

Counting liberties in Go capturing races

TEIGO NAKAMURA

ABSTRACT. Applications of combinatorial game theory to Go have, so far, been focused on endgames and eyespace values, but CGT can be applied to any situation that involves counting. In this paper, we will show how CGT can be used to count liberties in Go *semeai* (capturing races).

Our method of analyzing capturing races applies when there are either no shared liberties or only simple shared liberties. It uses combinatorial game values of external liberties to give an evaluation formula for the outcome of the capturing races.

1. Introduction

Combinatorial game theory (CGT) [1][2] has been applied to many kinds of existing games and has produced a lot of excellent results. In the case of Go, applications have focused on endgames [3][4][6][7][10] and eyespace values [5] so far, but CGT can be applied to any situation that involves counting. In this paper, we will show another application of CGT to Go, that is, to count liberties in capturing races. A capturing race, or *semeai*, is a particular kind of life and death problem in which two adjacent opposing groups fight to capture each other's group. In addition to the skills involved in openings and endgames, skills in winning capturing races are an important factor in a player's strength at Go. In order to win a complicated capturing race, various techniques, such as counting liberties, taking away the opponent's liberties and extending self-liberties, are required in addition to wide and deep reading. Expert human players usually count liberties for each part of blocks involved in *semeai* and sum them up. A position of capturing races can also be decomposed into independent subpositions, as in the cases of endgames and eyespaces.

We propose a method to analyze capturing races having no shared liberty or only simple shared liberties. The method uses combinatorial game values of

external liberties. We also prove an evaluation formula to find the outcome of the capturing races.

2. Capturing races

Terminology. Figure 1 shows an example of capturing races in Go. Both Black and White blocks of circled stones have only one eye and want to capture the opponent's block to make the second eye. In order to describe capturing races, we use the following terminology.

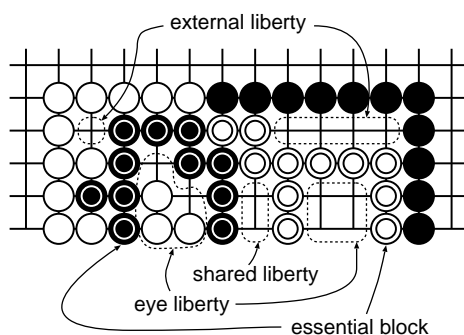


Figure 1. Example of a capturing race.

ESSENTIAL BLOCK: A *block* is a maximal connected set of stones of the same color and the adjacent empty points of a block are called *liberties*. If a block loses all its liberties, it is captured and removed from the board. An *essential block* is a block of Black or White stones which must be saved from capture. Capturing an essential block immediately decides a semeai.

SAFE BLOCK: A block which is alive or assumed to be safe. Nonessential blocks surrounding an essential block are usually regarded as safe blocks.

NEUTRAL BLOCK: A block other than an essential block and a safe block. Saving or capturing such blocks does not decide a semeai.

LIBERTY REGION: A region which is surrounded by at least one essential block and some other essential blocks and safe blocks. Liberty regions contain only empty points and neutral blocks. A liberty region is called an *external region* if its boundary does not consist of essential blocks of different color. A liberty region is called an *eye region* if its boundary consists entirely of essential blocks of the same color.

EXTERNAL LIBERTY: A liberty of an essential block in an external liberty region.

SHARED LIBERTY: A common liberty of a Black essential block and a White essential block.

PLAIN LIBERTY: A liberty of an essential block that is also adjacent to an opponent's safe block. A plain liberty can be filled without any additional approach moves by the opponent.

EYE LIBERTY: A liberty in an eye liberty region. An amount of liberty count in an eye liberty region is greater than or equal to the number of eye liberties [8]. Eye liberties behave like external liberties in capturing races.

ATTACKER AND DEFENDER: In an external liberty region, the player who owns the essential block is called the *defender* and the opponent is called the *attacker*.

Related research. Müller [8] gives a detailed discussion of capturing races. He classifies semeais into nine classes in terms of types of external and shared liberties and eye status. Figure 2 shows some examples of his classification.

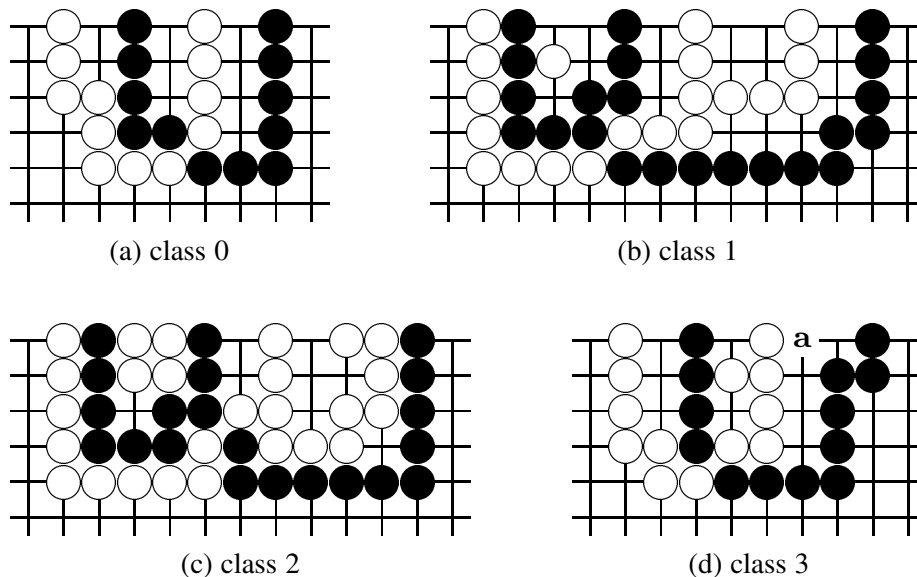


Figure 2. Müller's semeai classes.

He gives an evaluation formula for class 0 and class 1 semeais, which have exactly one essential block of each color and only plain external and shared liberties, and have no eye or only one eye in *nakade* shape.

Formula 1: Müller's semeai formula

Δ : Advantage of external liberties for the attacker
(difference in the number of liberties for each player's essential blocks)

S : Number of shared liberties

$$F = \begin{cases} S & \text{if } S = 0 \text{ or defender has an eye,} \\ S - 1 & \text{if } S > 0 \text{ and defender has no eye.} \end{cases}$$

The attacker can win the semeai if $\Delta \geq F$.

This is a really simple formula but it is enough to decide complicated semeais such as *one eye versus no eye semeais* and *big eye versus small eye semeais*. Nevertheless, it is only applicable to the restricted classes of semeais whose liberties are all plain liberties, that is, all the liberty counts are *numbers*. Generally, liberty counts are not numbers in the higher class of semeais, but are *games* whose values change with each player's move.

3. Analysis of capturing races using CGT

Although we must take into account the territory score to analyze a Go end-game using CGT, we don't need to assign the terminal score explicitly because the score comes out of CGT itself if we forbid pass moves, which are permitted in Go, and permit a return of one prisoner to the opponent as a move instead of playing on the board. In the case of eyespace values, we have to assign the terminal score explicitly as the number of distinct eyes, but we can easily find the terminal nodes and ignore plays beyond the nodes, that is, plays to *numbers*, because once the defender makes a secure eye space, the attacker cannot destroy it. But in the case of capturing races, even though the number of liberties of essential blocks is taken into account, there remains a subtle problem of how to assign the terminal scores. Unlike eyespace values, no secure liberty exists in capturing races, because the attacker can always fill the defender's liberties one by one. Even eye liberties are not secure. The attacker can fill liberties inside the eye.

How to count liberties. To model capturing races, we define the game **SemGo**, which has the same rules as Go except for scoring. In **SemGo**, the terminal score is basically the number of liberties of essential blocks, but it is exactly the number of the opponent's moves which are required to take away all the liberties. By convention, Black is Left and White is Right; Black scores are positive and White scores are negative.

Figure 3 shows some examples of CGT values of **SemGos**. In Figure 3(a), White's essential block has three liberties, but Black cannot directly attack White's external liberty, because if he simply fills the liberty, Black's attacking block gets to be in *atari*, and White can capture Black's five stones and White's essential block is alive. So, Black needs to spend one move to protect at the point of his false eye prior to attacking. Generally, liberty scores in external liberty regions are greater than or equal to the number of liberties of essential blocks. In Figure 3(b), if White plays first, the score is zero, but if Black plays first, the number of liberties becomes three. In Figure 3(c), Black can connect his two stones and the score becomes four, if he plays first.

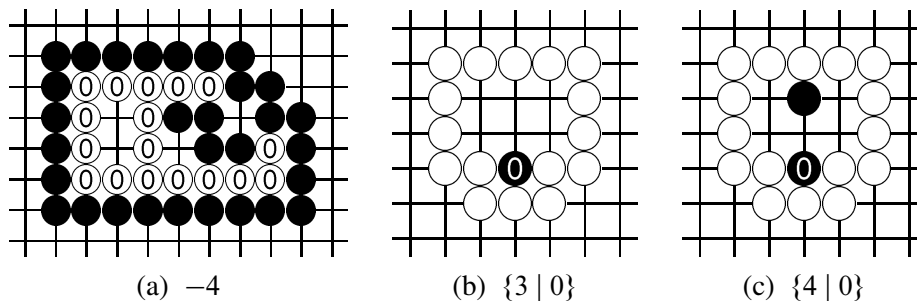


Figure 3. CGT values of **SemGos**.

Although it seems easy to obtain a CGT expression from a position of **SemGo**, we have to resolve the subtle problem of how we can find the terminal nodes and assign scores to them. As long as liberties of essential blocks exist, both players still have legal moves in **SemGo**, but not all moves are useful for both players. The attacker always has good moves, because he can fill the defender's liberties one by one. The defender, on the other hand, may not have any useful moves. If the defender cannot extend his own liberties by at least one, he should not play any more. It is always a bad move to take away his own liberties. In order to obtain CGT expressions, we have to prune the useless moves explicitly in contrast to endgames and eyespace values.

Figure 4 on the next page shows the key idea of pruning in **SemGo**. The position in Figure 4(a) has exactly one *dame* and the ordinary CGT value is $\{0 | 0\} = *$. In part (b) of the figure, Black's essential block has one liberty and Black's option of reducing his own liberty should be pruned. We assign the value one to the root node of the resulting game tree in part (c). Part (d) and (e) show the case of White's essential block.

The process for assigning the terminal value is summarized in the sidebar on the next page.

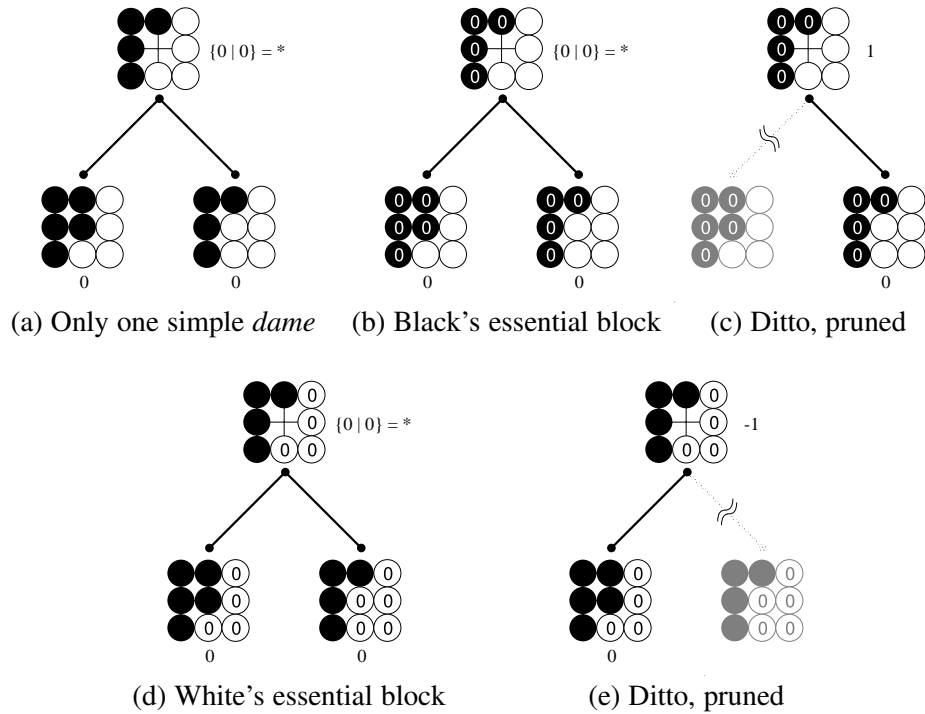


Figure 4. How should we assign the terminal score?

Process for Assigning Terminal Score

- (i) Play out all legal moves for both players until the essential block has no liberty.
 - At present, it doesn't matter whether the move is good or bad.
 - All the leaf nodes are zero.
 - In practice, however, we can count the number of liberties when they become plain liberties.
- (ii) Execute the following operations from bottom to top.
 - If the temperature of a node is less than or equal to one, prune the defender's branch.
 - If the resulting node becomes either of the following forms, replace it:

$$\begin{aligned} \{ | n \} &\implies n + 1, \\ \{ -n | \} &\implies -n - 1. \end{aligned}$$
 - This replacement operation is exactly contrary to conventional CGT.

Figures 5 and 6 illustrate the process for assigning the terminal score. A number between parentheses denotes the temperature of the node. All three positions in Figure 5 have the same score of 1, even if Black poses a threat to gain the number of liberties at the bottom position. In Figure 6 on the next page, the threat on White is immediately reversed by Black's fill, and White's option is pruned.

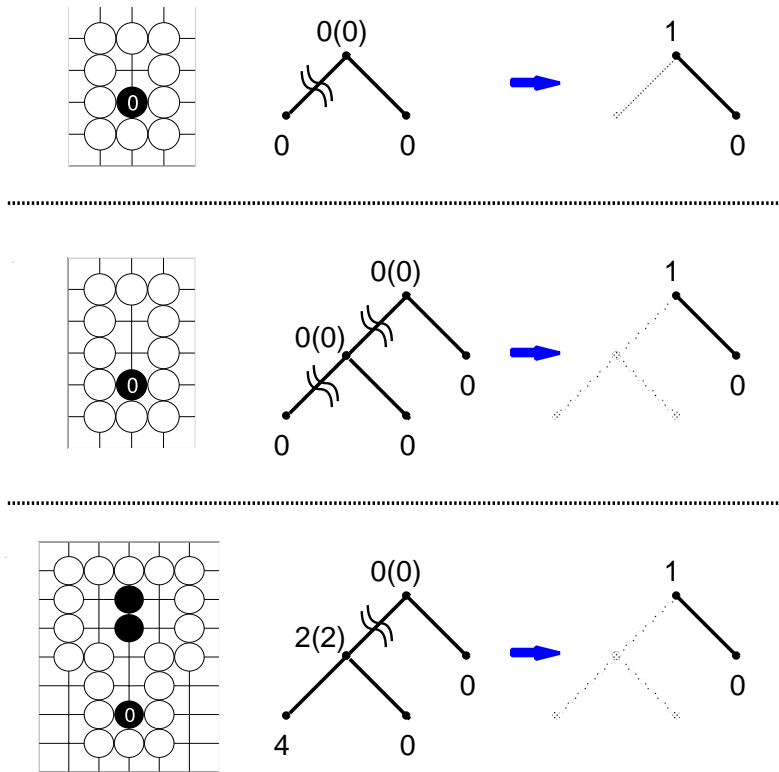


Figure 5. Process for assigning the terminal score: some positions having only one liberty.

Evaluation method. The winning condition of **SemGo** is different from that of endgames and eyespaces, because we cannot define the winner as the player who plays the last move. In **SemGo** the player who fills all the liberties of all of the opponent's essential blocks in all summands is the winner. Suicidal moves are forbidden except for the last winning move. In the case of endgames and eyespaces, the fact that the smallest incentive for a move is infinitesimal, that is *0-ish* and cooling by one, or *chilling*, plays a very important role in analyzing positions. But in the case of **SemGo**, the smallest incentive is *1-ish*, because the attacker always has a move to fill the opponent's liberties one by one. Therefore,

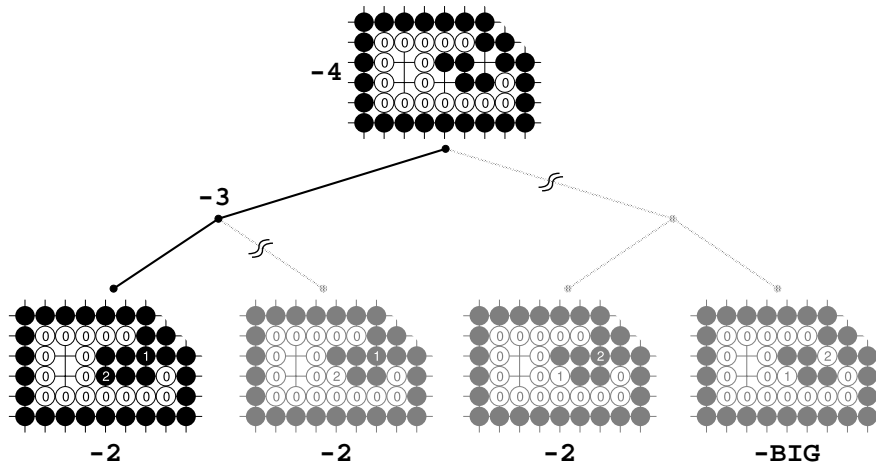


Figure 6. The game tree of Figure 3(a).

cooling by 1 makes **SemGo** 0-ish and another *cooling by 1* gives us the winner. That is to say, *cooling by 2* is the most important technique for analysis of **SemGo** positions.

The procedure below illustrates how to evaluate games of **SemGo**. We assume that each summand is a position of an external liberty region and no shared liberty region exists.

The resulting value rounded up and down in cases 2 and 3 is called the *adjustment value*.

Evaluation Method of SemGo

Suppose that G is **SemGo** and g is the game of “ G cooled by 2”.

CASE 1: g is an integer:

If $g > 0$, Black wins. If $g < 0$, White wins. If $g = 0$, the first player wins.



CASE 2: $n < g < n + 1$ (for an integer n):

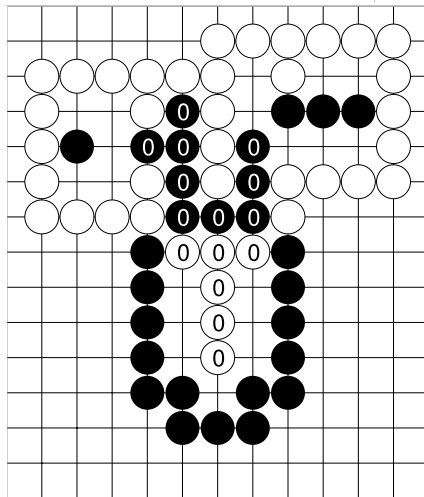
If Black plays first, he can round up g to $n + 1$ in keeping his turn.
 If White plays first, he can round down g to n in keeping his turn.
 Check the resulting value using the conditions of case 1.

CASE 3: $g \lessdot n$ (g is not comparable to an integer n):

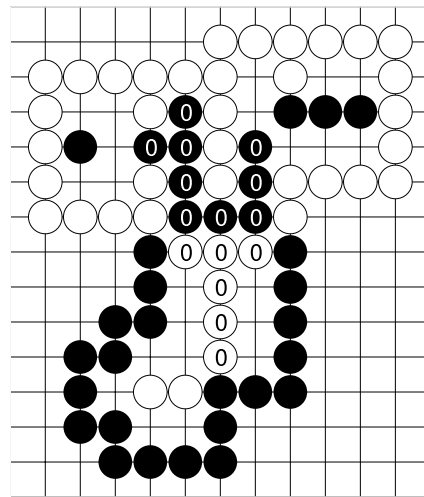
If Black plays first, he can round up g to $n + 1$ in keeping his turn.
 If White plays first, he can round down g to $n - 1$ in keeping his turn.
 Check the resulting value using the conditions of case 1.

4. Examples

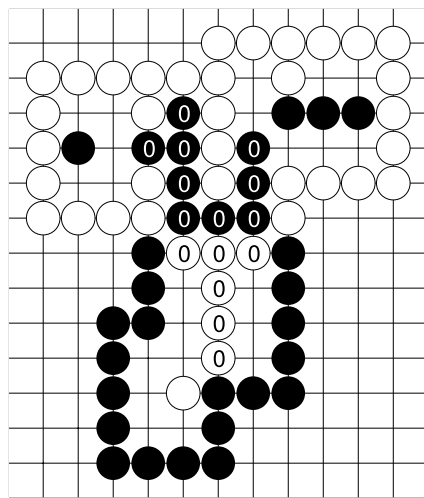
4.1. Partial board problems. Figure 7 shows some simple (!) problems of capturing races and Figure 9 shows the analyses of these problems using our method described in the previous section. The recommended winning move is  and the other winning move is  in Figure 9.



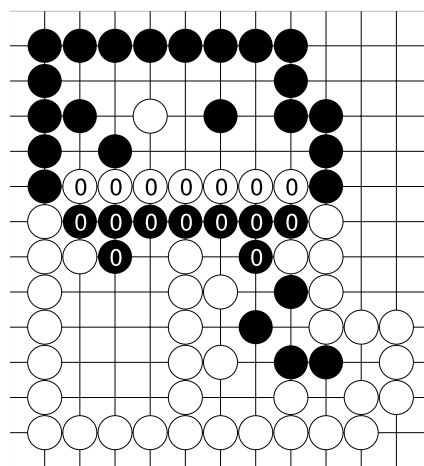
(a) Problem 1



(b) Problem 2



(c) Problem 3



(d) Problem 4

Figure 7. Semeai problems (Black plays first).

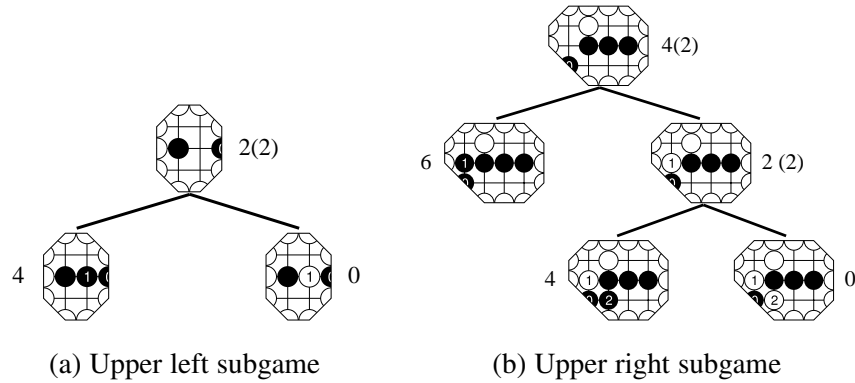
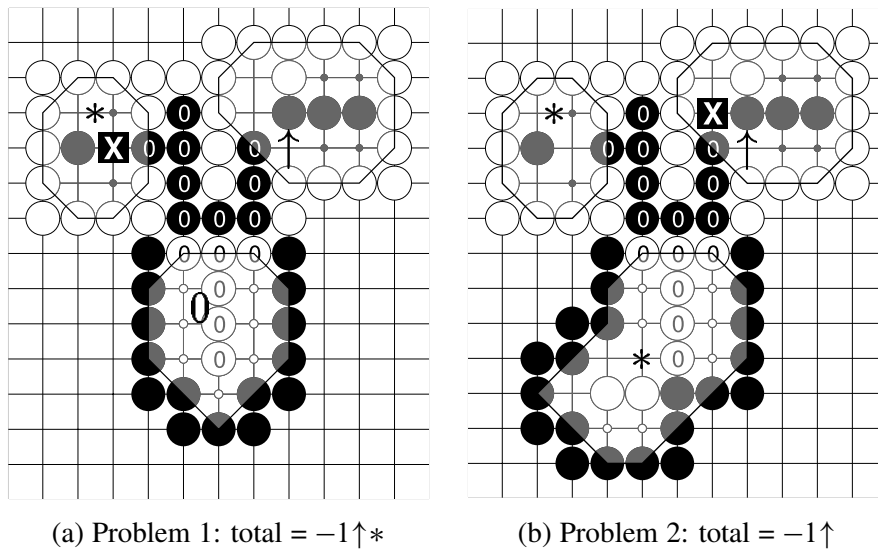


Figure 8. Game trees of the upper common parts.

The positions in parts (a)–(c) of Figure 7 all have the same upper part of Black’s essential block and are decomposed into three subgames. Figure 8(a) shows the game tree of the upper left subgame of Figure 7(a)–(c). The game is $\{4 \mid 0\}$ and after cooling by 2, the value $2*$ is obtained. Figure 8(b) shows the game tree of the upper right subgame. The game is $\{6 \mid \{4 \mid 0\}\}$ and after cooling by 2, the value $4\uparrow$ is obtained. The integer part of each subgame’s value is shown by small Black and White dots, or *markings*, in Figure 9, as used in [3]. In Problem 1, the cooled value of the lower part is -7 and the total value is $2* + 4\uparrow + (-7) = -1\uparrow*$. Black’s adjustment value of $-1\uparrow*$ is 0 and he can win if he plays first. The only winning move for Black is to play to $*$. Black can get *tedomari* as shown in Figure 10(a). But if Black plays to \uparrow in his



(a) Problem 1: total = $-1\uparrow*$

(b) Problem 2: total = $-1\uparrow$

Figure 9. Analysis using CGT (continued on next page).

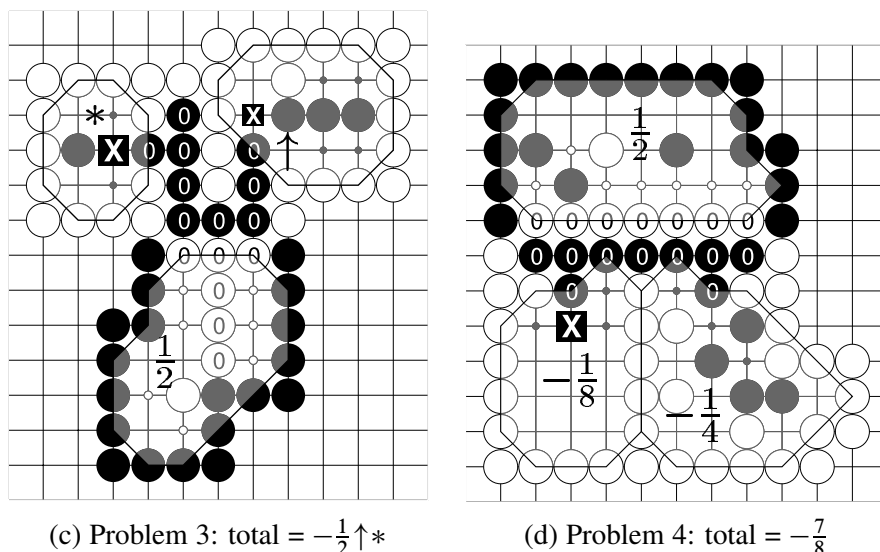


Figure 9 (continued). Analysis using CGT.

first move, White gets *tedomari* and White wins as shown in Figure 10(b). In Problem 2, the lower subgame is $\{-5 \mid -9\}$ and the cooled value is $-7*$. The total value is $2* + 4\uparrow - 7* = -1\uparrow$. Black can round up the value to 0 and win. Unlike Problem 1, Black should play to \uparrow , which is the only winning move in this problem. Figure 11(a) shows a winning sequence and Figure 11(b) shows a failure. In Problem 3, the lower subgame is $\{-5 \mid -8\}$ and the cooled value

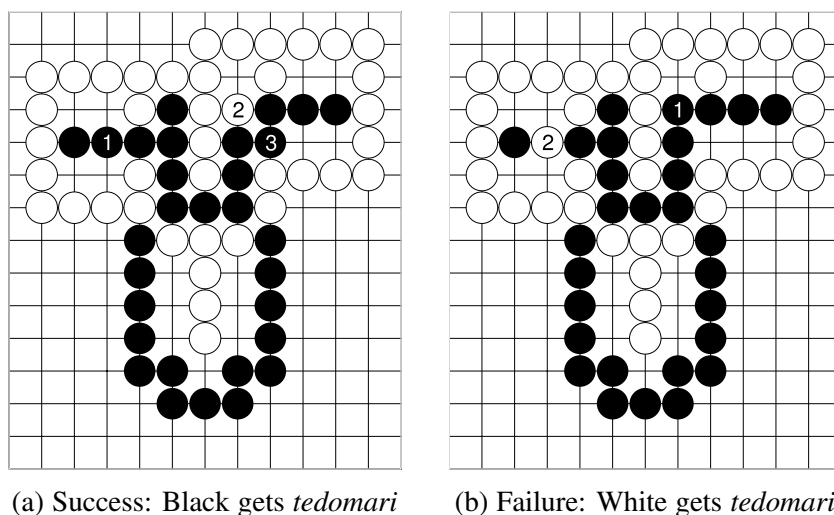
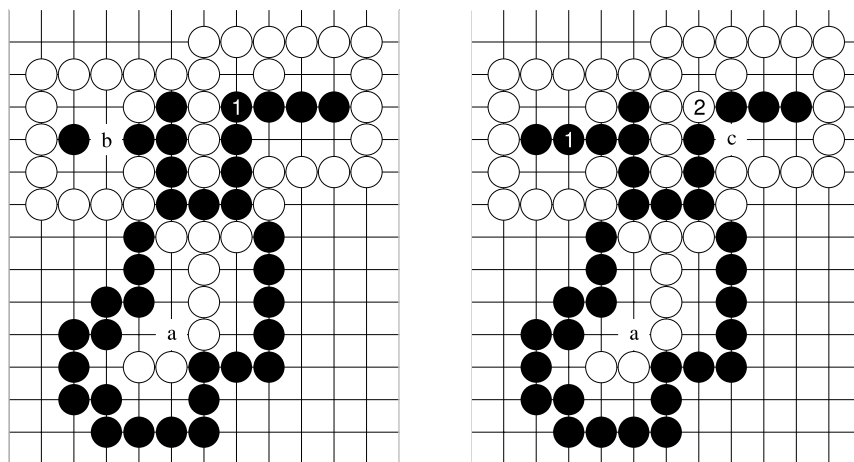


Figure 10. Solution of Problem 1.



(a) Success: Black gets *tedomari*, because *a* and *b* are *miai*.

(b) Failure: White gets *tedomari*, because *a* and *c* are *miai*.

Figure 11. Solution of Problem 2.

is $-6\frac{1}{2}$. The total value is $2* + 4\uparrow - 6\frac{1}{2} = -\frac{1}{2}\uparrow*$. Since $0 > -\frac{1}{2}\uparrow* > -1$, Black's adjustment value is 0 and he can win the semeai. In this problem, there are two winning moves for Black. Both plays to \uparrow and $*$ lead Black toward a win. In Problem 4, the upper subgame is $\{-5 \mid -8\}$ and the cooled value is $-6\frac{1}{2}$. The subgame of lower left is $\{\{8 \mid 5\} \mid 3\} \mid 1\}$ and the cooled value is $2\frac{7}{8}$. The subgame of lower right is $\{6 \mid 3\} \mid 1\}$ and the cooled value is $2\frac{3}{4}$. So the total value is $-6\frac{1}{2} + 2\frac{7}{8} + 2\frac{3}{4} = -\frac{7}{8}$ and Black's adjustment value is 0. Since all the subgames are *numbers*, Black's winning move is to play to the subgame with the largest denominator.

4.2. Whole board problems. Figure 12 shows two examples of whole board problems. Both problems are really complicated and may stump high-dan professionals. Problem 5 can be decomposed into six subgames whose values are not integers (Figure 13(a)). Figure 14 shows the game trees and cooled values of these subgames, and Figure 13(b) illustrates the values of all subgames and the winning moves. The total score is $-1\uparrow\leftarrow$, because Black has two more liberties and White has ten more liberties outside of the above six subgames. Since $0 > -1\uparrow\leftarrow > -1$, Black can round up the value to 0 and win the semeai if he plays first. The recommended winning move shown as **X** in Figure 13 is to attack \leftarrow , but in this case, attacking \downarrow also leads Black toward a win. Figure 15(a)–(h) shows a winning sequence. Although the actual battle ends at Black's move 11 (Figure 15(b)), a total of 49 moves is required by the end of the capturing race where the essential block is finally captured. Figure 15(i)

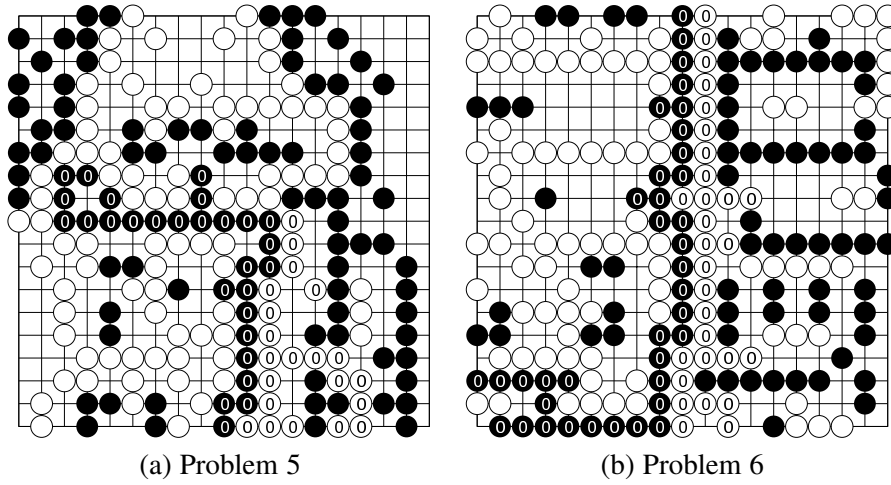


Figure 12. Whole board problems.

is an example of failure. If Black plays to $\uparrow*$, Black cannot win even if Black plays both \leftarrow and \downarrow afterward.

The position of Problem 6 is decomposed into thirteen subgames as shown in Figure 16(a). The game value and its atomic weight for each subgame is shown in Figure 16(b). The total value is $-1-ish$ and is fuzzy with -1 . Black's adjustment value is 0 and Black can win. Black's only winning move is to play to the subgame C. Figure 16(c) illustrates a winning sequence. The battle continues up to Black's move 27. At this point, the number of liberties of Black's

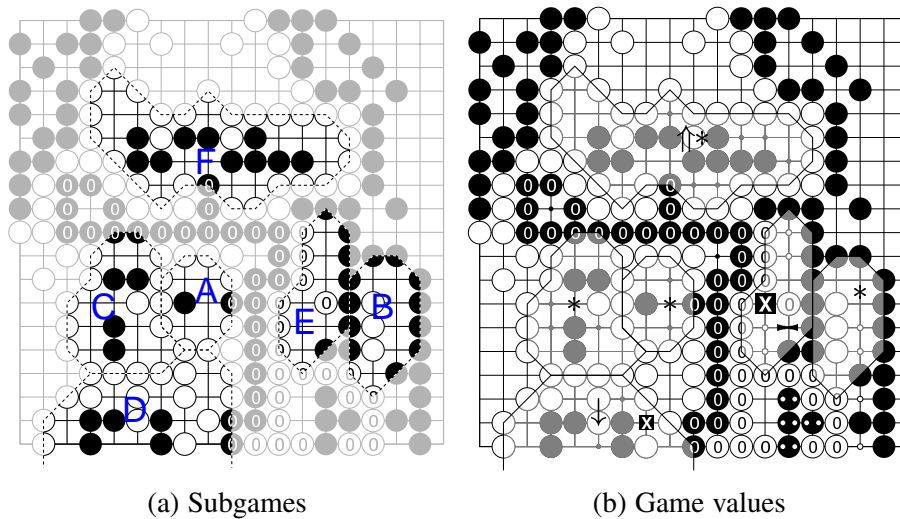


Figure 13. Solution of Problem 5.

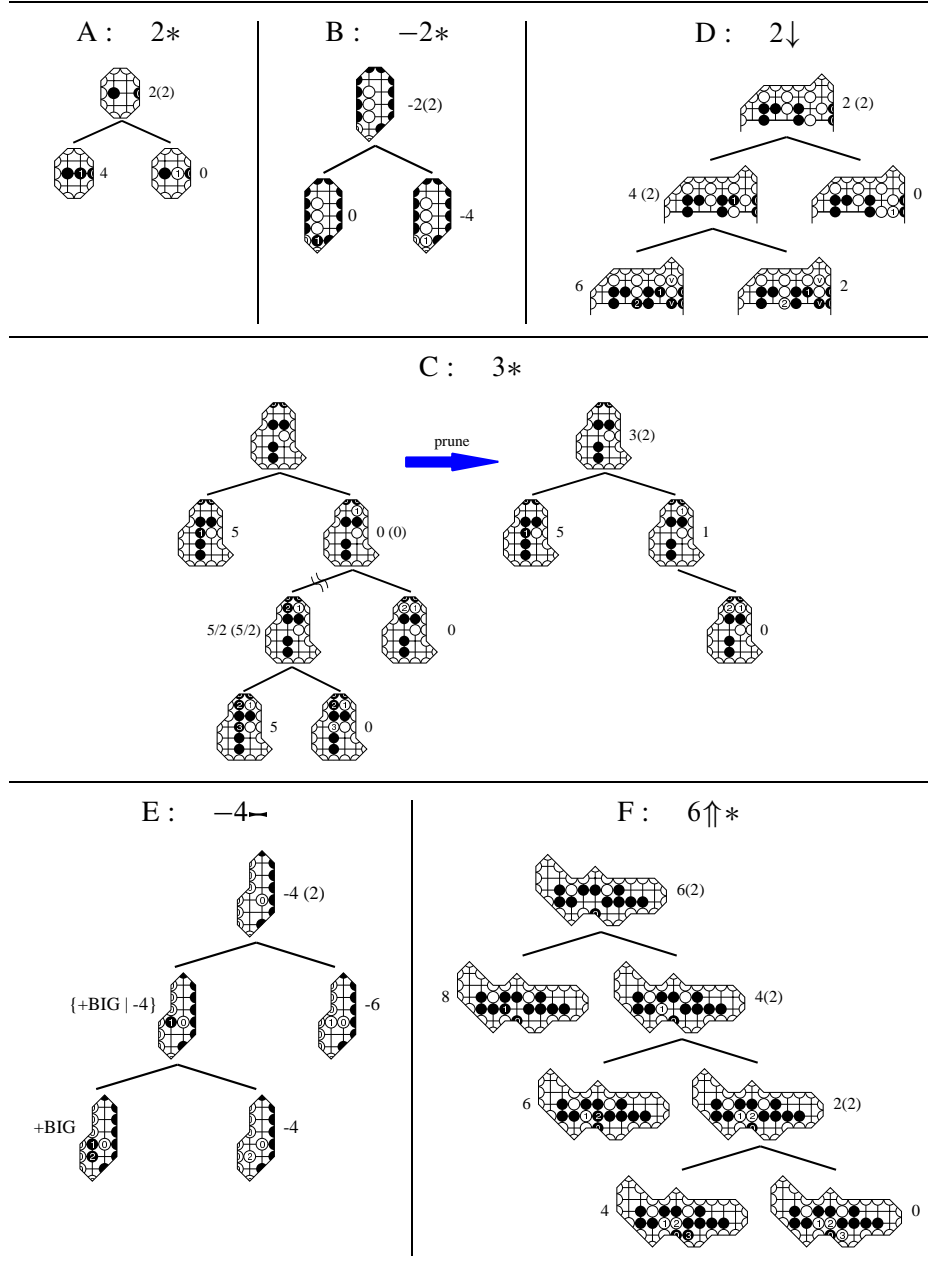


Figure 14. Game tree and cooled value of each subgame.

essential block is 22 and the number of liberties of White's essential block is 21, so Black wins the semeai regardless of who moves first from here. Figure 16(d) shows an interesting game tree whose cooled value has $\uparrow*$ as an infinitesimal.

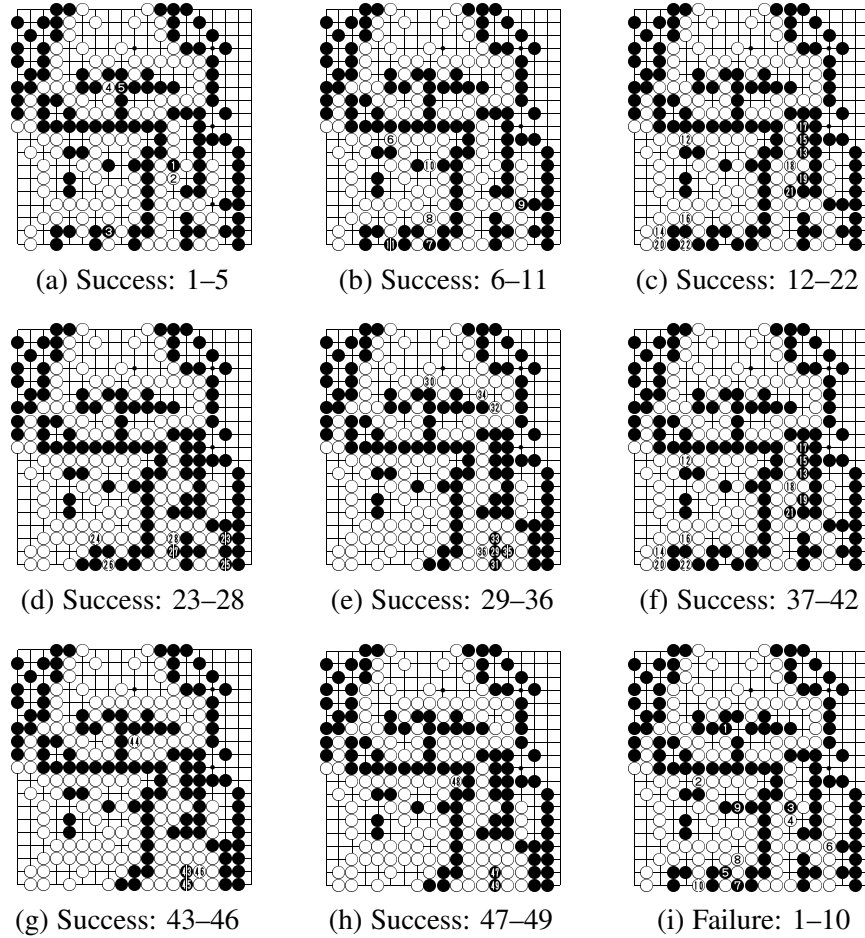
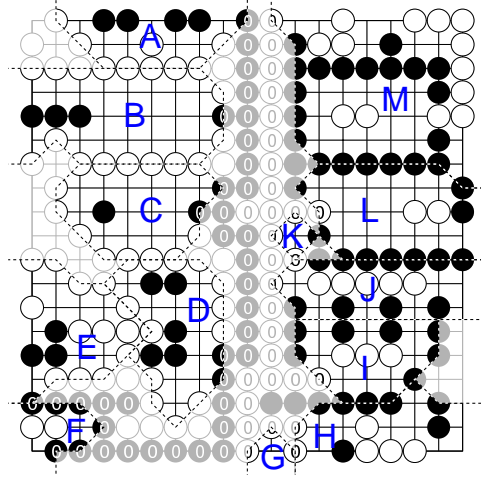


Figure 15. Example sequence to win.

5. Corridors

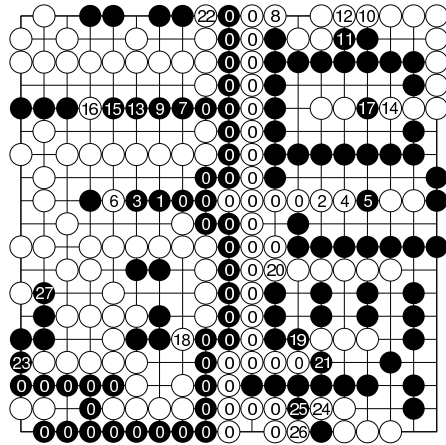
Corridors are simple positions that are precisely analyzable and hence give us good examples. In **SemGo**, corridors of width three exactly correspond to the corridors of width one in endgames, and corridors of width one or two have integer values. Figure 17(a) shows some examples of corridors of width three. The game tree of the corridor is shown in Figure 18 and all the corridors in



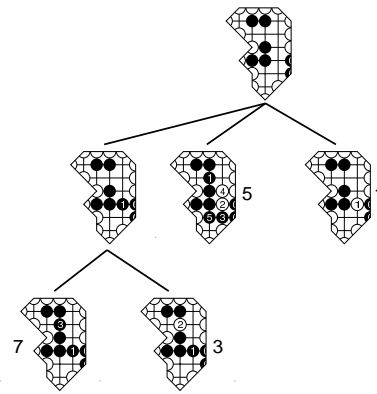
(a) Subgames

	value value	atomic weight
A	$1\frac{3}{4}$	0
B	$2\{-3 0^3\}$	-3
C	$3\{-1 0\}$	-1
D	$3\uparrow^*$	1
E	$1\frac{3}{4}$	0
F	3	0
G	-1	0
H	$-2\frac{1}{2}$	0
I	$-4\downarrow$	-1
J	-2^*	0
K	-1	0
L	$-3\{0 +2\}$	1
M	$-2\{0^2 +3\}$	2
Total	-1 ish	-1

(b) Game values



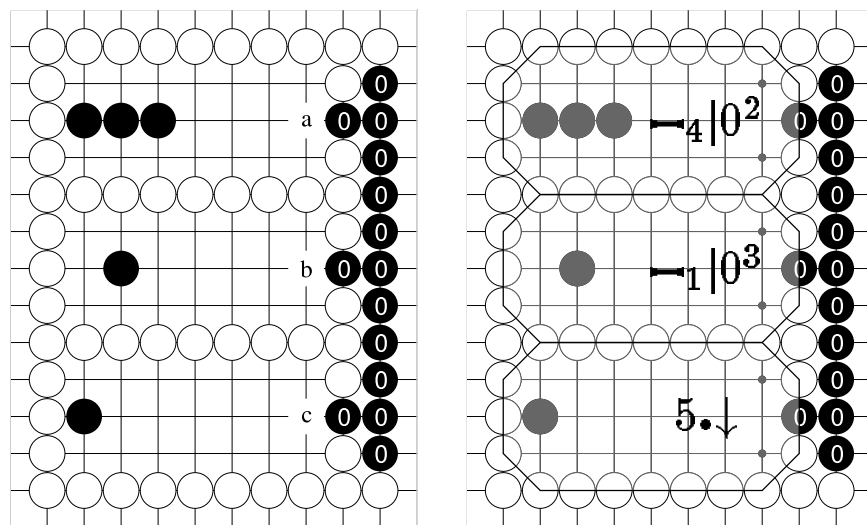
(c) Sequence to win



$$\{\{7 | 3\}, 5 | 1\} \xrightarrow{\text{cool by 2}} 3\{0, * | 0\} = 3\uparrow^*$$

(d) Game tree of subgame D

Figure 16. Analysis of Problem 6.



(a) Three corridors of width three (b) Cooled value of each corridor

Figure 17. Which move is the largest?

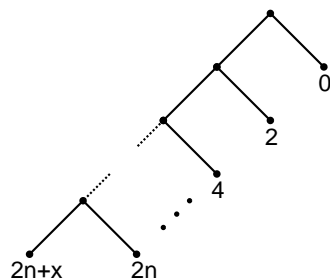
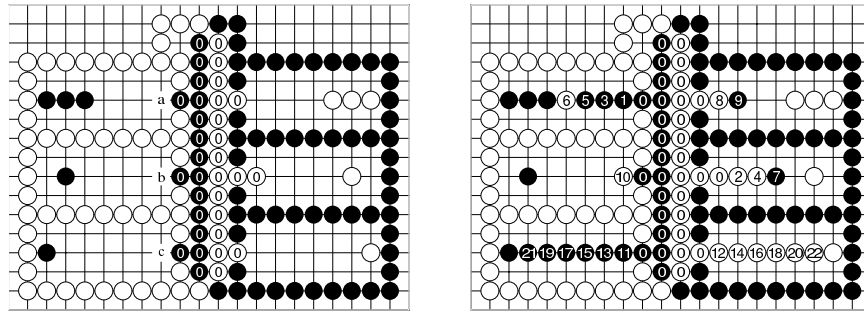


Figure 18. Game tree of the corridor.

Figure 17(a) are described using n and x : $n = 3$, $x = 8$ for the top corridor, $n = 4$, $x = 5$ for the middle, and $n = 5$, $x = 4$ for the bottom.

Now we can answer the question “Which move is best: a , b , or c ?” Using the cooled value of each corridor shown in Figure 17(b), we conclude $b > a > c$. In order to verify $b > a$, we use the difference game shown in Figure 19. In Figure 19(a), the right side is a mirror image of the left side except that Black’s essential block has one more liberty in the top center and White has just played at the point corresponding to b . It is obvious that Black can win the game if he plays at b in his first move and follows a symmetric strategy supposing that an essential block cannot make two eyes without capturing the opponent’s essential block. But if Black plays at a , White gets *tedomari* and Black loses. Figure 19(b)



(a) Problem (Black plays first) (b) An example of losing sequence

Figure 19. Difference semeai game.

shows an example of a losing sequence. Whatever sequence Black plays after his first move at *a*, White can win the difference game.

Figure 20 is a catalog of corridors in **SemGo**.

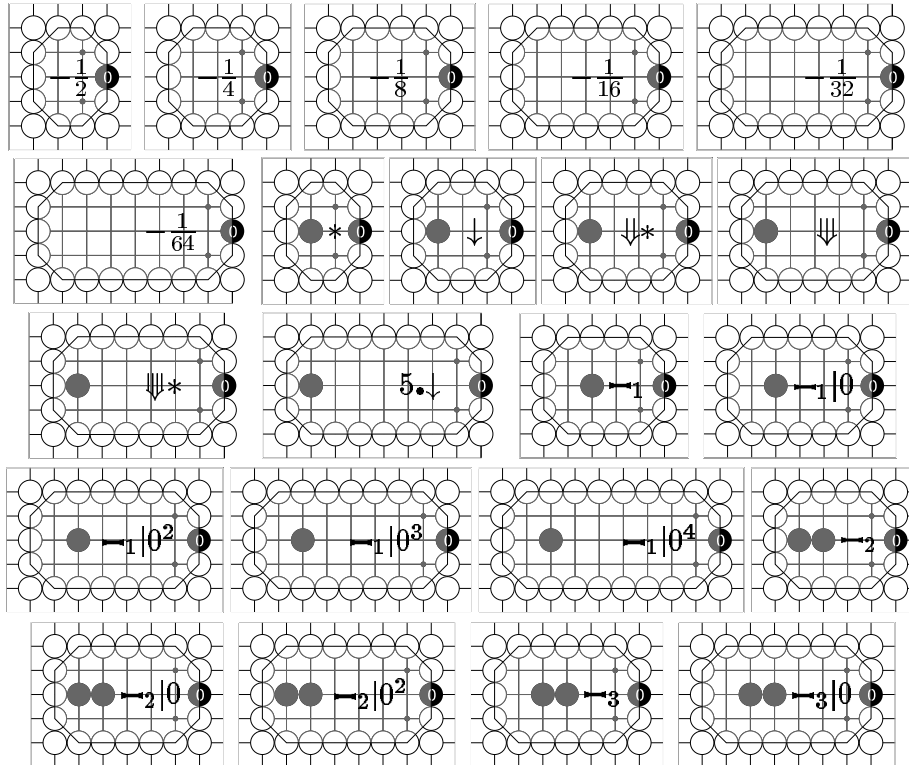


Figure 20. Catalog of corridors.

6. Summary and future work

In this paper, we described a new genre of applications of CGT to the game of Go. We proposed a method of counting liberties in capturing races using CGT and analyzed some complicated semeai problems which seem to be hard to solve even for the most experienced players. Although we have assumed that **SemGo** positions have no shared liberty regions so far, we can easily extend the applicability of our method to **SemGo** positions with some simple shared liberties.

Formula 2 : Extended semeai formula

G : Entire semeai game with the exclusion of shared liberty regions

Δ' : Adjustment value of $\text{Cool}(G, 2)$ for the attacker

S : Number of shared liberties

$$F = \begin{cases} S & \text{if } S = 0 \text{ or defender has an eye,} \\ S - 1 & \text{if } S > 0 \text{ and defender has no eye.} \end{cases}$$

The attacker can win the semeai if $\Delta' \geq F$.

Formula 2 is a straightforward improvement of Müller's semeai formula to be applied to the higher class of semeais. In Formula 2, we use the adjustment value of the sum of external liberty regions instead of a simple number of liberties.

Analyzing capturing races poses various problems. In shared liberty regions, a player can be an attacker and a defender at the same time. Properties of complicated shape of shared liberty regions are still unknown. Kos in capturing races have different characteristics from kos in endgames and capturing races with kos are also difficult to analyze. In order to truly analyze Go positions, we have to integrate all the analyses of eyespaces, capturing races and endgames into a unified description.

Acknowledgements

The basic idea of this research came into my mind in the summer of 2001, when I visited Berkeley to study Combinatorial Game Theory and its application to the game of Go with Professor Berlekamp and his research group. I am very grateful to Elwyn Berlekamp, William Fraser and William Spight for their kind advice and useful suggestions. I also want to thank Rafael Caetano dos Santos for many discussions about the game of Go and valuable comments on this paper.

References

- [1] John H. Conway: “On Numbers and Games”, Academic Press (1976).
- [2] Elwyn Berlekamp, John H. Conway and Richard K. Guy: “Winning Ways for your Mathematical Plays”, Academic Press, New York (1982).
- [3] Elwyn Berlekamp and David Wolfe: “Mathematical Go: Chilling Gets the Last Point”, AK Peters (1994).
- [4] Elwyn Berlekamp: “The economist’s view of combinatorial games”, *Games of No Chance*, Cambridge University Press, pp. 365–405 (1996).
- [5] H. A. Landman: “Eyespace Values in Go”, *Games of No Chance*, Cambridge University Press, pp. 227–257 (1996).
- [6] Martin Müller, Elwyn Berlekamp and William Spight: “Generalized Thermography: Algorithms, Implementation, and Application to Go Endgames”, International Computer Science Institute, TR–96–030, (1996).
- [7] Takenobu Takizawa : “Mathematical game theory and its application to Go endgames”, IPSJ SIG-GI 99–GI–1–6, pp. 39–46 (1999).
- [8] Martin Müller: “Race to capture: Analyzing semeai in Go”, Game Programming Workshop ’99 (GPW ’99), pp. 61–68 (1999).
- [9] Teigo Nakamura, Elwyn Berlekamp: “Analysis of composite corridors”, Proceedings of CG2002 (2002).
- [10] William L. Spight: “Evaluating kos in a neutral threat environment: preliminary results”, Proceedings of CG2002 (2002).
- [11] Teigo Nakamura: “Counting liberties in capturing races using combinatorial game theory” (in Japanese), IPSJ SIG-GI 2003–GI–9–5, pp. 27–34 (2003).

TEIGO NAKAMURA

KYUSHU INSTITUTE OF TECHNOLOGY, FUKUOKA, JAPAN

teigo@ai.kyutech.ac.jp