What Shogi Programs Still Cannot Do
- A New Test Set for Shogi -

Reijer Grimbergen and Taro Muraoka
Department of Informatics, Yamagata University
Jonan 4-3-16, Yonezawa-shi, 992-8510 Japan
E-mail: grim@yz.yamagata-u.ac.jp

Abstract

Sets of test positions are an important tool for finding problem areas in game programs. Compared to test sets used in chess, we feel that current test sets for shogi have some important shortcomings. In this paper we propose a new test set, which is more general than the ones proposed earlier and also points to certain problem areas in computer shogi that need to be addressed. The test set consists of 100 positions from Shukan Shogi that could not be solved by TODAI SHOGI 5, GEKISASHI 2 and AI SHOGI 2003. By using the analysis tools in these programs we discovered the following problem areas: an horizon effect due to consecutive checks, not calling the tsume shogi solver deep in the search tree, inaccurate evaluation function, problems with mate using unpromoted pieces, incorrect forward pruning, insufficient hardware speed and problems with the time allocation.

Keywords: Testing game programs, test set, shogi

1 Introduction

Testing is a vital software engineering tool. To properly test software, it is important that the software requirement specifications are clearly defined. However, in general such requirement specifications do not exist or they cannot be easily translated into test requirements. Here again, games provide a simplification of a complex problem. The requirements of game software can be easily defined: the program should play strongly. Game programmers usually take the reverse approach: the program should not play weakly, i.e. the number of bad moves should be minimized.

To meet this requirement, a test set should provide information about specific problem areas in the program, i.e. the types of positions that the program does not play well. To properly assess this, the test set ideally should cover all potential problem areas. To devise a test set that meets this requirement is not an easy task. Programs will not play flawlessly, so problems will appear in each game and an assessment must be made about which problems have priority. Most programmers save problematic positions from the games that were played. These positions become program-specific test sets representative for a class of positions that the program could not handle well. Keeping such positions as a future point of reference is important, because fixing one thing can break something else. Making sure that something that worked before is still working is another important feature of a test set.

Creating an incremental test set in this way is an important tool, but there are some drawbacks. The main drawback is that the test set is specific to a single program. The problem areas of one program are not necessarily the same as those of a different program, so these test positions cannot easily be used to compare different programs. Another drawback is that the positions are selected subjectively by the programmer(s). Even though they are the experts concerning the behavior of their program, it is possible that they wrongly assess the problem in a position or become too preoccupied with certain problems, neglecting others. It is also possible that important problem areas are overlooked. Especially in a game community with only a small number of strong competitors, playing the same opponent many times will only reveal a subset of the problems.

Therefore, it is important to have test sets that test a wide variety of potential problem areas and that have been designed without a specific program in mind. In chess there have been a number of proposed test sets, each with their own specific properties. Compared to this, only little work has been done for test design in other games. Especially in shogi, where the strongest programs now have reached a level where it is becoming hard for pro-

grammers to understand the problems of their programs, a good set of test positions could help to focus the research efforts on areas that require improvement.

Some test sets have been proposed for shogi, but in this paper we will argue that these test sets have some important shortcomings. To compare test sets in chess with test sets in shogi, we will start with a short overview of different test sets in chess and their features in Section 2. In Section 3 we will explain that test sets in shogi are either not general enough or cannot be expected to accurately point out problem areas in computer shogi programs. Our approach is to construct a test set that is difficult for computer programs. To assess this, we have assembled 100 positions that none of the current top shogi programs can solve. These positions are presented in Section 4. Furthermore, we are not only interested in the positions that cannot be solved, but we also want to know why these positions are difficult. We have used the analysis tools of strong computer shogi software to make assessments about the problem areas. These problem areas are explained in Section 5. During our tests, a new version of one of the programs we used became available and partial test results for this program are given in Section 6. Although it is not the main purpose of our test set, the positions can also be used for comparing human performance with computer performance. How this can be done is explained in Section 7. We end this paper with conclusions and suggestions for future work in Section 8.

2 Test Sets for Chess

The Bratko-Kopec test set is a famous example of a test set for chess [3]. Even though this test set of 24 positions was originally intended to compare human and computer performance in chess, it has been widely used to compare the strength of chess programs. The test is a mix of 12 tactical positions and 12 strategical positions. The tactical positions are usually rather easy for chess programs, but the strategic positions are still hard and as far as the authors know, no chess program has been able to correctly solve all 24 positions. Another test set that has often been used in computer chess are the 300 positions from Reinfeld's book *Win at Chess* [11]. These positions were clearly not designed with computer chess in mind and are mostly of a tactical nature. Almost all positions are rather easy to solve for strong programs. For example, the program LAMBCHOP, a strong but not a top program, reportedly solved 286 out of 300 positions [4]. Therefore, the Reinfeld positions are mostly used as a first test for new programs.

A more challenging test set is the LCT II test, designed by Frederic Louguet (author of the program CHESS WIZARD). This test set has 35 positions with a good balance between strategic, tactical and endgame positions. The program SHREDDER (the runner-up in the latest ICGA World Computer Chess Championship) is reported to have solved 94% of the LCT II positions [5]. An interesting feature of the LCT II test set is that an ELO rating can be calculated from the solved positions. Based on this calculation, the rating of SHREDDER is estimated at 2785 (maximum 2950). Using the answers to test positions for estimating ELO rating was already proposed in 1984 by Grottling [1].

It is not only important that a test set is general, but also that it is likely to provide information about specific problem areas of the programs. In chess, Lindner [6] proposed a set of positions that were assumed to be particularly hard for computers to solve. An important observation by Lindner was that finding the key move in a position does not automatically mean that the position is solved. We will later see that this is also a problem of test sets in shogi.

Other proposed test sets for chess can be found in [2, 10].

3 Test Sets for Shogi

For shogi, the importance of a set of test problems was recognized by Matsubara and Iida [7, 9, 8]. Also, on the web sites of Yamashita [13] (programmer of the CSA World Champion YSS) and Tanase [12] (main programmer of multiple CSA World Champion IS SHOGI), a small number of test positions is available to compare different programs.

The test set by Matsubara and Iida has been described in detail and is therefore the easiest to assess. The test set consists of 48 positions taken from the game scores of professional players. As far as we know the game scores from which the positions were taken have not been made public, but at most three positions were taken from a single game score, so more than 16 different games were used to compile the test positions.

The positions were selected by an expert player (Iida, a 6-dan professional shogi player). Each position represented some problem to be solved, the answer to this problem was unique and the position was unfamiliar. The set of positions was first given to human players to establish a connection between the number of solved positions and playing strength.

The aim of the Matsubara-Iida test set was to

judge the strength of computer programs. We think that making a good estimate of playing strength by having a small test of 48 positions is difficult. Playing strength can be established much more accurately by playing a large number of games on the Internet against different human opponents. Furthermore, if estimating playing strength is the aim of the test set, an ELO calculation based on this test set should be presented similar to the LCT II test in chess. Another problem of the test set is that the positions were selected by a human expert. Despite Iida's obvious knowledge about the inner workings of shogi programs, there is a difference between positions that are difficult to solve for human players and computer programs, so we doubt that such a selection method can lead to a well-balanced set of test positions. Also, because of the difference between what is difficult for human players and for computer programs, the connection between playing strength and the solved positions is unreliable.

The test sets by Yamashita and Tanase have the drawbacks that they are very small. Yamashita's test set has only 10 positions and Tanase's test set has 19. Our impression is that these were positions that were considered important problems for YSS and IS SHOGI. It is not even clear if the problems are correct (i.e. they have a single solution). The positions may test vital problem areas, but because of the correctness and generality issues, the use of these two test sets is limited.

4 A New Test Set for Shogi

As pointed out, we want to create a set of test positions that is both general and points to as many problem areas in computer shogi as possible. To achieve this, we decided to look for positions that could not be solved by any of the best shogi programs currently on the market. Therefore, our approach resembles the approach that Lindner took in chess and is different from the test set constructed by Matsubara and Iida. Instead of measuring strength, we want to find weaknesses.

This selection method has the extra advantage of objectivity. No assessment of the difficulty of a position is necessary. The general assumption is that if a position cannot be solved by any of the top programs, it is hard for computer programs in general.

To make it more likely that different problem areas are discovered, we selected the positions from the set of next-move problems from the weekly shogi magazine *Shukan Shogi*. In every issue of Shukan Shogi there are six next-move problems, divided in the categories *First Step*, *Upper Kyu*, *1-dan*, *2-dan*, *3-dan* and *4,5-dan*. The positions are middle game positions and endgame positions with different tactical themes like winning material, building a strong attack, defending against the attack of the opponent and mating. Because of the different themes and wide range of positions, we expected that positions taken from Shukan Shogi were likely to point to different problem areas in computer shogi. We wanted to make a test set with at least a 100 positions, to minimize the risk of creating a test set that focuses on only one or two problem areas.

We used AI SHOGI, TODAI SHOGI and GEKISASHI to solve the positions, as we believe that these three programs are currently the strongest. In both the 2003 and 2004 CSA Computer Shogi Championships these three programs finished first, second and third. At the time of testing, the following versions of the programs were available: AI SHOGI 2003, TODAI SHOGI 5 and GEKISASHI 2. These were the versions used in our analysis. Each program was given 30 seconds on a 2GHz Pentium 4 to solve a position. This is a rather arbitrary time limit. For most test positions in chess the programs are given 10 minutes and there is no time limit for human solvers of the Shukan Shogi problems. Still, this time limit is close to the average time per move a shogi program has in the CSA Computer Shogi World Championships. Strong shogi programs are expected to produce good moves in this amount of time. We will return to this issue later.

We should emphasize that the combined strength of the three programs was excellent. Finding 100 positions that none of the three programs could solve was much harder than we expected. More than 1500 positions needed to be examined to assemble a test set of 100 positions. The positions are given in Table 1. Each position is coded with the week number that Shukan Shogi assigns to the six problems (we started collecting in week 750 and ended with the next-move problems of week 1005) followed by a number giving the grade of the problem (*First Step* = 1, *Upper Kyu* = 2, *1-dan* = 3, *2-dan* = 4, *3-dan* = 5 and *4,5-dan* = 6).

An additional feature of the positions in Shukan Shogi is that for each position the percentage of respondents that solved the position is given. This is an indication of the difficulty of the position for human players and can give insights into the differences between positions that are difficult for human players and positions that are difficult for computers. These percentages are also given in Table 1.

The positions of our proposed test set can be downloaded here: http://gamelab.yz.yamagata-u.ac.jp/RESEARCH/shogitestset.zip. There are 2 copies of each position in this archive. 100 files have the name of the position in Shukan Shogi, while the

No	*Code*	*Sol*	*No*	*Code*	*Sol*	*No*	*Code*	*Sol*	*No*	*Code*	*Sol*	*No*	*Code*	*Sol*
1	750-3	16%	21	804-6	16%	41	852-5	26%	61	910-5	57%	81	964-6	12%
2	754-6	42%	22	808-3	26%	42	854-6	78%	62	920-6	26%	82	966-6	37%
3	755-3	51%	23	808-5	16%	43	856-5	28%	63	921-2	36%	83	968-4	16%
4	755-6	65%	24	809-3	75%	44	856-6	26%	64	922-6	41%	84	968-5	68%
5	762-6	47%	25	812-6	52%	45	864-6	64%	65	924-6	37%	85	968-6	15%
6	765-4	21%	26	818-6	30%	46	865-6	51%	66	925-5	52%	86	973-5	25%
7	765-6	44%	27	821-5	21%	47	871-4	82%	67	925-6	20%	87	977-3	76%
8	769-5	36%	28	821-6	76%	48	873-6	26%	68	934-6	73%	88	979-5	36%
9	772-4	83%	29	823-5	93%	49	877-5	51%	69	935-2	95%	89	988-2	58%
10	773-6	27%	30	823-6	46%	50	879-4	26%	70	942-6	89%	90	988-5	25%
11	779-6	46%	31	825-6	85%	51	887-6	60%	71	948-3	93%	91	990-6	57%
12	783-5	??	32	826-6	65%	52	891-5	35%	72	949-6	18%	92	991-6	16%
13	785-1	60%	33	828-6	86%	53	895-6	59%	73	951-6	15%	93	992-5	75%
14	785-4	44%	34	831-6	58%	54	898-6	28%	74	952-5	33%	94	993-6	11%
15	785-5	36%	35	833-5	36%	55	899-3	78%	75	952-6	47%	95	994-6	82%
16	786-2	22%	36	836-6	69%	56	900-4	82%	76	955-6	53%	96	995-4	73%
17	786-6	16%	37	838-6	78%	57	905-5	46%	77	957-5	47%	97	1003-6	43%
18	799-5	40%	38	839-6	91%	58	906-3	51%	78	963-4	43%	98	1005-2	66%
19	800-6	34%	39	842-3	90%	59	908-5	93%	79	963-6	42%	99	1005-5	30%
20	802-5	56%	40	846-5	62%	60	908-6	58%	80	964-5	82%	100	1005-6	30%

Table 1: The proposed set of test positions with the percentage of respondents that solved the position correctly. The percentage of 783-5 was unavailable.

other 100 are named *test1* to *test100* to make it easier to use the test for automatic testing in other computer programs.

5 Problem Area Analysis

By analyzing the test positions with the analysis tools available in TODAI SHOGI, GEKISASHI and AI SHOGI, we found seven problem areas that may have been the reason why these positions were hard to solve: An horizon effect due to consecutive checks, not calling the tsume shogi solver deep in the search tree, inaccurate evaluation function, incorrect forward pruning, problems with mate using unpromoted pieces, insufficient hardware speed and problems with the time allocation.

We will now discuss these problem areas with some examples.

Figure 1: Horizon effect example

5.1 Horizon effect and Tsume Shogi

The position in Figure 1 is number 750-3 and was solved by 16% of the human respondents. The solution to this position is 1.S*2d. After 2.Px2d 3.G*2c 4.Kx2c 5.B3b+ leads to mate and 2.K1d 3.G*3e leaves white without defense. There is no mate after 4.S*7i 5.K9g 6.N8e 7.K8f 8.+B5i 9.N7g.

In this position TODAI SHOGI selects 1.Px1e, GEKISASHI plays 1.B3b+ and AI SHOGI plays 1.G*3e. TODAI SHOGI's analysis shows that it thinks black is losing and GEKISASHI gives white a big advantage. In AI SHOGI it is only possible to use a hint mode, without the option of looking at the variations or the evaluation function value. GEKISASHI comes very close to solving this position. When given more time, after 40 seconds GEKISASHI changes its mind to S*2d, but still gives the position a big negative score, indicating that it is not able to find the winning variation. This is similar to the observation by Lindner in chess that finding the key move does not necessarily mean that the

position is solved.

We believe that the reason why the programs can not solve this position lies in the number of checks that white still has in the position after 1.S*2d 2.K1d 3.G*3e. These checks push the black win over the search horizon. To investigate this, we constructed a position in which white has no horizon checks (Figure 2). This position can be solved by all three programs.

Figure 2: No horizon checks

The position of Figure 1 not only indicates a problem with horizon checks. When GEKISASHI was given more time, it returns the following variation: 1.S*2d 2.K1d 3.G*3e 4.S*7i 5.Kx7i 6.Nx2e 7.Px1e 8.+Bx1e 9.Sx1e (Evaluation: −1192). In this variation, white has a 9-move mate after 5.Kx7i. Moreover, after 6.Nx2e black has a 3-move mate starting with Sx2c=. It seems clear that the program has problems to find mates that are deeper in the search tree.

5.2 Evaluation function and Forward Pruning

The position in Figure 3 is number 755-3 and was solved by 51% of the human respondants. The solution to this position is 1.G2b. The only reply is 2.Gx2b, after which 3.B2c+ 4.Gx3c 5.+Bx3c is winning.

TODAI SHOGI plays 1.Bx2a+, judging the position after 2.K4a 3.G*6a as winning for black. GEKISASHI selects 1.S6h and also thinks black is winning after 2.+S5f 3.N3g 4.+Sx6f 5.Nx2e 6.P5d 7.Bx2a+ 8.K4a 9.S2b+ 10.R4i+ 11.N3c+ 12.+Rx1i. Finally, AI SHOGI also plays 1.S6h, also judging this winning for black after 2.+S5h 3.Bx2a+ 4.K4a.

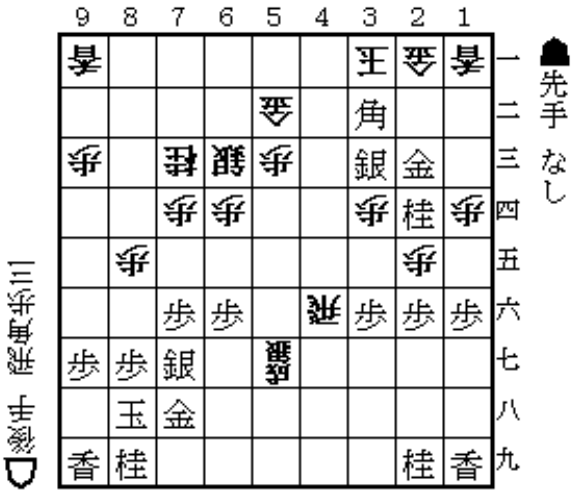

Figure 3: Inaccurate evaluation function example

The problem here seems to be an inaccurate evaluation of the position. After Bx2a+ K4a, the white king escapes and black has not enough pieces to prevent this. None of the programs is able to correctly assess this, evaluating that the black attack is strong enough to win. This is not a problem that can be solved with more search power. Correct evaluation of the chances of escaping an attack is necessary.

Another reason why this position is difficult might be the consecutive sacrifices of 1.G2b and 2.+B2c. There are other positions in the test set where it is necessary to sacrifice a number of pieces in a row to set up a winning attack. It is known that computer shogi programs do not give much priority to sacrifices. Most sacrifices are just handing over material and the search effort can better be spent elsewhere. Sacrifices are searched shallowly or not searched at all, especially at higher search depths. In most cases, a single sacrifice will eventually be searched deep enough to come up with the solution, but these forward pruning schemes can cause problems in the rare cases where multiple sacrifices are needed like in the position of Figure 3. Human players are good at recognizing goals in positions and playing moves to reach these goals, whether these moves are sacrifices or not. It is not easy to add the same awareness of goals into a shogi program, but the problem of forward pruning moves that are sacrifices needs to be attended to.

5.3 Unpromoted pieces

The position in Figure 4 is number 935-2 and was solved by 95% of the human respondants. The solution to this position is 1.P1c=. This is the rare

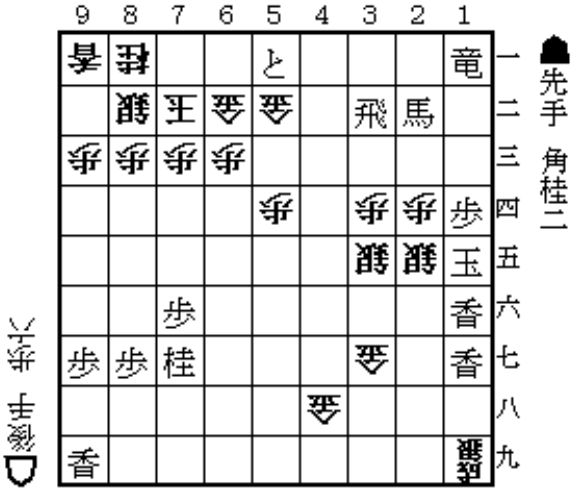

Figure 4: Mate with a pawn drop example

case where not promoting a pawn is better than promoting. 2.P*1d is now an illegal move (it would be winning after 1.P1c+) and after 2.S2e-2f 3.K1d or 2.S3e-2f 3.Kx2d, the black king escapes and black wins.

In this position TODAI SHOGI plays 1.+Px5b, acknowledging defeat. GEKISASHI plays the horizon check 1.N*8d, also indicating that black is losing. AI SHOGI takes the most drastic approach in this position: it resigns without playing a move!

The problem here is that promotion of a pawn (or bishop or rook) is almost always better than not promoting. Most shogi programs take their chances and improve the search efficiency by deleting non-promotions of pawns and major pieces. This is a problem that will not be easy to exploit it in actual game play, so it might not be a priority. There is one area that could cause future problems: thinking in opponent time. This is based upon guessing the opponent's move and searching these guesses while the opponent thinks about his move. When there is no difference between promotion and non-promotion of a move, a human player could select non-promotion because this would make the search in opponent time futile. After all, current programs do not generate such non-promotions and therefore this move will never be guessed. Our set of positions has 6 problems with non-promotion of pawn, rook and bishop, which seems a little bit too much. One for each non-promotion of a piece should be enough to cover this problem area.

5.4 Insufficient Hardware Speed

As mentioned earlier, setting the time limit to 30 seconds is arbitrary, especially since our tests were performed on a 2GHz Pentium 4 machine. The fastest PCs on the market are about two times faster and it is likely that a number of positions that could not be solved on our hardware can already be solved on faster hardware. To find the positions that are likely to be solved without requiring extra effort, we also gave the programs more thinking time to solve the positions. Forcing commercially available shogi programs to think long was unexpectedly complicated, but especially GEKISASHI's *Choukou* mode was very helpful. This mode does not have a time limit, but searches deeper. If a position could be solved using this mode, we assumed that hardware speed alone would soon be enough to solve that position without any further efforts into improving the search or the evaluation function. Our test set has 31 positions that can be solved using GEKISASHI's Choukou mode.

5.5 Time Allocation

In some positions, it was not possible to force the programs to think. Despite setting the time limit to 30 seconds (and often more than that), the programs would decide upon a move very quickly. We expected problems comparing this result with other programs and with future versions of the same program, so we deleted the positions where this happened. In hindsight, this might have been an error of judgment, as this behavior points to another problem area in shogi programs. When one move seems much better than all the other moves, the search is terminated early to avoid annoying the human opponent or to win time against a computer opponent. By doing so, there is the risk that a better move that can only be discovered at higher search depths is not found. In later versions of our test set we might re-introduce the positions where this early termination of the search occurred.

5.6 Overview

In Table 2, the problem areas for each position are given as far as we were able to assess them using the analysis tools in TODAI SHOGI and GEKISASHI. There are a 7 positions for which the reason why the programs could not find the solution is unclear. Further analysis is needed to determine why these positions are difficult. Note that some positions have multiple reasons, like the position of Figure 1.

In Table 3 the distribution of the problem areas is given. It seems that our test set has a reasonable balance between the important problem areas. Insufficient hardware speed is the main problem area, but the remaining 69 positions cannot be expected to be solved with faster hardware in the foresee-

Pos	*Prob*	*Pos*	*Prob*	*Pos*	*Prob*	*Pos*	*Prob*	*Pos*	*Prob*
750-3	1, 2	804-6	5	852-5	2, 3	910-5	4	964-6	1, 3
754-6	6	808-3	2, 3	854-6	4	920-6	1, 4	966-6	?
755-3	3	808-5	3	856-5	4	921-2	1	968-4	3
755-6	1	809-3	3	856-6	6	922-6	1	968-5	1
762-6	1	812-6	5	864-6	3, 4	924-6	1	968-6	3, 4
765-4	1, 2	818-6	3	865-6	5	925-5	4	973-5	6
765-6	6	821-5	5	871-4	4	925-6	6	977-3	6
769-5	6	821-6	4	873-6	2	934-6	1	979-5	3
772-4	1	823-5	6	877-5	2	935-2	5	988-2	4
773-6	2	823-6	6	879-4	4	942-6	5	988-5	3
779-6	?	825-6	4	887-6	3, 4	948-3	4	990-6	6
783-5	1	826-6	1	891-5	4	949-6	6	991-6	3, 4
785-1	?	828-6	1	895-6	6	951-6	6	992-5	6
785-4	?	831-6	2	898-6	3	952-5	3	993-6	3, 4
785-5	6	833-5	6	899-3	1, 2	952-6	6	994-6	6
786-2	2	836-6	6	900-4	6	955-6	1	995-4	6
786-6	2	838-6	3	905-5	6	957-5	6	1003-6	4
799-5	6	839-6	6	906-3	3	963-4	6	1005-2	?
800-6	6	842-3	4	908-5	?	963-6	1	1005-5	6
802-5	?	846-5	6	908-6	6	964-5	6	1005-6	3

Table 2: The problem areas for each position. 1 = horizon effect, 2 = tsume shogi, 3 = inaccurate evaluation function 4 = incorrect forward pruning, 5 = mate using unpromoted pieces, 6 = insufficient hardware speed, ? = reason unclear.

able future. If we consider mate by not promoting a pawn, rook or bishop of minor importance for actual game play, this leaves a minimal test set of 63 problems with the main problem areas.

Problem Area	*Posno*
Horizon effect	18
Tsume shogi	11
Inaccurate evaluation function	20
Incorrect forward pruning	19
Mate using unpromoted pieces	6
Insufficient hardware speed	31
Reason unclear	7

Table 3: The number of positions for each problem area

6 Todai Shogi 6 Results

Since the completion of our initial problem area analysis, new versions of the programs have become available. TODAI SHOGI 6, TODAI SHOGI 7, GEKISASHI 3 and AI SHOGI 2004 are now on sale. Of these, we currently only have TODAI SHOGI 6 available. There was no time for a thorough problem analysis using the program, but we did run the test set using TODAI SHOGI 6. TODAI SHOGI 6 was able to solve 6 of the 100 positions within 30 seconds. This indicates that Todai Shogi has been improved, but it also indicates that our test set is hard and that it will take a while before the majority of positions can be solved. Interesting is that the solved positions have different problem areas. The problem areas of the solved positions were inaccurate evaluation function (809-3, 921-2), insufficient hardware speed (823-5, 949-6), horizon effect (826-6) and reason unclear (1005-2). It seems that TODAI SHOGI 6 was improved in more than one area.

7 Human vs. Computer

Thus far we have not had the time to make a detailed comparison of the difference between positions that are difficult to solve for human players and positions that are difficult for computers. When looking at Table 1 it is clear that this difference exists. There are problems in our test set that can be solved by more than 90% of the human respondents. Table 4 gives a distribution of the percentages against the number of positions.

Almost half of the positions (46) can be solved by more than 50% of the human respondents. Furthermore, there are 14 positions that can not be solved by the computer programs but can be solved by more than 80% of the human respondents.

Percentage	*Pos*	*Percentage*	*Pos*
$0-10\%$	0	$51-60\%$	16
$11-20\%$	12	$61-70\%$	7
$21-30\%$	18	$71-80\%$	9
$31-40\%$	10	$81-90\%$	9
$41-50\%$	13	$91-100\%$	5

Table 4: The number of positions solved by human players for different percentage ranges

8 Conclusions

In this paper, we have pointed out that current sets of test positions for shogi have shortcomings. We have proposed a set of 100 positions that is general (none of the strong computer programs could solve them) and points to specific problem areas in computer shogi. We have analyzed some of the problem areas we discovered using the hint and analysis modes of the programs we used in our tests. We found problems with an horizon effect due to consecutive checks, not calling the tsume shogi solver deep in the search tree, inaccurate evaluation function, problems with mate using an unpromoted pawn, incorrect forward pruning, insufficient hardware speed and problems with the time allocation. We think these areas need to be improved to further improve the level of computer shogi programs.

Building a test set of positions that cannot be solved by strong shogi programs is by definition ongoing work. We already reported that TODAI SHOGI 6 can solve six of the positions in our test set so these can be deleted from the test set. Using other versions of the programs will delete more positions. We are particularly interested in the results of the versions that participated in the CSA World Computer Shogi Tournament.

Also, the increase of hardware speed will put more solutions within reach. We already know that a number of positions can be solved within 60 seconds, so considering the specifications of our hardware, these positions can already be considered solved.

On the other hand, there are new positions that can be added to the test set. We already pointed out that time allocation is a problem and adding positions which seem to suffer from this problem might be a good idea. Also, since our analysis was finished, new positions have been published in Shukan Shogi, which might turn out to be unsolvable for the top programs and point to new problem areas. Finally, we need to further investigate the 7 positions for which we could not determine the problem area.

Another line of research is the difference between what is difficult for humans and what is difficult for computers. If we look at Table 4, we see that even though most unsolvable positions are also difficult for human players, for many positions more than half of the human respondents found the correct move. So far, we have not investigated the reasons for this difference and for the moment this remains a future work.

References

[1] G. Grottling. The Truth About Their Strength. *ICCA Journal*, 7(4):221–226, 1984.

[2] J. Haring. A Problem to Solve. *ICCA Journal*, 8(3):181, 1985.

[3] D. Kopec and I. Bratko. The Bratko-Kopec Experiment: A Comparison of Human and Computer Performance in Chess. In M.R.B. Clarke, editor, *Advances in Computer Chess 3*, pages 57–72. Pergamon Press, Oxford, 1982.

[4] http://homepages.caverock.net.nz/~peter/chess4.htm.

[5] http://perso.wanadoo.fr/lefouduroi/test_lct_native.htm.

[6] L. Lindner. Experience with the Second Human-Computer Problem Test. *ICCA Journal*, 6(3):10–15, 1983.

[7] H. Matsubara and H. Iida. Some considerations on next-move tests of shogi. In *Game Programming Workshop in Japan '95*, pages 66–74, Kanagawa, Japan, 1995. (In Japanese).

[8] H. Matsubara and H. Iida. A next-move test set for shogi programs. In H. Matsubara, editor, *Computer Shogi Progress 2*, pages 61–111. Tokyo: Kyoritsu Shuppan Co, 1998. ISBN 4-320-02799-X. (In Japanese).

[9] H. Matsubara, H. Iida, and J. Uiterwijk. A next-move test set for shogi programs. In *Recent Advances in Computer Chess Workshop, 1996 ACM Computer Science Conference*, pages 139–146, New York, 1996.

[10] J.B. Nielsen. A Chess-Computer Test Set. *ICCA Journal*, 14(1):33–37, 1991.

[11] F. Reinfeld. *Win At Chess.* McKay, New York, 1945.

[12] http://www.platz.or.jp/~tanase/problems.html.

[13] http://www32.ocn.ne.jp/~yss/problems.html.