ELSEVIER

Information Sciences 122 (2000) 77–90

INFORMATION SCIENCES
AN INTERNATIONAL JOURNAL

www.elsevier.com/locate/ins

Mean and temperature search for Go endgames

Kuo-Yuan Kao *

10412 Old Bridge Lane, Charlotte, NC 28269, USA

Received 1 August 1998; accepted 1 August 1999

Abstract

The game of Go is one of the most popular games in the world. When playing games like Go, players usually consider the following two questions at each position:

- How many points ahead is BLACK (WHITE)?
- How big is the next move?

The strength of a Go player depends on his or her ability at answering these questions. The answers to these two questions, in terms of combinatorial game theory, are called the *mean* and *temperature* of a game and they come from the solution of a nested min–max equation. This article is intended to describe an algorithm for finding these values of a subclass of combinatorial games where each player can have only one option at each local non-terminal position. © 2000 Published by Elsevier Science Inc. All rights reserved.

1. Introduction

The calculus of Go endgame is an application of combinatorial game theory [1,3], which deals with sums of positions (or games). We use letters A, B, $C, \dots$ to denote different positions and the expression $A + B + C + \cdots$ to denote the sum of these positions. Near the end of a typical Go game, the whole board can be partitioned into several independent regions (i.e., positions). The entire game is a sum of these positions.

* Tel.: +1-704-559-7979.
E-mail address: kaos@fisonline.com (K.-Y. Kao).

0020-0255/00/$ - see front matter © 2000 Published by Elsevier Science Inc. All rights reserved.
PII: S 0 0 2 0 - 0 2 5 5 (9 9) 0 0 0 9 4 - 8

Let L and R be the two players of the game where player L's goal is to maximize and player R's goal is to minimize the outcome. If G is a position, we use the notation $G^L(G^R)$ to denote the set of the positions to which $L(R)$ can move from G, and the notations $LS(G)(RS(G))$ to denote the optimal outcome of the position G when $L(R)$ moves first and each of the players follows alternately. Let G and H be two positions. The following inequality [1] describes upper and lower bounds of $LS(G+H)$:

$$LS(G) + RS(H) \leqslant LS(G+H) \leqslant LS(G) + LS(H).$$

We use the notation nG, where n is an integer, to denote the sum of n copies of G. By the previous inequality, we get

$$LS(nG) \leqslant LS((n-1)G) + LS(G)$$

and

$$RS(nG) = LS((n-1)G + G^R) \geqslant LS((n-1)G) + RS(G^R).$$

Thus

$$\lim_{n\to\infty}(LS(nG) - RS(nG))/n \leqslant \lim_{n\to\infty}(LS(G) - RS(G^R))/n = 0.$$

The value $\lim_{n\to\infty} LS(nG)/n (= \lim_{n\to\infty} RS(nG)/n)$ is called the *mean* of G, denoted by $M(G)$. The meaning of this definition is that the mean of a position is the average outcome of the position, regardless of which player moves first.

Next, we try to define the term *temperature*, which measures the size of a move. Unless a position is a terminal position where neither of the players can gain more points, the player who moves first will gain more points. The idea is to associate each move with a tax t [2]. If the tax is too low then the first player will have some advantage. If the tax is high enough then neither of the players will move first. Let G be a position. For $t \geqslant 0$, define [1]

$$LS(G,t) = \max\{M(G), RS(G^L,t) - t\}, \quad \text{and}$$
$$RS(G,t) = \min\{M(G), LS(G^R,t) + t\}.$$

Note that $LS(G,t)$ is monotone decreasing and $RS(G,t)$ is monotone increasing with respect to t. The minimum solution for t of the equation $LS(G,t) = RS(G,t)$ is called the temperature of G, denoted by $T(G)$. The meaning of this definition is that the temperature of a position is the minimum tax that can eliminate the advantage of the first player.

Fig. 1 illustrates two positions A, B and their means and temperatures. BLACK is 8 points ahead at position $A(M(A) = 8)$ and 7 points ahead at

[1] In this paper, all the positions stopped at integers and we follow the convention in [1] that use 0 as the lower bound of the temperature. Since [2], many other authors have been extending the minimum temperature down to -1. In the later case, numbers have negative temperatures.

$M(A) = 8$, $T(A) = 3$ $M(B) = 7$, $T(B) = 4$

Fig. 1. Position A and B.

position $B(M(B) = 7)$. The size of the next move is 3 points at position $A(T(A) = 3)$ and 4 points at position $B(T(B) = 4)$. The method for calculating these values is explained in subsequent sections.

The mean and temperature of a sum have the following property [1]:

$$M(A + B) = M(A) + M(B) \text{ and } T(A + B) \leqslant \max\{T(A), T(B)\}.$$

If a player can calculate the mean of each position then he or she can derive the mean of the entire game, and, if a player knows the temperature of each position then he or she can derive an upper bound of the temperature of the entire game.

Fig. 2 exemplifies a position that is equivalent to the sum of the positions A and B in Fig. 1. Note how the mean and temperature of $A + B$ are derived.

Mean and temperature can be used to estimate the optimal outcome of a game. The following inequalities [1] show upper and lower bounds for $LS(G)$ and $RS(G)$.

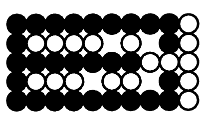
$$M(G) \leqslant LS(G) \leqslant M(G) + T(G) \text{ and } M(G) \geqslant RS(G) \geqslant M(G) - T(G).$$

The finding of the optimal outcome of a sum of positions is an NP-hard problem [4], whereby the above inequalities have already provided very practical solutions [5] in most games.

The concept of mean and temperature helps players to discover the good moves of a game. Berlekamp [2] introduced the following simple strategy for playing a sum of positions:

> If the opponent move at a position that raises the local temperature, then simply respond to it locally. Otherwise move at the position with the highest temperature.

Utilizing this strategy, if L moves first then he or she can always produce an outcome greater than or equal to the mean of the game, assuming R is an



$M(A + B) = 15$, $T(A + B) \leq 4$

Fig. 2. Position $A + B$.

optimal player. And, if L moves second then he or she can generate an outcome greater than or equal to the mean minus the temperature of the game.

2. The stable theorem of mean and temperature

In the remainder of this paper, we will study the algorithm for finding the mean and temperature of a position. We assume that each player has at most one option at each position. Each position is represented as a binary tree; each left branch corresponds to a move by L; each right branch corresponds to a move by R; each terminal node has a value corresponds to the score of the position.

If we start from a position G and let the two players move alternately, with L moving first, the sequence of positions visited will be $G^L, G^{LR}, G^{LRL} \ldots$ These positions are called the left alternating followers of G, denoted by $G^{L(1)}, G^{L(2)}, G^{L(3)}, \ldots$ Similarly, $G^R, G^{RL}, G^{RLR}, \ldots$ are called the right alternating followers of G, denoted by $G^{R(1)}, G^{R(2)}, G^{R(3)}, \ldots$ An alternating follower of G is called odd (even) if it can be reached from G by an odd (even) number of moves. Fig. 3 shows the complete game tree of position A (note that 4 white stones have been captured in position A^{RL}). A^L is an odd left alternating follower of A; A^R is an odd right alternating follower of A; A^{RL} is an even right alternating follower of A. Note that, at the terminal nodes of the tree, the means are known and the temperatures are zero. We will explain how to calculate the means and temperatures of the non-terminal nodes later.

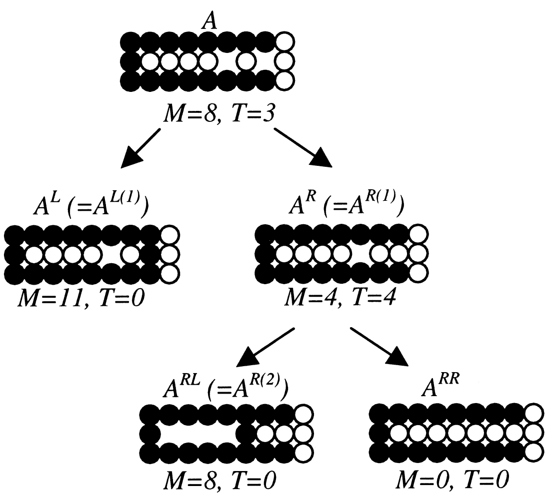


Fig. 3. The complete tree of position A.

The first left (right) alternating follower of G whose temperature is no more than the temperature of G is called the left (right) first stable alternating follower of G. For example, in Fig. 3, the left first stable alternating follower of A is $A^L(= A^{L(1)})$ and the right first stable alternating follower of A is $A^{RL}(= A^{R(2)})$. The importance of the left and right first stable alternating followers is that, once their means and temperatures have been determined, the mean and temperature of the root position can be determined.

Let $G^{L(m)}$ and $G^{R(n)}$ be the left and right first stable alternating followers of position G. That is

$$T(G^{L(i)}) > T(G), \quad 0 < i < m \quad \text{and } T(G^{L(m)}) \leqslant T(G),$$

$$T(G^{R(j)}) > T(G), \quad 0 < j < n, \quad \text{and } T(G^{R(n)}) \leqslant T(G).$$

If m is an odd number then

$$\begin{aligned} M(G) &= LS(G, T(G)) \\ &= RS(G^{L(1)}, T(G)) - T(G) \\ &\dots \\ &= RS(G^{L(m)}, T(G)) - T(G) \\ &= M(G^{L(m)}) - T(G). \end{aligned}$$

If m is an even number then

$$\begin{aligned} M(G) &= LS(G, T(G)) \\ &= RS(G^{L(1)}, T(G)) - T(G) \\ &\dots \\ &= LS(G^{L(m)}, T(G)) \\ &= M(G^{L(m)}). \end{aligned}$$

Similarly if n is an odd number then

$$M(G) = M(G^{R(n)}) + T(G).$$

And if n is an even number then

$$M(G) = M(G^{R(n)}).$$

Therefore, we have

- *Case* A: both m and n are odd

$$M(G) = (M(G^{L(m)}) + M(G^{R(n)}))/2,$$

$$T(G) = (M(G^{L(m)}) - M(G^{R(n)}))/2.$$

- *Case* B: m is odd and n is even

$$M(G) = M(G^{R(n)}),$$
$$T(G) = M(G^{K(m)}) - M(G^{R(n)}).$$

- *Case* C: m is even and n is odd

$$M(G) = M(G^{L(m)}),$$
$$T(G) = M(G^{L(m)}) - M(G^{R(n)}).$$

- *Case* D: both m and n are even

$$M(G) = M(G^{L(m)}) = M(G^{R(n)}),$$
$$T(G) = \max\{T(G^{L(m)}), T(G^{R(n)})\}.$$

The above result is called the stable theorem of mean and temperature [6]. In Fig. 3, A^{RL} is the left first stable alternating follower of A^R; A^{RR} is the right first stable alternating follower of A^R. Since both A^{RL} and A^{RR} are at odd levels relative to A^R, we have $M(A^R) = (M(A^{RL}) + M(A^{RR}))/2 = (8 + 0)/2 = 4$, and $T(A^R) = (M(A^{RL}) - M(A^{RR}))/2 = (8 - 0)/2 = 4$. A^L is the left first stable alternating follower of A. At first glance, it is not clear which one of A^R and A^{RL} is the right first alternating follower of A. Suppose A^R is stable in A then $T(A) = (M(A^L) - M(A^R))/2 = (11 - 4)/2 = 3.5 < 4 = T(A^R)$. This contradicts the assumption that A^R is stable in A. Hence, A^{RL} is the right first stable alternating follower of A. Since A^L is odd and A^{RL} is even relative to A, we have $M(A) = M(A^{RL}) = 8$, and $T(A) = M(A^L) - M(A^{RL}) = 11 - 8 = 3$.

We suggest the reader use these rules to verify the mean and temperature of position B in Fig. 1. Fig. 4 shows the complete tree of position B.

We end this section by studying one more example. Fig. 5 shows a position C and its left and right first alternating followers. C^R is the same as the position $A + B$ in Fig. 2. Suppose C^R is unstable in C. Then, $T(C) < T(C^R) \leqslant 4$, $M(C) = M(C^L) - T(C) > 24 - 4 = 20$. On the other hand, $M(C) \leqslant LS(C^R, T(C^R)) + T(C^R) = M(C^R) + T(C^R) \leqslant 15 + 4 = 19$, a contradiction. Therefore, we have C^R stable in $C, T(G) = (M(C^L) - M(C^R))/2 = 4.5$ and $M(G) = (M(C^L) + M(C^R))/2 = 19.5$. Note that, we can determine the mean and temperature of position C without expanding its tree completely. In many cases, to calculate the mean and temperature of a position, there is no need to expand the corresponding tree completely. We will discuss this more in the later sections.

3. Outline of the M-T search algorithm

The stable theorem in the previous section is very helpful for calculating the mean and temperature in general cases. Since the temperature is zero and the

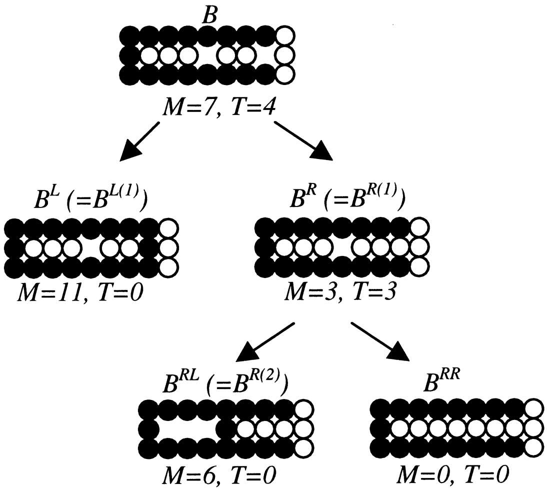


Fig. 4. The complete tree of position B.

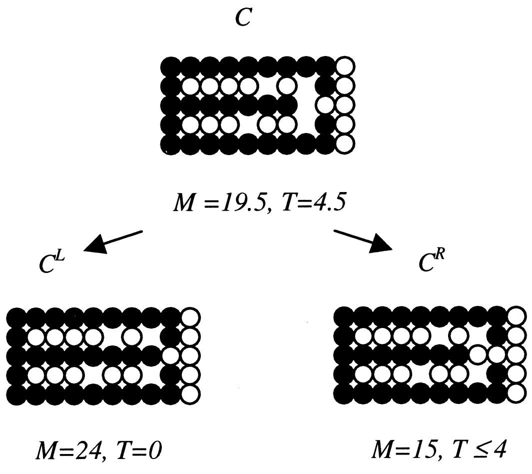


Fig. 5. Position C and its stable followers.

mean is known at each terminal node of the binary tree, by applying the theorem recursively and calculating the mean and temperature of each non-terminal node from the bottom up, one can derive the mean and temperature of the root position. This is a brute force approach; the drawback is that all the nodes of the tree must be visited and the approach is very time consuming. This situation is analogous to the pure min–max search [7] where all the nodes are visited in order to backup the root's value. A better algorithm should be able to eliminate some of the unnecessary search.

The outline of our search algorithm, *M-T* (Mean and Temperature) search, is similar to an alpha-beta search [8]. We use the notations $M^L(G)$, $M^U(G)$ and $T^L(G)$, $T^U(G)$ to denote the lower and upper bound of the mean and temperature of a position G. Before the search starts, these variables of all the positions have the following initial values: $M^L = -\infty$, $M^U = \infty$, $T^L = 0$, $T^U = -\infty$. During each run of the search, a new terminal node will be visited and the values of all the nodes on the path from the root to the terminal node will be updated. The search process continues until $M^L = M^U$ and $T^L = T^U$ at the root. Note that the search may terminate while there are still some unvisited nodes. In the worst case, all terminal nodes will be visited. Obviously, in the later case, we still will have $M^L = M^U$ and $T^L = T^U$. The following is the outline of the *M-T* search algorithm.

procedure MtSearch (root);
begin
  $M^L = -\infty$;
  $M^U = \infty$;
  $T^L = 0$;
  $T^U = -\infty$;
  while $((M^L < M^U)$ **or** $(T^L < T^U))$ **do**
  begin
    P = SelectNewTerminalNodeOf (root);
    $T^L(P) = T^U(P) = 0$;
    $M^L(P) = M^U(P)$ = ValueOfTerminalNode (P);
    UpdateMtValueOfAncestorsOf (P);
  end;
end;

The following three questions need to be answered when describing the details of the search process:
1. What order should the nodes of the tree be visited?
2. How do we update the values of the variables of a node after a new terminal node has been observed?
3. How do we determine which nodes can be eliminated from the search?

We will discuss each of these questions in details in each of the following sections.

4. Alternating first search

As in an alpha-beta search, the order of the nodes to be visited has a great impact on the searching efficiency (measured as the total number of nodes needed to complete the search) of the *M-T* search. In this section, we introduce a new binary tree traversing algorithm, alternating first search, in which the move sequences with higher alternating frequencies are visited before the ones with lower alternating frequencies.

We first define the term alternating rank. Let G be a node in a binary tree. We use the notation $A(G)$ to denote the collection of the alternating followers of G and define

$$A^{-1}(G) = \{\}$$
$$A^{0}(G) = \{G\} \text{ and}$$
$$A^{n}(G) = \{x : x \in A(y) \text{ and } y \in A^{n-1}(G)\}, \quad n > 0.$$

A node x is said to have an alternating rank n, $n \geqslant 0$, in G if $x \in A^{n}(G) - A^{n-1}(G)$. For example, in G, G^{LR} has an alternating rank of 1 and G^{LL} has an alternating rank of 2. A sequence of nodes $\{S_k\}$ in a binary tree is called alternating first if, for each $i < j$, there is no x such that the alternating rank of S_i is higher than the rank of S_j in x. Note that, the definition implies parent before children in an alternating first sequence. It turns out that, for a complete binary tree, there is only one alternating first sequence to traverse the left (right) sub tree. Fig. 6 shows a complete binary tree with depth 3 (assuming the tree is a right sub tree of its parent). The labels on the nodes are the alternating first orders. We suggest the reader verify the uniqueness of the order.

Alternating first search is well suited for the M-T search. Experience shows that, while studying a game, players usually look ahead first at those move sequences with lower alternating ranks. In many cases, the move sequences with high alternating ranks have little or no effect on the mean and temperature of the root position. Alternating first search follows this rule of thumb.

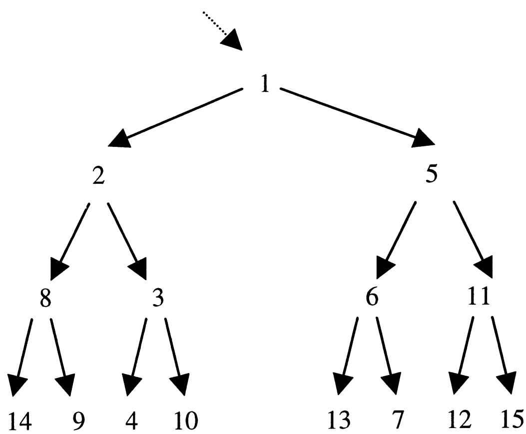


Fig. 6. An alternating first search example.

5. Mean and temperature of an incomplete tree

During the search, the game tree has not been expanded completely. However, the partial information of the tree still can be used to calculate M^L, M^U and T^L, T^U of the root. One can first construct two game trees G_U and G_L from the original tree G by the following rules:
- G_U is obtained from G by replacing each missing node (or sub tree) with the number which equals the upper bound of the node's value.
- G_L is obtained from G by replacing each missing node (or sub tree) with the number which equals the lower bound of the node's value.

Since G_U and G_L are complete game trees, we can calculate their means in the usual way. The mean of G is between the means of G_U and G_L. That is

$$M^L(G) = M(G_L) \leqslant M(G) \leqslant M(G_U) = M^U(G).$$

For example, Fig. 7 shows an incomplete game tree G. Note that G^{LL}, G^{RR} and G^{LRR} are missing.

The corresponding G_U and G_L are shown in Figs. 8 and 9. We have $0 \leqslant M(G) \leqslant 10$.

To calculate upper and lower bounds of the temperature of G, one can construct another two game trees G_H and G_c from G, as described below:
- G_H is obtained from G by
  - replacing each missing node (or sub tree) in the left sub tree of G with the number which equals the upper bound of the node's value, and
  - replacing each missing node (or sub tree) in the right sub tree of G with the number which equals the lower bound of the node's value.
- G_C is obtained from G by
  - replacing each missing node (or sub tree) in the left sub tree of G with the number which equals the lower bound of the node's value, and
  - replacing each missing node (or sub tree) in the right sub tree of G with the number which equals the upper bound of the node's value.

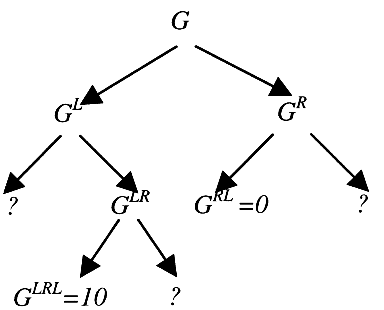


Fig. 7. An incomplete game tree G.

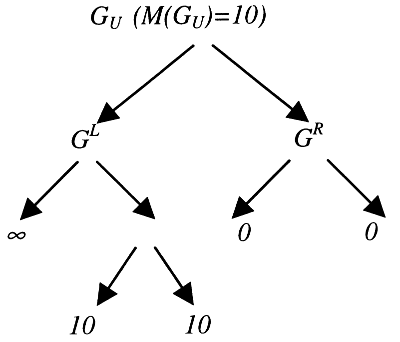


Fig. 8. G_U.

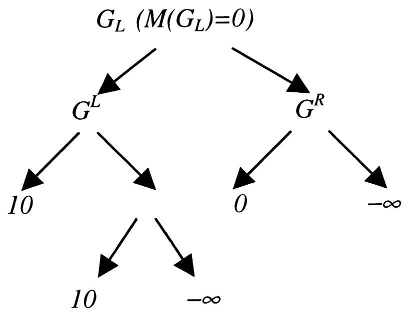


Fig. 9. G_L.

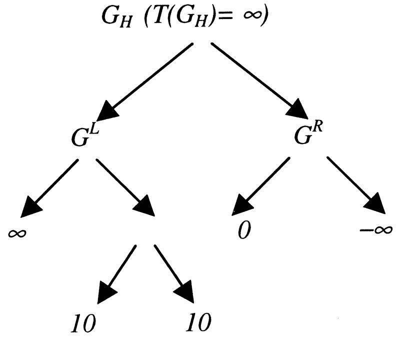


Fig. 10. G_H.

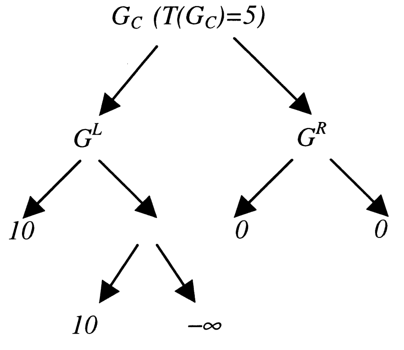


Fig. 11. G_C.

Since G_H and G_C are complete game trees, we can calculate their means in the usual way. The temperature of G is between the temperature of G_H and G_C. That is

$$T^L(G) = T(G_C) \leqslant T(G) \leqslant T(G_H) = T^U(G).$$

Figs. 10 and 11 show the corresponding G_H and G_C of the incomplete game tree in Fig. 7 and we have $5 \leqslant T(G) < \infty$.

6. *M*-cut and *T*-cut

In general, the search algorithm will visit nodes in the alternating first order described in the previous section. There are cases where some of the nodes can be eliminated from the search. We classify the elimination rules into two classes: *M*-cut and *T*-cut.

M-cut rule:

- If A is an odd level left (right) alternating follower of G, and, below A, the first stable left (right) alternating follower of G is odd in G, then one can cut the left (right) sub tree of A.

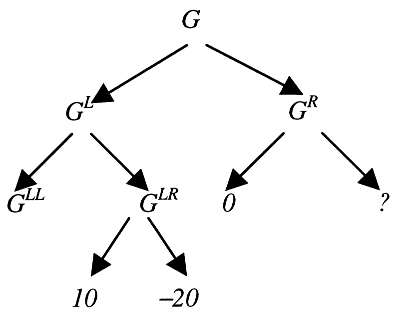


Fig. 12. An example of *M*-cut (G^{LL} can be cut).

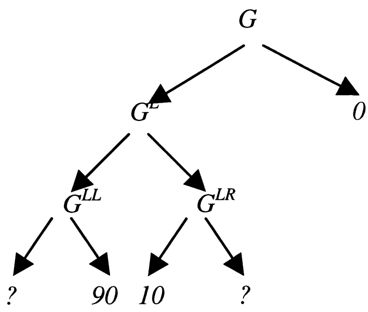


Fig. 13. An example of T-cut (G^{LL} can be cut).

- If A is an even level left (right) alternating follower of G, and, below A, the first stable left (right) alternating follower of G is even in G, then one can cut the right (left) sub tree of A.

For example, in Fig. 12, by applying the method introduced in the previous section, one can get $5 \leqslant T(G) \leqslant 10$, and $T(G^{LR}) = 15$. Hence G^{LR} is not stable in G. G^L is odd and, below G^L, the first stable alternating follower of G is $G^{LR\ L}$ which is odd also. According to the M-cut rule, one can cut G^{LL}. Note that we have pruned G^{LL} even though we do not know whether G^L is stable in G or not. The fact is that, regardless of whether G^L is stable in G or not, the first left alternating follower X (could be G^L or $G^{LR\ L}$) of G will be at odd level and $M(X) = 10$.

T-cut rule:

- If A is an unstable odd level left (right) alternating follower of G, then one can cut the left (right) sub tree of A.
- If A is an unstable even level left (right) alternating follower of G, then one can cut the right (left) sub tree of A.

Fig. 13 shows an example of T-cut. By applying the method introduced in the previous section, one can get $5 \leqslant T(G) \leqslant 10$, and $T(G^L) \geqslant 40$. Hence, G^L is unstable in G.

According to the T-cut rule, one can cut G^{LL}.

The proof of these rules follows directly from the stable theorem.

7. Conclusion

We have presented a search algorithm for finding the mean and temperature of a game. Although we used Go in the examples through out this paper, it should be mentioned that the result could be applied to general hot combinatorial games. The cornerstone of the M-T search is the stable theorem of mean and temperature, which Go players may find is very practical and easy to learn. In the worst cases, the algorithm may still need to visit all the nodes to finish the search. In the optimal cases, the number of visited nodes can be as few as two times the depth of the binary tree. Although the problem domains are different, there is some similarity between an *M-T* search and an alpha–beta search. The outline of an *M-T* search is similar to that of an alpha–beta search. An alpha–beta search can be regarded as a procedure for evaluating the value of a nested min–max expression, while *M-T* search can be regarded as a procedure for finding the solution to a particular type of nested min–max equation. Alpha-cuts and beta-cuts are used to reduce the unnecessary search when evaluating a nested min-max expression; *M*-cuts and *T*-cuts are used to reduce the unnecessary search when solving a nested min–max equation.

In this article, the assumption that each side has only one dominant option at each position is quite restrictive. We encourage future researchers to explore the circumstances and conditions (if any) under which this restrictive hypothesis might be weakened.

References

[1] E.R. Berlekamp, J.H. Conway, R.K. Guy, Winnign Ways, Academic Press, New York, 1982.
[2] E.R. Berlekamp, The economist's view of combinatorial games, in: Game of no chance, MSRI Publications, Cambridge University Press, Cambridge, 1996.
[3] J.H. Conway, On Numbers and Games, Academic Press, New York, 1976.
[4] David Moews, On some combinatorial games related to Go, Ph.D. thesis, UC Berkeley, 1993.
[5] Olof Hanner, Mean play of sums of positional games, Pacific Journal of Mathematics (1959) 81–99.
[6] K.Y. Kao, Sums of hot and tepid combinatorial games, Ph.D. thesis, UNC Charlotte, 1997.
[7] C.E. Shannon, Programming a computer for playing chess, Philosophical Magazine 41 (1950) 256–275.
[8] D.E. Knuth, R.R. Moore, An analysis of alpha–beta pruning, Artificial Intelligence 6 (4) (1975) 293–326.