ゲ ー ム 情 報 学 8－5
(2 0 0 2. 7. 2 4)

An Efficient Use of Piece-Square Tables in Computer Shogi

Jun Nagashima*, Masahumi Taketoshi*, Yoichiro Kajihara*,
Tsuyoshi Hashimoto** and Hiroyuki Iida*
* Departments of Computer Science and ** Systems Engineering
Shizuoka University
3-5-1 Johoku, Hamamatsu, 432-8011 Japan

Abstract

The play of the opening game must be improved in order for computer shogi to reach a higher level. Due to the poor long-term strategies, computer shogi often leads to a disadvantageous position in the opening. Making a good formation is required for keeping the positional balance even. Computer achieves this using the opening book and piece-square tables. In this paper, we propose an efficient use of the piece-square tables during a game. These tables help our shogi program TACOS play reasonable in opening and middle game.

1 Introduction

In computer shogi a prerequisite for a champion program is proper opening play. So far it is a serious impediment to achieving high performance. Hence this topic needs considerably more research attention[1].

In chess playing an out-of-book move in the opening is risky; in shogi even a grandmaster sometimes plays such a move, especially against an opening expert. Most shogi games do not follow book moves for very long. However, games played by grandmasters show that a poor formation or poor development can rapidly lead to a disadvantageous position. Making a good formation in shogi is very important for obtaining a positional advantage or for keeping the positional balance even. Most shogi programs have an opening database. Using the database, programs can build a grandmaster-like opening formation if both sides follow exactly the same opening line.

Therefore, our research questions in this paper are: How can a program build a good formation in the out-of-book case? and How can a program maintain such a good formation during the game?

The main idea is to continue the building of a good formation that would have arisen from the opening line played so far, even when the opponent does not exactly follow the normal opening sequence. This requires consideration of potentially tactical plays (i.e., the playing of one move of a move group) that might upset the creation of a previously-planned good formation. In practice it is not so easy to realize this idea, i.e., identifying whether the opponent's out-of-book moves require the choice of a different target formation. So far only a few programs address this problem, and consequently it is common to see strange positions arising from the opening in games played by computer programs.

Tacos, our shogi program uses an opening database only when the opponent exactly follow an opening line. As soon as an out-of-book position is reached, the program ends the exploitation of the database.

Tacos will continue to construct a castle using hill climbing, and develop pieces. Tacos has piece-square tables for this purpose for many kinds of opening lines.

In this paper, we propose an efficient use of the piece-square tables during the game. In Section 2 we describe the related works such as the hill climbing for constructing a castle and the castle-and-assault maps for guiding the opening and middle game. These idea have been incorporated into our shogi program TACOS, and then evaluate by performing several experiments on TACOS. Some problems that happen during implementation and result of this test are presented in Section 3. The experiments performed confirm the effectiveness of the piece-square tables and we will see another new problem to be solved. We tackle this problem in Section 5. In Section 6, conclusions and future works are given.

2 Related Works

To our best knowledge, there are two published contributions related to our research questions. One is the use of hill climbing for constructing a castle in a reasonable way. Another one is the use of the castle-and-assault maps for guiding both the opening and middle game. Some details of the two ideas are given here.

2.1 Constructing a castle using hill climbing

The opening line to be followed is determined at random or is selected depending on the position of the opponent's line. According to this method, a formation will be made based on the difference between the scores given to a piece occupying a square and given to the pieces on the neighbour squares of the square to which the piece would move. The bigger the difference, the sooner the piece will want to move that square. With this method a good formation will be made after a reasonable number of moves. This idea makes it possible to make a good formation even when the position is out-of-book. The method has originally been proposed by Yamashita[4] in his program YSS and followed by other strongest programs such as KANAZAWA and SHOTEST.

Let us show, in Figure 2.1, an example of hill climbing. The destination for the piece Gold on 6i is 7h, while it is 7g for Silver on 7i The movements to complete the formation will be made in the order: S7i-7h, S7h-7g, and G6i-7h by margin (+8, +6, +4).

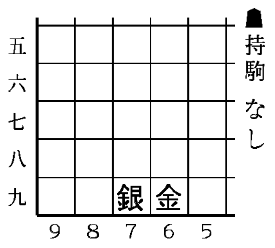
2.2 Castle-and-Assault Maps

Grimbergen and Rollason[3] proposed a new method for building castle formations and assault formations in the opening and middle game in shogi. To guide the building of castles, two notions called *castle maps* and *assault maps* were introduced. This is a straight extension of the method using hill climbing for building a castle to making a good formation for attacking as well.

3 Problems on Piece-Square Tables

Based on the two methods mentioned above, we implemented the piece-square tables in TACOS. In this process, we found several general problems to be solved. We discuss the problems below.

There are mainly four problems: (1) we meet the same scored position after different paths, (2) the move sequence led by piece-square tables is sometimes not stable in variable-depth search, (3) we meet the case where the best move cannot be selected due to the piece-square tables, and (4) we should make a good choice among piece-square tables depending on the opponent line.



```
// Table for Gold
{// 9  8  7  6  5  ...
  ...
  { 0, 0, 0, 0, 0, ... },
  { 0, 0, 0, 0, 0, ... },
  { 0, 0, 0, 0, 0, ... },
  { 0, 0, 4, 0, 0, ... },
  { 0, 0, 0, 0, 0, ... },
},
```

```
// Table for Silver
{// 9  8  7  6  5  ...
  ...
  { 0, 0, 0, 0, 0, ... },
  { 0, 0, 0, 0, 0, ... },
  { 0, 0,14, 0, 0, ... },
  { 0, 0, 8, 0, 0, ... },
  { 0, 0, 0, 0, 0, ... },
},
```

Figure 1: An example of hill climbing for the building of a castle.

3.1 Same position appear in search

When using the hill-clibming method, the same-scored position may appear after different paths. For example, we consider the case where a program looks a head by 3-ply in the position of Figure 2.1. In the position, the move S7i-7h has the highest priority. Then, expected moves would be S7i-7h, S7h-7g, and G6i-7h in this order. Another path to the same position is S7i-6h, G6i-7h, and S6h-7g. The target position is the same, but either of the two different paths should be appropriately selected depending on the situation.

For this purpose it is necessary to make the order into such distinct paths by which the same target position appears. Our solution to the problem is the addition of a margin that increases the priority with which the move is selected first. Indeed, this addition makes the order. For example, evaluation of position led by the move sequence S7i-7h, S7h-7g and G6i-6h obtains 26 points (position value 18 points plus margin 8 points), whereas the move sequence S7i-6h, G6i-7h and S6h-7g has 18 points (position value is 18 points, but without margin).

3.2 No stability under variable-depth search

When using not-fixed depth search such as realization probability search [7] or selective deepening like search extension, the move recommended with the highest priority by piece-square tables cannot be sometimes chosen. Figure 3 shows the progress of the changes of position evaluation in function of search depth. Here, ● is higher than ■ when positions are evaluated at the same ply. However, ■ is higher than ● when positions are evaluated at the different ply. Searching more deeply can solve this problem. But if the depth is not deep enough, the problem still remains.

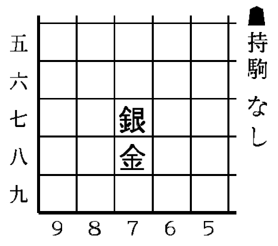


Figure 2: Destination Position

3.3 Problem with prior moves independent piece-square table

There are some cases that prior moves in piece-square table cannot make a suitable formation. The position shown in Figure 4 causes two problems: (1) program cannot play P9g-9f when playing Black side, and (2) program cannot play P9c-9d when playing White side.

These two problems are caused by the

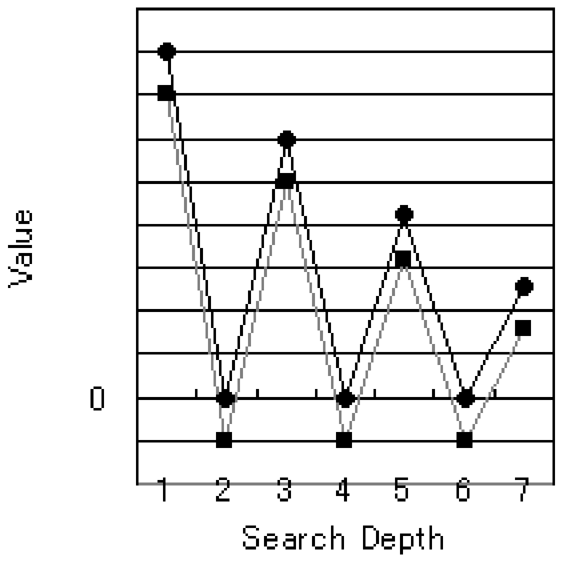


Figure 3: Changes of evaluation

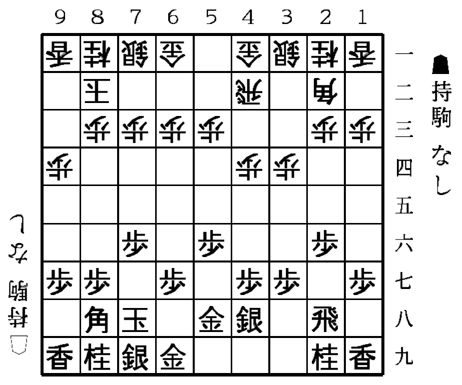


Figure 4: Prior moves and response moves

same problem as well as the previous problem, i.e., the best move (expected to play) cannot be selected. Other moves such as S4h-5g(Black) or S7a-7b(white) have the higher priority according to the piece-square table. Even though there is a better move to play currently than moves recommended by piece-square tables, the program cannot understand without looking ahead deeply to see the disadvantage of the choice of the piece-square tables.

To avoid this problem, we apply a method to decide priority without using piece-square tables. The method was originally proposed by Kanazawa [6] to evaluate moves directly. In our program, moves that have to be evaluate are listed before starting a game-tree search, and if such moves appear at the first ply, evaluation values of moves are added.

3.4 How to determine an appropriate piece-square table

When a program has more than two piece-square table candidates, one table to evaluate the formation must be determined. The piece-square table to be followed would be determined at random or selected depending on the progress of the game. In case, generic piece-square tables are to be used to play the very first stage of the opening. The point is that a program has to identify the current formation of the opponent as well as the target formation of the opponent.

Piece-square tables could be used to identify a formation and evaluate the formation. Then, the formation will be identified among similar formations by selecting a formation with the highest score. This method however has some shortcoming due to the pattern matching problem. To avoid this, sample formations for the comparison are prepared to determine a piece-square table in our program.

4 Performance

We perform several experiments to confirm the effectiveness of piece-square tables and the use of the table to evaluate formations. The performance of Tacos in which the ideas are incorporated is evaluated.

4.1 Experimental design

We prepare three programs as shown in Table 1. One is the version without the proposed ideas (Type-A). Second one has piece-square tables. The third one has piece-square tables as well as the routine to determine the tables for formation evaluation.

Type-A has the only one piece-square table. Type-B and Type-C has multiple tables and Type-B do not have the routine that selects a table to evaluate a formation. Thus Type-B cannot decide a table with detail information given by looking into the opponent formation. To identify the table, Type-B will see the location of Rook of the opponent since it is the strongest piece of assault. Using the information and rules given beforehand, Type-B will identify the formation. If there are not suitable tables when Type-C selects its information, Type-C also decides in this way.

For the experiments, we use the positions that appeared in the opening of the game scores played by shogi GMs. These positions seem to be even because GMs seldom make mistakes in the opening stage.

To compare with results of starting from ten moves positions, we also prepare the positions which appeared after 30 moves after the initial position. These positions seem to be in the late opening or early middle games. Therefore the results using these positions show how effectively piece-square tables works after the castle building is finished. Starting from these positions, different programs played each other in the experiments. A position is played twice as Black and White.

4.2 Results

The experimental results are given in Table 2.

Type3 outperformed Type1 (by 24 wins) and Type2 (by 14 wins) at positions on 10 moves from the initial position. The results clearly show that Type3 is the strongest among them, that is, the use of piece-square tables and the routine to identify the formation is effective.

Compared to the results of matches between Type-A and Type-C, the results of matches between Type-A and Type-B shows the importance of the routine to identify the formations. Checking up on game records, we find that Type-B failed to identify the right formation of Type-A. The reason is that Type-A made an eccentric formation because Type-A knows the only formation and always makes that formation, and Type-B identifies the opponent formation with the general rule. For this misunderstanding, Type-B cannot make a suitable formation against Type-A's assault and defeated. In contrast, Type-C identifies the opponent formation with the sample formations and even if the opponent makes an eccentric formation Type-C distinguishes properly.

The results of the games with positions that appeared after 30 moves from the initial position show that our piece-square tables do not work well in the middle game. In the next section, we discuss on this problem to consider the solution.

5 Piece-square tables in the middle game

5.1 Problems

Piece-square tables are used in two ways: one is for developing a formation, another one is for evaluating a formation. To develop a formation, intermediary squares have some value even if the location is worthless in evaluating a formation. It may not be a problem in the opening because the position is not tactical, but it can be so when a program must increase defensive pieces. In previous experiments performed in Section 4, our piece-square tables did not work so well and we think that it is the reason of this problem.

For this problem, we suggest that piece-square tables must be divided into two parts: one has values for making formation and the other has values for the evaluation of formations, and be used only when required. A

Table 1: Programs used experiment

Program name	piece-square table	selection routine
Type-A	×	×
Type-B	○	×
Type-C	○	○

Table 2: Results of experiments

	10	30	Result
Type-A vs. Type-B (Winning percentage of Type-B)	52 — 38 (42 %)	69 — 21 (23 %)	121 — 59 (33 %)
Type-A vs. Type-C (Winning percentage of Type-C)	7 — 83 (92 %)	56 — 34 (38 %)	63 — 117 (65 %)
Type-B vs. Type-C (Winning percentage of Type-C)	37 — 53 (59 %)	43 — 47 (52 %)	80 — 100 (56 %)

program uses the tables for making formations in the opening and the tables for the evaluation of formations when the formation completes.

We show these tables in Figure 5 - 8.

5.2 Experiments

To know whether the division of piece-square tables makes a program stronger, we perform experiments in the same way like the previous ones. We prepare three programs as shown in Table 3.

Type-E has piece-square tables with no division and Type-F has divided tables. To compare it with experiments in Section 4.1, Type-D which has the only one table like Type-A is prepared. In the experiments, we use again the same positions as used in the previous experiments. The results are given in Table 4.

The results of games with positions after 30 moves from the initial position show that Type-F is strong in the middle game and the division of tables works effectively in the middle game.

```
{ //  9    8    7    6    5  ···
  ···
  {   0,   0,   0,   0,   0, ··· }, // 5
  {   0,   0,   0,   0,   0, ··· }, // 6
  { -30, -30, -30, -30, -50, ··· }, // 7
  {  80, 100,  60,   5, -50, ··· }, // 8
  {   0,   0,  67,  31, -23, ··· }, // 9
},
```

Figure 5: Piece-square table for king in opening game

```
{ //  9    8    7    6    5  ···
  ···
  {   0,   0,   0,   0,   0, ··· }, // 5
  {   0,   0,   0,   0,   0, ··· }, // 6
  { -50,   0, -50, -50,-100, ··· }, // 7
  { -50, 125, -20, -50,-100, ··· }, // 8
  {   0,   0, -20, -50,-100, ··· }, // 9
},
```

Figure 6: Piece-square table for king in middle game

Table 3: Programs used in experiment

Program name	Tables for opening game	Tables for middle game
Type-D	×	×
Type-E	○	×
Type-F	○	○

Table 4: Result of experiments

	10	30	Result
Type-D vs. Type-E (Winning percentage of Type-E)	5 − 22 (81 %)	13 − 14 (52 %)	18 − 36 (67 %)
Type-D vs. Type-F (Winning percentage of Type-F)	13 − 15 (54 %)	12 − 16 (57 %)	25 − 31 (55 %)
Type-E vs. Type-F (Winning percentage of Type-C)	16 − 14 (47 %)	11 − 16 (59 %)	27 − 30 (53 %)

```
{ //  9    8    7    6    5  ···
  ···
  {   0,   0,   0,   0,   0, ··· }, // 5
  {   0,   0,   0,   0,   0, ··· }, // 6
  {   0,   0,  20,  44,   0, ··· }, // 7
  {   0,   0,  46,   0,  -7, ··· }, // 8
  {   0,   0, -20, -11, -40, ··· }, // 9
},
```

Figure 7: Piece-square table for gold in opening game

```
{ //  9    8    7    6    5  ···
  ···
  {   0,   0,   0,   0,   0, ··· }, // 5
  {   0,   5,   5,  10,   5, ··· }, // 6
  {   0,  10,  20,  40,   5, ··· }, // 7
  {   0,   0,  50,  25,   0, ··· }, // 8
  {   0,  15,  10,   0,   0, ··· }, // 9
},
```

Figure 8: Piece-square table for gold in middle game

6 Conclusions

In this paper, we have incorporated the piece-square tables and evaluated by performing several experiments on TACOS. The results obtained show that the piece-square tables make a program stronger in the opening and we meet a problem which happens in the middle game. Using the idea of the division of the piece-square tables, the problem is handled.

At present, our program cannot play a good opening yet. When a program completely plays a good opening, our next target will be an active opening play, i.e., it plays an aggressive opening to gain advantages as large as possible.

References

[1] Iida, H., Sakuta, M. and Rollason, J.: Computer Shogi. Artificial Intelligence 134(1-2), pp.121–144, 2002.

[2] H.Matsubara and T.Takizawa: How Shogi Programs Become Such Strong As Amateur 4-dan, Journal of Japanese Society for Artificial Intelligence Vol.16 No.3, pp.379–384, 2001. (In Japanese).

[3] R.Grimbergen and J.Rollason: Using Castle and Assault Maps for Guiding Opening and Middle Game Play in Shogi, Proceedings of The 6th Game Programming Workshop, editor, H.Iida, pp.102–109, October 2001.

[4] H.Yamashita: YSS-About its Datastructures and Algorithm, In H.Matubara, editor, Computer Shogi Progress 2, pp.112–142, Kyoritsu Shuppan Co, 1998. ISBN 4-320-02892-9. (In Japanese).

[5] Y.Tanase: Algorithm of IS Shogi, In H.Matubara, editor, Computer Shogi Progress 3, pp.1–14, Kyoritsu Shuppan Co, 2000. ISBN 4-320-02956-9. (In Japanese).

[6] S.Kanazawa: Algorithm of Kanazawa Shogi, In H.Matubara, editor, Computer Shogi Progress 3, pp.15–26, Kyoritsu Shuppan Co, 2000. ISBN 4-320-02956-9. (In Japanese).

[7] Y.Tsuruoka, D.Yokoyama, T.Maruyama, T.Chikayama: Game-Tree Search Algorithm Based on Realization Probability, Proceedings of The 6th Game Programming Workshop, editor, H.Iida, pp.17–24, October 2001.