八層三角殺棋之解法

Solving Eight Layer Triangular Nim

林宏軒
國立交通大學資訊科學與工程研究所
stanleylin.csie92@nctu.edu.tw

吳毅成
國立交通大學資訊科學與工程研究所
icwu@csie.nctu.edu.tw

單益章
國立交通大學資訊科學與工程研究所
shang@ms.ltcvs.ilc.edu.tw

摘要

在[3][4]論文中，許舜欽教授解掉所有七層三角殺棋。在本論文中，我們提出一種回溯分析演算法來解八層三角殺棋。這演算法有效利用三角殺棋勝負比例不均的狀況，來加快演算法速度；採用此法，只需要花 2.6 小時即可解掉所有八層盤面。此外，這演算法可以適用於 32 位元的電腦環境中。最近，白聖群[1]與我們同時獨立解出八層三角殺棋，但其方法需要 17.1 時，並需要利用 64 位元的環境來解。

關鍵詞: 三角殺棋、回溯分析、倒推法

Abstract

In the past, Hsu [3][4] solved all Triangular Nim positions with side size seven, In this paper, we propose a new retrograde algorithm to solve all Triangular Nim positions with side size eight. This algorithm makes use of unbalanced ratios of win and loss to reduce greatly the computation time to 2.6 hours. In addition, this algorithm also fits into the traditional 32-bit operating systems. In contrast to another independent result [1], they required 64-bit operating systems and took about 17.1 hours.

Keyword: Triangular Nim, Retrograde.

一、介紹

三角殺棋為一個兩人遊戲，在排成正三角形(如圖 1-1)的數顆棋子中兩人輪流取子，每次至少取一顆，取到最後一顆的人輸。

取子被限定為三個方向：橫向、左上-右下、右上-左下，取子必須相連，意即不可取到已被取走的子，每次取子數目不設限。

k 層三角殺棋是指此正三角形的邊長為 k。例如：傳統坊間的下法，大都是五層。五層三角殺棋共有 15 顆子；七層三角殺棋共有 28 顆子；八層三角殺棋共有 36 顆子。圖 1-1 是八層三角殺棋。

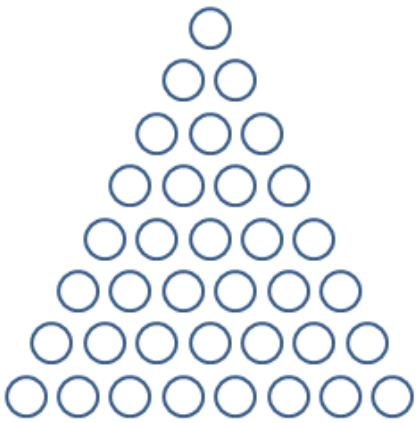

圖 1-1、八層三角殺棋完整盤面

在 1982 年及 1985 年由許舜欽教授發表了兩篇有關三角殺棋的論文[3][4]，利用倒推法解掉七層三角殺棋的勝負問題。到 2008 年，我們在「遊戲對局理論」課程中，即已得出此論文結果。在 2009 年白聖群[1]與我們獨立解出八層三角殺棋。

三角殺棋的程式解法皆是由下往上、由空盤面跑至全滿盤面，許舜欽教授與白聖群使用的倒推法在每個盤面需要往下抓取一層結果，而我們使用的回溯分析則是往上做更新。

回溯分析的優點是，我們只需對輸的盤面做往上更新的動作，而三角殺棋是一個敗率低的遊戲，由表 4-1 可以知道八層三角殺棋中輸的盤面只佔 6%，有 94%盤面可以略去更新的動作。

在八層三角殺棋中共有 36 顆子，需要 8G 的空間儲存所有結果，在倒推法中需要隨時對這些結果做存取，若是記憶體不足以儲存，效率將會降低許多。回溯分析則不用，在 2G 記憶體環境下仍有很好的效率。

白聖群[1]實作出的倒推法解掉八層三角殺棋需要 17.1 小時，而我們實作的回溯分析只需要 2.6 小時。

二、基礎理論

本章會介紹在寫三角殺棋時所需用到的一些基本程式架構，以及倒推法的實作。

2.1 基本程式架構

由於八層三角殺棋共有 36 顆子，可用 36 個 bits 記錄一個盤面的狀況，無子的 bit 設為 0，有子的 bit 設為 1，每個盤面皆對應到一個二進位數，如圖 2-1，其數字可以用一個長整數（兩個 32-bit 整數）來儲存。

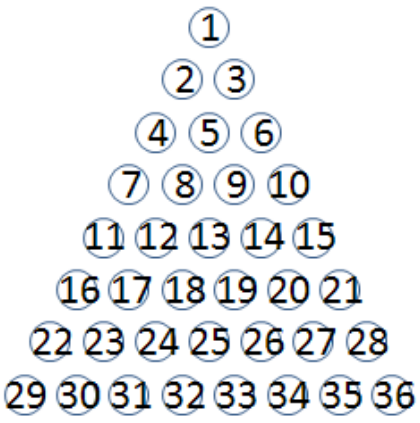

圖 2-1、每顆子分別對應到第幾個 bit

由於三角殺棋沒有和棋的可能，每個盤面不是勝就是負，因此可以用 1 個 bit 表示其勝負資訊，對於如八層三角殺棋共有 36 顆子，總計 2^36 種盤面，需要 2^36 個 bits 儲存所有結果，也就是 8G 的空間。

每種取子方式皆是屬於某種盤面，因此可以像儲存盤面一樣用一個長整數儲存。取子方式如七層三角殺棋最多有 196 種取子方式，八層三角殺棋最多有 288 種取子方式，這些可以在事前計算好，以節省程式時間。

每個盤面的母盤面及子盤面可用 AND 及 OR 運算得出。一個盤面經過合法的取子方式取子後可以得到其子盤面，而相反地若將子放回去可得母盤面，遊戲中雖然沒有取得母盤面的相關規則，但在程式中可能會用到。

對於合法的取子方式可以將盤面與取子的方式做 AND 運算，若與代表取子方式的長整數相同則表示此取子方式為合法。對於放子方式則是透過盤面與放子方式做 AND 運算，若為 0 則代表此放子方式為合法。

對於母盤面的取得可以將盤面及合法放子方式做 OR 運算，對於子盤面的取得可以將盤面及 NOT 後的合法放子方式做 AND 運算。

2.2 倒推法

在先前幾篇三角殺棋的相關論文研究中皆是使用倒推法來實作，按照每個盤面所對應到的二進位數大小，由小到大(由空盤面到全滿盤面)分別計算出每個盤面先手的勝負並存起來，每個盤面只需往下一層抓取其所有子盤面先手的勝負即可決定自己盤面是先手勝或先手負。

對於所有子盤面皆是先手勝的狀況可以得知自己為先手負。對於至少有一個子盤面為先手負的狀況則可以判定自己為先手勝。如圖 2-2 中，對應到數字為 7 的盤面必須往下抓取子盤面以決定自身勝負。

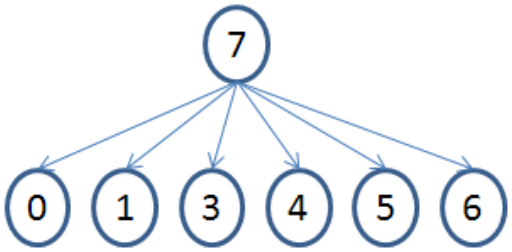

圖 2-2 倒推法中盤面需往下抓取子盤面

第三章、回溯分析演算法

在倒推法中需要往下抓取一層的子盤面先手勝負資訊，來判定自己盤面是先手勝或先手負。而在回溯分析演算法中則是根據自己盤面是先手勝或先手負，來做更新母盤面的動作，如圖 3-1 中對應到數字 7 的盤面更新其母盤面。

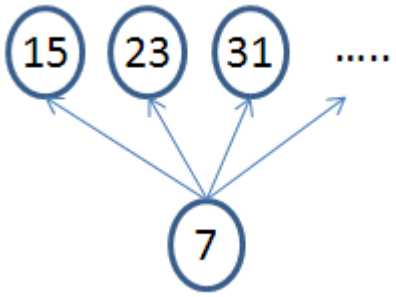

圖 3-1 回溯分析法中盤面往上更新

起始先設定所有盤面為先手負，並預設空盤面為先手勝。然後按照每個盤面對應到的二進位數字大小，由小到大(由空盤面至全滿盤面)做更新：

- 若自己盤面為先手勝，不做任何動作。
- 若自己盤面為先手負，更新所有母盤面為先手勝。

對於一個其真實結果為先手勝的盤面，一定有一子盤面為先手負，意即自己必會被此子盤面更新為先手勝。

對於一個其真實結果為先手負的盤面，由於所有子盤面皆為先手勝，因此自己不會被任何子盤面更新，維持起始值先手負。

在倒推法中若發現有一個子盤面為先手負，則可以直接判定自己盤面為先手勝，並不需要搜尋過所有子盤面，這是倒推法的優點。而回溯分析演算法則是遇到自己為先手勝的盤面時，可以省略做更新的動作。

由於三角殺棋為一個勝多敗少且差距頗大的遊戲，在倒推法中要遇到子盤面為先手負的機率較低，因此節省時間的效果不大，相對地回溯分析演算法則因為先手勝的盤面較多而較省時間。

另外，在實作八層時，由於所有結果需要 8G 大小來儲存，若記憶體不足 8G 時將必須對硬碟做存取的動作，若使用倒推法向下抓取一層所有子盤面的結果，將會很花時間。但若使用回溯分析的方法，對於更新至硬碟的情況可以直接地暫存起來，累積一定量再一次寫入，而在第四章的實作中我們也會提到另一種我們用來實作的方法。

第四章、實作、結果與數據

本章會介紹八層的實作方法，並附上解決七層與八層三角殺棋的結果與時間，以及各層數勝負的數據統計。

4.1 三角殺棋八層的實作

由於電腦記憶體只有 2G 大小，無法將結果全部儲存在記憶體，勢必用到硬碟，因此我們必須想方法將這部份的負擔降到最低。

我們將結果分為 16 個檔案做儲存，每個檔案大小為 0.5G，利用回溯分析的方法，依照每個盤面對應的數字大小由小到大來實作，但唯一的差別是我們每次只針對兩個檔案之間的結果做處理，詳細步驟如下：

1. 將第 1 檔案的結果只更新至第 1 檔案
2. 將第 1 檔案的結果只更新至第 2 檔案
3. 將第 1 檔案的結果只更新至第 3 檔案
4. 其餘以此類推

在第一步驟時只做第 1 檔案更新至第 1 檔案，對於第 1 檔案更新至其他檔案的都不做動作，如此第 1 檔案需要做 16 次將結果更新至第 1 至第 16 檔案，第 2 檔案需要做 15 次將結果更

新至第 2 至第 15 檔案，總共需要 16*17/2 = 136 次完成所有更新，因此我們可以預估對硬碟的存取動作最多只需 136 次，其餘動作皆在記憶體中進行。

某些各檔案之間不可能的更新可以省去，如第 2 檔案(如圖 4-2)，絕對不可能更新至第 3 檔案(如圖 4-3)，因為第 2 檔案是編號 33 的子必存在於盤面上，而第 3 檔案是編號 33 的子不存在於盤面，在回溯分析中並不存在取子的動作，所以第 2 檔案對第 3 檔案做的更新可以直接省略。

圖 4-1 第一檔案所儲存的盤面

圖 4-1 第二檔案所儲存的盤面

圖 4-2 第三檔案所儲存的盤面

對於八層的 288 種放(取)子方式我們也有增加效率的方法，由於每次是實作檔案對檔案之間的更新，有些放子方式是不可能合法或會超出我們的目標檔案的可以省略，只取有可能的幾個放子方式來跑即可。例如第 1 檔案(如圖 4-1)更新至第 2 檔案(如圖 4-2)，其放子的方法一定是編號 33 的子存在，而編號 34,35 及 36 的子必不存在，我們可以將這類的放子方式先行找出，以節省時間。

4.2 三角殺棋的結果與時間

程式的結果告訴我們，在七層與八層的完整盤面中皆是先手必勝，七層的數據統計也與許舜欽教授的論文[4]相符。

我們在七層的三角殺棋中有實作出倒推法與回溯分析的方法，倒推法需要 121 秒跑完所有盤面，而回溯分析的方法只需 29 秒，其中差距可以達到 4 倍之多。

而在八層三角殺棋中由於記憶體的限制，我們沒有實作倒推法，但由研究生白聖群的論文[1]中可知在 2.1GHz 的 CPU、擁有 8G 記憶體以及 64 位元的環境下大約需要 17.1 個小時將八層全解，而回溯分析的方法在 2.7GHz 的 CPU、擁有 2G 記憶體以及 32 位元的環境下僅花了 2.6 小時。

4.3 數據統計

對於回溯分析方法而言，輸的比例直接影響到其效率，由表 4-1 中可以發現在八層以內皆是勝多敗少的狀況，而且越多層的時候此輸贏比例的差距將越明顯，在八層的時候輸的比例只剩下 6%，也就是回溯分析演算法中將有 94% 的盤面不需做往上更新的動作。

	贏的個數	輸的個數	輸的比例

一層	1	1	50%
二層	5	3	37.5%
三層	50	14	21.9%
四層	882	142	13.9%
五層	28866	3902	11.9%
六層	1908063	189089	9%
七層	248938933	19496523	7.3%
八層	64595197284	4124279452	6%

表 4-1 各層數三角殺棋的勝負個數統計

第五章、結論與未來發展

在本論文中，針對三角殺棋的特性，提出了一些增加程式效率的方法，目前順利將八層的三角殺棋全解時間壓在 2.62 小時，但九層的三角殺棋僅在盤面數量複雜度部份，即已比八層複雜 512 倍，光是其結果的存放空間就至少需要 4T。而且，盤面間的交互運算亦增多，因此整體而言，不是單純的 512 倍這麼簡單，所以九層的三角殺棋很可能需要想出更多改善效率的方法。

參考文獻

[1] 白聖群，”八層三角殺棋的勝負問題之研究”，國立台灣師範大學資訊工程研究所，碩士論文，June 2009.

[2] 林宏軒，”對局理論 期中作業”，國立台灣交通大學資科工所，對局理論期中報告，November 2008.
http://java.csie.nctu.edu.tw/~stanleylin/nim/nim.doc

[3] 許舜欽，“三角殺棋的電腦解法及其實線”，電腦季刊，第十六卷，第 4 期(1982)，P.15-23.

[4] 許舜欽，”利用電腦探討七層三角殺棋的勝負問題”，Proc. Of 1985 NCS, pp.798-802，Dec. 1985.

[5] Ping-hsun Wu, Ping-Yi Liu and Tsan-sheng Hsu, "An External-Memory Retrograde Analysis Algorithm," *Proc. 4th International Conference on Computers and Games (CG)*, Springer-Verlag LNCS# 3846, pages 145--160, 2004.

[6] Tsan-sheng Hsu and Ping-Yi Liu, "Verification of Endgame Databases," *International Computer Game Association (ICGA) Journal*, volume 25, number 3, pages 132--144, 2002