Solving Problems with Uncertainty: A case study using Tsuitate-Tsume-Shogi

Makoto Sakuta and Hiroyuki Iida
Department of Computer Science, Shizuoka University
{sakuta, iida}@cs.inf.shizuoka.ac.jp

Abstract

This paper explores the possibility of computing imperfect-information games with focus on its endgame, namely, mating problems with uncertainty. We have chosen the domain of Tsuitate-Shogi, which is a Kriegspiel-like Shogi variant. Our present goal is to make a program that can solve mating problems of Tsuitate-Shogi (Tsuitate-Tsume-Shogi).

We have proposed an idea so-called metaposition, by which game-tree search for imperfect-information games turns out to be identical to generic AND/OR-tree search of metapositions. In order to solve Tsuitate-Tsume-Shogi, we introduce a search algorithm that we call double-nested iterative-deepening search. The algorithm has two iterations. The outer iteration is that of the search depth and the inner iteration is that of the allowed count of fouls in a problem.

We implemented a solver of Tsuitate-Tsume-Shogi based on the proposed ideas, and experiments using a test set of mating problems were performed. All problems with roughly less than 17 steps in the test set were solved.

1 Introduction

In the domain of computer-game research, most efforts have been devoted to programming a game with perfect information such as Chess. There are little literatures on computer-game research using imperfect-information chess-like games such as Kriegspiel. We know that it is extremely hard to make a strong program of a game with imperfect information, since there are some uncertain factors.

Our long-term challenge is to make a world-champion-level program of chess-like games with imperfect information. The current target is to make a program which can solve the endgame (mating problem) of such games. A mating problem of chess-like games with imperfect information contains much uncertainty that does not appear in computing games with perfect information. We have chosen the domain of Tsuitate-Shogi, which is a Kriegspiel-like Shogi variant. Our present goal in this paper is to make a program which can solve Tsuitate-Shogi mating problems so-called Tsuitate-Tsume-Shogi. In the domain of Kriegspiel, Ciancarini *et al.* reported on their works on some simple endgames of Kriegspiel while showing the performance of their program which can play KPK endgames by the knowledge-based approach[Ciancarini97]. We should note that the re-use rule of Shogi makes Tsuitate-Tsume-Shogi much more complicated than the endgames of Kriegspiel as well as in the case of orthodox games.

2 Tsuitate-Shogi

Tsuitate-Shogi is a Shogi variant which is one of the best known and most popular among all variants as well as Kriegspiel being in Chess[Pritchard94]. We read 'Tsuitate' (in Japanese) a screen, therefore we may call Tsuitate-Shogi (screen Shogi).

2.1 Rules of Tsuitate-Shogi

Let us give a short summary of the rules. Note that the rules of Tsuitate-Shogi seem mostly similar with Kriegspiel. Tsuitate-Shogi requires two players, an umpire, two Shogi boards with a set of pieces, and a screen. Both players are unable to see the other side because of the screen. The principle of the game is that each player moves normally but is not told the opponent's moves which he attempts to discover through judicious play.

Note that a move of Shogi includes not only 'capture' but also 'drop'. After one plays a move on his own board, the umpire approves the move to provide information to both players as required by the rules. Most important information is given by the umpire saying 'check', 'illegal', and 'Black/White has played'. Since the umpire cannot tell other information aloud, he, instead of players, performs capturing on the board involved.

Each player is allowed to make illegal moves within a certain number of times, typically eight times.[1] If one has made illegal moves more than the allowed number of times, he has to lose the game on fouls. A game also ends up when the kings of any two players are in mate.

2.2 Tsuitate-Tsume-Shogi

Tsuitate-Tsume-Shogi (TTS) is a mating problem of Tsuitate-Shogi[Kato95], which is also a variant of Tsume-Shogi, a mating problem of Shogi[Grimbergen98]. Here we give a summary of rules of TTS. Note that the rules of TTS seem almost similar with Tsume-Shogi except a few cases.

We distinguish two players in Tsuitate-Tsume-Shogi, as the attacker and the defender. In TTS, the attacker is unable to see the opponent's pieces and response except the initial position of a mating problem considered. The goal of solving a TTS problem is to mate the opponent's king after the sequence of check and its response as well as in Tsume-Shogi. However, in TTS, the attacker is unable to see the definite information on the defender's responses, where the defender has the perfect information on both sides. This corresponds to the best defense model of a game with imperfect information[Frank98]. In other words, it could be regarded that the defender who is unable to see the opponent always makes the best move at any position regardless of the probability. The attacker must lead to a checkmate whatever moves the defender may play. The attacker is allowed to try the illegal moves in a certain number of times, typically eight times or less.

The attacker should try to guess the position of the defender's pieces as the mating search progresses by trying moves that can be either legal or illegal with respect to the full position. In this sense, to make some (but limited) illegal moves which even may not be check, is useful to gain insight into the position. Namely, the attacker has to play in a context of uncertainty and not full but partial information.

In Tsuitate-Shogi, the illegal move is checked and announced by the referee. A player can even make an explicit nonsense move as a feint in order to confuse the opponent. In TTS, trying the illegal move is also allowed. A question arises: is the explicit nonsense move allowed in TTS?

Here we have chosen the more strict definition. We have defined the foul move as the subset of the illegal move. The *foul move* is the move that seems to be legal on the board only with the own pieces but is illegal on the board with pieces of both sides. In other words, the illegal moves on condition that only the own piece can be seen are not the foul moves.
Thus, the foul moves are:

1. The sliding piece (Rook, Bishop, Lance) jumps over the opponent piece
2. Drop a piece into the square where there is the opponent piece
3. The pawn-drop move that causes the dropped pawn mate

And for problems with double kings:

4. The king remains in check or has come to be in check after the move

There may be a foul move in violation of the rule that forbids the 4 times repetition of the position by the continuation of the check move and its response, but we have not yet implemented.

2.3 Classification

Here we summarize the features of chess-like games and puzzles in the viewpoint of uncertainty (see Table 1).

The ordinary chess-like games are two-person zero-sum games with perfect information, and for solving the endgame (mating problem) of such games AND/OR-tree search is used. A node in AND/OR-tree represents a position. The opposites are Kriegspiel and Tsuitate-Shogi, in which there is much uncertainty since one does not know the opponent's response during mating search process. It is too hard to search the game space of these games for computer. There are many problems of Kriegspiel as the intermediate type between Chess and Kriegspiel. These problems are the fairly restricted cases of Kriegspiel, for which a computer may be able to solve.

The problems of TTS are the intermediate type between Kriegspiel problems and Tsuitate-Shogi. In TTS, the initial position is definite and the perfect solution can be determined. A node of game tree for TTS is a metaposition (described later) and thus a game tree can be identical to a generic AND/OR tree, and then can be solved by computation. In a game tree of TTS, AND nodes and OR nodes do not necessarily appear by turns because of foul moves and the splitting of metapositions, while AND nodes and OR nodes appear by turns in an AND/OR tree for Tsume-Shogi.

[1] So, depending on the number of times for illegal moves, a strategy would be changed. When meeting the situation where there are no more chances to play illegal moves, can you drop a piece or move a sliding piece to some distance?

	how to solve	node	use clue	use probability	initial position	speculative play	computation
Tsume-Shogi, Chess endgame problem	AND/OR game tree	position	No	No	Definite	No	much faster than human
Tsuitate-Tsume-Shogi	more generic AND/OR tree	metaposition	Yes	No	Definite	No	
Kriegspiel problem	AND/OR tree or Minimax tree ?	metaposition of both sides	Yes	Yes or No	Uncertain or Definite	No	
Tsuitate-Shogi, Kriegspiel	-	metaposition of both sides	Yes	Yes	Definite	Yes	too hard

Table 1: Classification of chess-like games and puzzles in view of uncertainty

TTS has uncertainty, so the information sets[Owen95] of TTS often consist of many positions. Consequently, it is necessary to collect any clues of the current position for solving just the same as the problem of Kriegspiel. These are the features of the game with imperfect information. Though TTS can be solved by the search of AND/OR tree and this is the feature of the game with perfect information. Therefore, this is our remark that TTS has the hybrid features of perfect information and imperfect information.

3 Metaposition and its splitting by observations

To deal with problems with uncertainty, we have introduced the concepts of metaposition and metamove.

3.1 Metaposition

Metaposition is a composition of all possible positions, not a set of possible positions. In this meaning, a metaposition is similar to the wave function in quantum mechanics. In TTS, the initial position is definite, but after a few moves, it is diffused into the set of some metapositions. The solver must examine all the metapositions.
The count of positions in a metaposition can be regarded as the index of uncertainty.

3.2 Metamove

At a certain metaposition where the attacker is to move, possible moves are the moves that are either check or foul moves for each position in the metaposition. One of the moves is the metamove to the metaposition.
At a certain metaposition where the defender is to move, possible moves are all the legal moves of all positions in the metaposition. The composition of all possible moves is the metamove.

Metaposition is similar to the information set in game theory[Owen95]. However, we have handled this as a single node in the game tree and there is only one arc between two metapositions by a metamove, while there are probably several arcs between two information sets by several moves. This is an essential difference from the concept of the information set.

3.3 Clues (observations)

The attacker can guess the current position and the opponent moves by some clues (observations):

1. Whether the move of the attacker is a check move or a foul move
2. Whether the position is checkmated
3. Whether and where capturing occurs after the opponent move
4. Whether capturing occurs and what kind of piece is captured after the attack move

And for problems with double kings:

5. Whether the countercheck occurs after the opponent move

By these observations, the metaposition is split into some metapositions with less uncertainty. This is AND-splitting, so all the split metapositions are to be solved.
Finally TTS problem is resolved into the search of the generic AND/OR tree by use of the metaposition and its splitting.

3.4 High-level inference

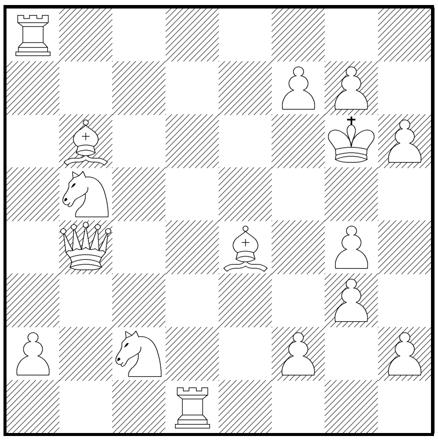


Figure 1: A Kriegspiel problem Mate in 2 (WTM); (G.R.Foster).

We cannot tell exactly whether the high-level inferences that can not be derived from the combination of the above primitive clues exist or not.

In Kriegspiel, there is a funny problem in which one can only see pieces of the attacker's side at the given position[Foster96] (see Figure 1). Surprisingly, exploiting the Chess-specific features, this problem is solved quite easily. From the placements of White's pawns we can see that there have been at least 14 captures and Black has been down to a king and maybe one pawn in file a. In addition, the Black king can only be on five squares: e2, e5, e6, h3, and a6. This should be so-called high-level inference.
However, all the knowledge that can be gotten from the problem position would also be able to be gotten from the combination of the primitive clues in the Kriegspiel game from the starting position, supposing that we could ignore the enormous time and memory space.

In our expectation, there is no high-level inference and no need for that in TTS problems.

3.5 A sample problem and solution of TTS

A sample problem of TTS and the essential part of its solution tree are shown in Figure 2.
The position at the upper left is the given one of the problem. The first move is Silver 2d-1e and there are four moves for its response. If the King moves into the square 1e and captures the Silver, the attacker knows the move because the piece at 1e disappears and so can lead to checkmate easily. If there is no capture, three moves are possible and the metaposition is diffused to one with three positions. The second attacker's move is dropping Rook at 2c. If it is a foul move, the King square is fixed at 2c and the position can be led to checkmate. Otherwise, the metaposition is reduced to one with two positions and there are four moves for its response.
If the King moves into the square 2c and captures the Rook, the attacker knows the move and the position and can lead to checkmate. If the King moves into the square 1e and captures the Silver, the attacker also knows the move and the position and can lead to checkmate. If there is no capture, seven moves are possible and the metaposition has come to have seven positions. In this figure, the interposing moves are represented by dropping a Pawn but actually there are six possible moves.

There is no common check move among the positions in this metaposition, but utilizing a foul move, Rook 3d-1d can be a metamove. If it is a foul move, the interposing move is confirmed and the positions can be led to checkmate. Otherwise, the metaposition has come to have only one position and there are six interposing moves and one capturing move. If there is no capture, the interposing move is confirmed and the positions can be led to checkmate. Otherwise, the metaposition has only one position and can be led to checkmate.

After all, the best sequence has nine plies and no foul whereas the allowed foul should be one for solving. The best sequence of the solution is:

1.銀1e 2.(-) 3.飛*2c 4.(-) 5.飛1d 6.玉x2c 7.飛2d 8.(-) 9.飛2b+ mate.

'(-)' indicates a metamove after which no capturing has occurred.

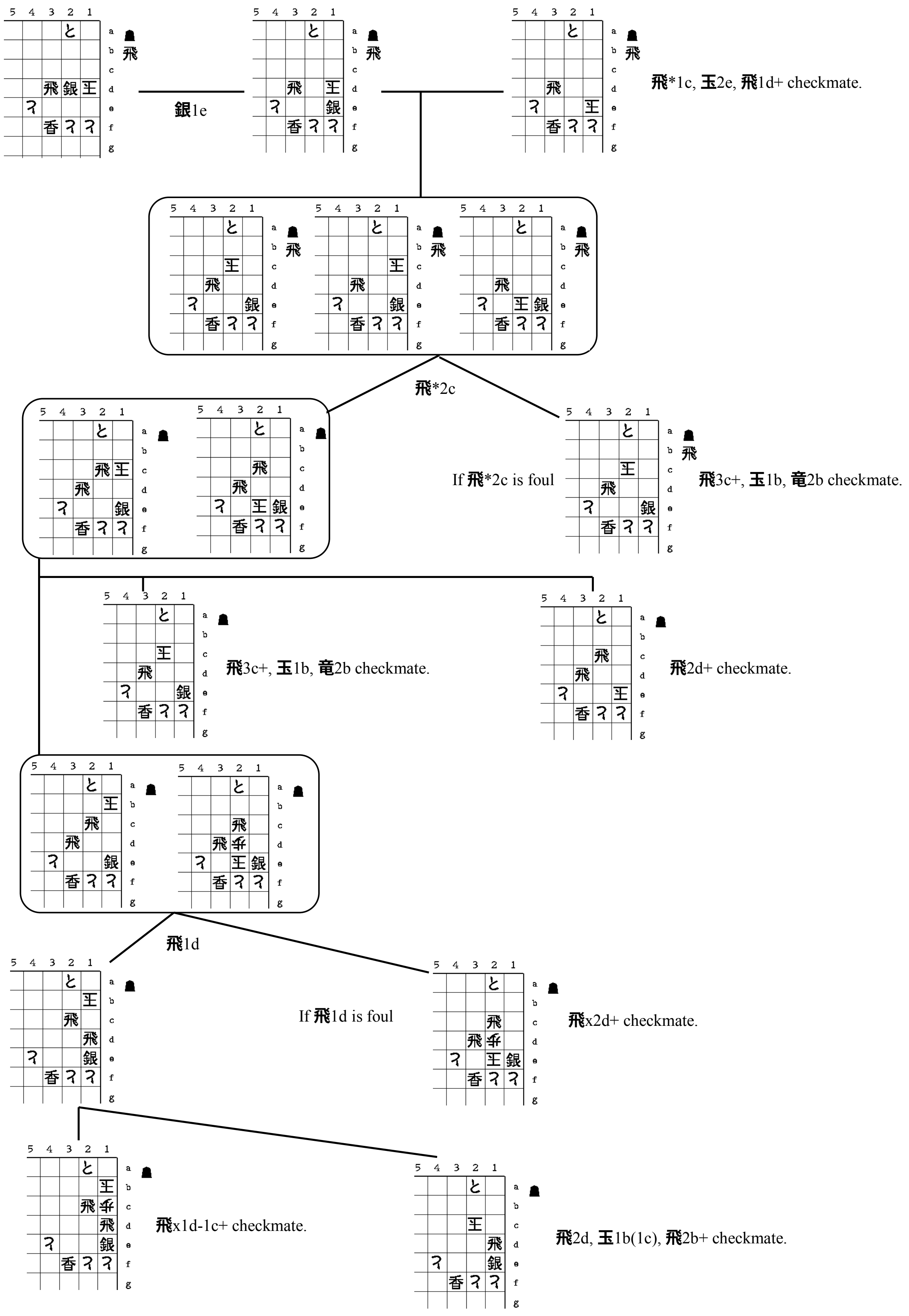
4 Implementation and Experiments of Solver

We have implemented the solver of TTS in computer.

4.1 Reduction of positions

In a metaposition, after a certain metamove, there happen to be some same positions. In this case, the redundant positions are discarded and the position count is reduced.
These reductions usually speed up solving, but in some problems it is faster not to perform the reduction of positions. So there is an option switch in our solver.

Figure 2: Solution of the sample TTS problem (composed by Kankuh Kobayashi and Katsuhiro Kamachi)


銀1e
飛*1c, 玉2e, 飛1d+ checkmate.
飛*2c
If 飛*2c is foul
飛3c+, 玉1b, 竜2b checkmate.
飛3c+, 玉1b, 竜2b checkmate.
飛2d+ checkmate.
飛1d
If 飛1d is foul
飛x2d+ checkmate.
飛x1d-1c+ checkmate.
飛2d, 玉1b(1c), 飛2b+ checkmate.

4.2 Iterative deepening

We have presently been using the quite inefficient iterative deepening search.
Iteration is doubly nested to find the solution with the shortest steps and the least fouls. The outer iteration is that of the search depth and the inner iteration is that of the allowed count of fouls.

4.3 Defining the best sequence of the solution

After deciding the steps and fouls, the search with the multiple iterative deepening at OR nodes is performed to find the best sequence of the solution. The sequence is recognized better when

The number of steps of one sequence is less than that of another sequence.
If it is same, the number of fouls is less than that of another sequence.
If it is same, the set of pieces in hand at the mated position is a subset of that of another sequence.

The metaposition that has the worst sequence is selected in the defender's turn.

4.4 Experimental design

Our solver was coded in C++ (Visual C++ 6.0). Experiments were done under the following environment:

Gateway2000, G6/GP6 Series (TB298-0109)
Pentium II 447MHz, RAM: 384MB
Windows 98.

We had 39 problems from the source[Kato95], but at first we tried to solve the first 15 problems shown in Appendix.

No	ST	F	TM	TP	M	P	MP	AP	BM	BP	TB	TT	validity of problem	validity of solution
1	7	0	5668	14844	608	1244	46	2.05	2.50	2.77	0.19	2.58		
2	13	1	236503	956259	28543	73494	138	2.57	2.20	2.37	13.29	153		
3	9	2	44978	280136	6263	26513	94	4.23	2.64	3.10	5.04	43.86		
4	9	0	6205	9683	1241	1761	9	1.42	2.21	2.29	0.24	1.30		
5	11	0	18534	33841	2901	4578	30	1.58	2.06	2.15	0.66	4.77		need to omit useless dropping pieces
6	11	0	62099	177146	11831	36773	59	3.11	2.35	2.60	5.52	26.48		
7	11	1	59834	140533	6088	10413	46	1.71	2.21	2.32	1.70	20.32		
8	13	3	1.77E6	1.39E7	166303	880752	214	5.30	2.52	2.87	264	2696	additional mate, longer variation	?
9	11	1	217325	409273	60581	107893	59	1.78	2.72	2.87	18.02	58.83		
10	13	1	40910	302404	24389	126538	592	5.19	2.18	2.47	26.40	49.55		
11	15	1	989726	7.41E6	393206	2.27E6	756	5.78	2.36	2.65	614	1358		
12	19	2	1.39E6	7.29E6	292411	1.85E6	1112	6.34	1.94	2.14	541	1286		
13	19	3	2.83E7	1.72E8	1.27E7	6.52E7	751	5.15	2.36	2.58	110877	164533		
14	19	2	2.02E7	8.27E7	1.06E7	3.01E7	1090	2.82	2.34	2.48	19842	39908		
15	11	3	220245	968951	50570	185196	423	3.66	2.68	3.01	32.84	147	earlier mate (intended 27 steps)	

Table 2: Results of solving TTS problems
ST:steps of the solution, F:fouls, TM:total count of the generated metapositions, TP:total count of the generated positions
M:metapositions, P:positions, MP:maximum value of position count of metaposition, AP:average value of position count of metaposition
BM:effective branching factor (based on metaposition), BP:effective branching factor (based on position)
TB:time for defining the best sequence, TT:total time

4.5 Results and Discussions

Results of the experiments are shown in Table 2. The solver could solve all problems and show the proper best sequences except #5, in which it is necessary to omit the useless interposing move.
M (metapositions) and P (positions) are the count of metapositions and positions after defining the solvable search depth and fouls respectively. MP (maximum position count) is the maximum count of positions in the metaposition.
The average count of positions in the metaposition, that is, the average uncertainty index roughly ranged from 1.4 to 6. And the maximum count of positions in the metaposition, that is, the maximum uncertainty index varied widely from 9 to 1112.
The effective branching factors were calculated supposing the worst cases and turned out to be roughly ranged from 1.9 to 2.7. For references the effective branching factors based on the counts of positions are also shown.

They are a little larger than those of the former are. Supposing the best cutoff cases, these values would be twice as large, ranged from 3.9 to 5.4. By allowing the attacker to try foul moves, the number of the possible moves is not so small. Consequently, a quantity of computing is also not so small. There seems to have the tendency that the branching factors are larger for the hard problems with long steps.

In our current implementation, the most of solving time is the part defining the solvable depths and fouls by the doubly nested iterative deepening. By searching more effectively, the total count of generated metapositions TM and positions TP would nearly come down to M and P by a factor of one, and consequently the total solving time TT would be reduced to the values of the time for defining the best sequence TB.

5 Conclusion and Future works

As the first step to challenge computing the games with imperfect information, we have chosen the problem of TTS. We have introduced the metaposition as the composition of the possible positions and the game tree of TTS has turned out to come down to the generic AND/OR tree of metapositions. Searching has been performed by doubly nested iterative deepening. The outer iteration is that of the search depth and the inner iteration is that of the allowed count of fouls.

Our solver could solve all the problems with roughly less than 17 steps. However, we have presently been using the quite inefficient iterative deepening search. There are some modifications to speed up solving.

First, the efficiency could be much higher to use the transposition table. Using the transposition table is the well-known and effective method in searching game tree and the Zobrist method, the fast method of coding positions using the pseudo-random numbers is known for board games. However, for TTS, it is not directly applicable for coding the metaposition. We have been considering the use of the CRC (or checksum) of position codes as the code of a metaposition.

We have succeeded to solve many problems in the source, but there still are several hard problems that have not yet been solved. These are problems with quite long steps, or problems that have fairly large branching factors. Secondly, to solve these in a short time, we think we should examine the other searches that have best-first manners, pn-search and PDS (or PDS*). This must be the coming subject.

Acknowledgements

We would like to thank Tetsu Kato who has pleasantly provided us the problems and explanations of TTS. In addition, we thank Katsuhiro Kamachi, Kankuh Kobayashi and Kato for permitting us carrying their pieces in this article.

References

[Ciancarini97] Ciancarini, P., Libera, F.D., and Maran F. (1997). Decision Making under Uncertainty: A Rational Approach to Kriegspiel. *Advances in Computer Chess 8* (ed. H.J. Herik and J.W.H.M. Uiterwijk), pp.277-298, Drukkerij Van Spijk B.V., Velno, Netherlands. ISBN 9062162347.

[Pritchard94] Pritchard, D.B. (1994). *The Encyclopedia of Chess Variants*.
Games & Puzzles Publications, Godalming, UK. ISBN 0-9524142-0-1.

[Owen95] Owen, G. (1995). *Game Theory*. Academic Press, New York, ISBN 0-12-531151-6.

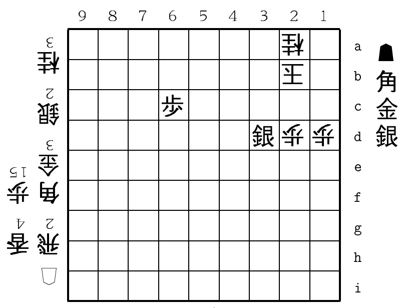
[Foster96] Foster, G. R. (1996). *Variant Chess*, issue 20, Summer 1996.
also available at A Kriegspiel Problem in Hans Bodlaender's Chess Variant Pages,
http://www.chessvariants.com/problems.dir/krieg1.html.

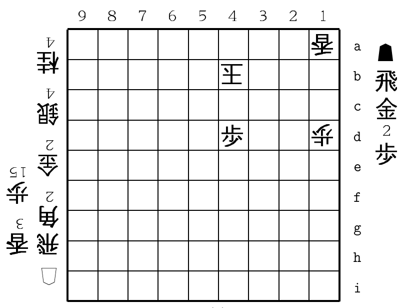
[Kato95] Kato T. ed. (1995). *Collection of Kapitein Documents* No.1, Explanations and Problems of Tsuitate-Tsume-Shogi. (in Japanese).

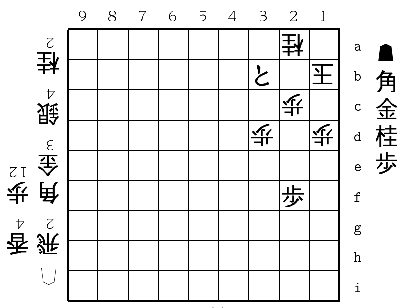
[Grimbergen98] Grimbergen R. (1999). A Survey of Tsume-Shogi Programs Using Variable-Depth Search.
in Jaap van den Herik and Hiroyuki Iida (Eds.) Proc. Internat. Conf. on Computers and Games, CG'98, *Lecture Notes in Computer Science*, vol.1558, Springer, Heidelberg, pp.300-317.

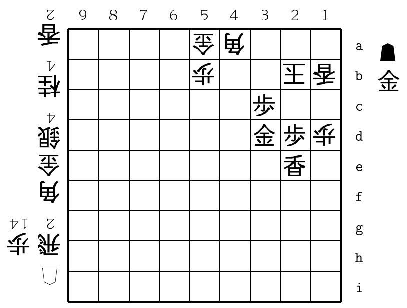
[Frank98] Frank, I., Basin, D. (1998). Search in games with incomplete information: a case study using Bridge card play. *Artificial Intelligence*, Vol.100, pp.87-123.

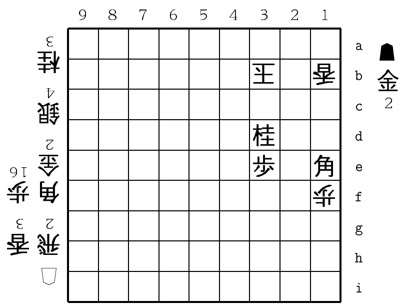
Appendix: Tested problems of TTS

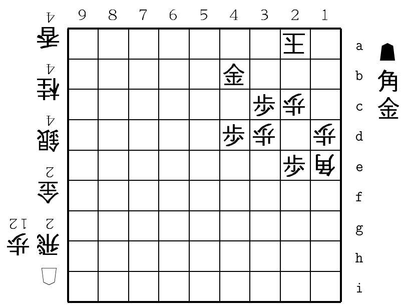

#1 *a

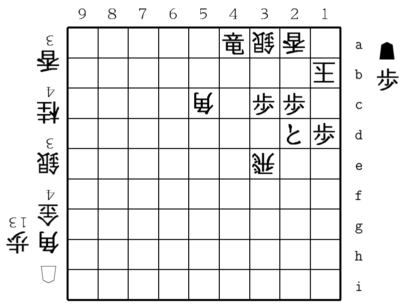

#2 *b

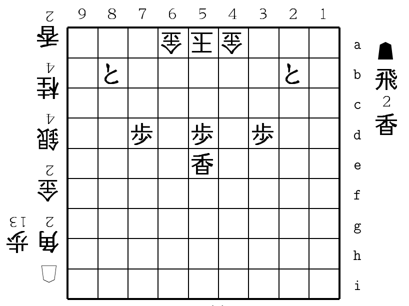

#3 *b

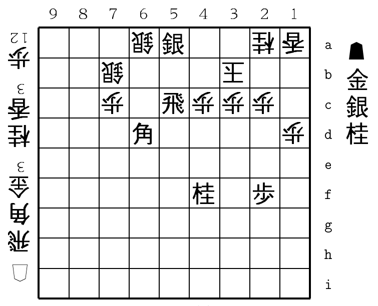

#4 *d

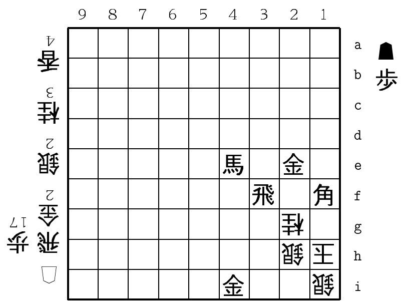

#5 *d

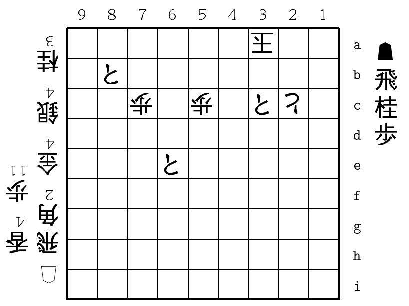

#6 *d

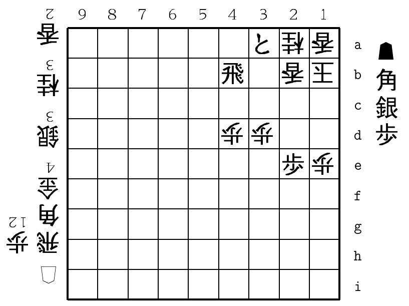

#7 *c

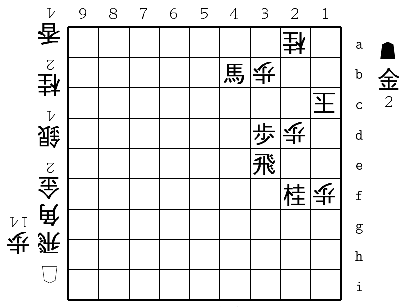

#8 *b

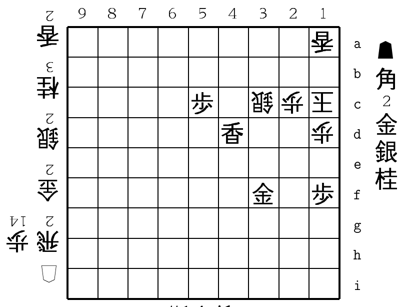

#9 *b

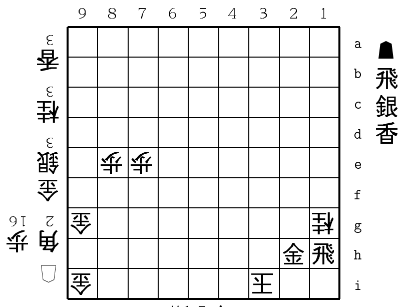

#10 *c


#11 *b


#12 *c


#13 *d


#14 *b


#15 *c

composed by *a:Tetsu Kato, *b:Katsuhiro Kamachi, *c:Kankuh Kobayashi, *d:Kamachi, Kobayashi, Kato