The Strategies for Simple One-Point Ko Situation of Computer Go

Shi-Jie Huang[1] Shun-Shii Lin[1] Shi-Jim Yen[2]

[1]Graduate Institute of Computer Science and Information Engineering,
National Taiwan Normal University, Taipei, Taiwan.
Email: aja@csie.ntnu.edu.tw
[2] Department of Computer Science and Information Engineering,
National Dong Hwa University, Houlian, Taiwan.
E-mail: sjyen@mail.ndhu.edu.tw

Abstract

Ko plays a very important role in Go, but most computer Go programs still cannot handle ko fights so far. Utilizing the principle of Minimax procedure, we obtain the best strategies for the simple one-point ko situation, enabling computer Go programs to gain maximum or loss minimum profit when dealing with the simple one-point ko situation. We also discuss in detail the strategies for using ko threats during the process of the ko fight.

Keywords: Computer Go, Ko, Ko Fight, Minimax procedure

1. Introduction

Go is a game of contesting wits which originated from China. The main feature of Go is that the rules are simple but the variations are almost infinite owing to very large search space ($3^{19\times19} \approx 10^{170}$) [5]. Go was proved to be P-Space Hard [1]. For detailed information on many other aspects of Go, please see [2]. Computer Go programming started from 1960, D. Lefkovitz designed the first computer Go program [3]. After eight years, Zobrist designed the first Go program which beat a human Go beginner [4]. During 1980s, due to the promotion of computer Go tournaments and the commercialization of Go programs, computer Go became a noticeable research field and the strength of Go programs was improved steadily every year. During 1990s, there were more than 40 programs taking part in a tournament from all over the world. For more surveys on computer Go, please see [5, 6].

Ko plays a very important role in Go. At present there are only a few Go programs, such as GNU Go [7] and Handtalk [8], which can handle ko fights in a very simple way. Previous researches on ko emphasized the discussion of patterns which is related to the combinatorial game theory [9]. In this paper, a study for Ko fights from strategic point of view is discussed. This strategy is practical and easy to be applied in a Go program.

2. The rules of ko and the value of a move

Ko is a temporal repetition situation in Go. The name ”ko” is from the pronunciation of Japanese. To avoid infinite repetitions, there is a basic ko rule [11]:

A shape in which the players can alternately capture and recapture one opposing stone is called a "*ko*." A player whose stone has been captured in a ko cannot recapture in that ko on the next move.

Human Go players use *forward and backward comparing territory method* [12] to compute the value of a move. For computer Go programs, “*mean*” of combinatorial game theory [13] can be used to compute the value of a move. The value of a ko can be computed in the same way and can be defined as the differences of territories between Black wins the ko and White wins the ko. The value of a ko threat is the sum of the values of two continuous moves in a local area where the ko threat lies.

3. The strategies for the simple one-point ko situation

It is estimated that the proportion of simple one-point ko situation to all kinds of ko is beyond 90%. The simple one-point ko situation is the most fundamental and important case. Other kinds of ko are variations of simple one-point ko situation. In this section we show our strategies for simple one-point ko situation and prove their correctness in the next section. We describe the strategies from the view of black player. The parameters for the strategies are defined below:

(1) The value of ko is represented by k.
(2) ***BKTVL*** (Black Ko Threat Value Lower Bound) and ***WKTVL*** (White Ko Threat Value Lower Bound) are used to obtain the number of ko threats whose values are greater than them. The ko threats whose value are greater than the lower bound are called *fitting-ko-threats*. The values of **BKTVL** and **WKTVL** may be different.
(3) Fitting-ko-threats for Black and White are represented by b_i ($1 \le i \le m$) and w_i ($1 \le i \le n$) respectively, where m(n) is the total number of Black's (White's) ko threats. b_1 is the first ko threat that Black will use when he determines to join in a ko fight and is determined by the following procedure:
   (a) If there exists ko threats with the property that only one player can use it, then b_1 is the one with smallest value among them.
   (b) If there exists local ko threats, then b_1 is the one with smallest value among them.
   (c) If case(a) and (b) do not happen, then b_1 is the one with smallest value among all fitting-ko-threats.
(4) The ko threat with highest value among the ko threats that don't belong to fitting-ko-threats is represented by b_u.
(5) Valuable moves are represented by x_1, x_2, x_3…, where $x_1 \ge x_2 \ge x_3 \ge \ldots > 0$, where x_1 is the move with highest value except the ko, x_2 is the move with highest value except the ko after x_1 is played. The others are defined similarly. In implementation, the number of valuable moves depends on the search depth and evaluation ability of the Go program. For simplicity, let X_{even} be $x_2 + x_4 + x_6 + \ldots$ and X_{odd} be $x_3 + x_5 + x_7 + \ldots$.

Two Assumptions are listed below:
(1) If one side uses a ko threat, the other side won't lose any profit if he responds to the ko threat.
(2) *Damage ko threat* is not considered in this paper.

Some information in a ko flight is also assumed to be known, including the kind of ko and its value, the kind of ko threats and their values and positions, the values of valuable moves and their positions. Our strategy proposes a best move for Black according to seven cases that Black may confront. Each case stands for a situation for White's last move.

3.1 The strategies

Case 1. White just took the ko

At first, Black examines the *simple one-point ko situation fitting condition* (a formula: $x_1 + X_{\text{even}} \le k + X_{\text{odd}}$) to determine whether he should join in the ko fight. If the condition holds, Black examines the distribution of ko threats of both sides in order to choose a ko threat to use. There are three kinds of ko threat distributions:

Distribution 1.
Let **BKTVL** be $2 \times (k+X_{\text{odd}} - X_{\text{even}})$ and **WKTVL** be $2x_1 + 1$. If $m > n$ then this distribution holds and Black plays at b_1. Black will gain at least $k + X_{\text{odd}} - (x_1 + X_{\text{even}})$ profit. Otherwise Black examines the next distribution.

Distribution 2.
This distribution is examined by decreasing **BKTVL** and increasing **WKTVL** step by step. The process is depicted by the following pseudo code:

```
for ( i = 1; i ≤ k+X_odd − (x_1+X_even); i++ )
{
    BKTVL = [2×(k+X_odd − X_even)] − i;
    WKTVL = (2x_1+1) + i;
    Obtain the values m and n;
    If (m > n) (break and plays at b_1);
 }
```

If m is larger than n in any iteration of the for loop, this distribution holds and Black plays at b_1. Black will gain profit between 0 and $k + X_{\text{odd}} - (x_1 + X_{\text{even}})$. Otherwise Black examines the next distribution.

Distribution 3.
Let **BKTVL** be $k + x_1 + X_{\text{odd}} - X_{\text{even}}$ and **WKTVL** be $k + x_1 + X_{\text{odd}} - X_{\text{even}} + 1$. If $m \le n$ then this distribution holds. If b_1 is a ko threat with the property that only one player can use it or a local ko threat, then Black play at b_1. Otherwise Black compares the value of b_1 and w_1. If $b_1 < w_1$ then Black plays at b_1. If $b_1 \ge w_1$, Black compares x_1 and $b_u - x_1$, if $x_1 \ge b_u - x_1$, Black plays at x_1, Otherwise he plays at b_u. The result is that Black will lose some profit.

Case 2. White just responded to Black's ko threat
Black retakes the ko.

Case 3. White just connected the ko
If Black's last move played at a ko threat then Black uses up the ko threat to gain the profit. Otherwise Black play at x_1.

Case 4. White just played at a ko threat
This case is similar to Case 1. The distributions of the ko threats of both sides are defined in Case 1.

The next move for Black can be determined according to the following rules:

Distribution 1.
If the value of the ko threat used by White is equal to or higher than **WKTVL**, then Black responds to the ko threat. In this situation, Black will gain at least $k - x_1 - x_2$ profit. Otherwise Black connects the ko.

Distribution 2.
If the value of the ko threat used by White is equal to or higher than **WKTVL**, then Black responds to the ko threat. Otherwise Black connects the ko. In this situation, Black will gain profit less than $k - x_1 - x_2$.

Distribution 3.
If the value w_i of the ko threat used by White is smaller than **WKTVL**, Black connects the ko. Otherwise Black compares $\max(x_1 + X_{\text{even}} - (k + X_{\text{odd}}), b_u + X_{\text{even}} - (k + x_1 + X_{\text{odd}}))$ and $k + x_1 + X_{\text{odd}} - (w_i + X_{\text{even}})$. If the former is larger, Black responds to the ko threat. Otherwise Black connects the ko. The result is that Black will lose some profit.

Case 5. White just played at a valuable move
Black connects the ko.

Case 6. White just used up the ko threat to gain the profit if Black's last move didn't respond to White's move
Black plays at x_1.

Case 7. White just played at an invaluable move, a blind ko threat or passed
Black connects the ko.

Fig. 1 shows the proof sketch of the ko strategy. The parameters used in the flowchart are defined below:

(1) The value of the ko is represented by k.
(2) The ko threats used by Black are represented by $b_1, b_2, b_3, \ldots$, in usage order.
(3) The ko threats used by White are represented by $w_1, w_2, w_3, \ldots$, in usage order.
(4) Valuable moves are represented by $x_1, x_2, x_3, \ldots$, where $x_1 \geq x_2 \geq x_3 \geq \ldots$.
(5) $X_{\text{even}} = x_2 + x_4 + x_6 + \ldots$.
(6) $X_{\text{odd}} = x_3 + x_5 + x_7 + \ldots$.

3.2 Exchanging ko threats and reducing profit loss

When ko threats belong to distribution 3, Black will lose some profit anyway. In this situation, Black should make the efforts to reduce profit loss. If the value of Black's ko threat whose value is greater than **BKTVL** and is also greater than the value of White's ko threat whose value is greater than **WKTVL**, then exchanging ko threats is effective. For example, the *fitting-ko-threats* of both sides are listed below:

B = (25, 30, 40, 60); W=(42, 45, 50, 55, 80)

The Black's ko threat with value 25 is smaller than the White's ko threat with value 42. If Black uses the ko threat with value 25, White will respond and proceed to use the ko threat with value 42 to gain maximum profit. After Black exchanges ko threats with smaller values, the final outcome will be:

B=(60); W=(55, 80)

Consider another ko fight after this one. Assume that the number of ko threats of both sides doesn't increase meantime and **BKTVL** and **WKTVL** are both 45 for this ko. The difference of the number of ko threats between both sides decreases from 3 to 1. This shows that exchanging ko threats is effective.

After exchanging ko threats, Black turns to choose to play at x_1 or b_u to reduce profit loss as much as possible. According to the flowchart, the profit differences after Black plays at x_1 and b_u are $x_1+X_{\text{even}} - (k+X_{\text{odd}})$ and $b_u+X_{\text{even}} - (k+x_1+X_{\text{odd}})$, respectively. As a result, Black computes which is smaller for determining how to play for the next move. Besides, when White uses a ko threat whose value is smaller than WKTVL, Black can compare the profit difference between connecting ko right away and the move of fewer profit loss between x_1 and b_u to reduce profit loss. In other words, Black just needs to compare $\max(x_1+X_{\text{even}} - (k+X_{\text{odd}}),\ b_u+X_{\text{even}} - (k+x_1+X_{\text{odd}}))$ and $k+x_1+X_{\text{odd}} - (w_i+X_{\text{even}})$.

4. Conclusions and future work

In this paper, we discuss the entire process of *simple one-point ko situation* and utilize the principle of the Minimax procedure. Assuming that all information are available for a ko fight, we are able to obtain the optimal strategies, enabling computer Go to gain maximum or loss minimum profit when dealing with *simple one-point ko situation*. We abstract our strategies to rules for greatly reducing search time. We also discuss the strategies of using ko threats and integrate them into our strategies for *simple one-point ko situation*. The first challenge followed is to consider *damage ko threats* which makes the ko fight

problems more complicated. According to the knowledge of Go playing, *damage ko threats* should be used after all other kinds of ko threats are exhausted. But this convention has not been proved yet. Besides, another future work is to obtain the strategies for other more complicated ko fights, such as *multiple kos*, *triple kos*, *approach ko*, *two-stage ko*, *three-stage ko* and *making ko*. *Enlarging ko* is also an interesting and important topic. Nevertheless, the groundwork, such as determining the type of a ko, determining whether a move is a ko threat or not and computing its value, and computing the values of valuable moves, is also the critical key for our strategy to succeed in ko fights and is an important factor to improve the strength of Go programs.

5. References

[1] Lichtenstein. D.and Sipser. M.,"*GO is Polynomial-Space Hard*", Journal ACM 27, 2(April 1980), 393-401. (Also: IEEE Symp. on Foundations of Computer Science, (1978),48-54).

[2] R. Bozulich, The Go Players Almanac, Ishi Press, Tokyo, 1992.

[3] D. Lefkovitz,"*A strategic pattern recognition program for the game of Go*", University of Pennsylvania, the Moore school of Electrical Engineering, Wright Air Development Division, Technical note 60-243,1-92, 1960.

[4] Zobrist, A. L,"*A Model of Visual Organization for the Game of Go*", Proc. AFIPS 1969 Spring Joint Computer Conf. 34 (Boston, Mass., May 14-16, 1969), 103-112. AFIPS Press, Montvale, NJ, 1969.

[5] Martin Müller, "*Computer Go*", Artificial Intelligence, 134(1-2):145-179, 2002.

[6] B. Bouzy, T. Cazenave,"*Computer Go: An AI-Oriented Survey*", Artificial Intelligence, Vol. 132(1), pp. 39-103, October 2001.

[7] GNUGo,http://www.gnu.org/software/gnugo/gnugo.html.

[8] Chen Zhixing, http://www.wulu.com.

[9] Elwyn Berlekamp, and YoungHoan Kim,"*Where is the Thousand-Dollar ko?"*, Games of No Chance, MSRI Publications Volume 29, 1996.

[10] Murashima Yoshinori, *Ko Dictionary*, 1983.

[11] *The Japanese Rules of Go*. April 10, 1989 (Effective May 15, 1989).

[12] Kano Yoshinori, *Endgame Dictionary*, 1974.

[13] John H. Conway, "*On Numbers and Games*". Adademic Press, London/New York, 1976.

Fig. 1. Proof ske