Comparative evaluation of strategies based on the values of direct threats

Tristan Cazenave

Labo IA, Université Paris 8, 2 rue de la Liberté, 93526, St-Denis, France

Abstract. Taking into account the value of threats is not standard in computer Go programs. Many strategies can be used to take the value of threats into account. We made some experiments to compare relevant strategies. The value of five sub-games are randomly chosen, and the strategies play in these independent sub-games. Each sub-game has four leaves: one leave after two Left moves, one after a Left-Right combination, one after Right-Left and one after Right-Right. A brute force approach tries all possible moves. It is time consuming, but it finds the best move. An approach that does not take threats into account will play the best move, only taking into account the opponent local answer. We have defined two such strategies: BMove that do not take Right options into account and MaxMove that very roughly approximates the temperature. An approach based on the notion of sente classifies a move as sente when the value of the threat associated to the move is greater than the value of the best move. We have tested two strategies based on sente. Sente multiplies by 4 the value of double sente moves, and by two the sente moves. SenteQ always choose double sente moves first, then sente moves, then gote moves. The MaxThreat strategy select the best sub-game by comparing sub-games taking threats into account. And finally we also used the ThermoStrat and the HotStrat strategies as defined by Berlekamp. Experimental results show that MaxThreat, Thermostrat and HotStrat perform well. BMove the strategy commonly used in current Go programs scores only 87% of Optimal.

1 Introduction

In Winning Ways [1], Conway, Berlekamp and Guy define Thermostrat as a graphical tool that can be used to evaluate the temperature of a position. They also define two strategies to combine thermostrats of different games so as to choose the best game to play in. The more simple one is Hotstrat. It consists in playing in the game with the highest temperature. The other one is Thermostrat. It consists in adding thermographs to choose the game to play in. The goal of this paper is to analyze the interest of these two strategies for current computer Go programs. Many Go programs, including some of the best ones, do not take the temperature into account for choosing a move. We have experimented with different strategies, that correspond to the strategies used in current programs.

In the second section, the direct threats that were used in our experiments are presented. The third section describes the threat strategies we have experimented. The fourth section details experimental results.

2 Direct Threats

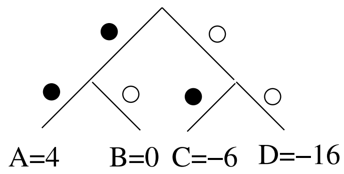

Fig. 1. Representation of a direct threat.

There are 22 points of territory for Black in the Figure 2, which corresponds to the root of the tree representing the direct threat of the Figure 1. There are 15 points of territory for White. Territory is counted with the Chinese style, both empty intersections and alive stones are counted. As Black is the Left player, the local balance of territory is 7 points for Black. This reference value is used to compute the differences after the four possible sequences that are analyzed.

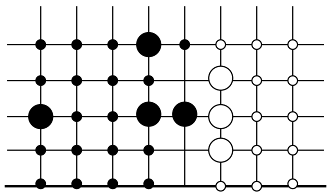

Fig. 2. The position at the root of the tree.

In the figure 3, there are 25 points of territory for Black after the sequence following the two Black moves in a row. There are 14 points of territory for White. The local balance of territory is 11 points for Black. When it is compared to the territory value at the root, the difference is 4 points for Black. Therefore the corresponding leaf in the tree, the leftmost leaf labeled A, is associated to the value 4.

In the figure 4, there are 23 points of territory for Black after the sequence following the Black and the White moves. There are 16 points of territory for

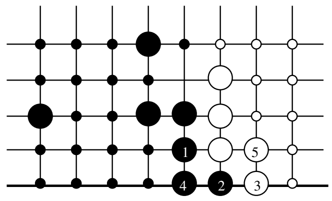

Fig. 3. The sequence following two Black moves.

White. The local balance is 7 points for Black. The difference with the situation at the root is 0, B=0 in the Figure 1.

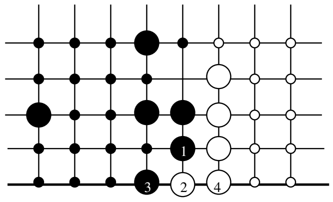

Fig. 4. The sequence after a Black and a White move.

In the figure 5, there are 20 points of territory for Black after the sequence following the White and the Black moves. There are 19 points of territory for White. The local balance is 1 points for Black. The difference with the situation at the root is -6, C=-6 in the Figure 1.

In the figure 6, there are 15 points of territory for Black after the sequence following the two White moves (we only count the same intersections as for the position at the root so as to be coherent with the other calculated values). There are 24 points of territory for White. The local balance is 9 points for White. The difference with the situation at the root is -16, D=-16 in the Figure 1.

Direct threats are composed of four values, A, B, C and D. We assume we always have $A \geq B \geq C \geq D$. A direct threat is noted (A,B,C,D). For example, the direct threat of the Figure 1 is noted (4,0,-6,-16).

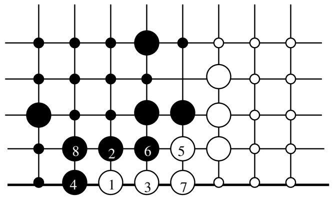

Fig. 5. The sequence after a White and a Black move.

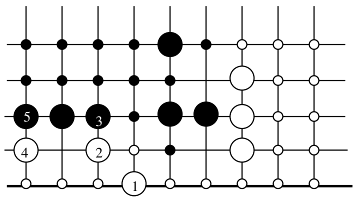

Fig. 6. The sequence after two White moves.

3 Threats Strategies

In the game of Go, there are multiple local independent situations that can be modeled using direct threats. Given multiple available direct threats on a Go board, we propose different strategies to choose among the threats. A direct threat is also called a sub-game. We only consider the case where the different sub-games are independent. The presented strategies are inspired by the current strategies implemented in computer Go programs.

3.1 Thermograph, Temperature and Mean value

Once the direct threat is computed for a position, the associated thermograph can be computed as explained in [1]. The thermograph in the Figure 7 has been built with the (4,0,-6,-16) threat. It is built by drawing two line with $\pi/4$ and $-\pi/4$ angles from the values of A and B (A=4 and B=0 here), stopping when the two lines intersect. The same drawing is done for C and D (C=-6 and D=-16 here). Then the left part of the (C,D) thermograph is rotated of $\pi/4$, and the right part of the (A,B) thermograph is rotated of $-pi/4$. The intersection of these two lines gives the temperature of the direct threat on the vertical axis, and the mean of the direct threat on the horizontal axis. The (4,0,-6,-16) direct threat has a temperature of 6.5 and a mean value of -4.5.

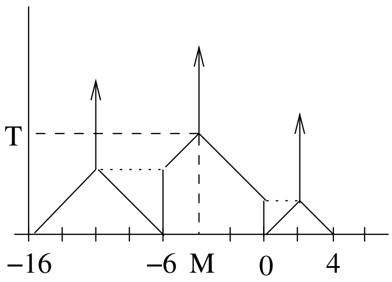

Fig. 7. The thermograph for the direct threat (4,0,-6,-16).

3.2 The Optimal strategy

The Optimal strategy takes a brute force approach to the problem of finding the best move among many independent games. The brute force approach develops a tree trying all possible moves at each node of the tree, and it develops the tree until no more moves are available. The number of possible moves varies between 5 at the beginning of the tree and 0 at the leaves. The depth of the tree is uniform and is 10 as two moves are played in each of the five sub-games. Exhaustively searching the tree is time consuming, but it finds the best move.

3.3 The BMove strategy

An approach that does not take threats into account will play the best move, only taking into account the best Left move in each sub-game. This is the strategy used by most of the computer Go programs. We call it BMove because for an (A,B,C,D) sub-game is only take into account the B value.

A direct threat where a move is possible might be in three possible different states. When a Left move has been played, it is in the state (A,B), when a Right move has been played, it is in the state (C,D), and when no moves have been played in the sub-game, it is in the (A,B,C,D) state. In the (A,B,C,D) state, the value of moving in the sub-game according to the BMove strategy is B. In the (A,B) state it is A. In the (C,D) state, it is C.

BMove plays in the sub-game that has the maximum value according to the rule above.

3.4 The MaxMove strategy

An approach that very roughly approximates the temperature is MaxMove. According to MaxMove, in the (A,B,C,D) state, the value of moving in the sub-game is B-C. In the (A,B) state it is A-B. In the (C,D) state, it is C-D.

MaxMove plays in the sub-game that has the maximum value according to the rule above. MaxMove models a strategy used in some of the Go programs that try to approximate the temperature to choose their moves.

3.5 The Sente strategy

An approach based on the notion of sente classifies a move as sente when the value of the threat associated to the move is greater than the value of the best move. The Sente strategy multiplies by 4 the value of double sente moves, and by two the sente moves.

It starts with finding the value MaxV of the best move according to the MaxMove strategy. Then for each sub-game in the (A,B,C,D) state, it compares MaxV with the value of A-B. If $A - B > MaxV$ then the move is sente, if $C - D > MaxV$ the move is reverse sente. When a move is not sente nor reverse sente, it is gote.

If a move is both sente and reverse sente, its value according to the MaxMove strategy is multiplied by 4. If a move is only sente or reverse sente, its value is multiplied by 2. Otherwise the move keep the value given by MaxMove. The move with the best value among the sub-games is chosen.

3.6 The SenteQ strategy

The SenteQ strategy classifies sente and reverse sente moves in the same way as the Sente strategy. But on the contrary of Sente, SenteQ always choose double sente moves first, then sente or reverse sente moves, then gote moves. When multiple sub-games have double sente moves, the double sente sub-game with the move which has the greatest value is chosen. When there are no double sente sub-games, the move that has the greatest value among the sente and reverse sente sub-games is chosen. When all sub-games are gote the move with the greatest value is chosen.

3.7 The MaxThreat strategy

The MaxThreat strategy select the best sub-game by comparing sub-games taking threats into account. The two sub-games to be compared are L and R. The function that compares two sub-games works as follows considering the side to move is Left:

1. if state(L)=(A,B,C,D),
   (a) if state(R)=(A,B,C,D), $L > R$ if $min(B_L + B_R, max(A_L + D_R, B_L + C_R)) > min(B_R + B_L, max(A_R + D_L, B_R + C_L))$.
   (b) else if state(R)=(A,B), $L > R$ if $min(B_L + A_R, A_L + B_R) > C_L + B_R$.
   (c) else if state(R)=(C,D), $L > R$ if $min(B_L + C_R, A_L + D_R) > C_L + C_R$.
   (d) else if no move is available in R, $L > R$.
2. else if state(L)=(A,B)
   (a) if state(R)=(A,B,C,D), $L > R$ if $C_R + A_L > min(B_R + A_L, A_R + B_L)$.
   (b) else if state(R)=(A,B), $L > R$ if $B_R + A_L > B_L + A_R$.
   (c) else if state(R)=(C,D), $L > R$ if $D_R + A_L > B_L + C_R$.
   (d) else if no move is available in R, $L > R$.
3. else if state(L)=(C,D)
   (a) if state(R)=(A,B,C,D), $L > R$ if $C_R + C_L > min(B_R + C_L, A_R + D_L)$.
   (b) else if state(R)=(A,B), $L > R$ if $B_R + C_L > D_L + A_R$.
   (c) else if state(R)=(C,D), $L > R$ if $D_R + C_L > D_L + C_R$.
   (d) else if no move is available in R, $L > R$.
4. else $L < R$.

When Right is to move, the symmetric cases are used. The cases above correspond to a local MinMax performed on the addition of two sub-games. The MaxThreat strategy consists in going through all the sub-games, comparing them to the current best sub-game, and to update the best sub-game only if the current sub-game is greater. The move of the best sub-game is played. It is intended to test if performing an optimal search on sub-games taken two by two is similar or not to an optimal search on all the sub-games.

3.8 The HotStrat strategy

The HotStrat strategy consists in evaluating the temperature of all the sub-games, and then to play in the sub-game that has the highest temperature.

3.9 The ThermoStrat strategy

The Thermostrat Strategy consists in adding all the thermographs of all the sub-game so as to choose the best sub-game to play in. It is well described in [1].

4 Experimental Results

In our experiments, five sub-games are randomly chosen. They all respect the inequality $A \geq B \geq C \geq D$. A game consists in making two strategies play against each other on the five sub-games. All the strategies are played against all the other strategies on the same five sub-games. As we have seven strategies, we play 56 different games with different strategies on the same five sub-games. These 56 different games are called a tournament between the strategies. For each game, the result of the game for the first player is added to the result of the strategy that has played the first player.

We repeat one hundred times this tournament. Each time with five new sub-games randomly chosen. The result of a strategy is the sum of its scores on the one hundred tournaments.

For each sub-game, D is chosen randomly between 0 and 50 (noted D=rnd(50)). C is D plus a random number between 0 and 50 (noted C=D+rnd(50)). We also have B=C+rnd(50) and A=B+rnd(50).

The table 1 gives the results of the program testing the different strategies (this is a corrected table from the original paper, thanks to Martin Mueller and Zhichao Li who found an inaccuracy in the original program).

Table 1. Results for the different strategies.

Strategy	*Result*	*%ofOptimal*
BMove	194898	88.39%
MaxMove	210366	95.40%
SenteQ	215278	97.63%
Sente	215455	97.71%
MaxThreat	218675	99.17%
ThermoStrat	219627	99.60%
Optimal	220504	100.00%
HotStrat	221231	100.33%

5 Conclusion

According to our result, the current Go playing program could gain approximately 15% more points in their games using the HotStrat strategy with direct threats. It is a simple strategy to implement, and many Go programs already have the the necessary information to compute direct threats. Some of our future work is to test the different strategies in a real Go program to see the practical improvement in strength they enable.

References

1. Conway, J.H., Berlekamp, E., Guy, R.K.: Winning Ways. Academic Press (1982)