電腦圍棋的發展概況：

周政緯 私立輔仁大學資訊工程所研究生
顏士淨 國立東華大學資訊工程系副教授

摘 要

電腦對局是人工智慧領域中相當重要的一個分枝。而在圍棋方面，由於它本身的特質，使得電腦圍棋在繼西洋棋、象棋之後，成為人工智慧中一個相當引人注目的新挑戰。

在本篇文章當中，我們首先會介紹近年來引起電腦圍棋界大地震的新演算法—UCT，其次則介紹 2007 年較為活躍的圍棋程式，最後則是基於上面的敘述，總結對於電腦圍棋的未來展望。

一、 序論

電腦圍棋自 Zobrist 在 1970 年設計出第一個可與人對奕的程式以來[1]，至今已有約三十年的歷史。由於圍棋本身的特質，使得電腦圍棋在繼西洋棋、象棋之後，成為人工智慧中一個相當引人注目的新挑戰。

然而電腦圍棋的難點之一，便在於缺乏良好的審局函數[2]，使其不能與西洋棋或象棋一般，運用設計良好的審局函數、搜尋樹以及優秀的剪枝法，即可獲得不錯的棋力；電腦圍棋大多藉由一些經驗法則，以靜態的評估為主，而動態的搜尋則僅用於局部的、目標明確的棋串攻殺，較少全局的搜尋。因此，人類的經驗如何用於電腦圍棋，就成了設計的重點。

自 2003 年起，Bouzy[3]試圖打破這種情況。他運用蒙地卡羅法作為評估函數，並且試圖運用此一評估函數，作全局性的搜尋，然而在棋力上始終沒有太大的突破。直到 2006 年，同樣使用蒙地卡羅法的程式 Crazy Stone[4,5]，才在杜林舉行的第 11 屆電腦奧林匹亞的九路圍棋項目中奪得金牌。雖然如此，Crazy Stone 僅在 19 路圍棋項目中奪得第五名，仍未撼動以人類思維為主的圍棋程式在 19 路圍棋的地位。

然而，隨著基於蒙地卡羅的搜尋法「UCT」[16]的出現，以 UCT 為基礎的圍棋程式 MoGo[6,7,8]也逐漸在一些較非正式的比賽中展露頭角。2007 年 6 月，第 12 屆電腦奧林匹亞於阿姆斯特丹舉行，上屆冠軍 GNUGO、亞軍 GO Intellect 以及前文介紹過的 Crazy Stone 等程式均有參賽，MoGo 在強敵環伺之下，以全勝戰績奪得了 19 路圍棋項目的金牌，Crazy Stone 也拿到了第二名，GNUGO 退居第三。這象徵著 UCT 的成功，也代表一個嶄新的局面即將到來。

在第二章中，我們將介紹這個引起電腦圍棋界大地震的演算法「UCT」；第三章則針對 2007 年電腦奧林匹亞 19 路圍棋項目的前三名作簡單的介紹。第四章則

是本文總結，並對電腦圍棋的未來作一展望。

關於電腦圍棋的規則，可參考[13]，電腦圍棋的歷史則見於[14]。另，附錄A為1997年至今重要比賽的結果。

二、 UCT

2.1 蒙地卡羅法

將蒙地卡羅法應用於圍棋，最早是由Bruegmann[9]所提出。其核心的概念，在於透過統計許多模擬棋局的結果，進行局面的優劣判斷。亦即將蒙地卡羅法做為一審局函數，以決定著手的好壞。

其中，所謂的「模擬棋局」，指的是對某一目標盤面，由電腦隨機落子，直到終盤而可以判定勝負為止。Bruegmann的方法裡，在隨機落子時，除了基本的圍棋規則外，只有一個限制：不得自填眼位，這個限制是防止棋局無法結束而設的。模擬棋局的結果，與目前常見的只判斷黑勝或白勝不同，而是會判斷輸贏目數，在決定著手優劣時，則是統計此著手下所有模擬棋局平均的輸贏目數來決定的。

2.2 UCT

UCT的全名是UCB for Tree Search，是UCB(Upper confidence Bound)[10]在Tree Search上的應用。而UCB本來是為了解決吃角子老虎問題(Bandit Problem)而產生的。所謂的吃角子老虎問題，簡述如下：目前有若干台吃角子老虎機，每台機器可以投錢並拉動操縱桿，此時會得到收益(reward)，投錢、拉桿、得到收益的過程，稱之為一個Play。每台吃角子老虎機有不同的收益率，倘若玩家想要在這若干次的Play裡獲得最大總收益，那麼玩家該怎麼作?

一般來說，玩家會開始動手玩，並且依照目前累積的經驗來決定下一次的Play要選擇哪一台機器，這稱之為開發(exploitation)。相對地，如果玩家不斷地依照目前所獲得的經驗來決定，而不試圖嘗試其他的機器，則可能會忽略收益率更高的機器，因此適度地嘗試其他機器是必須的，這稱之為探險(exploration)。如何在開發與探險之間保持平衡，就是UCB試圖解決的ExE(exploitation vs. exploration)問題。

UCB根據目前獲得的資訊，配合上一個調整值，試圖在開發與探險間保持平衡。大致上而言，每一次Play時，UCB會根據每一台機器目前的平均收益值(亦即其到目前為止的表現)，加上一額外參數，得出本次Play此台機器的UCB值，然後根據此值，挑選出擁有最大UCB值的機器，作為本次Play所要選擇的機器。其中，所謂「額外參數」，會隨每一台機器被選擇的次數增加而相對減少，其目的在於讓選擇機器時，不過份拘泥於舊有的表現，而可以適度地探索其他機器。這個公式可以這樣表示：

$$UCB_i = X_i + \sqrt{\frac{2\ln N}{T_i}}$$

其中，UCBi代表第i台機器經由UCB公式運算後所得到的值，Xi代表第i台機

器到目前為止的平均收益，Ti 表第 i 台機器的總 Play 次數，N 代表到目前為止全部機器加起來的總 Play 次數。前項即為此台機器之「過去表現」，後項則是「調整參數」。

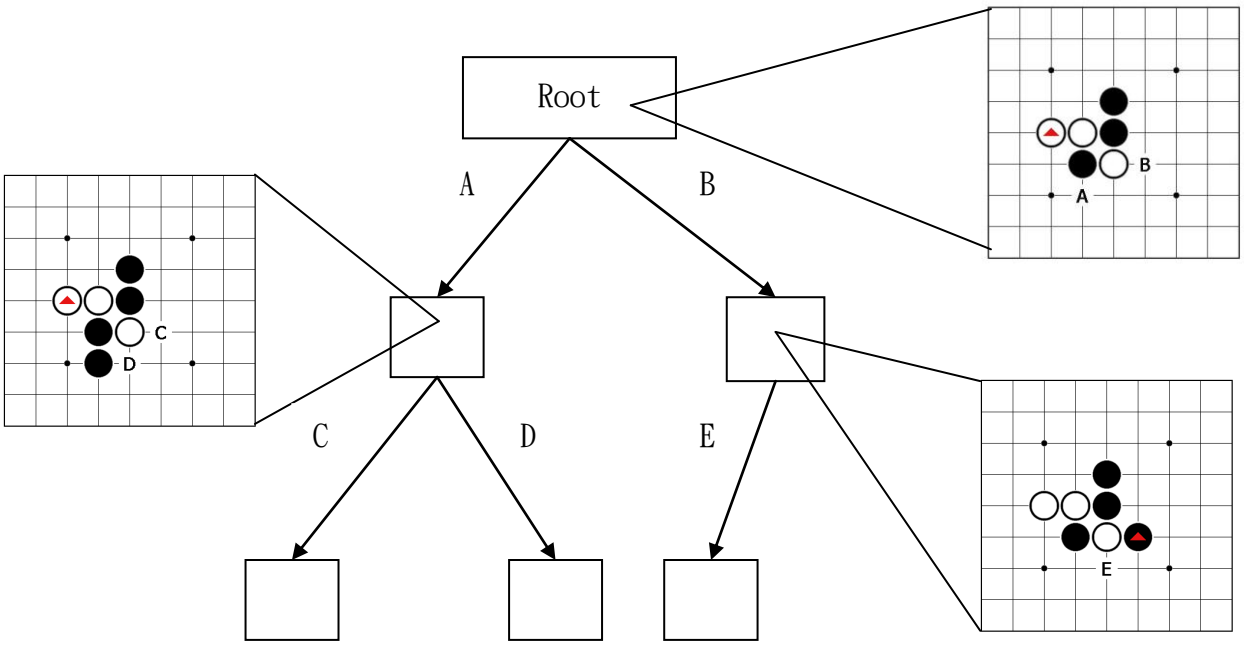
UCT 其實就是把 UCB 的公式運用於 Tree Search 上的一個方法。以概念而言，UCT 把每一個節點都當作是一個吃角子老虎問題，而此節點的每一個分支，都是一台吃角子老虎的機器。選擇分支，就會獲得相對應的收益。
Tree Search 開始時，UCT Method 會建立一棵 Tree，然後：

1. 從根節點開始
2. 利用 UCB 公式計算每個子點的 UCB 值，選擇 UCB 值最高的子點
3. 若此子點並非葉節點(從未拜訪過的節點)，則由此點開始，重複 2
4. 直到遇到葉節點，則計算葉節點的收益值，並依此更新根節點到此一葉節點路徑上所有節點的收益值
5. 由 1 開始重複，直到時間結束，或達到某一預設次數
6. 由根節點的所有子點中，挑選平均收益值最高者，作為最佳點

此一最佳點，就是 UCT 的結果。

2.3 UCT 在圍棋上的應用

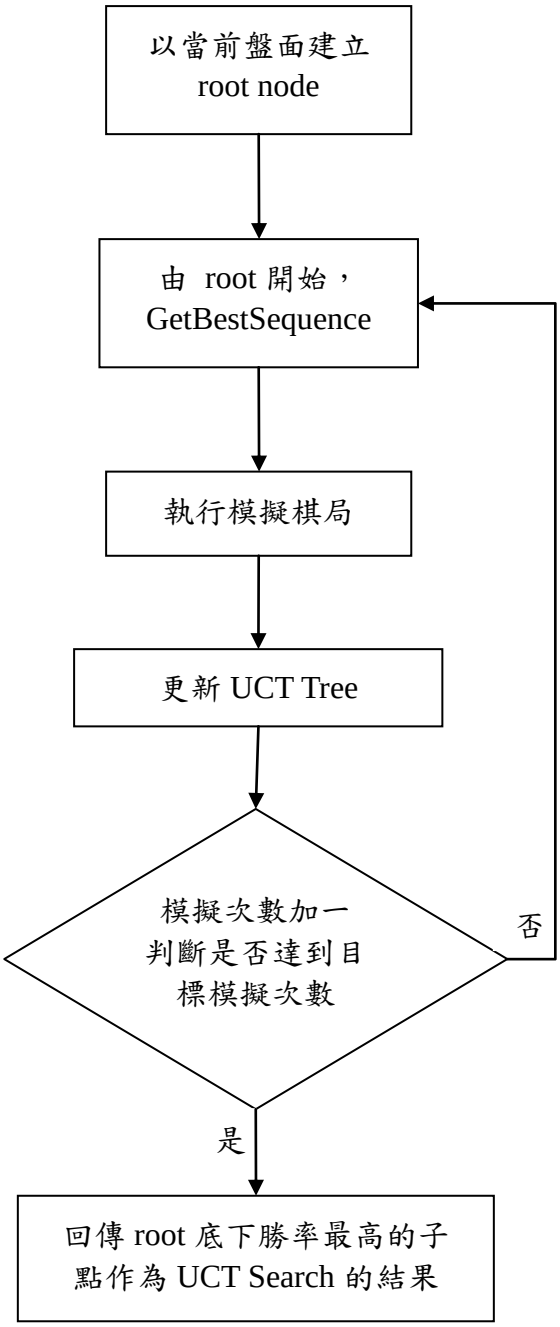
在 2.2 節中，我們已經介紹了 UCT 方法的理論。UCT 使用在圍棋上，主要的概念，就是每個節點代表一個盤面，此節點的分支代表此盤面下的合法著手，每個分支連結到的子節點，就是原盤面加上分支代表的著手後，所產生的新盤面。一般而言，目前盤面為根節點，根節點的分支代表目前盤面下的合法著手，根節點的子節點代表根節點的盤面加上分支代表的著手後所形成的新盤面，如圖所示：



圖一：UCT Tree 的概念表示，其中每個節點均會紀錄拜訪次數、勝利次數、形成此節點之著手、著手顏色等等資訊。

此外，UCB 公式的收益值，如前所述，就是依照蒙地卡羅的概念，用模擬棋局的結果來決定。上面 UCT Method 第 4 個步驟中，「計算收益值」，在此應該改為「執行模擬棋局」。所謂的模擬棋局，就是給定一個盤面(在此給的是葉節點所代表的盤面)，由電腦接手落子，直到終局，然後判定並回傳勝負結果(白勝或黑勝)。在此要注意的是，UCT 會據此結果，更新葉節點到根節點路徑上所有節點的收益值，也就是說，一個節點會概括承受所有子點模擬棋局的結果。對於任一節點，其收益值為「此一節點之模擬棋局勝利數/此點拜訪(經過)次數」，亦即其勝率。上式中所謂的勝利數，指的是指向此節點的分支所代表的顏色(黑棋或白棋)在模擬時的勝利次數。

在實際上執行 UCT Search 時，其流程可以下圖表示：



圖二：UCT Search 的流程圖

上圖中，所謂「GetBestSequence」，指的是從根節點開始，根據 UCB 公式向下搜尋，直到找到葉節點為止的過程。所謂葉節點，就是未曾拜訪過的節點。當搜尋時，若拜訪到的節點雖非葉節點，但仍無子點，則會產生子點，子點的多寡，直接影響到搜尋的深度與棋力的高低，因此對於產生出來的子點，必須加以裁減。

第三個步驟是模擬棋局。所謂的模擬棋局，就是由上一個步驟所找到的葉節

點開始，由電腦接手下完，並回傳勝負結果，以更新 UCT Tree。模擬棋局的核心就是其決定著手的方法。最簡單的方法，就是純粹隨機落子(除了非法著手或自填眼位以外)，這種方法的好處就是快速、簡單，且符合蒙地卡羅法的精神。但缺點則是用這種方法做出來的圍棋程式，棋力低落。Gelly[6]指出，要使棋力進步的重點，在於棋局的手順要有意義，而不是像隨機落子一樣，天南地北地亂下。

有了模擬棋局的結果，我們就可依此更新 UCT Tree。所謂的更新，指的是把從根節點到第二步驟中所找到的葉節點所形成的路徑上的所有點，依照模擬棋局的結果來更新勝場數與拜訪次數，亦即若此點為黑，且模擬棋局之結果為黑勝，則此節點的勝利次數加一，反之亦然。拜訪次數則是此路徑上的所有節點都加一。

若總模擬次數達到我們所設定的門檻，或限制時間已用完，則結束 UCT Search，並且挑出根節點下勝率最高的子點，作為此次搜尋的最佳著手。

三、 2007 年活躍的電腦程式

本章介紹 2007 年第 12 屆電腦奧林匹亞 19 路圍棋項目中的前三名程式：MoGo、Crazy Stone 以及 GNUGO 的歷史與特性。

3.1 MoGo

MoGo 的作者為 Sylvain Gelly，這個程式是第一個使用 UCT 為基礎的電腦程式，他落子的特性為：佈局很差，甚至沒有佈局，但是擅長於近距離接觸戰，也擅長收官，這可以說是所有以蒙地卡羅為基礎的圍棋程式共同的特性。另一個特別之處在於，此程式若贏棋，則必贏半目；若輸棋，則必輸得極慘。此外，若給予其越多的思考用時，則棋力會越強。

如前所述，UCT 主要分為搜尋部份以及模擬棋局部分。就搜尋部份而言，MoGo 運用 CFG[11]來區分棋塊，並且將距離上一手以及上上手的棋塊太遠的著手予以裁剪，以適應過大的棋盤；就模擬棋局而言，MoGo 運用基本的 Heuristic 以及手工打造的 Pattern 以使模擬棋局的著手更有意義，以增強棋力。這個程式曾在九路的非正式對局中勝過職業棋士[15]。

3.2 Crazy Stone

Crazy Stone 的作者為 Rémi Coulom，此程式曾於 2006 年第 11 屆電腦奧林匹亞的九路圍棋項目奪得金牌，也是運用蒙地卡羅法成功的案例之一。其特性與 MoGo 相當類似，在此不再贅述。

Crazy Stone 引人注目之處在於，他跳脫既有的傳統 Pattern 思維。傳統的 Pattern，指的是棋子間的分布關係，也就是所謂的「棋形」；對於棋形以外的資訊，著墨甚少。Crazy Stone 針對這部份的缺陷，提出了新的看法。首先，他對每一個著手，找出其特徵，例如長、提、叫吃、與盤端距離、與上一手距離…等等，然後從業餘高段棋士棋譜裡學習，分析每一種特徵的重要性，並據此得出了一份重要性的表格。執行模擬棋局時，就分析每一個可能著手的特徵，得出其重要性的分數，並依此算出選擇此著手的機率，依機率落子。使用這種新方法，CrazyStone 對 GNUGO 的勝率，從 38%提升到了 68%。

除此之外，他還採用了「Progressive Widening」的技術。這個技術跟 2.3.2 節介紹的「Progressive Unpruning」有異曲同工之妙，其出發點都是為了減少搜尋數的分支度。首先，CrazyStone 先依著手機率排序，隨著模擬次數逐漸增多，再逐漸開放子點以供選擇。此方法使其勝率從 68%提升到了 90%。

3.3 GNUGO

GNUGO 最特別之處在於，這個軟體沒有特定的作者，因為這是一個自由軟體，遵守 GNU 規範，其作者通稱為 GNUGO Development。這個軟體自 2001 年開始發展以來，由世界各地的作者們通力合作，棋力日強，曾獲得 2003、2006 年電腦奧林匹亞 19 路圍棋項目的冠軍，其餘大小比賽亦獲獎無數。

這個軟體是典型的以人類思維為基礎的程式，他有一個相當精緻的 Pattern 資料庫 [12]，他的棋風與 MoGo 等以蒙地卡羅為基礎的程式完全不同，而更接近人類的下法，且由於 Pattern 的關係，此程式的棋形一般而言都很漂亮。

關於這個程式的資訊與程式碼，可參考
http://www.gnu.org/software/gnugo/

四、 結論與未來展望

由於以蒙地卡羅為基礎的程式，其棋力會隨著模擬次數增加，在硬體速度日益增加的今日，電腦圍棋實力的進步已指日可待，MoGo 就曾在 9 路圍棋打敗過職業棋士。目前最強的電腦程式大約 2 級左右，綜合目前電腦圍棋進步的速度，以及硬體設備發展的程度，我們預估兩年內就會有業餘初段水準的程式產生。

參考文獻

[1] Zobrist, A. L. “Feature Extraction and Representation for Pattern Recognition and the Game of Go” , Ph.D. Dissertation, University of Wisconsin, 1970.

[2] Jay Burmeister and Janet Wiles. “An introduction to the computer Go field and associated Internet resources.” Technical Report CS-TR-339, Department of Computer Science, University of Queensland, 1995.

[3] Bouzy, B. and Helmstetter, B. “Monte-Carlo Go Developments” , in H. J. van den Herik, H. Iida and E. A. Heinz (eds.), Proceedings of the 10th Advances in Computer Games Conference (ACG-10) (Kluwer Academic), pp. 159–174.

[4] Rémi Coulom. “Efficient selectivity and backup operators in Monte-Carlo tree search.” Submitted to CG 2006, 2006.

[5] Rémi Coulom. “Computing Elo ratings of move patterns in the game of Go.” Draft, submitted to ICGA Computer Games Workshop 2007, 2007

[6] Sylvain Gelly, Yizao Wang, Rémi Munos, and Olivier Teytaud. “Modification of UCT with patterns in Monte-Carlo Go.” Technical Report 6062, INRIA, France, November 2006.

[7] Sylvain Gelly and Yizao Wang. “Exploration exploitation in Go: UCT for Monte-Carlo Go” , December 2006.

[8] Sylvain Gelly and David Silver. “Combining online and offline knowledge in UCT.” In *International Conference on Machine Learning, ICML 2007*, 2007.

[9] B. Bruegmann. “Monte carlo go.” 1993.

[10] L. Kocsis and C. Szepesvari. “Bandit based monte-carlo planning.” In 15th European Conference on Machine Learning (ECML), pages 282–293, 2006.

[11] Thore Graepel, Mike Goutrie, Marco Krüger, and Ralf Herbrich. “Learning on graphs in the game of Go.” In *International Conference on Artificial Neural Networks (ICANN-01)*, Vienna, Austria, 2001.

[12] Tanguy Urvoy and gnugo team. “Pattern matching in Go with DFA” , 2002.

[13] 顏士淨、許舜欽，”電腦圍棋的發展概況”，*Communications of the Institute of Information and Computing Machinery*，Vol. 3, NO. 2, April, 1997, pages 21 -- 28

[14] 顏士淨、許舜欽，”電腦圍棋近兩年來的發展概況”，*Communications of the Institute of Information and Computing Machinery*，Vol. 1, NO. 2, April, 1999, pages 23 – 30

[15] Sylvain Gelly and Yizao Wang , “MOGO WINS 19x19 GO TOURNAMENT” , ICGA Journal Volume 30 : Number 2 , p.111~112

[16] L. Kocsis and C. Szepesvari. “Bandit based monte-carlo planning.” In 15th European Conference on Machine Learning (ECML), pages 282–293, 2006.

附錄 A. 97 年至今重要比賽對局結果

表一、應氏杯 1997 年之後歷年之比賽結果

年份	地點	冠軍	亞軍	季軍
1997	舊金山	Handtalk	Go4++	Go Intellect
1998	倫敦	Many Faces	WULU	Go4++
1999	上海	Go4++	Goemate	KCC Igo
2000	貴陽	Wulu	Goemate	Go4++

表二、FOST 杯 1997 年之後歷年之比賽結果

年份	地點	冠軍	亞軍	季軍
1997	那古野	Handtalk	Go4++	Many Faces
1998	東京	Silver Igo	Hamlet	Goemate
1999	東京	KCC Igo	Go4++	Many Faces

其中，FOST 杯與應氏杯先後於 1999 年與 2000 年停辦，目前較大型的比賽為奧林匹亞杯、Gifu 杯(因經濟因素，於 2007 年停辦)以及 CGF 杯，此外隨著網路的興起，還有舉辦於網路上的 KGS 杯，以下就奧林匹亞杯以及 KGS 杯作一簡單介紹。

奧林匹亞杯電腦圍棋賽

第五屆電腦奧林匹亞，於上一屆的八年後，亦即西元 2000 年再度舉辦，此後逐年舉辦。除了第六屆(2001)以外，均有包括圍棋項目。在應氏杯與 FOST 杯不再舉辦後，這個比賽逐漸成為了最具公信力的電腦圍棋比賽。他與應式杯、FOST 杯的最大不同處，在於其並無獎金，純粹是榮譽之爭。

這個比賽舉辦的時間大約在每年的七八月左右，每次舉辦的地點都不同，2004 年甚至移師以色列舉行。有趣的是，當年的比賽隊數是歷年來最少的，只有五隊，而前一年有十一隊。

此比賽採用中國規則，黑貼 6.5 目，每方用時一小時。關於比賽的細節可參考 ICGA 網頁，此比賽的舉辦地點與比賽結果，可參考下表。

表三、奧林匹亞杯電腦圍棋賽歷年之比賽結果

年份	地點	冠軍	亞軍	季軍
2000	倫敦	GoeMate	Go4++	Aya
2002	馬斯垂克	Go4++	Go Intellect	GNU GO
2003	格拉茨	GNU GO	GoAhead	Go Intellect
2004	拉馬特甘	Go Intellect	Many Faces	Indigo
2005	台北	HandTalk III	Go Intellect	Aya
2006	杜林	GNU GO	Go Intellect	Indigo
2007	阿姆斯特丹	MoGo	Crazy Stone	GNU GO

KGS 杯

隨著網路的興起，網路對奕平台也逐漸成熟，KGS(Kiseido GO Server)便是一個網路圍棋伺服器，供世界各地的圍棋愛好者在網路上下棋。KGS 與眾不同的是，它提供了一組介面，讓圍棋程式也可以連上這個伺服器與人類或其他程式對弈。

KGS 杯是 KGS 所主辦的一個網路電腦圍棋大賽，自從 2003 年四月開始舉辦以來，每個月舉辦一次，未曾間斷。截至 2008 年 4 月為止，已舉辦了 37 屆。特別的是，這個比賽交錯舉辦 9 路、13 路、19 路比賽，基本上，9 路與 13 路會在同一屆比；每舉辦兩屆 9 路與 13 路比賽，才舉辦一次 19 路比賽。

表四、近幾屆 KGS 杯比賽結果

年/月	屆數	棋盤大小	冠軍	亞軍	季軍
2008/04	37	9	Steenvreter	Valkyria	Aya
2008/03	36	19	CrazyStone	Go Intellect	Leela
2008/02	35	13	Aya GNUGO		Valkyria Goanna
2008/01	34	9	Valkyria	MonteGNU	Leela
2007/12	33	19	CrazyStone	MoGo	Leela
2007/11	32	13	Leela	GNUGO	FirstGo
2007/10	31	9	MoGo	greenpeep	MonteGNU