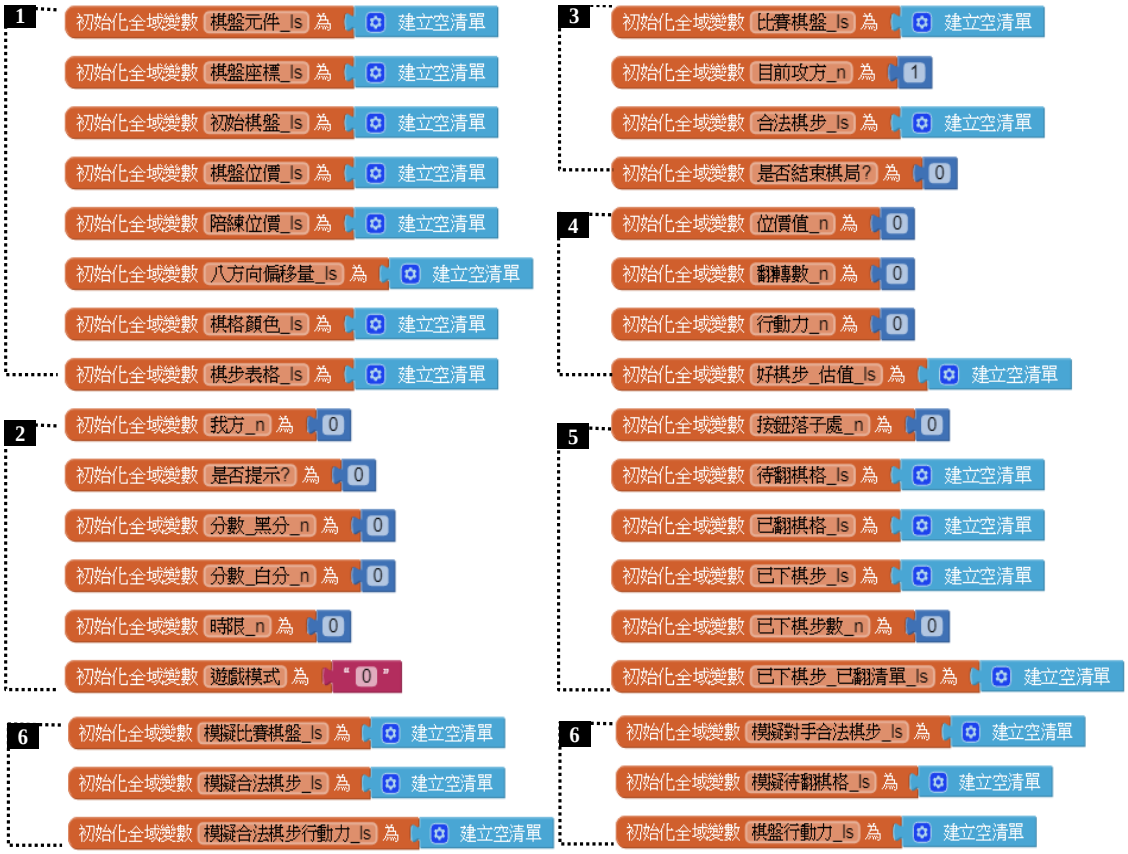


圖 2A

1 各類常數清單與儲存格清單，是本程式最主要的資料結構與實作依據，於後詳細說明。

2 **3** 遊戲流程控制用的相關變數。

3 比賽棋盤清單用於紀錄當前局面狀態。合法棋步清單用來記錄當時局面可下位置的棋子編號。目前攻方設為 1 則由黑子下(進攻)、目前攻方設為 2 則換白子下(進攻)。

4 AI 棋手評估局面用的主要參數，再依不同的加權作為選定走法的依據。

位價值：某一棋步落子位置的價值，其值越高就對後續發展越有好處故應優先選取。

翻轉數：是因應某一棋步而可翻子的棋子總數。

行動力：是指某一局面的合法棋步總數。

5 確認棋手落子的棋子編號後，待翻棋格清單用以紀錄因應所走棋步而需要翻子的棋子編號。已下棋步-已翻清單 用以記錄所有棋步的棋子編號及已翻棋子清單，提供顯示及回溯之用。

6 模擬比賽棋盤清單用於取得當前局面狀態做模擬前瞻一步。模擬對手合法棋步清單用來記錄模擬前瞻一步後對手可下位置的棋子編號。模擬合法棋步行動力清單用以記錄己方所有合法棋步所對應的行動力。

B. 螢幕按鍵輸入所對應的程序：

1. 進入遊戲模式選單頁

圖 2B-1

1 當跳入第二頁，則遊戲初始化，先打開模式選單頁面，並關掉比賽棋盤頁面。

2 準備遊戲中要用到的【初始化各類清單】，並【更新顯示棋盤】為正式起始狀態，再進入【開局設定】等候玩家按鈕選定遊戲模式。

2. 遊戲模式選擇按鈕

圖 2B-2

1 當遊戲模式的按鈕被按下後，便關掉遊戲模式選單頁面，並打開比賽棋盤頁面。一人遊戲的對手為 AI 棋手。

2 兩人遊戲則輪流操作。

3 電腦對戰則由兩個不同設定的 AI 程序輪流決定落子下棋處。

3. 棋子按鈕

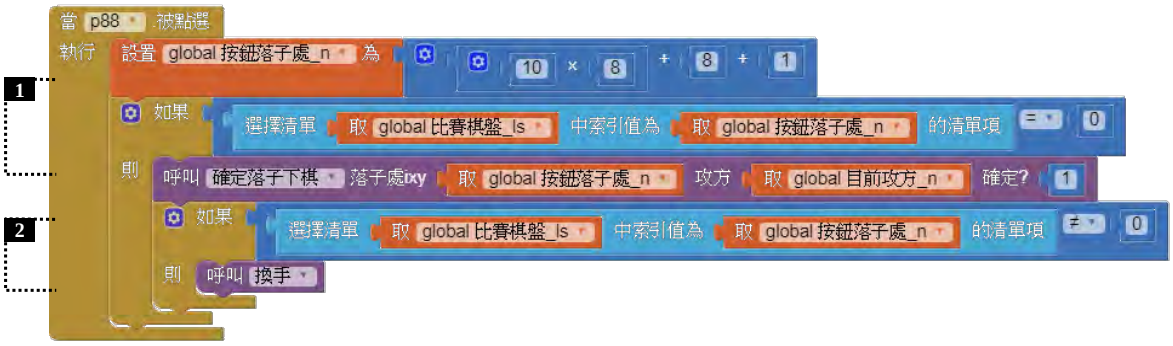


圖 2B-3

1 當代表棋子的按鈕被按下時，會先計算對應的棋子編號，然後查找比賽棋盤清單，確定此處尚未有棋子，便呼叫【確定落子下棋】。

2 若該格為可以翻子的合法棋步，便依目前執棋的棋子顏色，修改比賽棋盤清單資料項。並呼叫【換手】，攻守互易。

4. 結束按鈕

圖 2B-4

❶ 遊戲結束準備遊戲中要用到的【初始化各類清單】，並【更新顯示棋盤】為正式起始狀態，再次進入【開局設定】等候玩家按鈕選定遊戲模式。

5. 回上一步按鈕

圖 2B-5

若不小心下錯棋子或思慮不周，可按回上一步，悔棋一步，初學者學習時很好用。

❶ 須確定已走過棋步，才有步可退。

❷ 如果已進行到遊戲結束的狀態，便不提供「回上一步」功能。

確認可退就依照遊戲模式：若為 2 人模式，僅退玩家自己的一步，呼叫【回溯一步】一次，需要呼叫【換手】是因為是從對手要進攻，換回自己進攻。

❸ 若為 1 人遊戲，需先將 AI 棋手已落子處先退去，所以需要退 2 步，呼叫【回溯一步】兩次。同樣回到自己進攻，僅需呼叫【取得合法棋步】。

C. 初始化各類常數清單：

圖2C-1

1初始棋盤清單：

如圖 2C-1 為棋盤及棋子狀態的模型，本程式皆已改為一維清單，以模擬 Warren Smith 棋盤模型的實作，以簡化程式方塊的操作，於此以二維說明較為方便，以下的清單皆是如此。外圍的白色方框處為棋盤邊界以代號 3 標記。中央 8X8 的範圍為棋格的位置，以 0 標記空位。在正中央 4 格黃色圓角框處，為初始棋盤上的 4 顆棋子：用 1 標記黑棋、2 標記白棋。

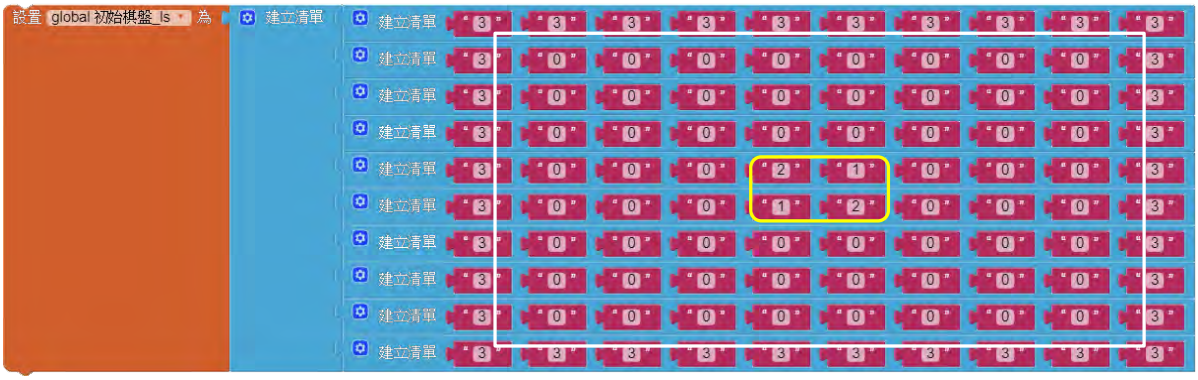


圖 2C-2

2八方向偏移量清單：選定一棋子編號每次加上第 1 項偏移量(－11)，便往左上方移 1 格。其他方向依序為：向上(－10)、右上(－9)、左(－1)、右(1)、左下(9)、向下(10)、右下(11)。

3棋盤元件清單：對應於各棋格的按鈕元件，用於提供觸控的棋子以確認走哪一棋步。用顏色來顯示目前棋局黑子、白子及空位的狀態，還可供以文字提示可能的合法棋步。

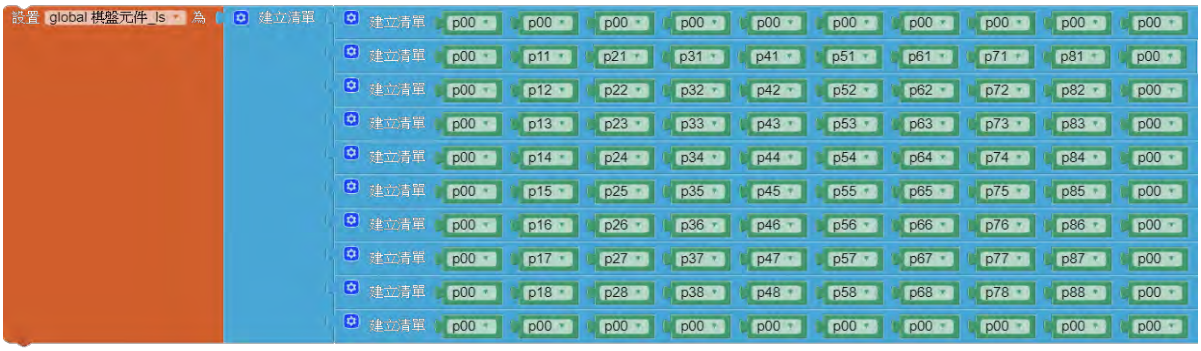


圖 2C-3

■4棋盤座標清單： 對應於各棋格的座標名稱，用於將已經走過棋步表列顯示。

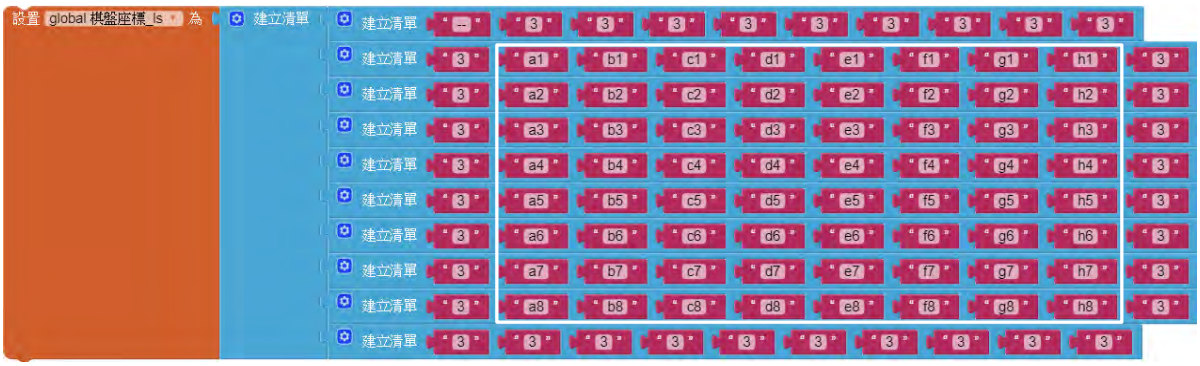


圖 2C-4

■5棋盤位值清單：

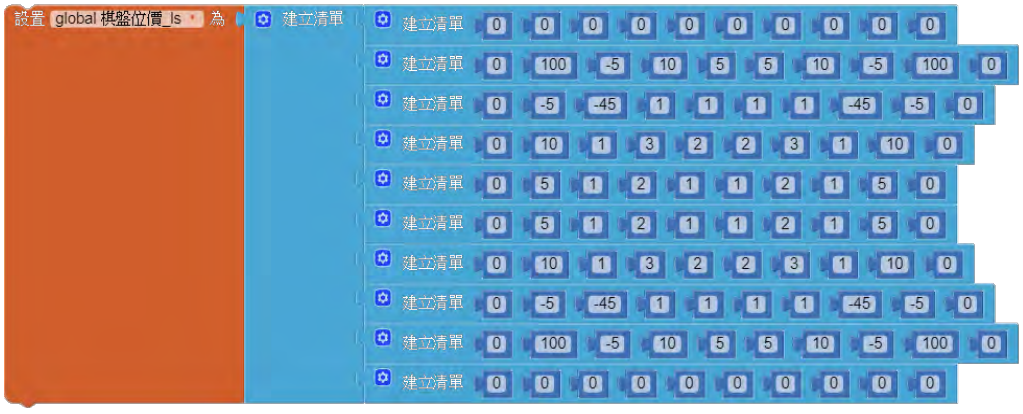


圖 2C-5

對應棋盤上的位置，取得角位：a1，a8，h1，h8 是無法被對手翻子的穩定狀態，稱為穩定子，多數的情形下應優先取得，所以給定的位置優先值較高設為 100 分。而取得 C 位：a2，a7，b1，b8，h2，h7，g1，g8 和星位：b2，b7，g2，g7 等位置，卻容易造成失去角位，故位置優先值設為負分，並以星位最嚴重設為負 45 分，而東邊(行 h)、西邊(行 a)、南邊(列 8)、北邊(列 1)等四邊上的棋子也是較容易形成穩定子，依照位置優劣遞等設為 10 分，5 分。其餘棋盤中央地帶較無絕對的位置優劣差別，再依次酌給 3 分，2 分，1 分。文獻資料顯示此配分的方式大致已於許多優秀的黑白棋程式實作驗證，但還可再自行實驗微調。棋盤中央地帶應考量的是實戰中：若在已下棋子集合的邊緣線上，稱為邊緣子，易被落子於空位而內外夾擊翻了過去，邊緣子的集合又稱邊界，所以邊界不可太長，否則便提高了被對手攻擊的機會。相反的，不在已下棋子集合的邊界上，而在內部未接觸空格的稱為內部子，又若內部子周圍被對手棋子圍繞，便形成穩定子。

D. 更新顯示棋盤：

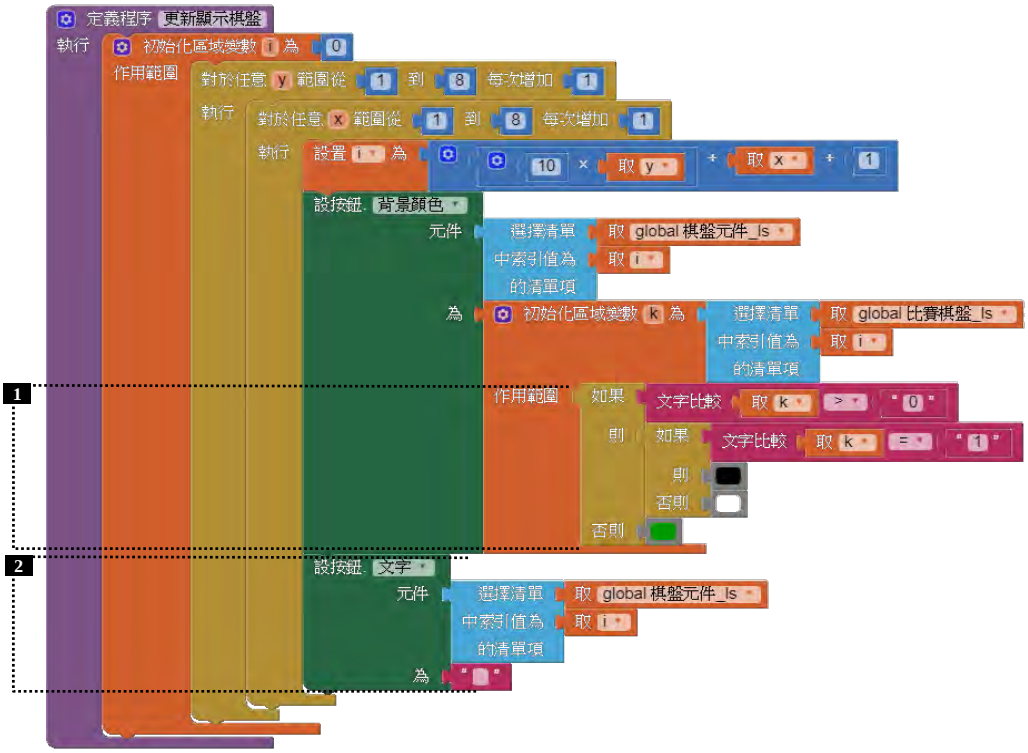


圖 2D

1 依據比賽棋盤清單的內容，設定棋盤 8X8 的範圍內棋子按鈕元件的背景顏色。

2 並將棋子按鈕元件的文字清除，預備作為提示合法棋步用。

E. 開局設定：

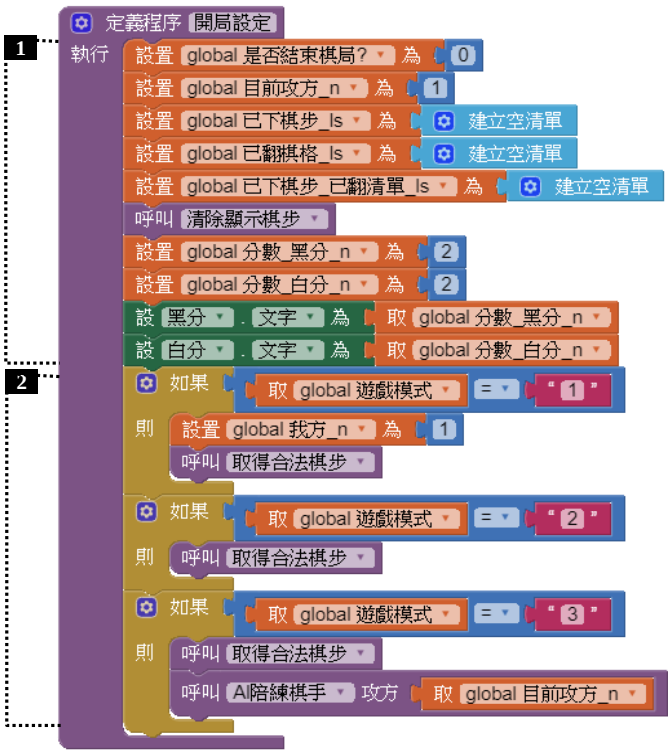


圖 2E

1 遊戲前初始化各項動態清單，並清除螢幕上舊的棋步資料。

2 依照所選模式，呼叫【取得合法棋步】，並交由棋手(或呼叫【AI 陪練棋手】)選定落子處。

F. 取得合法棋步：

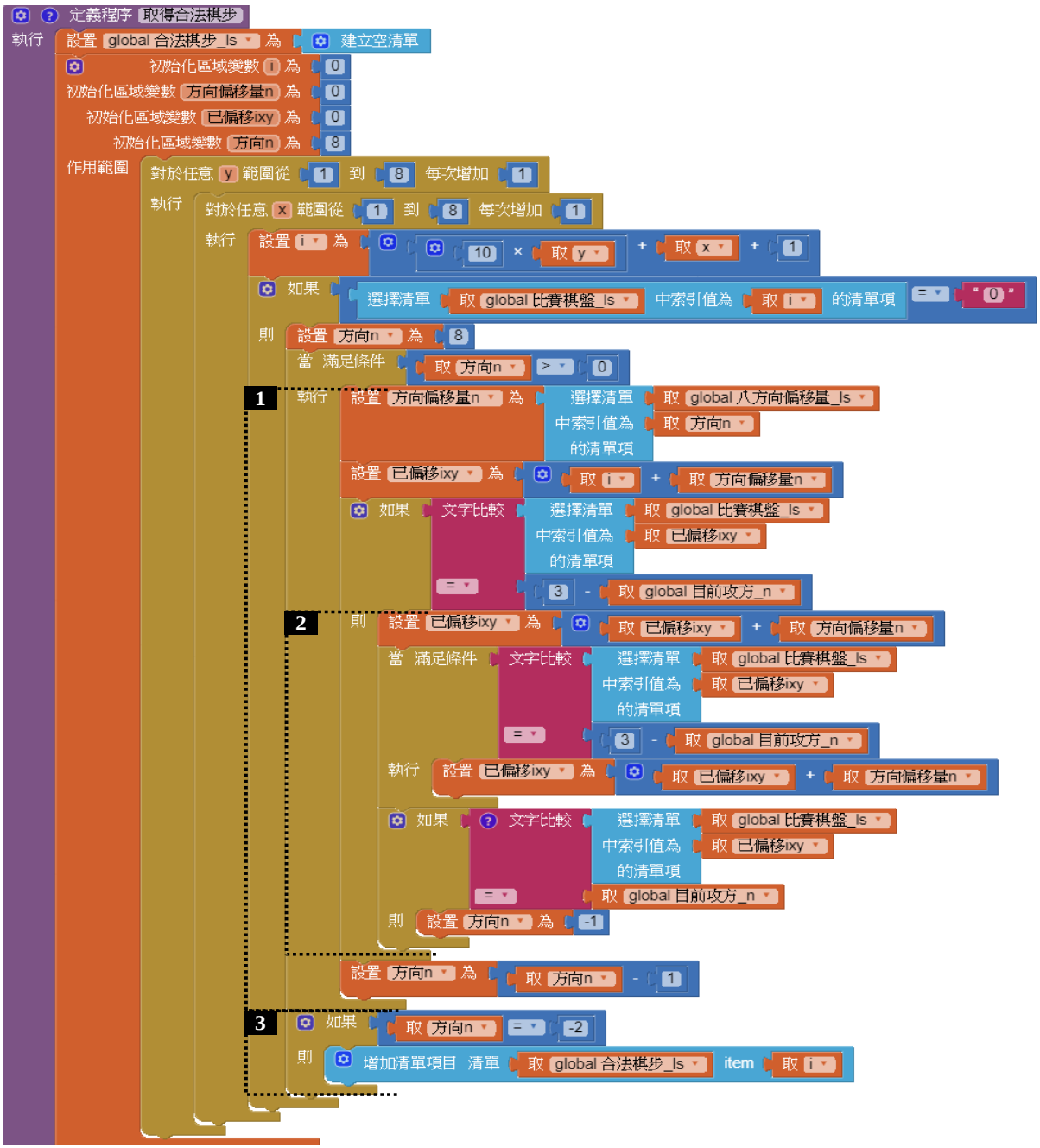


圖 2F-1

❶逐一找出棋盤上的空白棋格，在 8 個方向上檢視是否可翻子。首先朝第 1 個方向，其第 1 階段先偏移一步，檢驗是否為對手棋子，若是：則下一階段。

❷第 2 階段持續往前移動並檢視直到出現非對手棋子。檢驗是否為我方棋子，若是：則判定可翻子，設定跳脫不用再檢查其他方向。若非：則繼續檢視剩餘方向。

❸第 3 階段，是可翻子的情況跳脫則將棋子編號加入清單，不是：則免加。繼續檢視其他空白棋格。

❹全部空白棋格都檢視完畢後，依照黑子，白子之區別設定顯示合法棋步記號，如圖 2F-2。

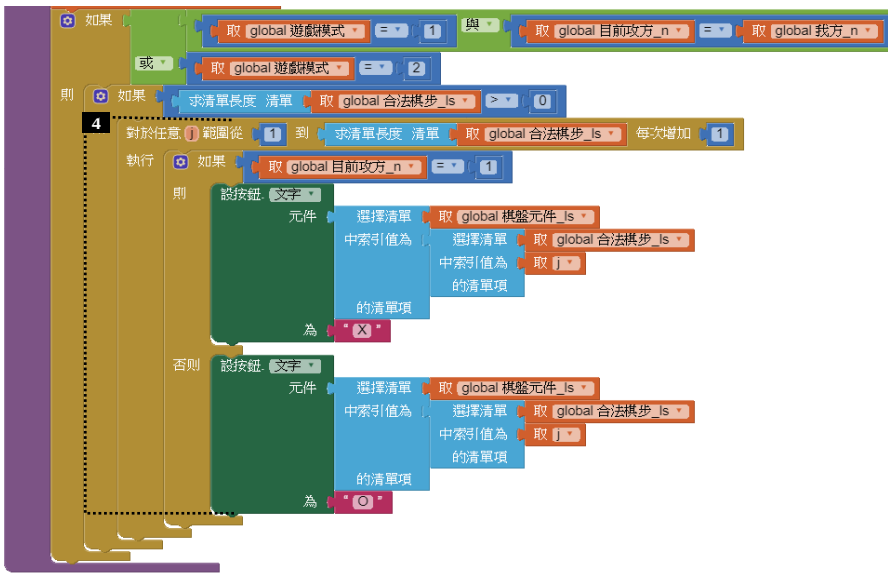


圖 2F-2

G. 取得行動力：

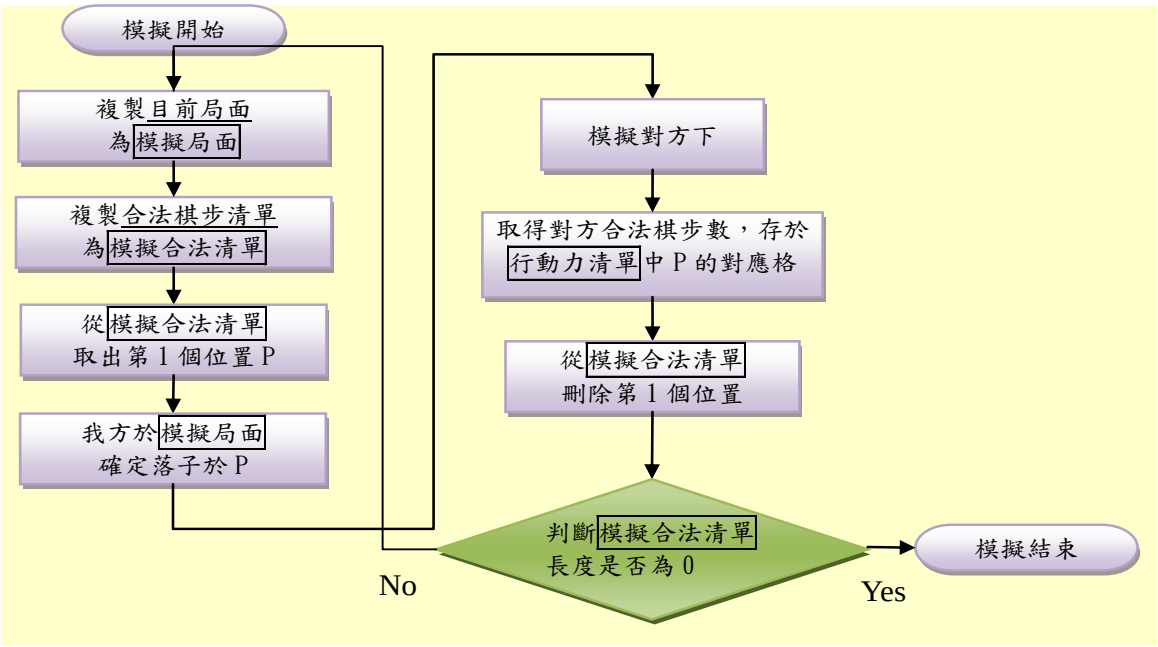


圖 2G-1 模擬前瞻一步以取得行動力流程圖

圖 2G-2

H. 確定落下棋子：(說明於下一頁)

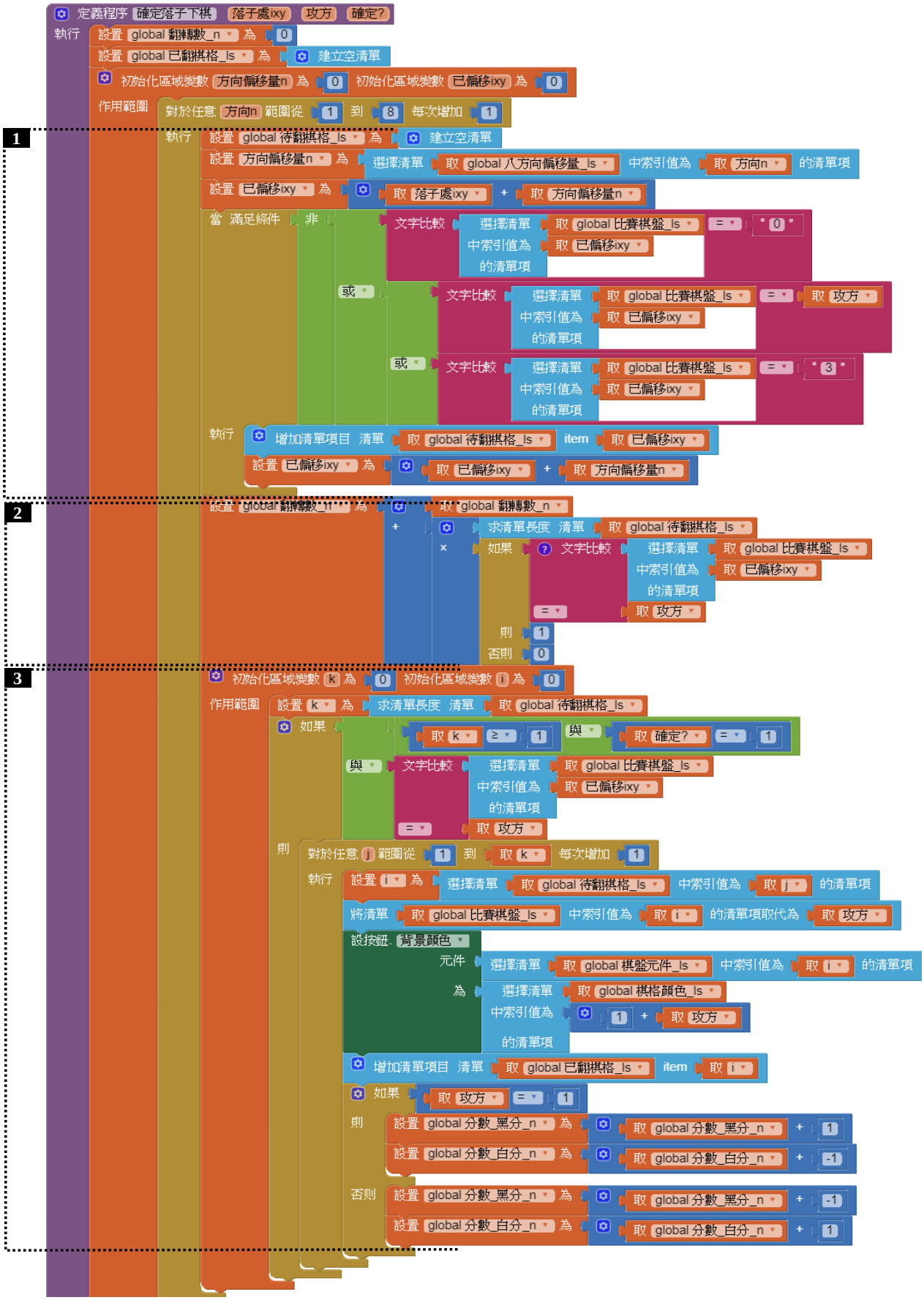


圖 2H-1

❶以確定落子處為中心點，在 8 個方向上檢視是否可翻子。首先朝第 1 個方向，其第 1 階段先偏移一步，檢驗是否為空位，或同色棋子，亦或棋盤邊界，否則為待翻棋子的候選清單。

❷將 1 個方向的可翻轉數加入翻轉數，等 8 個方向都累加起來後，便是此棋步的翻轉數。

❸將待翻棋格全部翻轉，並修正對應的棋子顏色以及比分，並記錄於已翻棋格清單。

❹修正落子處對應的比分，以及棋子顏色。

❺已確定落子，所有的合法棋步提示必須清除。

❻將已下棋步及已翻清單存起來並更新顯示。將比分顯示出來。

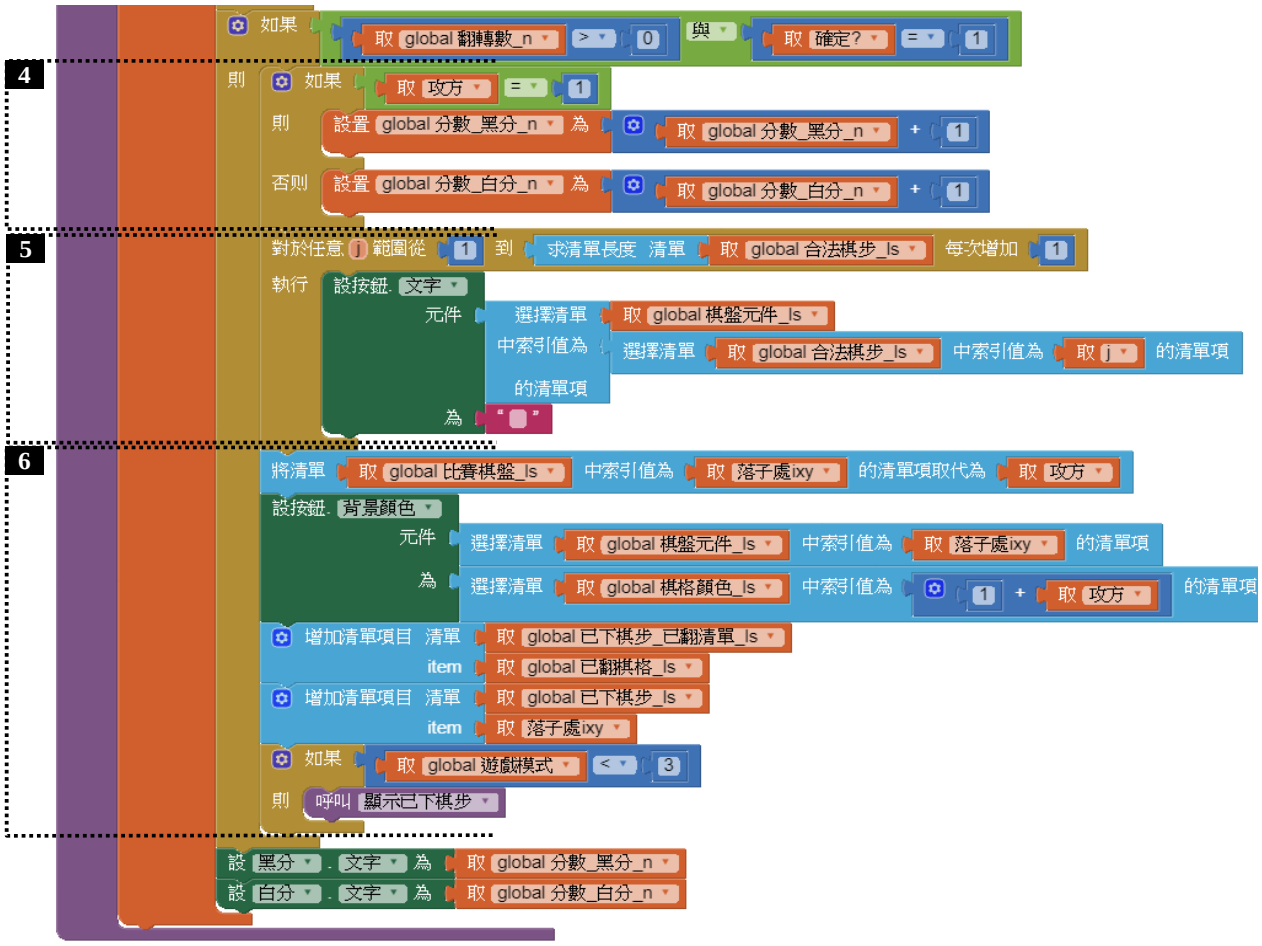


圖 2H-2

I. 換手：

❶確定落子後，先設定對手成為攻方並取得合法棋步清單，若對手有步可走便呼叫【AI 對手】決定落子，要是確定無步可走則應棄一手 PASS 。

❷PASS 後，再設定原來的一方為攻方並取得合法棋步清單，若原來的攻方有步可走便呼叫【AI 對手】決定落子，要是確定也無步可走，便是要結束棋局判定輸贏了。在此，【AI 對手】會依遊戲模式設定 AI 決定落子，還是等待真人棋手輸入決定。

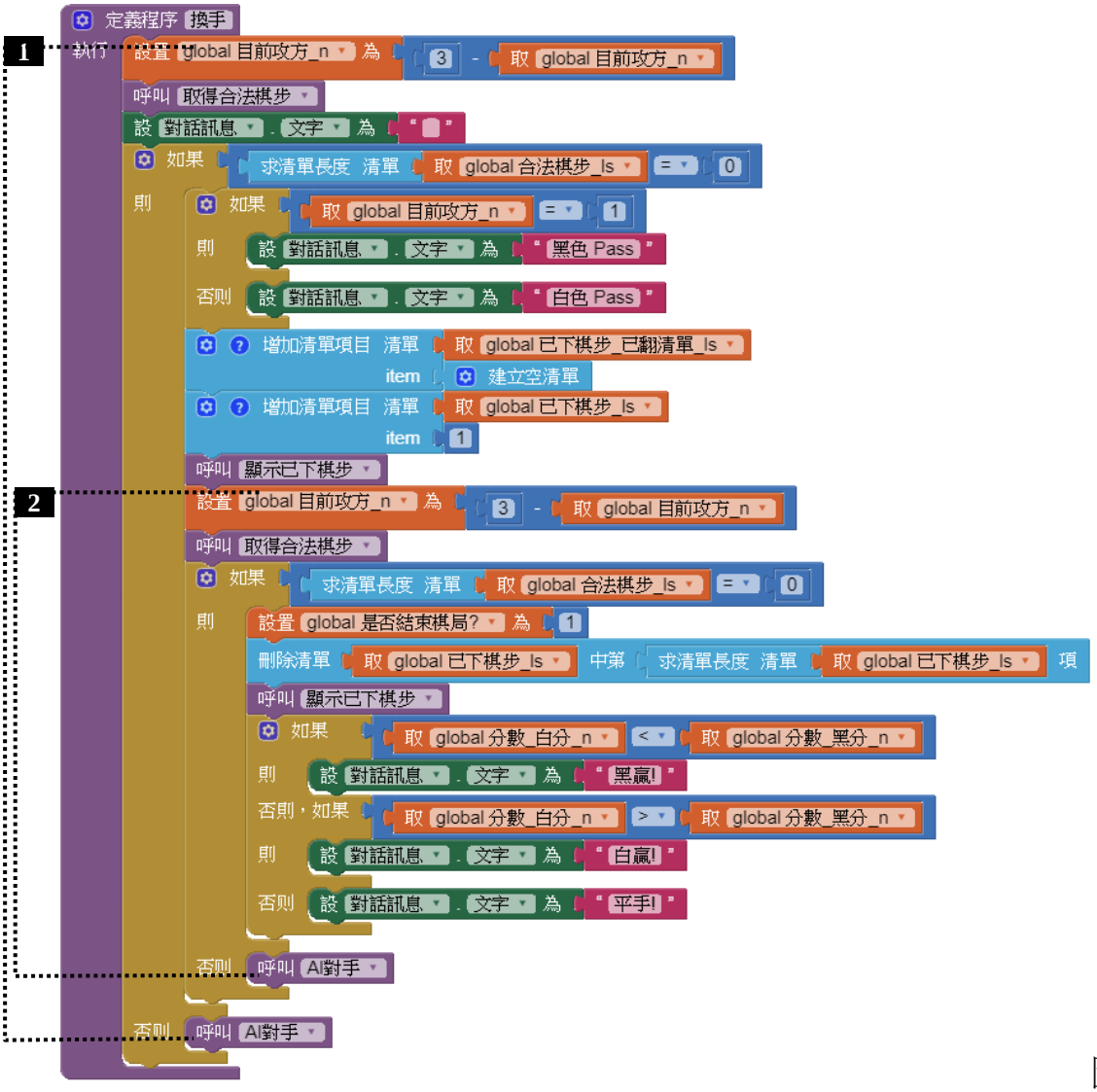


圖 2I

J. AI 陪練棋手：

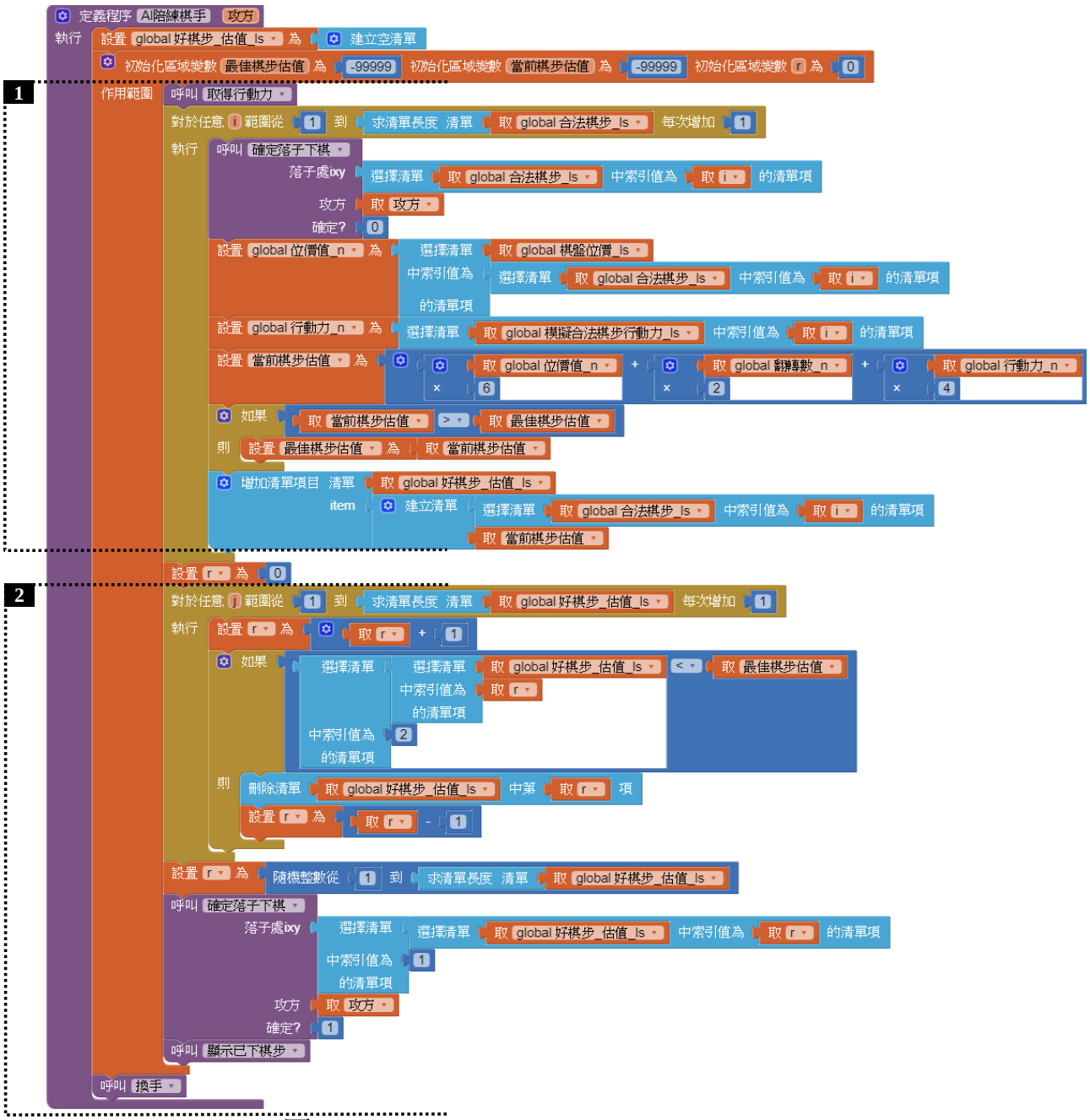


圖 2J

1 AI 陪練棋手，用合法棋步清單的棋子編號，呼叫【取得行動力】，並呼叫【確定落子下棋】但設定僅取得翻轉數，而不是真要確定落子。再查取對應的位價值及行動力，依照加權比例計算估值。依次將當下最佳的棋子編號及估值，存入清單。

2從好棋步_估值清單中留下有最佳棋步估值。若有很多個則已亂數擇一選取，並呼叫【確定落子下棋】，完成落子後呼叫【換手】

K. AI 最佳棋手：

1 AI 最佳棋手的演算程序和 AI 陪練棋手大致相同，唯一的不同是其依照棋局的進度，分成開局、盤中、和終局，在計算當前棋步估值時，給予位價值：翻轉數：行動力使用不同的加權，於開局前 20 手設為 6:1:4，中局第 21 手到 40 手設為 6:2:4，終局第 41 手以後設為 1:4:4。我們便可以安排了實驗讓兩個 AI 估值函式來對決，並統計兩者勝率。

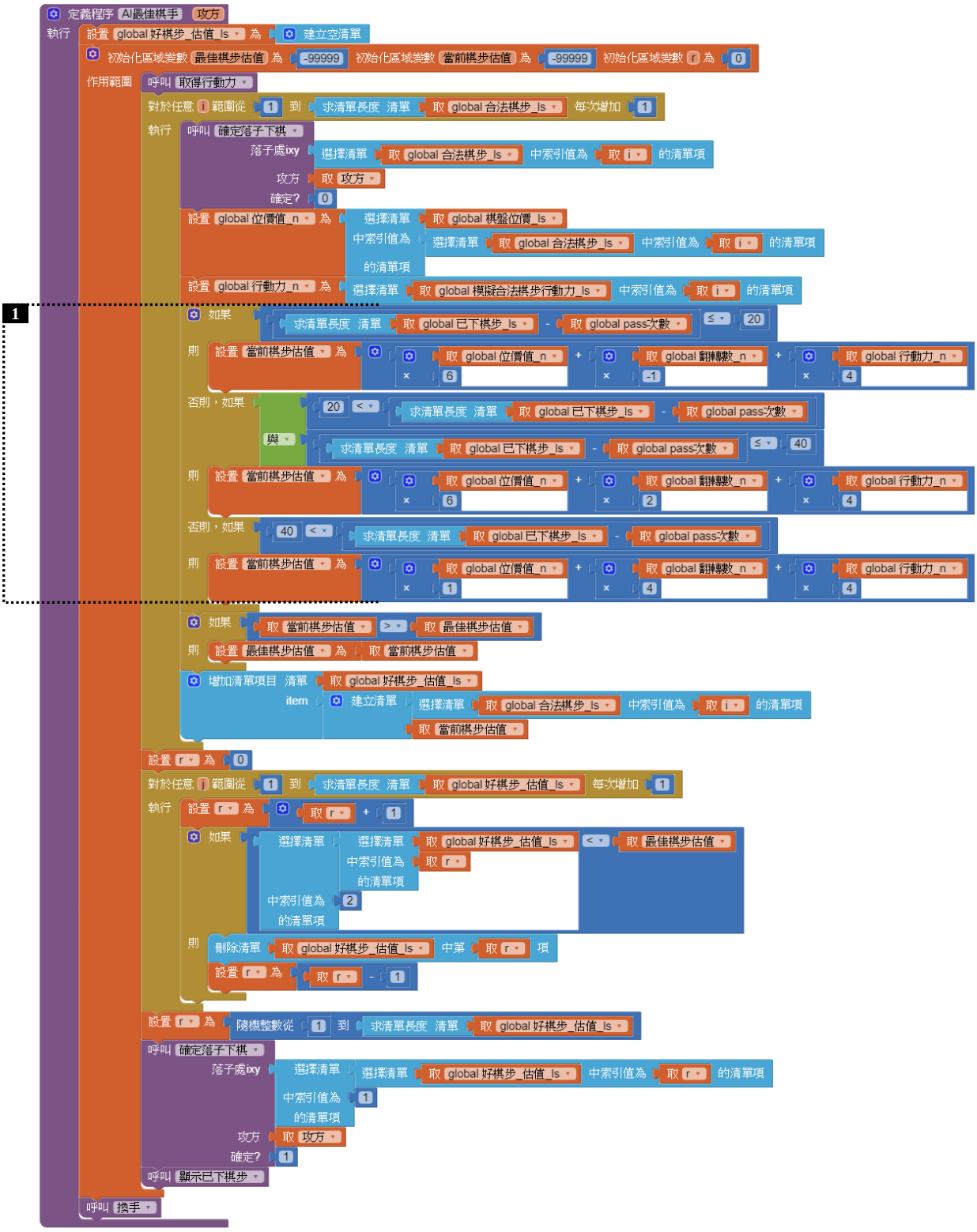


圖 2K

L. AI 對手：用以決定各個遊戲模式中的對手是誰，應由誰來選定下一手落子處。

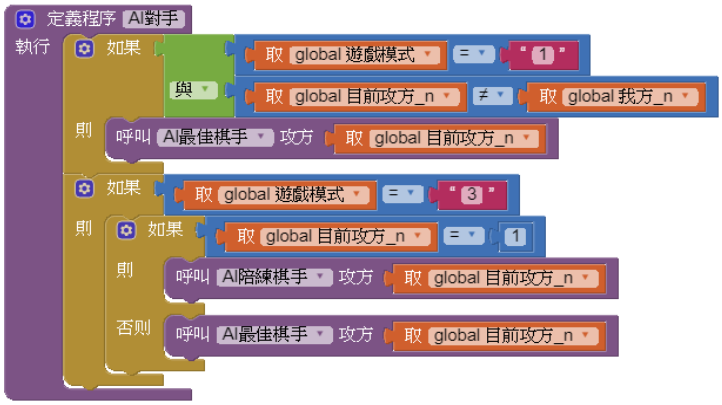


圖 2L

M. 清除顯示棋步：用遊戲結束後清除上一局的棋步顯示。

圖 2M

N. 顯示已下棋步：

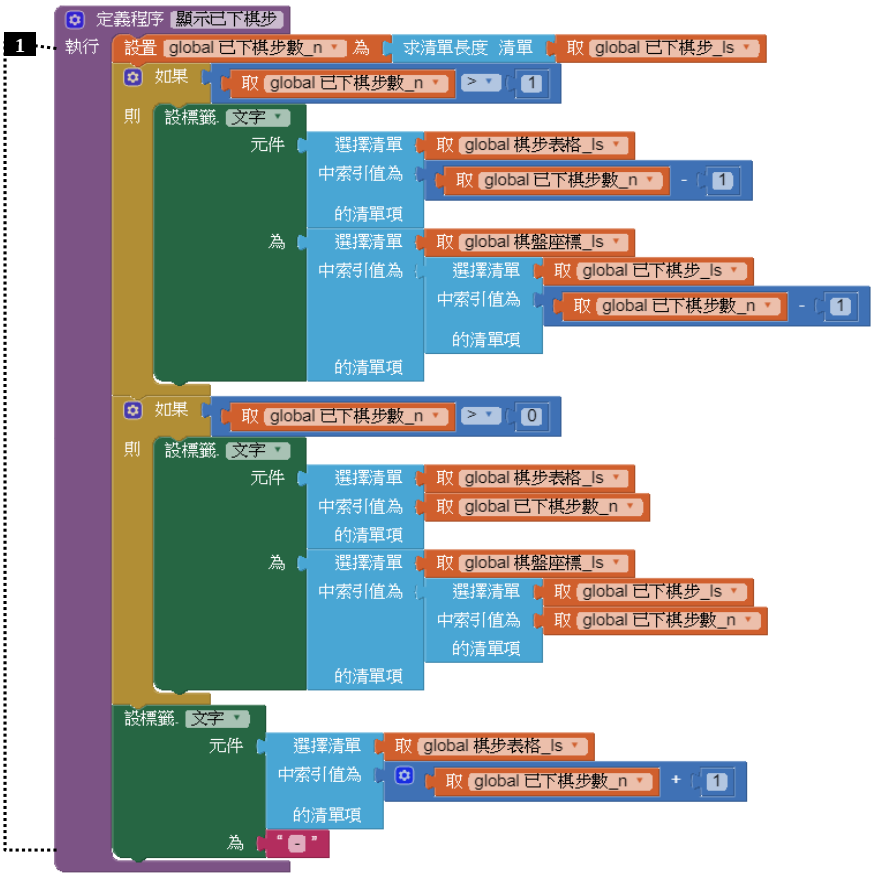


圖 2N

❶當已下棋步增加或減少一步，會同時更新最後一步和它下一步的顯示格。而僅有一步時又退一步，清單長度歸零，便只需清除第 1 格。

O. 回溯一步：❶回溯一步的第一件事先要將已經顯示的合法棋步提示清除。

圖 2O-1

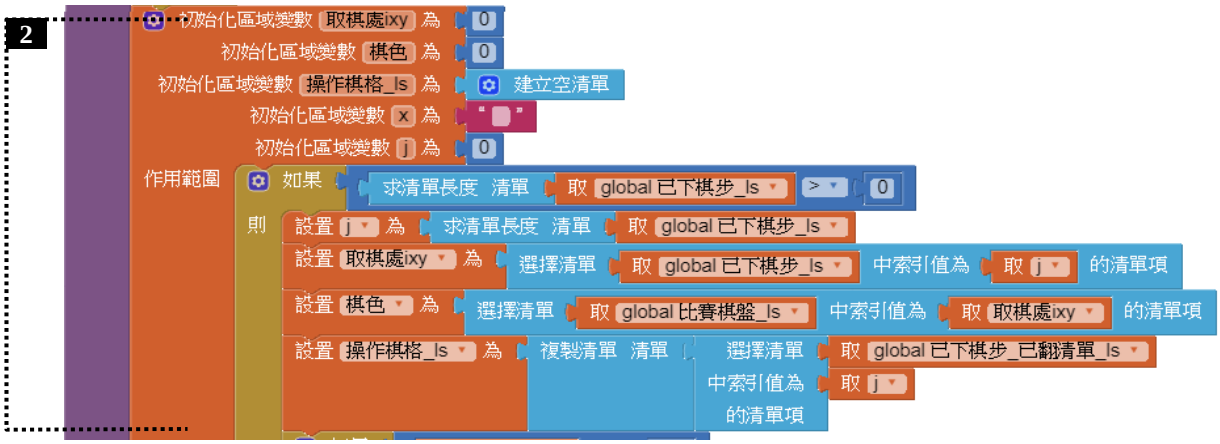


圖 2O-2

❷要取出的棋子是已下棋步的最後一手，並從比賽棋盤確認棋子顏色。還要將其對應的已翻清單取出，複製給操作棋格清單。

❸由操作棋格清單項目逐一將棋子由比賽棋盤清單修正，棋子顏色和比分也一併修正。

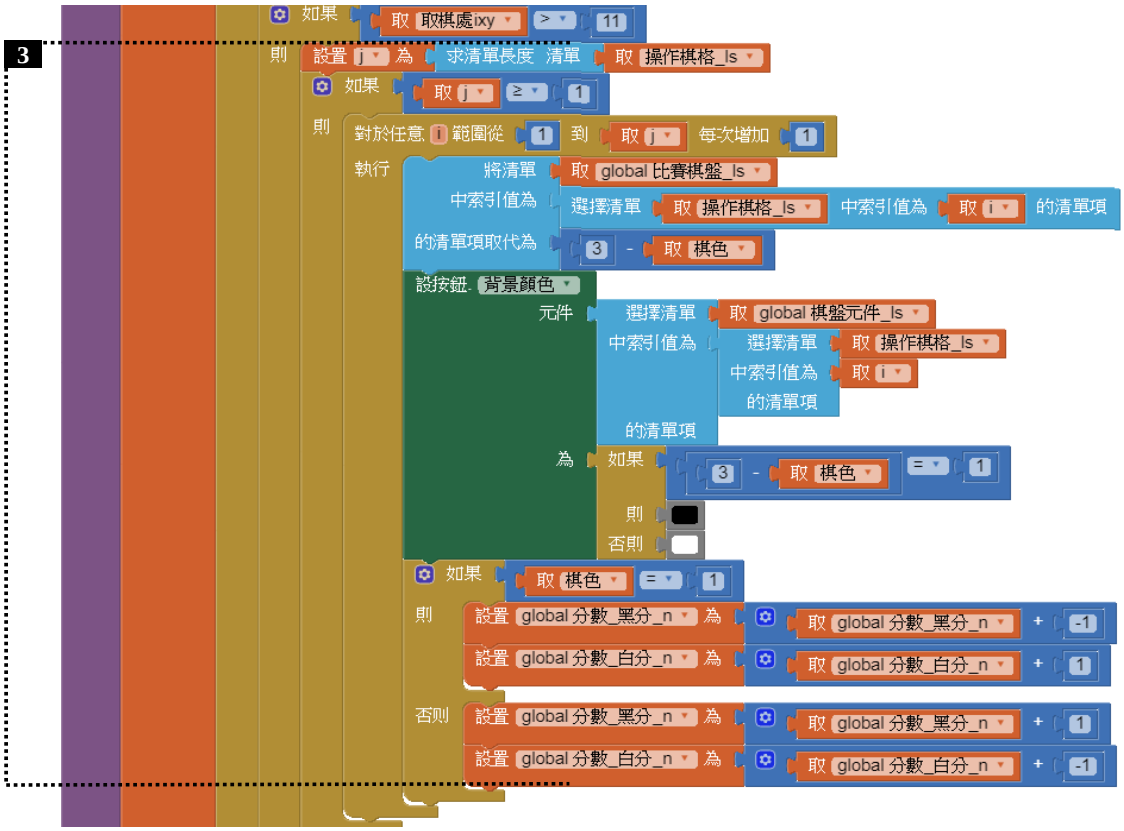


圖 2O-3

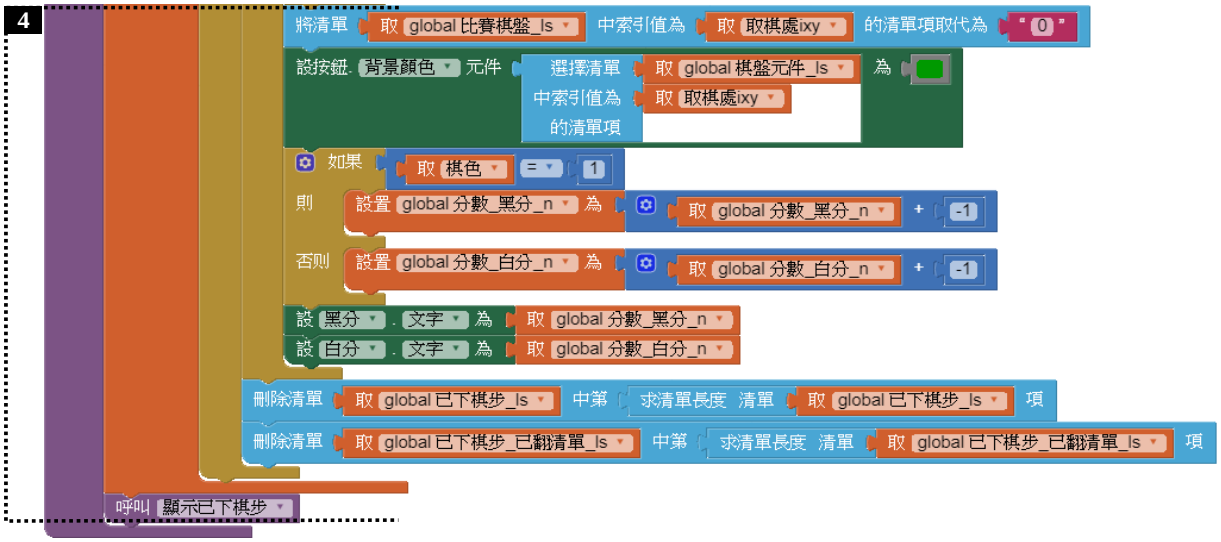


圖 2O-4

4將取棋處的棋盤格設為空格，並更正其顏色為空格的綠色，並依照取出得棋子顏色由比分中扣除，並顯示比分。最後同時將已下棋步清單及對應的已翻清單刪除最後一項，再呼叫【顯示已下棋步】就大功告成。

陸、 實驗與結果

實驗目標：依照實戰經驗開局應拓展行動力，終局應累進翻轉數，而位價值的準則應貫穿全場。計畫由對戰實驗中找出最佳的參數權值比例，並驗證。

實驗方法：給予兩個 AI 程式不同的參數權值設定使其對戰，並紀錄 100 組 AI 對戰的結果來計算勝率，用以比較優劣。

一、 AI 程式對戰實驗 1： 3 組參數權值功效實驗

黑方 位價值：翻轉數 100：1 單組參數權值 對戰

白方 位價值：翻轉數 100：1, 100：2, 100：4 三組參數權值

參數權值設定說明：

黑方：在計算當前棋步估值時，給予位價值：翻轉數 100：1。

白方：其依照棋局的進度，分成開局、盤中、和終局，在計算當前棋步估值時，給予 (位價值：翻轉數) 的比例使用不同的加權，於

開局第 1 手到第 20 手設為 位價值：翻轉數 100：1，

盤中第 21 手到第 40 手設為 位價值：翻轉數　100：2，

終局第 41 手以後設為位價值：翻轉數　100：4。

結果

單組參數權值的 AI 棋手(執黑子) 勝率為 36%，

設定前中後 3 組參數權值的 AI 棋手(執白子)的勝率為 61%。如圖 3-1。

結果：設為 3 組參數權值的 AI 棋手 可謂智能較高。

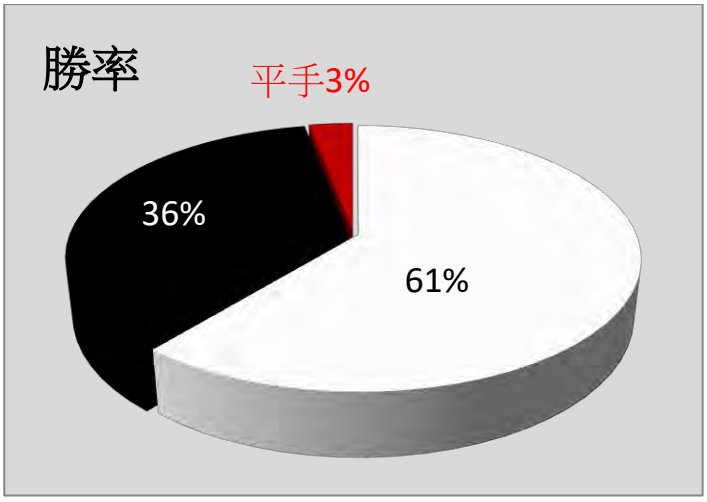


圖 3-1

二、 AI 程式對戰實驗 2：調整 (位價值：翻轉數) 之比例實驗

黑方 位價值：翻轉數 100：1 單組參數權值 對戰

白方 位價值：翻轉數 10：1 單組參數權值

結果

黑方：勝率為 43%， 白方：勝率為 57%，如圖 3-2。可知位價值：翻轉數不宜過高。

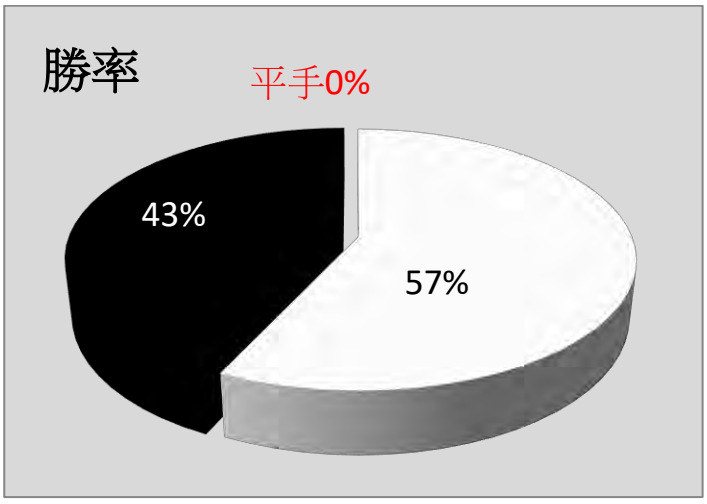


圖 3-2

三、 **AI 程式對戰實驗 3**：加入行動力參數實驗

黑方 位價值：翻轉數：行動力 10：1：0 單組參數權值 對戰

白方 位價值：翻轉數：行動力 10：1：1 單組參數權值

結果

黑方：勝率為 35%，白方：勝率為 62%，如圖 3-3。可知加入行動力可提升 AI 智能。

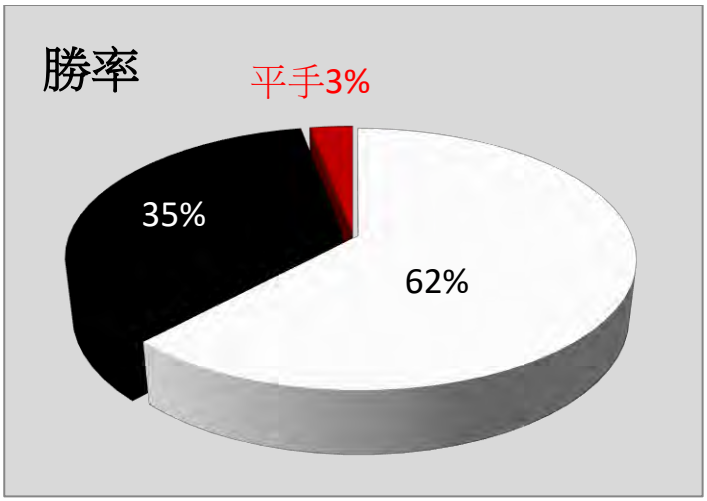


圖 3-3

四、 **AI 程式對戰實驗 4**：調整 (位價值：翻轉數：行動力) 之比例實驗

黑方 位價值：翻轉數：行動力 10：1：1 單組參數權值 對戰

白方 位價值：翻轉數：行動力 6：2：4 單組參數權值

結果

黑方：勝率為 32%，白方：勝率為 68%，如圖 3-4。

可知位價值：翻轉數：行動力 再微調 3 個參數權值比例可提升 AI 智能。

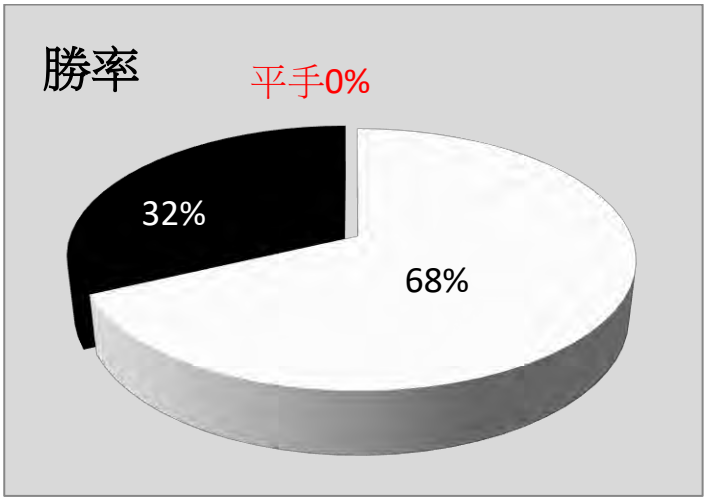


圖 3-4

五、 **AI 程式對戰實驗 5**：3 組參數權值 實驗

黑方 位價值：翻轉數：行動力 6：2：4 單組參數權值 對戰

白方 位價值：翻轉數：行動力 6：1：4, 6：2：4, 1：4：4 三組參數權值

參數權值設定說明：

黑方：在計算當前棋步估值時，給予位價值：翻轉數：行動力 6：2：4 單組參數權值。

白方：其依照棋局的進度，分成開局、盤中、和終局，在計算當前棋步估值時，給予 (位價值：翻轉數：行動力) 的比例使用不同的加權，於

開局第 1 手到第 20 手設為 位價值：翻轉數：行動力 6：1：4，

盤中第 21 手到第 40 手設為 位價值：翻轉數：行動力 6：2：4，

終局第 41 手以後設為 位價值：翻轉數：行動力 1：4：4。

結果

單組參數權值的 AI 棋手(執黑子) 勝率為 24%，

設定前中後 3 組參數權值的 AI 棋手(執白子)的勝率為 74%。如圖 3-5。

結果：設為 3 組參數權值的 AI 棋手勝出 具有較高的智能。

目前 最佳 AI 棋手參數權值比例：

開局第 1 手到第 20 手為 位價值：翻轉數：行動力 6：1：4，

盤中第 21 手到 40 手為 位價值：翻轉數：行動力 6：2：4，

終局第 41 手以後為 位價值：翻轉數：行動力 1：4：4。

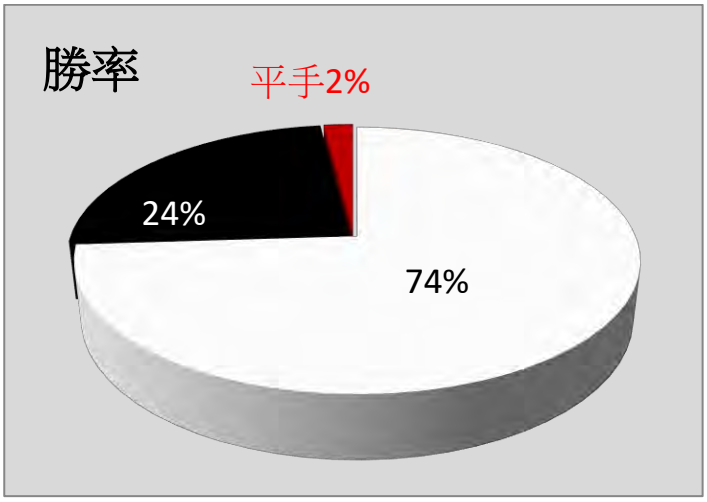


圖 3-5

柒、 討論與結論

一、MIT App Inventer 2 積木式程式語言的設計概念類似 Scratch 以程式積木塊堆疊而成，適合無程式碼編寫基礎的初學者，適合於國中小學階段學習使用。

二、AI 程式的估值函數參數有位價值、翻轉數和行動力。位價值是由預編的位置價值表清單查找，而翻轉數是核算合法棋步的翻子數量，行動力是核算前瞻一步時我方與敵方的合法棋步數差值。

三、App 下棋操作介面的設計，具有人與人、人與 AI 程式及 AI 與 AI 對戰等三種執行模式。

四、App 提供完整的棋步紀錄，合法棋步的提示、回溯上一步等功能，方便學者使用。

五、以內建的 AI 程式對戰進行實驗，用勝率來評估人工智慧程度及優劣，可藉以調校各參數的權值比例。

六、其中依棋局的前、中、後階段，給予參數不同的權數組合，藉以提高將 AI 棋手程式評估選棋的能力。依照實戰經驗開局應拓展行動力，終局應累進翻轉數，而位價值的準則應貫穿全場。

七、以內建的 AI 程式對戰作實驗，用勝率來評估人工智慧程度及優劣，結果可知：具有棋局前、中、後遊戲階段走法有不同加權者的 AI 智能較高。

八、本 App 最佳 AI 棋手參數權值比例：開局前 20 手為 位價值：翻轉數：行動力 6：1：4，盤中第 21 手到第 40 手為 位價值：翻轉數：行動力 6：2：4，終局第 41 手以後為 位價值：翻轉數：行動力 1：4：4。

捌、 未來展望

一、AI 程式可再試更多不同的階段區分，給予參數不同的權數組合，以提高評估選棋的能力。

二、未來還可將遊戲樹等相關 AI 走法搜尋演算法加入實作，來提高 AI 程式的評估選棋的能力。如：Negamax 負極大值搜尋法、α-β 剪枝搜尋法、Zobrist 雜湊置換表…等。

玖、 參考資料

1.黑白棋簡介。取自 http://www.twword.com/wiki/%E9%BB%91%E7%99%BD%E6%A3%8B。

2. Wzebra 黑白棋用詞及致勝策略。取自 http://radagast.se/othello/。

3.王曉華(2015)*演算法的樂趣：23 個程式設計必學主題與應用實例*。碁峰資訊。

4.文淵閣工作室(2016)。*手機應用程式設計超簡單—App Invento r 2 專題特訓班*。碁峰資訊。

【評語】032811

1. 使用 MIT App Inventer 2 實作黑白棋 APP。掌握黑白棋的數學模型，以資料結構的形式來建立下棋模式。該作品以現今 AI 概念應用於製作 MIT App Inventer 2 積木程式之黑白棋 APP，提供多方功能供學習者使用，其演算程式透過瀏覽器在雲端完成。

2. 了解黑白棋的規則與策略後，用 MIT App Inventor 做出三種對弈模式(人對人，人對 AI，AI 對 AI)的黑白棋 APP，有能力撰寫程式執行。

3. 能將黑白棋下棋策略分析後，以位價值、翻轉參數、行動力函數評分計算已決定下棋位置。模擬並記錄對手的合法棋步以計算出合適的棋步，具有不同組參數權值(位價值，翻轉數，行動力)的 AI 勝率較高。這三種評估函數在下棋的先後期有不同比例變化。利用實戰已獲得此三種函數之較佳權重比例。

4. 電腦下棋已是眾所皆知本作品主要以黑白棋為研究範圍。可惜沒有融合自動學習調整參數權重之功能在 AI-AI 對戰應可取的大量數據自動改變參數。未來能夠進一步加入機器學習，提高選棋能力。

作品海報

壹、研究動機

近幾年來人工智慧已然變成顯學，而最近圍棋程式 AlphaGo 戰勝職業冠軍棋士的消息更是令人感到驚艷。AlphaGo 是使用了蒙地卡羅樹搜尋，並藉助估值網路與走棋網路這兩種深度神經網路，它透過估值網路來評估大量選點，並透過走棋網路選擇落點。AlphaGo 的崛起，已經引發了 AI 人工智慧的學習熱潮。於是我們想試試，藉由實作另一種棋類程式黑白棋 App 來初步探索 AI 人工智慧的奧秘。

貳、研究目的

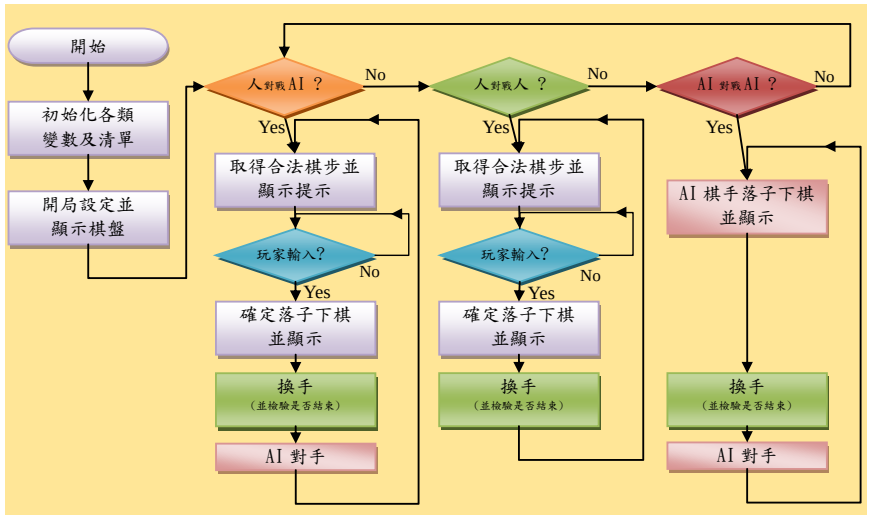
一、 探討黑白棋致勝策略，並以 App Inventor 2 實作。
二、 探討並實作估值函數。
三、 探討並調校估值函數之最佳參數權值。

參、研究器材

一、 Windows 10 作業系統、Google Chrome 網路瀏覽器、iOS 作業系統、Safari 網路瀏覽器、Word(文書軟體)、Excel(試算表軟體)。
二、 MIT App Inventor 2 積木式 Android 程式線上開發環境、Internet、Android 手機。

肆、研究及實作方法

系統流程圖



一、 主要介面配置說明：

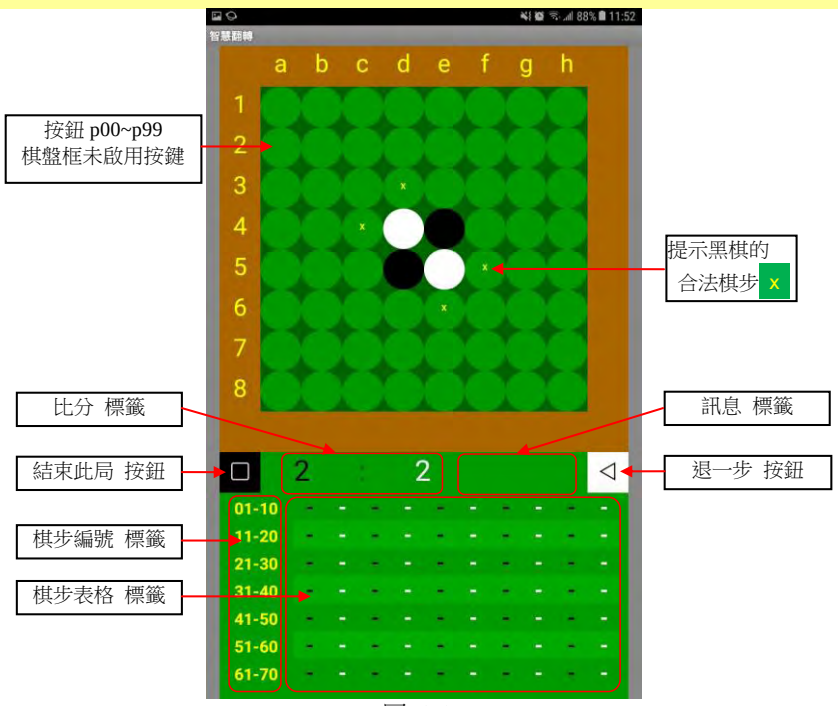


圖 1-1

圖 1-2 遊戲模式選單頁面
三種執行模式

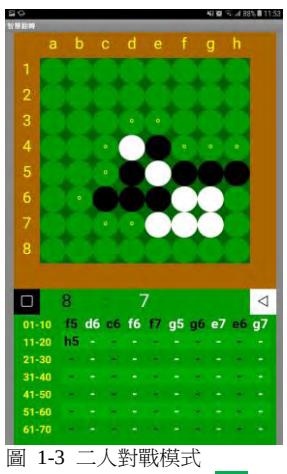


圖 1-3 二人對戰模式
提示白棋的合法棋步 o

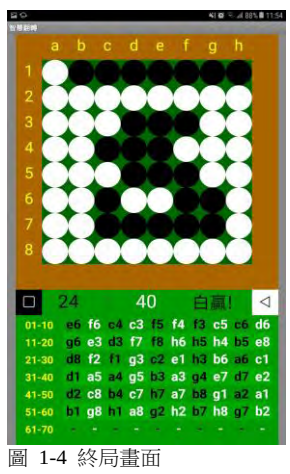


圖 1-4 終局畫面
判定最後比分及輸贏

二、程式功能說明：

A. 宣告全域變數：

1 各類常數清單與儲存格清單，是本程式最主要的資料結構與實作依據，於後詳細說明。

2 3 遊戲流程控制用的相關變數。

3 比賽棋盤清單用於紀錄當前局面狀態。

合法棋步清單用來記錄當時局面可下位置的棋子編號。

目前攻方設為 1 則由黑方下、反之目前攻方設為 2 則換白方下。

4 AI 棋手評估局面用的主要參數，再依不同的加權作為選定走法的依據。

位價值：落子位置的價值。

翻轉數：落子後對方棋子轉為己方的個數。

行動力：(落子前我方的合法棋步數)
減去(落子後對方的合法棋步數)

5 確認棋手落子的棋子編號後：

待翻棋格清單用以紀錄因所走棋步而需要翻子的棋子編號。

已下棋步-已翻清單 記錄所有棋步的棋子編號。

已翻棋子清單提供顯示及回溯之用。

6 模擬比賽棋盤清單用來取得當前狀態做模擬前瞻一步。

模擬對手合法棋步清單用來記錄前瞻一步後對手可下處。

模擬合法棋步行動力清單用以記錄己方所有合法棋步所對應的行動力。

B. 初始化各類變數及清單：

1 初始棋盤清單：

如圖 B1 為棋盤及棋子狀態的模型，本程式皆為一維清單，以模擬 Warren Smith 的模型實作棋盤，於此以二維說明。

外圍的白色方框處為棋盤邊界以 3 標記。中央 8x8 的範圍為棋格的位置，以 0 標記空位。在正中央 4 格黃色圓角框處，為初始棋盤上的 4 顆棋子：用 1 標記黑棋、2 標記白棋。

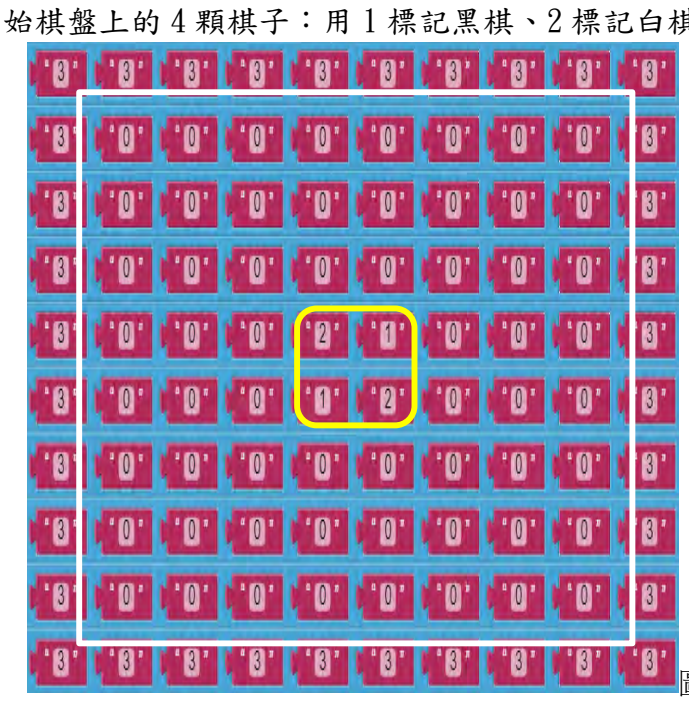


圖 B1

2 八方向偏移量清單：如圖 B2。

圖 B2

3 棋盤座標清單：對應於各棋格，用於將已下棋步表列顯示。

4 棋盤位價值清單：如圖 B3，依照棋盤上各棋格位置的重要性，預先編定位價值表。

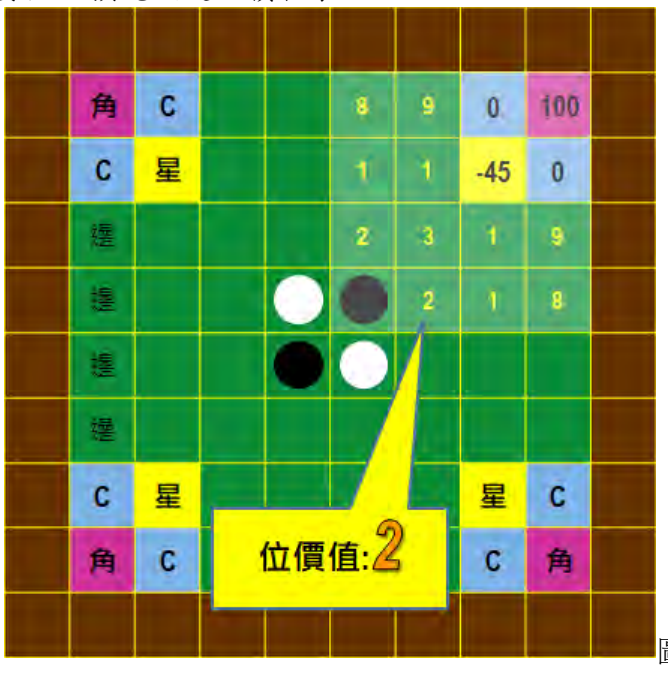


圖 B3

肆、研究及實作方法(續)

二、程式功能說明(續)：

D. 更新顯示棋盤：依據比賽棋盤清單的現況，更新棋盤畫面(圖略)

E. 開局設定：初始化各項動態清單，並顯示合法棋步(圖略)

F. 取得合法棋步：

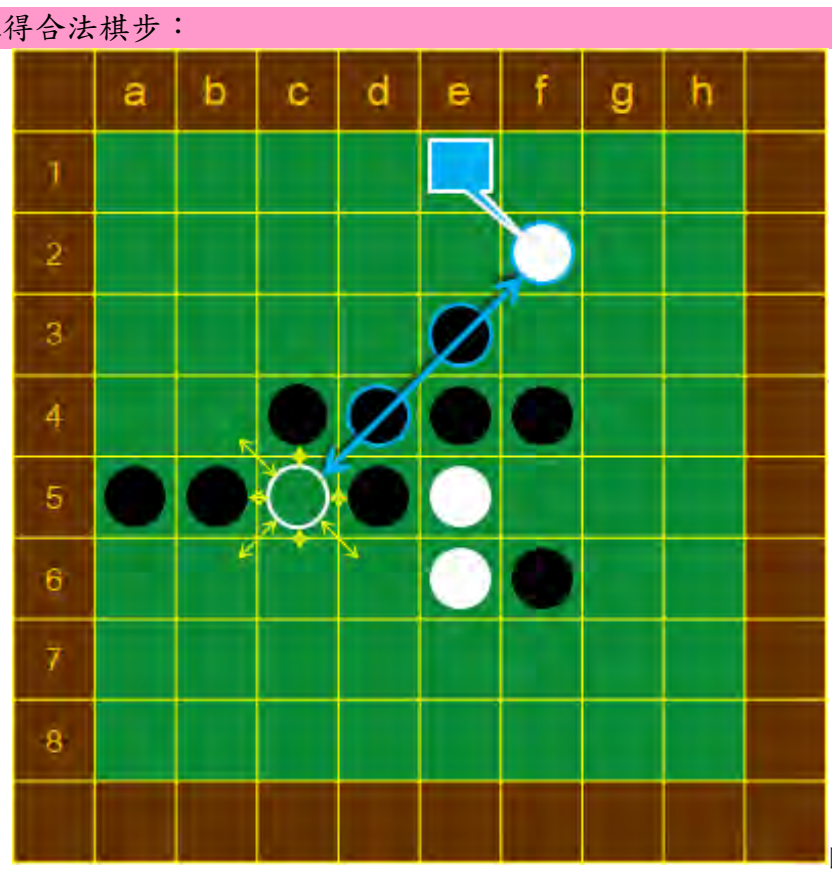


圖 F

❶逐一找出棋盤上的空白棋格，在 8 個方向上檢視是否可翻子。首先朝第 1 個方向，其第 1 階段先偏移一步，檢驗是否為對手棋子，若是：則下一階段。

❷第 2 階段持續往前移動並檢視直到出現非對手棋子。檢驗是否為我方棋子，若是：則判定可翻子，設定跳脫不用再檢查其他方向。若非：則繼續檢視剩餘方向。

❸第 3 階段，若是可翻子的情況跳脫則將棋子編號加入清單，若非則免加。繼續檢視其他空白棋格。

❹全部空白棋格都檢視完畢後，依照黑子，白子之區別設定顯示合法棋步記號。

G. 取得行動力：

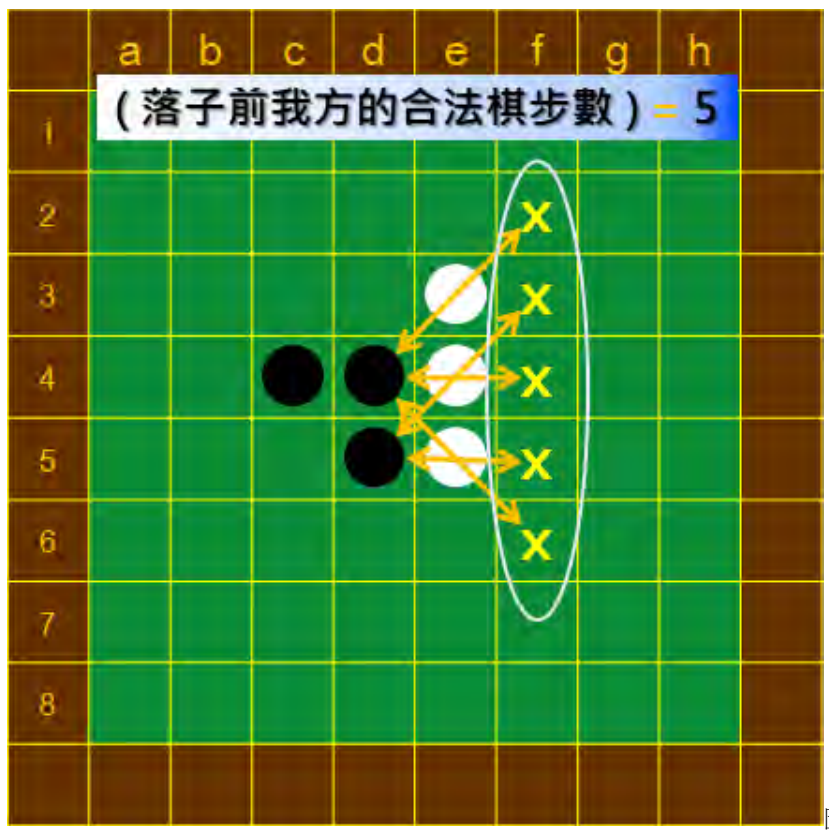


圖 G1

圖 G2

H. 確定落下棋子：

圖 H

❶以落子處為中心點，在 8 個方向上檢視是否可翻子。
首先朝第 1 個方向，其第 1 階段先偏移一步，檢驗若是對方棋子則列入待翻棋子的候選清單並繼續偏移。

❷若再偏移一步變成攻方棋子則將此方向的待翻棋子數加入翻轉數，等 8 個方向都累加起來後，便是此棋步的翻轉數。

❸將待翻棋格翻轉，修正棋子顏色以及比分，並記錄於已翻棋格清單。

❹修正落子處對應的比分，以及棋子顏色。

❺已確定落子，所有的合法棋步提示必須清除。

❻將已下棋步及已翻清單存起來並更新顯示並將比分顯示出來。

I. 換手：檢驗換手後是否無步可走，若雙方皆無步則結束此局(圖略)

J. AI 陪練棋手： 位價值：翻轉數：行動力為 6:2:4 (圖略)

K. AI 最佳棋手：

❶ AI 最佳棋手的演算程序和 AI 陪練棋手大致相同，唯一的不同是其依照棋局的進度，分成開局、盤中、和終局，在計算當前棋步估值時，給予位價值：翻轉數：行動力使用不同的加權，於開局前 20 手設為 6:1:4，盤中第 21 手到 40 手設為 6:2:4，終局第 41 手以後設為 1:4:4。我們便可以安排了實驗讓兩個 AI 估值函式來對決，並統計兩者勝率。

圖 K1

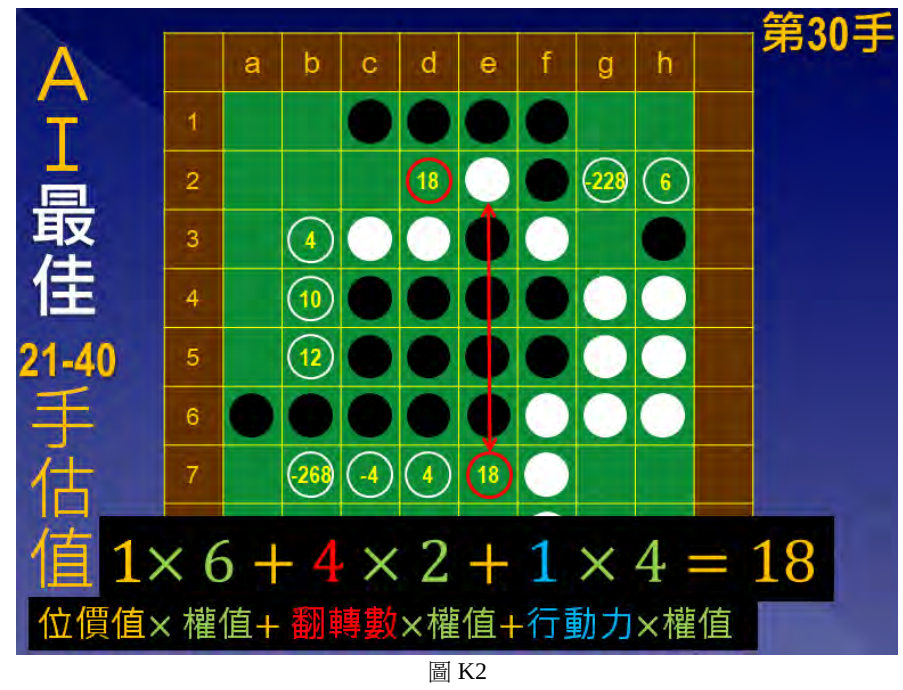


圖 K2

L. AI 對手：用以決定遊戲中的 AI 對手是誰來選定落子處 (圖略)

M. 顯示已下棋步：(圖略)

N. 回溯一步：(圖略)

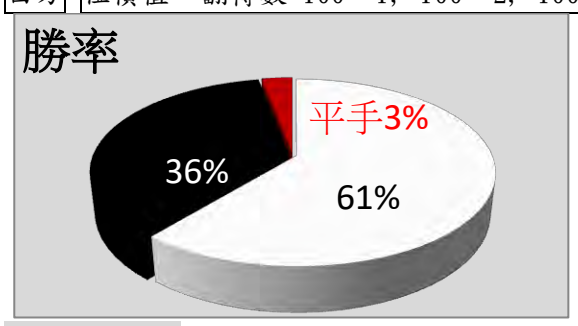
伍、實驗與結果

實驗目標：

依照實戰經驗行動力的準則應貫穿全場，開局主要參考位價值，而終局應積極累進翻轉數。計畫由對戰實驗中找出最佳的參數權值比例，並驗證。給兩個 AI 程式不同的參數權值設定使其對戰，並紀錄 100 組結果來計算勝率以比較優劣。

實驗 1 三組參數權值功效實驗

黑方 位價值：翻轉數 100：1 單組參數權值 對戰
白方 位價值：翻轉數 100：1，100：2，100：4 三組參數權值



黑方 勝率為 36%

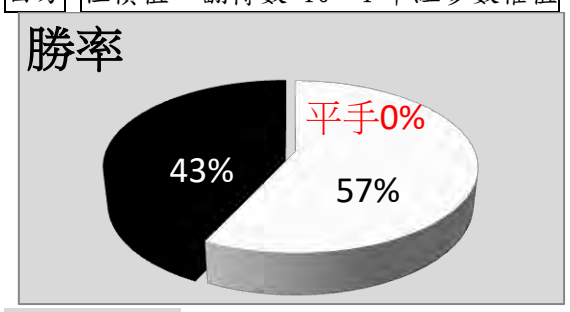
白方 勝率為 61% 勝出

實驗 1 結果: 翻轉數在遊戲的越後段，所佔的比例應越重。在不同的遊戲階段，使用不同的權數比例，確實可提升 AI 智能。

討論: 將翻轉數的比例值漸提高後，可提高 AI 智能。所以我們認為也許是 100:1 的位價值所占比例太高。

實驗 2 調整（位價值：翻轉數）之比例實驗

黑方 位價值：翻轉數 100：1 單組參數權值 對戰
白方 位價值：翻轉數 10：1 單組參數權值



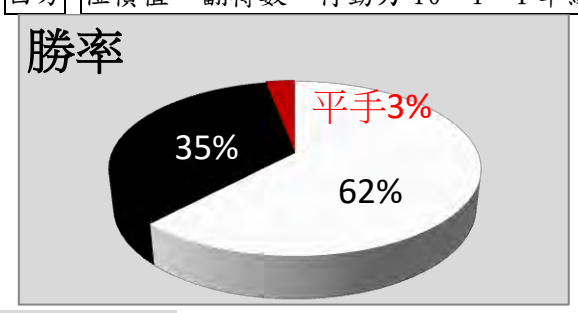
黑方 勝率為 43%

白方 勝率為 57% 勝出

實驗 2 結果:在實驗 2 之前，有測試單組 100:4、100:8 勝率有提高的趨勢，所以我們以 100:10 作為實驗的比例。結果確認位價值比例應調到 10:1 左右為最佳。

實驗 3 加入行動力參數實驗

黑方 位價值：翻轉數：行動力 10：1：0 單組參數權值 對戰
白方 位價值：翻轉數：行動力 10：1：1 單組參數權值



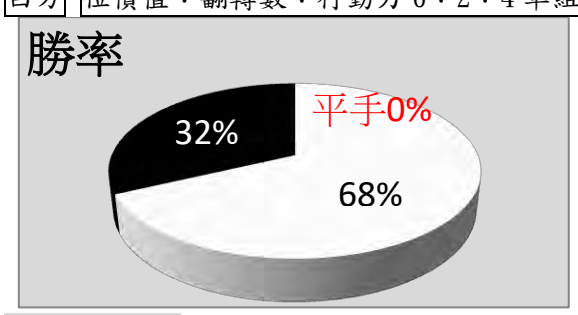
黑方 勝率為 35%

白方 勝率為 62% 勝出

實驗 3 結果: 加入行動力之後，能確實提高 AI 的智能。

實驗 4 調整(位價值：翻轉數：行動力）之比例實驗

黑方 位價值：翻轉數：行動力 10：1：1 單組參數權值 對戰
白方 位價值：翻轉數：行動力 6：2：4 單組參數權值



黑方 勝率為 32%

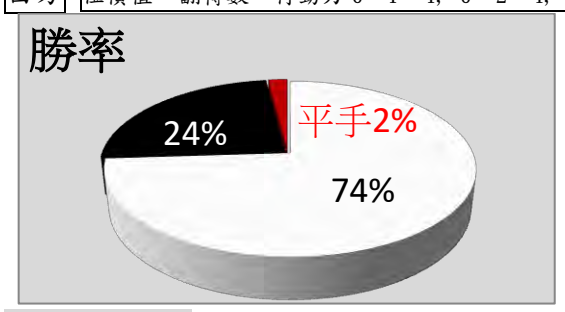
白方 勝率為 68% 勝出

實驗 4 結果: 微調之後，勝率又再次提升。

討論: 首先我們將行動力的比例由 10：1：2，10：1：4 逐漸調高，也得到逐漸增加的勝率。最後發現加入行動力之後，位價值與翻轉數的比例可再微調為 6：2 來提升勝率。

實驗 5 最終調校配成三段參數權值 實驗

黑方 位價值：翻轉數：行動力 6：2：4 單組參數權值 對戰
白方 位價值：翻轉數：行動力 6：1：4，6：2：4，1：4：4 三組參數權值



黑方 勝率為 24%

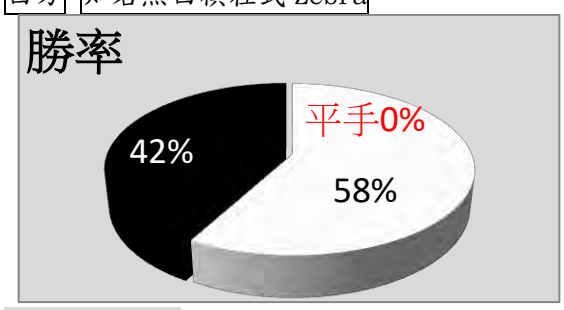
白方 勝率為 74% 勝出

實驗 5 結果：設為 3 組參數權值的 AI 棋手勝出 具有較高的智能。

討論:我們將盤中設為與原本單組相同的比例 6:2:4，其他部分的全值再做相對應的調整。由於行動力目前只前瞻一步，參考價值不如預期，目前以設為 4 為最佳。在開局時翻轉數應比盤中低故設為 1；終局時翻轉數應比盤中高故設為 4。最後位價值要降低，因為棋盤所剩的空格並不多且應積極搶分，所以位價值設為 1，翻轉數提升為 4。

實驗 6 AI 最佳棋手與坊間黑白棋程式對戰實驗

黑方 AI 最佳棋手 3 組參數權值 對戰
白方 知名黑白棋程式 Zebra



黑方 勝率為 42% 落敗

白方 勝率為 58%

實驗 6 結果:實驗後，我們發現 AI 最佳棋手與搜尋深度為 1 步的 Zebra 其勝率只差 16%。

實驗 7 AI 最佳棋手與坊間黑白棋程式 Zebra 評等實驗

實驗 7 結果: 以 Othello quest 評等，其核心為 ELO 系統，同樣搜尋深度為 1 在全部 30 個等級跨距中，Zebra 等級為 25，本 App 的 AI 最佳棋手等級為 26，棋力相近。

總結：

經過一連串的對戰實驗，我們一開始發現有前、中、後三組權數的 AI 較單組的為佳。而位價值與翻轉數之間的比例經調校後可得最佳的 10:1，但加入第三個參數行動力之後須再微調位價值與翻轉數的比例來更加提升 AI 智能。最後再調配成前、中、後三段便得到目前最佳 AI 棋手參數權值比例為:

開局第 1 手到第 20 手為 位價值：翻轉數：行動力 6：1：4，
盤中第 21 手到 40 手為 位價值：翻轉數：行動力 6：2：4，
終局第 41 手以後為 位價值：翻轉數：行動力 1：4：4。

並經由 Othello quest 進行評等，本 App 的 AI 最佳棋手等級為 26，同樣搜尋深度為 1 的 Zebra 等級為 25，在全部 30 個等級跨距中棋力可稱相近。

陸、結論

1. 本研究的 AI 程式的估值函數有位價值、翻轉數和行動力三種參數。
2. 本研究的 App 下棋操作介面的設計，具有人與人、人與 AI 程式及 AI 與 AI 對弈三種執行模式。
3. 本研究的 App 提供完整棋步紀錄，合法棋步提示、回上一步等功能，方便學習者使用。
4. AI 程式對戰作實驗結果可知：棋局前、中、後遊戲階段走法有不同加權者的 AI 智能高於單一階段的。
5. 本 App 最佳 AI 棋手參數權值比例：
   開局前 20 手為 位價值：翻轉數：行動力 6：1：4，
   盤中第 21 到第 40 手為位價值：翻轉數：行動力 6：2：4，
   終局第 41 手以後為位價值：翻轉數：行動力 1：4：4。
6. 以 Othello quest 進行評等，同樣搜尋深度 1 的時候，本 App 的 AI 最佳棋手等級為 26，Zebra 的等級為 25，棋力相近。

柒、未來展望

1. 未來可以做其他的階段劃分及加權數調配，以實驗出更佳的估值函數。
2. 還可將遊戲樹等相關 AI 走法搜尋演算法加入實作，如：Minimax 極大極小值搜尋法、$\alpha-\beta$ 剪枝法…等，以提升 AI 的智能。

捌、參考資料

1. 王曉華(2015)。*演算法的樂趣：23 個程式設計必學主題與應用實例*。碁峰資訊。
2. 文淵閣工作室(2016)。*手機應用程式設計超簡單—App Inventor 2 專題特訓班*。碁峰資訊。