Plausible Move Generation Using Move Merit Analysis with Cut-Off Thresholds in Shogi

Reijer Grimbergen

Complex Games Lab, Electrotechnical Laboratory
1-1-4 Umezono, Tsukuba-shi, Ibaraki-ken, Japan 305-8568
grimberg@etl.go.jp

Abstract. In games where the number of legal moves is too high, it is not possible to do full-width search to a depth sufficient for good play. Plausible move generation (PMG) is an important search alternative in such domains. In this paper we propose a new method for plausible move generation in shogi. During move generation, Move Merit Analysis (MMA) gives a value to each move based on the plausible move generator(s) that generated the move. These values can be used for different cut-off schemes. We investigate the following alternatives: 1) Keep all moves with a positive MMA value; 2) Order the moves according to their MMA value and use cut-off thresholds to keep the best N moves. PMG with MMA and cut-off thresholds can save between 46% and 68% of the total number of legal moves with an accuracy between 99% and 93%. Tests show that all versions of shogi programs using PMG with MMA outperform an equivalent shogi program using full-width search. It is also shown that MMA is vital for our approach. Plausible move generation with MMA performs much better than plausible move generation without MMA. Cut-off thresholds improve the performance for $N = 20$ or $N = 30$.

Keywords: Plausible move generation, move merit analysis, cut-off thresholds, shogi.

1 Introduction

Full-width search has been very successful in two-player complete information games. Deep Blue in chess [20], Chinook in checkers [19] and Logistello in Othello [5] are examples of well-tuned full-width search programs that perform at the level of the human world champions.

In full-width search all legal moves in any given game position are generated. However, this does not mean that all legal moves are searched to the same depth. Based on domain-dependent heuristics, selectivity is added: some moves will be searched deeper than other moves. Examples of methods to add selectivity to the full-width search are *quiescence search* [3], *singular extensions* [1] and *futility pruning* [8].

Full-width search has not always been the main approach. *Plausible Move Generation* (PMG) was very important in the early days of chess research. A plausible move generator would select a small number of moves using domain-specific knowledge [15, 4, 7]. The remaining candidates were then searched as deep as possible with alpha-beta search. For example, Bernstein's chess program [4] generated only 7 plausible moves in any position. Plausible move generation is the ultimate form of selectivity: discarding moves without any search. In chess, the risk of discarding a good search candidate was too high and full-width search has been the dominant approach since the CHESS 4.5 program in the early seventies [23].

However, there are games in which it is impossible with current technology to search deep enough with standard full-width search to get a high performance program. Examples are games with a large average number of legal moves like *Go* and *shogi* [14] and single agent search problems with extremely long solution sequences such as *sokoban* [10]. To make a high performance program in these domains, some method for plausible move generation is needed [6, 24, 12, 26, 11]. Especially in Go, most of the available time per move is spent on generating promising looking moves, leaving little time for search [6].

In this paper we propose a new method of plausible move generation. Even though the method has been designed for shogi, we will present a framework for plausible move generation that will be applicable to other two-player complete information games as well. We think that one of the problems of plausible move generation in current programs is the lack of a combined effort to develop a general and satisfactory method. Shogi is a good example of this, and we will see in Section 7 that there are big differences between the plausible move generation methods used in some of the top programs. With the plausible move generation method that we present in this paper, we hope to make it easier to compare the different methods that are currently used and also make it easier for others to implement and improve their own plausible move generation method.

In section 2 we will explain why plausible move generation is a good alternative to full-width search in shogi. Then, a set of plausible move generators for shogi will be defined in Section 3. In Section 4 we will explain how analysing the merit of a move using these plausible move generators can improve move ordering and reduce the set of candidate moves generated by the set of plausible move generators. With this analysis of move merit additional cuts in the number of candidate moves can be made. These additional cuts are explained in Section 5. In Section 6 we will show that shogi programs based on plausible move generation with move merit analysis outperform an equivalent program using full-width search. In Section 7 we will compare our method with other plausible move generation methods used in shogi. We will end with some conclusions and ideas for future work in Section 8.

2 Why is Plausible Move Generation Necessary in Shogi?

The main difference between chess and shogi is the possibility of re-using pieces. A piece captured from the opponent becomes a *piece in hand* and at any move a player can *drop* a piece he captured earlier on a vacant square instead of moving a piece on the board. As a result of these drop moves, the number of legal moves in shogi is on average much larger than in chess. The average branching factor of the search tree in chess is about 35, while in shogi the average branching factor is about 80 [13].

In shogi the average branching factor does not tell the whole story. In chess the branching factor rapidly decreases towards the endgame and finally gets to a point where the exact theoretical game value can be retrieved from endgame databases [25]. This is not the case in shogi, where the branching factor of the search tree increases as the game progresses. To illustrate this behaviour, we have analysed the number of legal moves in 100 expert shogi games. The games have been selected to give a good coverage of the different types of positions that occur in shogi. The games therefore involve many different expert players (112) and have many different opening strategies (15).

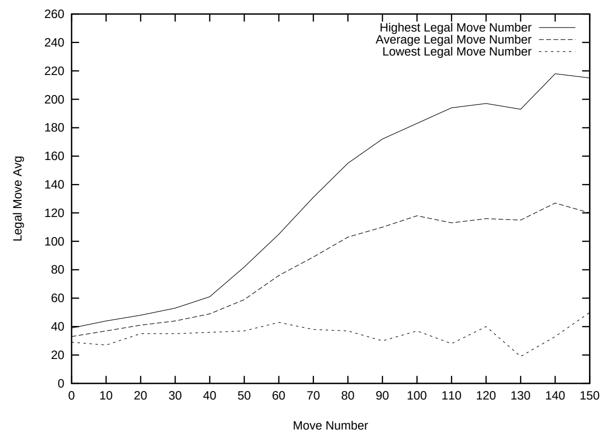

Fig. 1. Highest number of legal moves, average number of legal moves and lowest number of legal moves by move number in 100 test games.

The number of legal moves in the test games is given in Fig. 1. This figure shows that the number of legal moves in shogi increases as the game progresses. As more pieces get captured, the number of possible drop moves increases, leading to an average number of legal moves that is higher than 100 in the endgame. The top line in Fig. 1 shows that peaks of more than 200 can also be expected. The result of a shogi game is often decided in the endgame, so being able to deal with search trees that have such a high branching factor can mean the difference between winning and losing.

Not only the high average number of legal moves is a problem for building a strong shogi program. There is also the problem of strict time constraints. In shogi, the available time for finding the next move under tournament conditions is much less than in chess. There are two reasons for this. First, the average game length of shogi is about 115 ply [9], while the average game length of chess is about 80 ply. Therefore, even under the same conditions, a shogi program will have 30% less available time per move. Second, the tournament conditions for shogi programs are much stricter than in chess. In the annual CSA tournament, the computer shogi world championships, the available time per game is only 20 minutes. Therefore, even with the help of an opening book, on average only about 30 seconds per move are available for search.

To deal with large search trees under strict time constraints it is necessary to make good decisions about which moves the available search time should be spent on. Plausible move generation is a method to make these decisions and therefore can be an important alternative to full-width search in shogi.

3 A Set of Plausible Move Generators for Shogi

As the general game-playing system METAGAMER [17] shows, a wide range of games have the notion of goal, threat and positional improvement in common. The goal of a game can be to win material (e.g. chess, checkers), to occupy the largest territory (e.g. Go, Othello), or to reach a certain board configuration (e.g. five in a row, sokoban). Some goals are more important than other goals. In chess, for example, mating (winning the king) is more important than winning a queen, which in turn is more important than winning a pawn. A threat is a move that, if not defended against, will reach a goal on the next move or after a forced sequence of moves. Finally, there are moves to improve the player's position without actually threatening to reach a goal. An example is too improve the mobility of a piece. It is also possible to defend against this type of move by playing a move that makes such a positional improvement impossible. An example in chess is pinning a piece.

For each of these move classes a plausible move generator can be build which generates the moves in this class:

1. **PMG-Goal**:
   Moves that reach a goal.
2. **PMG-Th**:
   Moves that threaten to reach a goal.
3. **PMG-DefTh**:
   Moves that defend against a threat.
4. **PMG-PIm**:
   Moves that improve the position.
5. **PMG-DefPIm**:
   Moves that make it impossible for the opponent to improve the position.

For each game in which this basic set of plausible move generators is used, the PMGs have to be refined to reflect the features of that specific game. In shogi, the goal of the game is different than in Go and *PMG-Goal* is therefore different as well. For shogi, we have split the five basic PMGs above in 21 shogi specific PMGs:

1. **PMG-Goal:**
   - *Capture material.*
   - *Promote piece.*
2. **PMG-Th:**
   - *Check.*
   - *Attack king.*
   - *Attack material.*
   - *Discovered attack.*
     Moving a blocking piece leads to check or to a material attack.
   - *Threaten promotion.*
3. **PMG-DefTh:**
   - *Defend against checks.*
   - *Defend king.*
   - *Defend material.*
   - *Defend discovered attacks.*
   - *Defend against promotion threat.*
4. **PMG-PIm:**
   - *Defend pins.*
   - *Tie improvement.*
     If a piece can not move because it is tied to the defence of another piece P, defend piece P.
   - *Defend undefended pieces.*
   - *Defend against the exchange of pieces.*
   - *Cover squares in own camp.*
     Moves that gain control over a square in one's own camp.
   - *Develop pieces.*
     Patterns and move sequences taken from expert games for 1) standard opening sequences, 2) building castles, and 3) positional pattern moves.
5. **PMG-DefPIm:**
   - *Pin piece.*
     Moving the pinned piece puts the king in check or loses material.
   - *Cover squares in opponent camp.*
     Moves that gain control over a square in the opponent camp.
   - *Avoid development.*
     Moves that do not allow the opponent to develop its position.

4 Move Merit Analysis

Generating each plausible move only once is faster than having the same move generated several times. Therefore, having multiple plausible move generators with the possibility of generating moves more than once might not be efficient. However, the possibility of generating a move by more than one PMG is vital for our approach. If a move is generated by more than one PMG, it is often better than a move that is generated only once. For example, moving a piece away from an attack is in general more powerful if it is attacking an opponent piece at the same time.

Table 1. Piece values of shogi pieces as used for MMA value calculation.

Piece	Value
King	10000
Promoted rook	1300
Promoted bishop	1200
Rook	900
Bishop	800
Gold/Silver	500
Knight/Lance	300
Pawn	100

Knowledge about which PMGs generated a move can be used to analyse the merit of the move. In our method, each PMG assigns a value to the generated moves based on the importance of the PMG. We will now give a detailed description of the values that the *Move Merit Analysis* (MMA) gives to the moves generated by the PMGs in shogi. As a reference, we have given the values of the shogi pieces that are used as the basic units of calculation of the MMA values in Table 1. It should be noted that because of the dynamic nature of shogi, there is no agreement on the values of the pieces as there is in chess.

1. **PMG-Goal:**
   - *Capture material*:
     Estimate of the material gain after a capture sequence on a square. For this estimate, MMA uses a static evaluator for sequences of piece captures on the same square.

   - *Promote piece*:
     Promotion value of the piece. Table 1 shows that this is the equivalent of 4 pawns for rook, bishop and pawn and 2 pawns for knights and lances (knights and lances promote to gold).

2. **PMG-Th:**
   - *Check*:
     A constant based on whether the check is on a safe square or a sacrifice:
     - Safe square: $V_{MMA} = 3 \times V_{Pawn}$.
     - Sacrifice: $V_{MMA} = 1\frac{1}{2} \times V_{Pawn}$.

     where V_{Pawn} is the value of a pawn.

   - *Attack king*:
     A constant based on whether the attack is on a safe square or a sacrifice:
     - Safe square: $V_{MMA} = 3 \times V_{Pawn}$.
     - Sacrifice: $V_{MMA} = 1\frac{1}{2} \times V_{Pawn}$.

- *Attack material*:
  The MMA value is an estimate of how strong the attack is. An attack on a high value piece like a rook has a higher MMA value than an attack on a low value piece like a pawn. The basic piece attack value is:

  $$AttackVal = \frac{1}{10} \times V_{AtteckedPiece}$$

  where $V_{AttackedPiece}$ corresponds to the piece value as given in Table 1.

  For pinned pieces, a bonus is given based on the value of the piece that would be lost if the pinned piece would move:
  - King (check): $AttackValBonus = \frac{1}{6} \times V_{PinnedPiece}$.
  - Major piece: $AttackValBonus = \frac{1}{7} \times V_{PinnedPiece}$.
  - Gold and silver: $AttackValBonus = \frac{1}{8} \times V_{PinnedPiece}$.
  - Knight and lance: $AttackValBonus = \frac{1}{9} \times V_{PinnedPiece}$.
  - Pawn: $AttackValBonus = \frac{1}{10} \times V_{PinnedPiece}$.

  where $V_{PinnedPiece}$ is the piece value of the pinned piece.

  In case of attacking a dead piece the piece attack value is doubled.

- *Discovered attack*:
  A constant based on which piece the discovered attack is attacking.
  - King: $V_{MMA} = 1\frac{1}{2} \times V_{Pawn}$.
  - Other piece: $V_{MMA} = \frac{1}{10} \times V_{AttackedPiece}$.

- *Threaten promotion*:
  - $V_{MMA} = \frac{1}{10} \times V_{Prom}$.

  where V_{Prom} is the expected value of the promotion.

3. **PMG-DefTh:**
   - *Defend against checks*:
     No MMA value is given to a defence against a check, as there are no other moves generated, so any MMA value would be given to each move anyway.

   - *Defend king*:
     The MMA value is an estimate of how strong the attack of the opponent on a square SQ adjacent to the king is:
     - SQ controlled by the opponent: $V_{MMA} = 3 \times V_{Pawn}$.
     - SQ controlled by the player to move: $V_{MMA} = \frac{1}{2} \times V_{Pawn}$.
     - SQ controlled by neither player: $V_{MMA} = 1\frac{1}{2} \times V_{Pawn}$.

   - *Defend material*:
     - $V_{MMA} = \frac{1}{2} \times V_{ExpectedLoss}$.

     where $V_{ExpectedLoss}$ is the value of the expected material loss after the capture sequence.

– *Defend discovered attacks*:
  Estimated improvement of the discovered attack. These values are the same as for playing the discovered attack.

– *Defend against promotion threat*:
  A constant based on which piece was threatening to promote. These values are the same as for promotion threats.

4. **PMG-PIm:**
  – *Defend pins*:
    The MMA value is based on the value of the piece that is being pinned. These values are the same as the bonus values in *Attack material*.

  – *Tie improvement*:
    The MMA value is based on the value of the piece that is tied and the material loss that would be the result if the tied piece should move:

$$V_{MMA} = \frac{V_{TiedPiece} + V_{TieLoss}}{40}$$

    where $V_{TieLoss}$ is the material loss resulting from moving the tied piece.

  – *Defend undefended pieces*:
    • $V_{MMA} = \frac{1}{8} \times V_{UndefendedPiece}$.

  – *Defend against exchange of pieces*:
    • $V_{MMA} = \frac{1}{10} \times V_{ExchangePiece}$
    where $V_{ExchangePiece}$ is the value of the piece that can be exchanged.

  – *Cover squares in own camp*:
    • $V_{MMA} = \frac{1}{10} \times V_{Pawn}$.

  – *Develop pieces*:
    • Piece development moves: $V_{MMA} = \frac{1}{2} \times V_{Pawn}$.
    • Castle moves: $V_{MMA} = 1\frac{1}{2} \times V_{Pawn}$.
    • Pattern moves: $V_{MMA} = \frac{1}{4} \times V_{Pawn}$.

5. **PMG-DefPIm:**
  – *Pin piece*:
    The MMA value is based on the value of the piece that is being pinned. These values are the same as the bonus values in *Attack material*.

  – *Cover squares in opponent camp*:
    • $V_{MMA} = \frac{1}{10} \times V_{Pawn}$.

  – *Avoid development*:
    • $V_{MMA} = \frac{1}{8} \times V_{Pawn}$.

The values of the constants and weights used in the calculation of the MMA values have all been tuned by hand. An interesting future work is to investigate if these values can be learned automatically.

Negative MMA values are also possible because MMA can give a negative value to three types of moves:

- Material sacrifices: The penalty for sacrifices is the value given by the static exchange evaluator.
- Drops far from the kings: A drop move is given a penalty of $5 \times \min(BlackKingDistance, WhiteKingDistance)$
- Passive moves, i.e. moves that are not a threat: These moves are given a penalty of half a pawn.

If the penalty is higher than the expected merit of the move, this will result in a negative MMA value.

After move merit analysis, the plausible moves can be ordered according to their MMA value. If this MMA based ordering is a good estimate of the importance of a move, this will improve the performance of alpha-beta search. However, the main advantage of using MMA is to make additional cuts in the number of candidate moves generated by the PMGs. A natural cut-off is to discard all candidate moves with a negative MMA value. In this paper, plausible move generation without MMA will be called *PMG-All* and plausible move generation that cuts all moves with a negative MMA value will be called *PMG-MMA*.

5 Cut-Off Thresholds

The MMA values can be used for further cuts in the number of candidate moves. Assuming that the move ordering based on MMA values is a good indication of the quality of a move, the number of moves to search can be further reduced by taking only the best N moves after MMA based move ordering. We have investigated the *cut-off thresholds* $N = 20, 30, 40, 50$. The PMGs with cut-off thresholds will be called *PMG-N*.

Some modifications are needed in the application of these cut-off thresholds. First, moves with a high MMA value should not be discarded, even if they fall outside the first N moves. Second, in shogi the merit of checks and captures is hard to judge statically. A captured piece can be dropped back on almost any vacant square, and static evaluation of good drop points is difficult. Therefore, checks and captures of pieces higher than pawns are included regardless of their MMA value. These two modifications can increase the number of candidate moves above the basic threshold N.

The third modification helps to improve the number of cut-offs. If moves have an MMA value that is much smaller than the top moves, they are discarded even if they fall in the range N.

6 Results

We have analysed the behaviour and performance of our method of plausible move generation with four tests:

1. Plausible move generation test.
2. Move ordering test.
3. Search comparison test in tactical shogi problems.
4. Self play experiment.

With the plausible move generation test and the move ordering test the accuracy and savings of *PMG-All*, *PMG-MMA* and *PMG-N* were analysed by comparing the moves generated by the PMGs with moves played by expert players. Of course, expert performance is not equivalent to perfect performance, but there are several reasons why the results of these tests are interesting. First, they give a first indication about the performance of each PMG method. Second, the differences between the PMG methods will be clearer. We will see that the balance between accuracy and savings varies considerably between *PMG-All*, *PMG-MMA* and *PMG-N*. Third, the results will make it easier to compare our PMG method with other methods.

The search comparison test and the self play experiment compared the search performance of shogi programs using *PMG-All*, *PMG-MMA* and *PMG-N* with an equivalent program using full-width search. This is necessary to answer the most important question: is our PMG method cost-effective, i.e. is the time spent on plausible move generation worthwhile?

6.1 Plausible Move Generation Test

First, we compared the savings and move generation accuracy of *PMG-All*, *PMG-MMA* and *PMG-N*. For this comparison, we used the 100 test games described in Section 2. These 100 test games have a total of 12097 positions. We tested the accuracy of *PMG-All*, *PMG-MMA* and *PMG-N* by checking if the move played by the expert was generated by the plausible move generators. We also calculated the savings of our approach, i.e. the difference between the total number of legal moves and the total number of moves generated by the PMG versions. The savings for *PMG-All*, *PMG-MMA* and *PMG-N* are given in Fig. 2. This figure shows that there are small areas of good and bad results, but that the majority of the savings are close to the average. It is also clear from the figure that the savings of *PMG-MMA* are much better than the savings of *PMG-All*. *PMG-N* gives a gradual increase in savings for smaller N. *PMG-20* and *PMG-30* give much more savings than *PMG-MMA*, but the difference between *PMG-40*, *PMG-50* and *PMG-MMA* is much less prominent.

Vital is the balance between the savings of the plausible move generation and the accuracy. The savings and accuracy results of *PMG-All*, *PMG-MMA* and *PMG-N* are summarised in Table 2. *PMG-All*, the basic plausible move generation without any additional cuts, can generate 99.4% of all expert moves

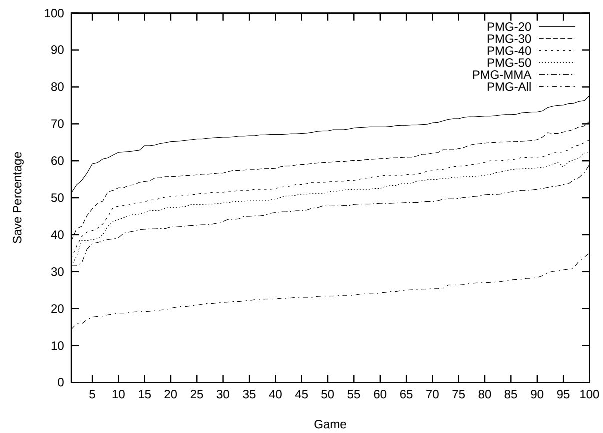

Fig. 2. Savings of different versions of plausible move generation in 100 test games, ordered by percentage of moves saved.

in the test games. This version on average reduces the number of moves with 23.7% compared to the total number of legal moves. The savings can be almost doubled if MMA is used. *PMG-MMA* saves 46.5% of all moves at the cost of 0.5% accuracy compared to *PMG-All*.

The results indicate that *PMG-50* is a good improvement over *PMG-MMA* as it improves the savings with 4.7%, while the accuracy is only reduced by 0.3%. There is also a good improvement of savings between *PMG-40* and *PMG-30* at 5.0%. However, the loss of accuracy compared with *PMG-40* is 1.4%. Finally, savings of 68% can be achieved with *PMG-20*, but the accuracy will then drop to 92.7%. Other tests are needed to show if the relatively low accuracy or the high savings are the deciding factor in the performance of *PMG-20*.

Table 2. Results of the different versions of plausible move generation on 12097 positions. *NG* is the number of moves played by the expert, but not generated by the PMG; *Ac* is the accuracy of the PMG; *Sv* are the savings of the PMG.

Version	NG	Ac(%)	Sv(%)
PMG-All	81	99.4%	23.7%
PMG-MMA	144	98.9%	46.5%
PMG-50	178	98.6%	51.2%
PMG-40	231	98.1%	54.2%
PMG-30	401	96.7%	59.2%
PMG-20	892	92.7%	68.0%

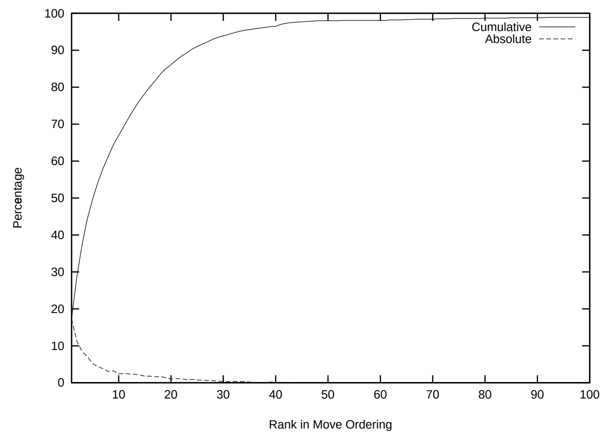

Fig. 3. Absolute and cumulative move ordering results in 100 test games.

6.2 Move Ordering Test

We have also looked at the move ordering results of MMA for each position in the set of test games. The results are given in Fig. 3. MMA orders the expert move first in 17.3% of the positions. Almost half of the expert moves (49.6%) are ordered among the best five moves. If the first ten moves in the ordering are considered, 66.9% of the expert moves are produced by the PMGs.

The absolute move ordering curve in Fig. 3 shows that there are only very few expert moves ordered lower than 30. These results support our assumption that MMA leads to a reasonable move ordering. The results also make it highly unlikely that a *PMG-N* for *N* higher than 50 will give better results than the *PMG-N* we investigate here.

6.3 Search Comparison Test in Tactical Shogi Problems

As a first test to compare the search performance of shogi programs using *PMG-All*, *PMG-MMA* and *PMG-N* with an equivalent program using full-width search, we compared the performance of different versions of the same shogi program on a set of tactical shogi problems. The PMG based programs all use plausible move generation at every node in the search tree.

The shogi problems in the test were taken from the weekly magazine *Shukan Shogi*. The test set consists of 300 problems published in issues 762 (November 4th 1998) to 811 (October 20th 1999). The problems in each issue are divided into six classes, ranging from starting level to expert level. It should be noted that the starting level is already quite advanced and is too hard for beginners. Two of the problems in the test set are incorrect and have been removed from the test set.

Table 3. Results of full width search and different versions of plausible move generation on 298 tactical shogi problems.

Cat	Tot Pos	Full width	PMG					
			All	MMA	50	40	30	20
1	50	17	21	22	21	23	23	22
2	50	9	8	13	12	12	13	10
3	50	10	10	11	11	12	10	11
4	50	9	7	8	8	9	9	8
5	50	5	4	4	4	4	5	7
6	48	4	5	6	6	6	6	6
Tot	298	54	55	64	62	66	66	64

The basic program that we used for all versions has the following features, which are common in most shogi programs:

– Iterative alpha-beta search.
– Principal variation search [16].
– Quiescence search [3].
– History heuristic and killer moves [18].
– Null-move pruning [2].
– Hashtables for transposition and domination [22].
– Specialised mating search [21].

All three programs have the same evaluation function. The evaluation function has the following features:

– Material.
– King safety.
– Piece mobility.
– Pinned pieces.
– Discovered attacks.
– Promotion threats.
– King distance of pieces.
– A number of piece patterns to evaluate good and bad piece formations.

All versions of the program were given 30 seconds per problem on a 700 MHz Pentium II, which is about the same time as can be expected to be available under tournament conditions.

The results of this test are given in Table 3. The categories in the table correspond to the categories in Shukan Shogi. The table shows that MMA is vital to our approach. There is almost no difference between the number of solved problems by full-width search and *PMG-All*. However, *PMG-MMA* and *PMG-N* solve significantly more problems than full-width search and *PMG-All*.

This test shows no significant improvement of the performance when cut-off thresholds are used. *PMG-N* solves about the same number of problems as *PMG-MMA*.

Table 4. Results of a self play experiment between a full-width search shogi program and shogi programs using different plausible move generation versions.

No	Version	1	2	3	4	5	6	7	P	W	L
1	PMG-20	x	10-10	11-9	11-9	11-9	16-4	17-3	5.5	76	44
2	PMG-30	10-10	x	11-9	13-7	10-10	16-4	20-0	5	80	40
3	PMG-MMA	9-11	9-11	x	12-8	12-8	15-5	20-0	4	77	43
4	PMG-40	9-11	7-13	8-12	x	11-9	18-2	19-1	3	72	48
5	PMG-50	9-11	10-10	8-12	9-11	x	17-3	18-2	2.5	71	49
6	PMG-All	4-16	4-16	5-15	2-18	3-17	x	16-4	1	34	86
7	Full-width	3-17	0-20	0-20	1-19	2-18	4-16	x	0	10	110

6.4 Self Play Experiment

As a final experiment to compare full-width search, *PMG-All*, *PMG-MMA* and *PMG-N* we played different versions of the same shogi program against each other. One program was using full-width search and the other programs were using different versions of our plausible move generation method. Each program version played the other versions twenty times with a time limit of 20 minutes per side per game. This is the same time limit as used in the annual CSA tournament. The results of this tournament are given in Table 4.

The results show that a PMG based program is playing better than a program without plausible move generation. Full-width search won only 10 out of 120 games. The results also show how important move merit analysis is for our approach. *PMG-All* only managed to win against the full-width search program and lost all other matches by a wide margin, on average winning less than 4 games against any of the MMA based programs. Move merit analysis is clearly an improvement of the general plausible move generation approach.

From the results of this tournament, it is difficult to draw conclusions on the importance of cut-off thresholds. *PMG-MMA* did surprisingly well, losing against *PMG-20* and *PMG-30* with only the narrowest possible margin of 11-9. Actually, *PMG-MMA* scored more points in the tournament than *PMG-20*, which won most matches. It seems that *PMG-50* and *PMG-40* are not an improvement over *PMG-MMA*, but *PMG-30* and *PMG-20* might be an improvement. Further tests with more games and against different opponents are needed to evaluate the importance of cut-off thresholds.

7 Related Work

In this section we will present different approaches for plausible move generation in shogi and compare them with the method presented in this paper. We will look at the plausible move generation in the shogi programs IS SHOGI, winner of the 1998 and 2000 CSA computer shogi championships; KANAZAWA SHOGI, winner in 1996 and 1999; YSS, winner in 1997; and KAKINOKI SHOGI, winner of the Computer Shogi Grand Prix in 1999.

7.1 IS Shogi

IS SHOGI [24] uses the following plausible move generators:

- Best move of the previous iteration.
- Capture opponent piece that just moved.
- Move piece that was attacked on the previous move to a safe square.
- Killer move.
- Null move.
- Attack king or attack material.
- Discovered attacks.
- Defend piece that was attacked on the previous move.
- Defence moves.
- Other special moves.

The categories are strictly ordered. If a move in a category leads to an alpha-beta cut-off or has a sufficiently high evaluation, none of the moves in the categories ordered below it will be generated. It is unclear which moves are generated by *Other special moves* as the description of this PMG is very short. There is only one example given of a special defence move to shut out pieces from attack.

The advantage of this plausible move generation method compared to our method is that no time is spent on generating moves that do not influence the search. The disadvantage is that it is not possible to make use of the extra information of moves that are generated by multiple plausible move generators.

7.2 Kanazawa Shogi

KANAZAWA SHOGI uses plausible move generation only in the following special cases [12]:

- Best move of the previous iteration.
- Take piece with the highest value.
- Move attacked piece with the highest value.
- Killer move.
- Null move.

If none of these moves are good enough to stop the search, all remaining legal moves are generated. After this, all moves are played and evaluated. Based on the evaluation, a decision is made on which moves to search and which moves to discard.

Strictly speaking, KANAZAWA SHOGI does not use plausible move generation, since in most cases all legal moves are generated. However, some legal moves are discarded without any search, so the method is very similar to plausible move generation. No detailed data on the proportion of moves that is being discarded without search is being given, so it is difficult to compare the savings of the method KANAZAWA SHOGI uses to MMA. The advantage of having all legal moves available is that it is easier to recover from a bad decision about the moves to search. The disadvantage is that all moves need to be evaluated. The number of legal moves in a position can be high and evaluation in shogi is expensive, so the extra number of evaluations can slow down the search.

7.3 YSS

Yamashita's YSS [26] uses 30 move categories. His plausible move generation is strongly related to the search depth. Moves are only generated if the remaining search depth is enough to show that the move can actually reach the goal implied by the move category. For example, a move that attacks a piece is not generated at depth 1, because it is not possible to show that the attack will have a positive effect on the position. Also, some moves are only generated at the start of the search and based on the stage of the game (opening, middle game or endgame). Here are YSS's move categories in detail:

- **Remaining search depth is at least 1:**
  - Capture opponent piece that just moved.
  - Capture undefended piece.
  - Promote piece.
  - Checks that do not sacrifice material.
  - Move attacked piece with highest value.
- **Remaining search depth is at least 2:**
  - Defend against strong threat.
  - Attack material.
  - Discovered check.
  - Attack king from the front.
  - Discovered attack.
  - Attack pinned pieces.
  - Drops of bishop and rook in the camp of the opponent.
- **Remaining search depth is 3 or higher:**
  - Attack pieces around the opponent king.
  - Attack tied defending pieces.
  - Capture material that has a higher value than a pawn.
- **Moves only generated at the first ply of search:**
  - Develop inactive pieces.
  - Sacrifices with check.
  - Pawn drops far from the promotion zone.
- **Moves only generated at the first two ply of search:**
  - Pawn pushes in front of rook and lance.
  - Material sacrifices that lead to a fork.
  - Pawn promotion sacrifice.
  - Dangling pawn.
  - Block opponent rook or bishop.
  - Move gold sideways in the camp of the opponent.
  - Attack opponent piece that just moved with a pawn drop.
  - Attack opponent piece that just moved.
  - Move the king.
- **Moves only generated at the first two ply of search in the opening:**
  - Attack a pawn with a piece.
  - Drop a pawn to make an attacking base (covering the opponent camp).
  - Develop inactive pieces.

YSS uses more plausible move generators than our method (30 PMGs instead of 21 PMGs), which might improve the quality of the plausible move generation, but also takes more time. Also, control of the search seems difficult with a plausible move generation that depends heavily on the search depth.

Still, YSS's practical results are very good, so this method deserves further investigation. It is not difficult to use our plausible move generation method in the same way. Only minor modifications are needed to relate plausible move generation to the search depth. Further research is needed to investigate if this improves the performance of our method and this is a future work.

7.4 Kakinoki Shogi

Kakinoki Shogi uses 8 basic move categories [11]:

- Capture material.
- Defend material.
  - Move away from attack.
  - Cover attacked piece.
  - Take attacking piece.
  - Interpose piece between attacker and attacked piece.
- Promote piece.
- Defend against promotion threat.
- Attack king.
- Defend king.
- Other attacks.
- Other defences.

Most of Kakinoki's move categories correspond to the PMGs in Section 3. Although not clear from his description, we assume that moves like *Attack material* and *Pin piece* fit in the category *Other attacks*, while moves like *Defend undefended pieces* are part of the category *Other defences*. Absent from Kakinoki's move category description are non-tactical moves for piece development. Search in KAKINOKI SHOGI is a tactical search only and moves for piece development are handled differently. They become plans that can be played if there are no tactical problems detected by the search.

8 Conclusions

Full-width search has been the dominant approach in most game playing programs and has been the subject of much scientific research into two-player complete information games. In this paper we have argued that plausible move generation is an important alternative that deserves further investigation. There are games where the full-width search paradigm can not be successfully applied because of a large average number of search alternatives.

We have proposed a new plausible move generation method for shogi. This plausible move generation method uses five basic move categories that are general for a wide range of games. For each game these basic move categories need to be refined to match the specific features of the game under investigation. For shogi the five basic move categories resulted in 21 different plausible move generators. Each generated move is given a value based on the plausible move generator(s) that generated the move. This *Move Merit Analysis* (MMA) is an indication of the importance of a move. The MMA value of moves can be used for different cut-off schemes. In this paper we compared *PMG-All* (no cut-offs), *PMG-MMA* (cut moves with a negative MMA value) and *PMG-N* (cut moves that are ranked lower than N after MMA; $N = 20, 30, 40, 50$).

Results in shogi show that plausible move generation with move merit analysis gives important savings in the search candidates without compromising accuracy. Savings between 46% and 68% can be achieved, losing only between 1% and 7% of the moves chosen by expert players.

Using plausible move generation with MMA also significantly improves the performance of a shogi program. In tactical shogi problems, a shogi program using plausible move generation without MMA performs only slightly better than an equivalent program using full-width search, but PMG with MMA can solve significantly more problems. More importantly, programs based on *PMG-MMA* and *PMG-N* beat *PMG-All* and a full-width search program by a wide margin under tournament conditions. Our tests also suggest that further cutting the number of moves by using thresholds based on the MMA value can improve the playing strength of a shogi program. In this case rigorous extra cuts are needed, keeping no more than 30 moves. However, the results are not conclusive on this point and further testing is needed to support this.

References

1. T. Anantharaman, M.S. Campbell, and F. Hsu. Singular Extensions: Adding Selectivity to Brute-Force Searching. *Artificial Intelligence*, 43:99–109, 1990.
2. D. Beal. Experiments with the Null Move. In D.Beal, editor, *Advances in Computer Chess 5*, pages 65–79. Elsevier Science Publishers: The Netherlands, 1989.
3. D. Beal. A Generalised Quiescence Search Algorithm. *Artificial Intelligence*, 43:85–98, 1990.
4. A. Bernstein and M. de V. Roberts. Computer v Chess-Player. *Scientific American*, 198:96–105, 1958.
5. M. Buro. The Othello Match of the Year: Takeshi Murakami vs. Logistello. *ICCA Journal*, 20(3):189–193, September 1997.
6. K. Chen. Some Practical Techniques for Global Search in Go. *ICGA Journal*, 23(2):67–74, June 2000.
7. R. Greenblatt, D. Eastlake III, and S. Crocker. The Greenblatt Chess Program. In *Proceedings of the Fall Joint Computer Conference*, pages 801–810, 1967.
8. E. Heinz. Extended Futility Pruning. *ICCA Journal*, 21(2):75–83, June 1998.
9. Japanese Shogi Federation. *Heisei 10 Nenban Shogi Nenkan*. Nihon Shogi Renmei, 1999.
10. A. Junghanns and J. Schaeffer. Domain-Dependent Single-Agent Search Enhancements. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 570–575, 1999.

11. G. Kakinoki. The Search Algorithm of the Shogi Program K3.0. In H. Matsubara, editor, *Computer Shogi Progress*, pages 1–23. Tokyo: Kyoritsu Shuppan Co, 1996. ISBN 4-320-02799-X. (In Japanese).
12. S. Kanazawa. The Kanazawa Shogi Algorithm. In H. Matsubara, editor, *Computer Shogi Progress 3*, pages 15–26. Tokyo: Kyoritsu Shuppan Co, 2000. ISBN 4-320-02956-9. (In Japanese).
13. H. Matsubara and K. Handa. Some Properties of Shogi as a Game. *Proceedings of Artificial Intelligence*, 96(3):21–30, 1994. (In Japanese).
14. H. Matsubara, H. Iida, and R. Grimbergen. Natural developments in game research: From Chess to Shogi to Go. *ICCA Journal*, 19(2):103–112, June 1996.
15. A. Newell, C. Shaw, and H. Simon. Chess Playing Programs and the Problem of Complexity. *IBM Journal of Research and Development*, 2:320–335, 1958.
16. J. Pearl. *Heuristics: Intelligent Search Strategies for Computer Problem Solving.* Addison Wesley Publishing Company: Reading, Massachusetts, 1984. ISBN 0-201-05594-5.
17. B. Pell. A Strategic Metagame Player for General Chess-like Games. *Computational Intelligence*, 12(2):177–198, 1996.
18. J. Schaeffer. The History Heuristic and Alpha-Beta Search Enhancements in Practice. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(11):1203–1212, 1989.
19. J. Schaeffer. *One Jump Ahead: Challenging Human Supremacy in Checkers.* Springer-Verlag New York, Inc., 1997. ISBN 0-387-94930-5.
20. J. Schaeffer and A. Plaat. Kasparov Versus Deep Blue: The Rematch. *ICCA Journal*, 20(2):95–101, June 1997.
21. M. Seo. The C* Algorithm for AND/OR Tree Search and its Application to a Tsume-Shogi Program. Master's thesis, Faculty of Science, University of Tokyo, 1995.
22. M. Seo. On Effective Utilization of Dominance Relations in Tsume-Shogi Solving Algorithms. In *Game Programming Workshop in Japan '99*, pages 129–136, Kanagawa, Japan, 1999. (In Japanese).
23. D. Slate and L. Atkin. Chess 4.5: The Northwestern University Chess Program. In P. Rey, editor, *Chess Skill in Man and Machine*, pages 82–118. Springer Verlag, New York, 1977.
24. Y. Tanase. The IS Shogi Algorithm. In H. Matsubara, editor, *Computer Shogi Progress 3*, pages 1–14. Tokyo: Kyoritsu Shuppan Co, 2000. ISBN 4-320-02956-9. (In Japanese).
25. K. Thompson. 6-Piece Endgames. *ICCA Journal*, 19(4):215–226, December 1996.
26. H. Yamashita. YSS: About its Datastructures and Algorithm. In H. Matsubara, editor, *Computer Shogi Progress 2*, pages 112–142. Tokyo: Kyoritsu Shuppan Co, 1998. ISBN 4-320-02799-X. (In Japanese).