Knowledge Inferencing on Chinese Chess Endgames

Bo-Nian Chen[1], Pangfeng Liu[1], Shun-Chin Hsu[2], and Tsan-sheng Hsu[3,⋆]

[1] Department of Computer Science and Information Engineering,
National Taiwan University, Taipei
{r92025,pangfeng}@csie.ntu.edu.tw
[2] Department of Information Management,
Chang Jung Christian University, Tainan
schsu@mail.cjcu.edu.tw
[3] Institute of Information Science, Academia Sinica, Taipei
tshsu@iis.sinica.edu.tw

Abstract. Several Chinese chess programs exhibit grandmaster playing skills in the opening and middle game. However, in the endgame phase, the programs only apply ordinal search algorithms; hence, they usually cannot exchange pieces correctly. Some researchers use retrograde algorithms to solve endgames with a limited number of attack pieces, but this approach is often not practical in a real tournament. In a grandmaster game, the players typically perform a sequence of material exchanges between the middle game and the endgame, so computer programs can be useful. However, there are about 185 million possible combinations of material in Chinese chess, and many hard endgames are inconclusive even to human masters. To resolve this problem, we propose a novel strategy that applies a knowledge-inferencing algorithm on a sufficiently small database to determine whether endgames with a certain combination of material are advantageous to a player. Our experimental results show that the performance of the algorithm is good and reliable. Therefore, building a large knowledge database of material combinations is recommended.

1 Introduction

Several Chinese chess programs are playing at a par with human masters or grandmasters [14]. Most algorithms that are incorporated in Western computer-chess programs are also suitable for Chinese chess programs. In the opening game, the most popular strategy involves building an opening book, either by collecting a large number of games or by inputting only master-level opening moves. The strategy is successful, in particular for general opening play. If a position is not in the book, the most important component, the search engine, takes over and computes the best move by evaluating hundreds of millions of positions.

⋆ Corresponding author.

H.J. van den Herik et al. (Eds.): CG 2008, LNCS 5131, pp. 180–191, 2008.

Some programs can search more than 14 plies deep with today's computers. Although some computer-chess games end in the middle game, the endgame tends to be the key phase for strong programs.

However, in the endgame, the search performance is not comparable to the playing strength of master-level players. There are two reasons for this. The first reason is that players need more moves to finish the game than the search depth allotted to the program. The second reason is that the result of the endgame is not always related to the amount of material. For example, KR and KGGMM usually end in a draw, even though the former has the advantage of a rook. Hence, a program that uses the material advantage as the main evaluation feature often misinterprets it as a huge advantage to the attacking side.

To solve endgame problems, van den Herik and Herschberg suggested the concept of the retrograde strategy in 1985 [1]. Subsequently, van den Herik, Herschberg, and Nakad constructed a six-man endgame database of chess in 1987 [2]. Thompson proposed an improved retrograde algorithm in 1986 [6] and solved 6-piece chess endgames in 1996 [7]. Subsequently, Schaeffer (2003) created a 10-piece endgame database of Checkers [4]. Some games, like Checkers, used the retrograde method successfully [8]. For instance, Gasser solved Nine-Men's Morris in 1996 [11]. For the full game of Western chess, which is a complex game, the retrograde strategy has so far not been very successful. In 2000, Nalimov used an efficient partitioning of subgames to build all 3-to-5-men endgames [9]. In summary, we may state that the endgame research is still in progress.

In Chinese chess, Fang used the retrograde method to construct an endgame database in 2000 [3], and in 2002 Ren Wu [12] used a memory efficient strategy to build large endgames, including KGMCPKGGMM. In 2006, Wu, Liu, and Hsu proposed using an external-memory strategy for building a retrograde algorithm for a large endgame database [10]. Nowadays, there are also web sites that provide the exact values of endgame databases [5]. However, there are serious time and space limitations when constructing a practical endgame database of materials with sufficient attack pieces. The current largest endgame database of Chinese chess comprises no more than two strong attack pieces on each side. We remark that many useful endgames that contain two strong attack pieces on both sides cannot be solved by retrograde strategies.

In a typical grandmaster game, before a grandmaster applies his[1] endgame knowledge, he usually performs a series of material exchanges at the end of the middle game. In each material exchange, he gradually obtains an advantage. The advantage may not derive from accumulating more materials, but from a combination of materials that has proven to be better based on prior experiences. For example, it is generally believed that a combination of one rook, one horse, and one cannon is better than a combination of two horses and two cannons, although their material values are roughly equal. The goal of this paper is to determine whether a material combination is good by performing knowledge inferencing on a small dataset of kernel knowledge. To this end, we define two phases in the endgame: (1) *the prior phase*, during which many attack pieces are

[1] For brevity we use 'he' and 'his' whenever 'he or she' and 'his or her' are meant.

still in position and retrograde strategies cannot be applied to them; and (2) *the posterior phase*, which can be solved completely by retrograde algorithms.

In particular, we propose a novel strategy that applies a knowledge-inferencing mechanism on a small knowledge database of material combinations to generate a database of material for the prior phase of a practical endgame.

The remainder of this paper is organized as follows. In Sect. 2, we describe the knowledge database of material combinations and the implemented knowledge-inferencing technique. In Sect. 3, we introduce a probabilistic model for predicting unknown material states. In Sect. 4, we build a practical knowledge database of material combinations. In Sect. 5, we take the data used by CONTEMPLATION [15] as our experimental data and report the results of applying our model to it. Then, in Sect. 6, we present our conclusions.

2 Constructing a Knowledge Database

To construct a practical knowledge database of material combinations, henceforth called a *material database*, we first need to construct a basic database. Instead of adding all data manually, we utilize knowledge-inferencing techniques in the construction phase to reduce the workload and the time required for the task.

2.1 Knowledge Database of Material Combinations

The word *material* denotes all pieces that appear in a specific position in both Western and Chinese chess. The *material state* of a position is an evaluation measurement that only considers material in the given position, not with respect to different locations.

For simplicity, we assume that two players in an endgame play either the attacking role or the defending role. The attacking role, which is called the *attacking player*, is defined as the player that has more attack power than the player with the defending role, who is called the *defending player*. We define 5 categories of material states for a material combination.

WIN: The score when the attacking player usually wins.
EASY_WIN: The score when the attacking player wins in many cases, but draws in some cases.
CHANCE_WIN: The score that ends in a draw in most cases, but the attacking player wins in some cases.
HARD_WIN: The score when the attacking player seldom wins.
UNKNOWN: The score when the attack power of either side is strong enough to capture the king of the opposite side; hence information about the material is not very useful.

A knowledge database of material combinations consists of the defending materials that players use. Each item of defending material is mapped to an attack file that includes all possible attack materials. Attack material is defined as the pieces that belong to the attacking player. The possible number of materials held

by a player in Chinese chess can be computed by combinatorics as follows. First, there are 27 combinations of strong pieces, including rooks, horses, and cannons. Second, pawns are divided into three categories, as defined in Subsection 2.3. By using combinations with repetition of all possible numbers of pawns, we retrieve the combinations of all categories of pawns, which total 56. Third, there are 9 combinations of defending pieces, including guards and ministers. Totally, a player can have 13,608 ($= 27 \times 56 \times 9$) possible material combinations; and the total number of possible material combinations on both sides is 185 million.

We have designed two useful knowledge inferencing strategies. The first strategy, *redundant attacking material checking and elimination*, which is described in Subsection 2.2, can be applied when creating both the basic database and database queries. The second strategy, called *pawn inferencing*, can only be used when creating the basic database. It is described in Subsection 2.3.

2.2 Redundant Attacking Material Checking and Elimination

This knowledge-inferencing tool can find and remove all attack material that is not necessary. The idea is that if we already know a material state is a WIN, material states to which attack material is added by one or more pieces are also WIN states because the attacking player has a bigger advantage in the WIN state. Similarly, if a material state is a HARD_WIN, material states from which attack material is taken by one or more pieces are also at most HARD_WIN states.

By using this algorithm, we can eliminate redundant attack materials when creating the basic database. For database queries, the same concept is used when there are some gaps between the attack power of two players. If the state of attack material found in the database is a WIN and the material is a subset of the query attack material, we can also report a WIN state. We call this inferencing algorithm *material state extension*.

A knowledge database of material combinations is said to be complete if all the database items that record defense materials have all the necessary information about attack materials. Generally, the time complexity of a query is $O(NM)$, where N is the number of defending materials in the database, and M is the maximum number of attacking materials among all defending materials in the database. However, if we use a complete material database, we do not need to search the whole database for the answer to a query. Instead, we only search the desired attacking file so that the time complexity becomes $O(M)$. We remark that the time so saved leads to more computation when searching.

2.3 Pawn Inferencing

In Chinese chess, as in Western chess, it is illegal to move a pawn backwards, but in Chinese chess a pawn is not promoted when reaching the final rank. Since the opposite player's king can only stay somewhere in the last 3 ranks, the distance between a pawn and the final rank decides the pawn's power. In the common definition, there are three types of pawns:

1. Top-pawn: the pawn stays behind the river line or the pawn line of the opposite side and has yet to cross the river. It moves forward 3 steps at most.
2. Low-pawn: the pawn moves forward 4 or 5 steps.
3. Bottom-pawn: the pawn reaches the final rank. Note that a pawn must move forward 6 steps to reach the final rank.

In general, a top-pawn is more useful than a low-pawn and a low-pawn is more powerful than a bottom-pawn. Furthermore, if we know the state of material with one bottom-pawn, we cannot obtain a better result by adding more bottom-pawns in all cases.

There is a similar rule for low-pawns. If we know the state of material with two low-pawns, we cannot obtain a better result by adding more low-pawns in most cases. There are two possible reasons for this. First, if low-pawns can win, then, based on past experience, only two low-pawns are sufficient to win. Second, if low-pawns cannot move into the palace or are lower than the king, adding low-pawns will not solve the problem. For example, the results of the material combinations KPPKGGMM and KPPPKGGMM are a CHANCE_WIN when all pawns are low-pawns. In our basic database, there are 16,705 material combinations where the attacking player has two low-pawns, and there are only 361 combinations where the result of corresponding material with three low-pawns is different. However, when there is one top-pawn in the material, the attacking player can always gain an advantage by adding another top-pawn.

The pawn-inferencing algorithm is a game-specific inferencing scheme that is only suitable for Chinese chess. It uses the knowledge of bottom-pawns and low-pawns. If we have the result of material containing one bottom-pawn or two low-pawns, we can use the algorithm to copy the results to more bottom-pawns or low-pawns until the number of bottom-pawns plus low-pawns equals 5. The algorithm reduces the work involved in creating the basic database by almost half. This is because the combinations of materials with more than one bottom-pawn or more than two low-pawns that can be generated automatically are approximately equal to the combinations of materials with one bottom-pawn or less than or equal to two low-pawns.

3 Predicting Unknown Material States

Although a large number of original unknown material states can be inferred by methods stated in Sect. 2, we still need a systematic strategy for handling arbitrary unknown materials. The algorithm that predicts arbitrary unknown materials is called the *unknown state predictor*.

3.1 Human Evaluation of Unknown Positions

By exchanging pieces, human experts can accurately infer the results of material combinations that were previously unheard of. For example, KHKGGMM is generally a draw. When the result of the material combination KRHKRGGMM

is in question, we may see the following: if the defending player has a rook, he can exchange it directly with the rook of the attacking player, and the result will be a draw. This strategy is called *material reduction*.

A second example is the material combination KRPKHGGMM. If the attacking side exchanges a pawn for two guards of the defending player, the resulting material KRKHMM can win easily, but it would not be an absolute win. However, if the pawn is exchanged for two ministers of the defending player, the resulting material, KRKHGG would be an absolute win.

The two examples show that making a correct exchange of pieces is important during the endgame phase.

3.2 Material Exchange Table

We have designed a probabilistic model that predicts the results of unknown material states by exchanging pieces. Both sides can exchange pieces when necessary. A material exchange table is introduced to compute the probabilities of exchanging pieces.

The mobility of many types of pieces is different. The ability to exchange a certain piece for pieces of another type is also different. A *helper piece* can be any piece that is not being exchanged, but it can be used to facilitate an exchange. Each player can select one piece as the helper piece. Generally, actively exchanging pieces with the assistance of a helper piece will increase the player's exchange ability. Similarly, passively exchanging pieces with the aid of a helper piece may reduce the chance of pieces being exchanged. Hence, we manually construct a two-dimensional material exchange table to record the probabilities of exchanging each type of piece with the assistance of helper pieces.

There are 6 types of pieces in addition to the king. To map a table to each active/passive piece pair, we use 36 tables for all possible types. Each table contains the probabilities of the specified active piece with all possible helper pieces and the specified passive piece with all possible helper pieces.

3.3 Determining the Score of an Unknown Material State

For an unknown material combination, we can try to make any exchange and to make a reference to the database for the material state. The strategy of an expert player is to choose the possible best way to make an exchange. We can accept an exchange that has a high probability, but we cannot accept an exchange with a low probability.

An *acceptable exchange* is formally defined as an exchange of which the material state is the most advantageous to the active player in all feasible situations and of which the probability is higher or equal to a lower bound. To achieve an acceptable probability of exchange and to avoid wasting time on searching for exchanges with low probability, we define the probability lower bound, PLB, to filter out situations with very low probability that seldom occur in practice. In our test, the best value of PLB is 10%. After an exchange, we make a reference to the database to retrieve the result of the reduced material. If two or more

exchanges result in the same material state, we choose the one with the highest probability. If we cannot find the result in the database, the material state of the specified material remains UNKNOWN.

The algorithm computes two acceptable exchanges: (1) the attacking player exchanges pieces actively, and (2) the defending player exchanges pieces actively. Each exchange reaches its own material state. We define five numerical score values, 0, 1, 2, 3, and 4, which correspond to UNKNOWN, WIN, EASY_WIN, CHANCE_WIN, and HARD_WIN, respectively. If the material states of both sides are known, the final score of the query material is computed by the formula $V = \lfloor (V_a + V_d)/2 \rfloor$. The values V_a and V_d represent the results of the attacking player and the defending player exchanging pieces actively, respectively. V is the final score. If one of the material states is unknown, we choose the known state as our result. If both are unknown, the result remains unknown. This formula simply computes the average of the two results. It is worth noting that, because we use division on integers, the result leans towards WIN rather than HARD_WIN, due to the setting of the numerical scores.

4 Constructing a Practical Knowledge Database of Material Combinations

We use two algorithms, material state extension and unknown state predictor, to determine the advantage of unknown materials.

To construct a knowledge database of material combinations, we simply generate each material pair as input for the material state extension algorithm, which can only be applied to WIN and HARD_WIN in the basic database. If the algorithm cannot find the answer, we input the material pair to the unknown state predictor algorithm to retrieve an approximate result value.

However, the value of some materials may still be unknown after applying the unknown state predictor algorithm. Finally, we use a heuristic algorithm to identify the advantage or disadvantage of the input material. We compute a player's attack power by the formula $10 \times Rook + 5 \times (Horse + Cannon) + 1 \times Pawn$. In the formula, Rook, Horse, Cannon, and Pawn are the numbers of the attacking pieces. The *difference between the attack power* of the two players is calculated as the formula $D = RedPower - BlackPower$. RedPower is the attack power of the attacking player, and BlackPower is that of the defending player. When D is more than or equal to 10, we reduce the value of the material state by one. When D is less than 7 and the predicted result is UNKNOWN, we set it to be CHANCE_WIN. This simple algorithm is used to fine tune the materials when the attacking player has a clear advantage or the value of the materials cannot be derived by the unknown state predictor algorithm.

The most practical usage of the knowledge database of material combinations is to retrieve material scores as a part of the evaluation function during the search phase. When a middle game position changes to an endgame position due to piece exchange, the search algorithm can select better endgame positions with the aid of our material database. However, there may be some positions

where the attack power of both sides is strong; or one player is disadvantaged in terms of material, but still represents a great threat to the opposite player's king. The former can be handled by assigning UNKNOWN states to the positions when both sides are strong enough to attack each other's kings. The latter can be handled by increasing the weight of special locations of piece combinations in the evaluation function.

5 Experiment Design and Results

To demonstrate the performance of our algorithm, we generate a basic database. It is a complete database of defense materials with at most one strong attacking piece plus one pawn and all defending pieces. We use a practical data set as our test data and compare it with the results obtained by our algorithm.

5.1 Experiment Design

We use the endgame knowledge table used by CONTEMPLATION as our test data. There are 17,038 combinations of materials that have been manually annotated by a 4-Dan expert. Since the data is symmetric, that is, if a material combination is in the database, information about exchanges between the attacking player and the defending player is also in the database, the actual number of test data combinations is 8,519. The scoring scheme used by the test data is different to that of our method. The score of the test data is divided into 10 values. The values 0 and 1 are mapped to WIN in our method, which means the attacking player usually wins. The value 2 is mapped to EASY_WIN, 3 is mapped to CHANCE_WIN, 4 is mapped to UNKNOWN, and 5 is mapped to HARD_WIN. The values from 6 to 9 indicate that the attacking player changes places with the defending player. The value 6 is mapped to CHANCE_WIN; 7 is mapped to EASY_WIN; and 8 and 9 are mapped to WIN. A second difference relates to the definition of pawns. In the test data, all pawns are the same, with no category information. As a result, our program must compute the approximate values of materials and then compare them with the test data. Because bottom-pawns are not considered by the test data, we only compute the approximate values of materials with top-pawns and low-pawns. The approximation formula is $V_{app} = \lfloor (V_{top} + V_{low})/2 \rfloor$, where V_{app} represents the approximated result; V_{top} represents the result of defining all pawns of both players as top-pawns; and V_{low} represents the result of replacing all pawns of the attacking player with low-pawns. There are 6,396 entries that are not in our basic database. We use the difference between the attack powers to filter out unreasonable annotations, which means that the attacking player has less attack power than the defending player, and is assigned the grade of better than or equal to CHANCE_WIN.

There are 1,621 annotations where the attacking player, who has the advantage of at least CHANCE_WIN, has less attack power than the defending player. The remaining data, containing 4,775 entries, becomes our test set, called END4775. We use two algorithms in the test: (1) material state extension, and

(2) unknown state predictor. The first experiment demonstrates the result of combining the two algorithms. The second experiment demonstrates the result of only using the unknown state predictor algorithm.

5.2 Experimental Results

In our results, we denote UNKNOWN by U, WIN by 1, EASY_WIN by 2, CHANCE_WIN by 3, and HARD_WIN by 4. The descriptions and results are shown in Table 2. We define the following variables to measure our model's performance: (1) *total_correct_number*, which records the number of cases where the output scores are equal to the transformed answer; (2) *tolerant_correct_number*, which ignores the error between WIN and EASY_WIN and also between CHANCE_WIN and HARD_WIN; and (3) *slight_error_number*, which records the errors between WIN and EASY_WIN and also between CHANCE_WIN and HARD_WIN.

For our algorithm, we need to choose a suitable value of PLB, described in Subsection 3.3. Table 1 shows the relationship between different PLBs and the ratio of tolerant_correct_number to the total number of data items, i.e., 4775. This is the most important measurement, when using only the unknown state predictor algorithm. As the results show, the value 10% is the best for our test data. We suggest that users set the PLB value between 10% and to 30%.

The total_correct_number is 2,169 or 45.42%. The tolerant_correct_number is 4,200 or 87.96%. The slight_error_number is 2,031 or 42.53%.

In practical usage, the most important measurement is tolerant_correct_number because it identifies the categories of either WIN and EASY_WIN, which are

Table 1. The relationships between PLBs from 0 to 100 and the corresponding ratio of tolerant_correct_number to the total number of data items

PLB	0	10	20	30	40	50	60	70	80	90	100
%	82.07	84.50	84.13	83.12	82.28	81.53	80.04	76.04	65.13	39.25	39.04

Table 2. Comparison of human annotated answers and the algorithm generated results for END4775. The horizontal axis represents the number of human annotated material states. The vertical axis represents the number of material states generated by the algorithm. U represents an unknown state.

	U	1	2	3	4	Sum
U	0	35	55	195	40	325
1	0	990	402	52	0	1444
2	0	1278	663	120	17	2078
3	0	31	30	233	330	624
4	0	0	0	21	283	304
Sum	0	2334	1150	621	670	4775

considered winning materials, or CHANCE_WIN and HARD_WIN, which are considered draw materials. The value 87.96% indicates the percentage of how well our algorithm fits a human expert's endgame knowledge. Moreover, the percentage shows the accuracy of the material part of the evaluation function used by a search algorithm in prior phase of the endgame, as defined in Sect. 1. Hence, the result shows that the search algorithm using our method will make as good an exchange as CONTEMPLATION in most cases.

A material combination KCPGGKPPP in our test data is assessed as EASY_WIN by a human expert, but reported as HARD_WIN by our algorithm. The discrepancy is due to the different opinions about the defense ability of a defending player who has three pawns. Since even masters have different opinions about hard endgames, a slightly different human annotated answer is reasonable.

The performance of the individual algorithms is as follows. The number of material combinations that can be inferred by the material state extension algorithm is 2,614. The total_correct_number among 2,614 entries is 1,379 (52.75%); the tolerant_correct_number is 2,562 (98.01%); and the slight_error_number is 1,183 (45.25%).

By using the heuristic strategy described in Sect. 4, we did not obtain any unknown material states in this test. The number of the entries that could not be handled by the material state extension is 2,152. However, they can be predicted by our predictor algorithm or the heuristic algorithm. The total_correct_number is 781 (36.29%); the tolerant_correct_number is 1,814 (84.29%); and the value of slight_error_number is 1,033 (48.00%).

Although the ratio of total correctness is reduced by using the heuristic strategy compared to that of combining two algorithms, we believe that our predictor algorithm is reliable for the following three reasons. First, the input of 2,152 entries is the most complex data among all data sets. Second, the total correctness ratio shows that, for the given material, the algorithm can distinguish the true advantage or disadvantage in endgames. Third, even master players cannot clearly identify the difference between WIN and EASY_WIN and between CHANCE_WIN and HARD_WIN based only on information about the material. For example, Y. C. Xu, a Chinese chess grandmaster, gave his opinions about a practical endgame in his publication "YinChang Chess Road." He criticized his opponent, G. L. Wu, who is also a Chinese chess grandmaster [13].

The human annotated answer for the material combination KHCPKHCM in our test data is an EASY_WIN; however, our algorithm reports CHANCE_WIN, which has different advantage. If a situation like this occurred during a real game, a grandmaster would not usually exchange an attack piece with his opponent because any exchange would result in a draw. These kinds of material combination problems cannot be solved by reducing the amount of material.

To evaluate the performance of using the unknown state predictor algorithm alone, we apply it to all 4,775 material combinations. The detailed experimental results are presented in Table 3.

The total_correct_number is 2,110 (44.19%); the tolerant_correct_number is 4,035 (84.50%) and the slight_error_number is 1,925 (40.31%).

Table 3. Results using only the unknown state predictor algorithm

	U	1	2	3	4	Sum
U	0	76	30	188	31	325
1	0	1302	30	112	0	1444
2	0	1615	281	158	24	2078
3	0	35	80	261	248	624
4	0	4	2	32	266	304
Sum	0	3032	423	751	569	4775

Note that the ratios of the tolerant_correct_numbers to the total number of data items are similar among the two tests, and so do the slight error number values. This shows that the data input to the unknown state predictor algorithm in the first experiment is really hard. The difference between the ratio of the tolerant_correct_number to the total number of data items of the two experiments is 3.46%. This indicates that the advantage predicted by our predictor algorithm is still reliable, even for hard data.

The space used to store all defense materials up to one strong piece plus a pawn and all combinations of defense pieces and their corresponding attack materials is 2.44M bytes.

6 Conclusions

Endgame problems represent a difficult issue in both Western chess and Chinese chess. The largest Chinese chess endgame database built by a retrograde algorithm currently contains only two strong attack pieces on each side. However, the endgame results show that many strong attack pieces exist. We have designed a knowledge inferencing scheme to build a practical material database for the initial phase of the endgame. In addition, we use the material state extension algorithm and the unknown state predictor algorithm to construct endgames with many strong attack pieces. Our experimental results show that the performance of our algorithms is good and reliable. When predicting the advantage of a material combination with a large number of pieces, we may conclude from the results above that our material state extension algorithm is an effective approach. However, if the extension algorithm fails, the predictor algorithm takes over and reports an inferred solution. This strategy can be used to solve the problem when a complete knowledge database of an endgame with a large amount of material cannot be built using conventional computer methods, and only advantage information is required to know for the material state.

Acknowledgments. This research was partially supported by National Science Council, Grants 95-2221-E-001-004 and 96-2221-E-001-004.

References

1. van den Herik, H.J., Herschberg, I.S.: The construction of an omniscient endgame data base. ICCA Journal 8(2), 66–87 (1985)
2. van den Herik, H.J., Herschberg, I.S., Nakad, N.: A six-men-endgame database: KRP(a2)KbBP(a3). ICGA Journal 10(4), 163–180 (1987)
3. Fang, H.R., Hsu, T.S., Hsu, S.C.: Construction of Chinese chess endgame databases by retrograde analysis. In: Marsland, T., Frank, I. (eds.) CG 2001. LNCS, vol. 2063, pp. 96–114. Springer, New York (2000)
4. Schaeffer, J., Björnsson, Y., Burch, N., Lake, R., Lu, P., Sutphen, S.: Building the checkers 10-piece endgame databases. In: van den Herik, H.J., Iida, H., Heinz, E.A. (eds.) Advances in Computer Games: Many Games, Many Challenges, vol. 10, pp. 193–210. Kluwer Academic Publishers, Dordrecht (2003)
5. Pai, J.T.: Chinese Chess Endgame Databases Query System, `http://lpforth.forthfreak.net/endgame.html`
6. Thompson, K.: Retrograde analysis of certain endgames. ICCA Journal 9(3), 131–139 (1986)
7. Thompson, K.: 6-piece endgames. ICCA Journal 19(4), 215–226 (1996)
8. Lake, R., Schaeffer, J., Lu, P.: Solving large retrograde analysis problems using a network of workstations. In: Advances in Computer Chess 7, Maastricht, The Netherlands, pp. 135–162 (1994)
9. Nalimov, E.V., Haworth, G.M., Heinz, E.A.: Space-efficient indexing of endgame databases for chess. In: van den Herik, H.J., Monien, B. (eds.) Advances in Computer Chess 9, pp. 93–113 (2001)
10. Wu, P.S., Liu, P.Y., Hsu, T.S.: An external-memory retrograde analysis algorithm. In: van den Herik, H.J., Björnsson, Y., Netanyahu, N.S. (eds.) CG 2004. LNCS, vol. 3846, pp. 145–160. Springer, Heidelberg (2006)
11. Gasser, R.: Solving nine men's morris. In: Nowakowski, R. (ed.) Games of No Chance. MSRI, vol. 29, pp. 101–113. Cambridge University Press, Cambridge (1996)
12. Wu, R., Beal, D.F.: Fast, Memory-Efficient Retrograde Algorithms. ICGA Journal 24(3), 147–159 (2001)
13. Xu, Y.C.: YinChang Chess Road, Special Column 44-45. Yan Chen Ti Yu Newspaper Office (1997)
14. Yen, S.J., Chen, J.C., Yang, T.N., Hsu, S.C.: Computer Chinese Chess. ICGA Journal 27(1), 3–18 (2004)
15. CONTEMPLATION, A Chinese chess program, `http://www.grappa.univ-lille3.fr/icga/program.php?id=112`