Solving screen-chess-type problems: Examining the ability to distinguish solvable and unsolvable problems

Makoto Sakuta and Hiroyuki Iida
Department of Computer Science
Shizuoka University
3-5-1 Johoku, Hamamatsu
432-8011 Japan
Phone & Fax: +81-53-478-1456
E-mail: {sakuta,iida}@cs.inf.shizuoka.ac.jp

Abstract

We recently proposed a deterministic approach for solving problems with uncertainty, called the Uncertainty Paradigm. Under this paradigm, deterministic solving of such a problem is resolved into a plain AND/OR-tree search. The search under this paradigm is denoted by the Uncertainty Paradigm Search (UPS). As an application, we have chosen the domain of Tsuitate-Tsume-Shogi (TTS), which is the mating problem of screen-chess-like variant of Shogi. This paper examines the ability of the TTS solver to distinguish whether a problem is to be proved or disproved. 498 normal Tsume-Shogi problems are examined by our TTS solver using proof/disproof numbers. Since the solver has the high disproving ability as well as the proving ability, it shows the high ability to distinguish solvable and unsolvable problems. When no fouls are permitted, only 128 problems (26%) are solvable. However, when the number of permitted fouls is 3, 4 or 8, 179 problems (36%) are solvable. The number of solvable problems does not increase even if the number of permitted fouls is 16.

1 Introduction

In the domain of computer-game research, most efforts have been devoted to the programming of games with complete information such as chess. There is less literature on computer-game research using imperfect-information games.

We know that developing a program of an incomplete-information game is usually much more difficult than developing the program of a perfect-information game.

However, there are some kinds of problems with uncertainty or some kinds of endgames of the imperfect-information games that can be solvable deterministically. Nevertheless, there are a few studies on the deterministic solving of problems with uncertainty.

We recently proposed a deterministic approach for solving problems with uncertainty, called the Uncertainty Paradigm [6]. Under this paradigm, deterministic solving of such a problem is resolved into a plain AND/OR-tree search. The search under this paradigm is denoted by the Uncertainty Paradigm Search (UPS). As an application, we have chosen the domain of Tsuitate-Tsume-Shogi (TTS), which is the mating problem of Kriegspiel-like variant of Shogi. We have exploited a transposition table in a search [5] and developed some search algorithms using proof/disproof numbers. Consequently, one of our developed programs has solved all hard problems of TTS [4].

In TTS, a given initial position is definite the same as in normal Tsume-Shogi[1]. Therefore, a set of TTS problems is a subset of normal Tsume-Shogi problems. Since our TTS solver that uses proof/disproof numbers has the high disproving ability as well as the proving ability, we can distinguish whether a Tsume-Shogi problem can be solvable (to be proved) or unsolvable (to be disproved) by the TTS solver.

In this contribution, we have tried to solve many normal Tsume-Shogi problems using our TTS solver, and confirmed the high distinguishing ability.

2 Tsuitate-Tsume-Shogi

Below we give a summary of the rules of TTS, which can be considered as a kind of puzzle derived from the game of Shogi. The rules of Shogi are available on the Internet, for instance, at http://www.jwindow.net/LWT/SHOGI/INTRO/

[1]This is not an essential feature of TTS. If we modify our TTS program a little, we can make the program solve a problem in which an uncertain initial position is given.

shogi_intro.html.

There are two agents in TTS: the *attacker* as the *solver*, and the *defender* as the *opponent*. Because the attacker's King often has no meaning in TTS, most TTS problems only have a defender's King but do not have the attacker's King. Nevertheless, there are some problems that have both Kings on the board; they are called *problems with double Kings*. In TTS, the attacker is unable to see the opponent's pieces and responses, except for the initial position of a mating problem (cf. in Kriegspiel, the initial chess position when play starts). The goal of solving a TTS problem is to mate the opponent's King after a sequence of checks and their responses. However, the attacker is not allowed to see the information that follows from the defender's responses, while the defender has perfect information on both sides. The attacker must find a checkmate whatever moves the defender would play. The attacker is allowed to try so-called *foul moves* up to a certain number of times, typically eight times or less. A foul move is defined as a move that belongs to a subset of possible illegal moves. It is a move that is legal if only the attacker's pieces are on the board, but is illegal when all pieces are considered. Foul moves are:

1. a sliding piece (Rook, Bishop, Lance) jumping over an opponent piece;
2. dropping a piece onto a square that is occupied by an opponent piece;
3. a pawn-drop move that causes the dropped Pawn mating the opponent King;
4. a move after which the King of the attacker remains in check or brings itself in check (for problems with double Kings).

If the attacker has tried a foul move, the count of fouls increases by one; the attacker only knows the count and that it is a foul move. He does not know the type of the foul move. The position does not change at all and the attacker has to choose another move that is possibly a check move. The attacker is not allowed to make a foul move if the number of fouls has reached the predetermined limit. Notice that the attacker may not cancel a move that has been tried; moreover the move has to be a check move if it has turned out to be a legal move. If the attacker made a legal move that does not check the opponent King, the solving is unsuccessful.

3 Experimental Design

The used test set that contains 498 Tsume-Shogi problems is from the sources [7], [8], [9] and [10]. The distribution of steps of problems in the test set is shown in Table 1. Every problem has a solution as a Tsume-Shogi problem. A large part of problems range from 5 to 13 steps.

As for the TTS solver, we used a program with the UPDS algorithm. UPDS is a kind of the PDS* algorithm [2, 3] specialized for the Uncertainty Paradigm [4]. UPDS sets the initial value of proof-number/disproof-number of each unexpanded node to the value according to the formulae listed below, rather than 1 in the normal PDS.

$$\mathrm{proof}_{\mathrm{init}} = \begin{cases} \lceil (N/D)^{\gamma_p} \rceil & \\ \mathrm{IPD}_{\mathrm{max}} & \text{if } \lceil (N/D)^{\gamma_p} \rceil > \mathrm{IPD}_{\mathrm{max}} \end{cases}$$

$$\mathrm{disproof}_{\mathrm{init}} = \begin{cases} \lceil (N/D)^{\gamma_d} \rceil & \\ \mathrm{IPD}_{\mathrm{max}} & \text{if } \lceil (N/D)^{\gamma_d} \rceil > \mathrm{IPD}_{\mathrm{max}} \end{cases}$$

Here N is an uncertainty index of a node (that means, a node has N possible positions). γ_p, γ_d, D, and $\mathrm{IPD}_{\mathrm{max}}$ are the parameters of the constant values. γ_p or γ_d indicates a degree trying to suppress the combinatorial explosion of uncertainty. As γ_p or γ_d is set to a larger value, the search is more likely to proceed in the way to select a node with less uncertainty. D is the damping factor of positive values. As it is set to a larger value, the effect of γ_p or γ_d is suppressed. $\mathrm{IPD}_{\mathrm{max}}$ is the upper bound of initial values. In the experiments, we set γ_p, γ_d, D, and $\mathrm{IPD}_{\mathrm{max}}$ to 0.8, 0.3, 4, and 50, respectively. We assigned half an hour to each stage of solving. We recognized the problems that were not solved in half an hour as unsolved.

We tried to solve all problems in the test set changing the number of permitted fouls. The number of permitted fouls examined was set to 0, 1, 2, 3, 4, 8, and 16.

4 Results and Discussions

The summary of results is shown in Table 2 and the percentages of proved, disproved or unsolved problems against the permitted fouls are plotted in Figure 1.

When no fouls were permitted, 128 problems (26%) were proved and 368 problems were disproved. Only 2 problems were unsolved. When the number of permitted fouls was set to a larger value, the number of solvable problems increased up to about 36%. This is because the solutions of many problems have a few fouls. However, at the same time, the number of unsolved problems also increased. This is because the search space becomes larger as the number of permitted fouls gets larger. The number of solvable problems did not increase even if the number of permitted fouls was 8 or 16. The maximum number of solvable problems was 179 (36%) when the number of permitted fouls was 3, 4 or 8. In the default number (8) of permitted fouls, 92% problems were distinguished as solvable or unsolvable. The solver, which

Table 1: Distribution of steps of problems in the test set

steps	5	7	9	11	13	15	17	19	21	23	25	27	29	≥31	total
tsukada	15	25	44	51	34	14	9	2	1	1	1	0	1	4	202
sato dai	40	38	0	0	0	0	0	0	0	0	0	0	0	0	78
sekine	12	99	82	25	0	0	0	0	0	0	0	0	0	0	218
all	67	162	126	76	34	14	9	2	1	1	1	0	1	4	498

Table 2: Summary of results

steps	5		7		9		11		13		15		≥17		total	
permitted fouls	**0**															
proved	27	(40)	59	(36)	27	(21)	13	(17)	1	(3)	1	(7)	0	(0)	128	(26)
disproved	40	(60)	102	(63)	98	(78)	63	(83)	33	(97)	13	(93)	19	(100)	368	(74)
unsolved	0	(0)	1	(1)	1	(1)	0	(0)	0	(0)	0	(0)	0	(0)	2	(0)
permitted fouls	**1**															
proved	35	(52)	73	(45)	41	(33)	22	(29)	6	(18)	1	(7)	0	(0)	178	(36)
disproved	32	(48)	85	(52)	83	(66)	54	(71)	28	(82)	13	(93)	19	(100)	314	(63)
unsolved	0	(0)	4	(2)	2	(2)	0	(0)	0	(0)	0	(0)	0	(0)	6	(1)
permitted fouls	**2**															
proved	35	(52)	74	(46)	40	(32)	22	(29)	6	(18)	0	(0)	0	(0)	177	(36)
disproved	30	(45)	80	(49)	80	(63)	54	(71)	27	(79)	13	(93)	19	(100)	303	(61)
unsolved	2	(3)	8	(5)	6	(5)	0	(0)	1	(3)	1	(7)	0	(0)	18	(4)
permitted fouls	**3**															
proved	35	(52)	74	(46)	41	(33)	23	(30)	6	(18)	0	(0)	0	(0)	179	(36)
disproved	27	(40)	79	(49)	76	(60)	51	(67)	26	(76)	12	(86)	19	(100)	290	(58)
unsolved	5	(7)	9	(6)	9	(7)	2	(3)	2	(6)	2	(14)	0	(0)	29	(6)
permitted fouls	**4**															
proved	35	(52)	74	(46)	41	(33)	23	(30)	6	(18)	0	(0)	0	(0)	179	(36)
disproved	27	(40)	77	(48)	75	(60)	51	(67)	26	(76)	12	(86)	19	(100)	287	(58)
unsolved	5	(7)	11	(7)	10	(8)	2	(3)	2	(6)	2	(14)	0	(0)	32	(6)
permitted fouls	**8(default)**															
proved	35	(52)	74	(46)	41	(33)	23	(30)	6	(18)	0	(0)	0	(0)	179	(36)
disproved	27	(40)	76	(47)	73	(58)	50	(66)	24	(71)	12	(86)	19	(100)	281	(56)
unsolved	5	(7)	12	(7)	12	(10)	3	(4)	4	(12)	2	(14)	0	(0)	38	(8)
permitted fouls	**16**															
proved	35	(52)	73	(45)	41	(33)	23	(30)	6	(18)	0	(0)	0	(0)	178	(36)
disproved	27	(40)	76	(47)	72	(57)	50	(66)	24	(71)	12	(86)	18	(95)	279	(56)
unsolved	5	(7)	13	(8)	13	(10)	3	(4)	4	(12)	2	(14)	1	(5)	41	(8)

The numbers in parentheses indicate the percentages.

uses proof/disproof numbers, showed the high ability to distinguish whether each problem is to be proved or disproved. It should be noted that the standard depth-first searches do not have such a distinguishing ability. They generally solve the problems to be proved but hardly disprove the problems. The ability to distinguish solvable and unsolvable problems is an excellent feature of the searches that uses both proof and disproof numbers.

As for the relation of the ratios of solvable problems to the problem steps, nearly a half of the problems with short steps (5, 7) were proved. In our experiments, no solvable problems with more than 15 steps were found. On the other hand, there are several hard problems with more than 30 steps made by some famous Tsume-Shogi composer [1]. The longest-step problem among them is a problem with 43 steps and 4 fouls. This fact shows that we need to specialize for TTS in order to make a long-step TTS problem.

We have examined whether there is a certain difference depending on the composer of problems. The results classified by composer are shown in Table 3, only in a case where the number of permitted fouls is 3. These results suggest that the problems composed by Sekine are more likely to be proved than ones composed by Tsukada or Sato. However, we cannot state the further conclusions because the size of test set is not so large for a statistical analysis.

References

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

[2] A. Nagai. A new AND/OR tree search algorithm using proof number and disproof number. In

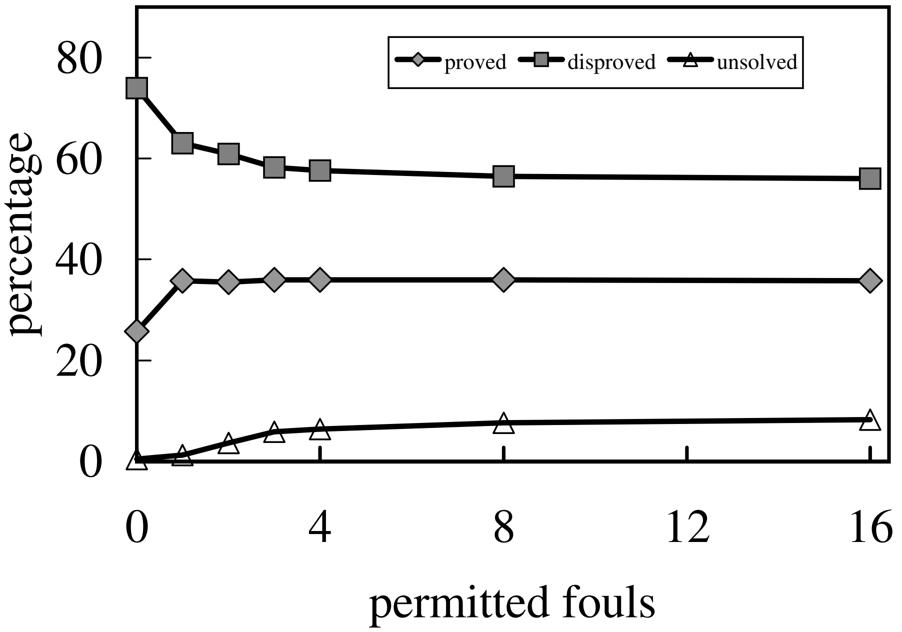

Figure 1: Percentages of proved, disproved or unsolved problems

Table 3: Results classified by composer (number of permitted fouls:3)

composer	steps	5		7		9		11		13		15		≥17		total	
tsukada	proved	4	(27)	8	(32)	8	(18)	11	(22)	6	(18)	0	(0)	0	(0)	37	(18)
	disproved	10	(67)	16	(64)	32	(73)	38	(75)	26	(76)	12	(86)	19	(100)	153	(76)
	unsolved	1	(7)	1	(4)	4	(9)	2	(4)	2	(6)	2	(14)	0	(0)	12	(6)
sato dai	proved	24	(60)	15	(39)	0	(-)	0	(-)	0	(-)	0	(-)	0	(-)	39	(50)
	disproved	13	(33)	15	(39)	0	(-)	0	(-)	0	(-)	0	(-)	0	(-)	28	(36)
	unsolved	3	(8)	8	(21)	0	(-)	0	(-)	0	(-)	0	(-)	0	(-)	11	(14)
sekine	proved	7	(58)	51	(52)	33	(40)	12	(48)	0	(-)	0	(-)	0	(-)	103	(47)
	disproved	4	(33)	48	(48)	44	(54)	13	(52)	0	(-)	0	(-)	0	(-)	109	(50)
	unsolved	1	(8)	0	(0)	5	(6)	0	(0)	0	(-)	0	(-)	0	(-)	6	(3)

The numbers in parentheses indicate the percentages.

Proceedings of Complex Games Lab Workshop, pages 40–45, ETL, Tsukuba, Nov. 1998.

[3] A. Nagai. A new depth-first-search algorithm for AND/OR trees. M.Sc. thesis, Department of Information Science, University of Tokyo, Japan, 1999.

[4] M. Sakuta and H. Iida. A deterministic approach for solving Kriegspiel-like problems: Examining efficient search methods of high solving ability. In *Working Notes of Second International Conference on Computers and Games*, pages 349–368, 2000.

[5] M. Sakuta and H. Iida. Solving Kriegspiel-like problems: Exploiting a transposition table. *ICGA Journal*, 23(4), 2000. to appear.

[6] M. Sakuta, H. Iida, and J. Yoshimura. Solving problems under Uncertainty Paradigm. In *Proceedings of International Conference on Advances in Infrastructure for Electronic Business, Science and Education on the Internet (SSGRR 2000)*, L'Aquila, Italy, Aug. 2000. (in CD-ROM).

[7] D. Sato. Daigoro's refreshing 5 and 7-step problems: PART 5. *Shogi Sekai*, 63(8), 1999. Supplement No. 1 (in Japanese).

[8] D. Sato. Daigoro's refreshing 5 and 7-step problems: PART 6. *Shogi Sekai*, 63(11), 1999. Supplement No. 1 (in Japanese).

[9] S. Sekine. *Clear and refreshing Tsume-Shogi.* Japan Shogi Association, 1999. (in Japanese).

[10] M. Tsukada. *Representative pieces of Tsukada Tsume-Shogi.* Japan Shogi Association, 1999. (in Japanese).