Indefinite Sequence of Moves in Chinese Chess Endgames

Haw-ren Fang[1]*, Tsan-sheng Hsu[2]**, and Shun-chin Hsu[3]

[1] Department of Computer Science
University of Maryland
A.V. Williams Building, College Park, Maryland 20742, USA
hrfang@cs.umd.edu
[2] Institute of Information Science
Academia Sinica
No 128, Section 2, Academia Road, Nankang, Taipei 115, Taiwan
tshsu@iis.sinica.edu.tw
[3] Department of Computer Science and Information Engineering
National Taiwan University
No 1, Section 4, Roosevelt Road, Taipei 106, Taiwan
schsu@csie.ntu.edu.tw

Abstract. Retrograde analysis has been widely used to solve many problems. In western chess, it has been successfully applied to construct 6-piece endgame databases. This classical algorithm first determines all terminal win or loss positions, i.e., those that are either checkmate or stalemate, and then propagates the values back to their predecessors until no propagation is possible. The un-propagated positions are then declared draws.

However, in Chinese chess, there are special rules other than checkmate and stalemate to end a game. Therefore, some terminal positions cannot be determined in the initialization phase of a typical retrograde analysis algorithm. If these special rules are ignored in the construction of the endgame databases by retrograde analysis, the resulting databases may contain incorrect information.

In this paper, we not only describe our approach in abstracting the special rules of Chinese chess and its consequent problems while retrograde analysis is applied, but also give a solution to construct complete endgame databases complying with the most important special rule in which one player checks the opponent all the time in order to avoid losing the game. Forty-three important endgames with both sides having attacking pieces are constructed. We believe this problem has not been tackled successfully before.

---

* This work was mainly done while this author is with Institute of Information Science, Academia Sinica, Taiwan.

** Corresponding author.

1 Introduction

Retrograde analysis has been widely used to solve many problems. It has been successfully applied to construct 6-piece endgame databases for western chess [4]. This classical algorithm first determines all terminal win or loss positions, i.e., those that are either checkmate or stalemate, and then propagates the values back to their predecessors until no propagation is possible. The un-propagated positions are then declared draws.

In both western chess and Chinese chess, checkmate and stalemate are the two principal rules to end a game. They are also the foundations for propagation while retrograde analysis is applied. In western chess, if a game continues periodically with a series of moves, it ends in a draw. On the contrary, such a game may result in a win/loss/draw depending on the characteristics of the indefinite move patterns involved. These rules are called *special rules* throughout this paper.[1]

The most influential special rule is *checking indefinitely*. If only one player checks his opponent continuously,[2] he loses the game. Therefore, the endgame databases of Chinese chess constructed by retrograde analysis may have errors if this special rule is not taken into account. Other special rules may also spoil the endgame databases in a similar way. It is known that endgames, in which only one side has attacking pieces, are not affected by these special rules [2]. Using this fact, endgame databases with attacking pieces on one side only are constructed [2, 6]. This paper studies the problem of constructing Chinese chess endgame databases with both sides having attacking pieces. We believe this problem has not been tackled successfully before.

2 Notations and Rules of Chinese chess

In Chinese chess, the two sides are called Red and Black. Each side has one King, two Guards, two Ministers, two Rooks, two Knights, two Cannons and five Pawns which are abbreviated as K, G, M, R, N, C and P, respectively. The pieces Rook, Knight, Cannon and Pawn are called *attacking pieces* since they can move across the *river*, the imaginary stream between the two central horizontal lines of the board. In contrast, Guards and Ministers are called defending pieces because they are confined in the domestic region.[3] Moreover, a side is called *armless* if it

[1] The detailed special rules of Chinese chess are very complicated. Some minor rules differ in the Asian, Chinese, and Taiwanese versions, and might be revised as time goes on. Our discussions in this paper are all based on the rule book [1] published in 1999.

[2] In real games, each indefinite checking pattern is determined by the appearance of the same position for three times in a series of moves in which one side checks his opponent all the time.

[3] The information of Chinese chess such as notations and rules in English can be found in FAQ of the Internet news group `rec.games.chinese-chess`, which is available at `http://www.chessvariants.com/chinfaq.html`. More detailed rules can be found at `http://txa.ipoline.com/`.

has no attacking pieces. A *position* in Chinese chess is an assignment of a subset of pieces to distinct addresses on the board with a certain player-to-move.

In Chinese chess, a game also ends in a position of checkmate or stalemate. In addition, there are other end positions caused by the special rules of indefinite move patterns. Therefore, the databases constructed by retrograde analysis may have incorrect information if these special rules are ignored. In the following chapters, we describe how the problems caused by the special rules occur and our approach to solve them.

3 Special Rules in Chinese Chess

Retrograde analysis algorithms are widely used to construct the databases of finite, two-player, zero-sum and perfect information games [5]. In the endgame databases of western chess, the bottom is established by the checkmate and stalemate positions. In Chinese chess, the resulting endgame databases have potential errors if the special rules were to be neglected.

3.1 The Rule of Checking Indefinitely

In Chinese chess, if only one player checks his opponent continuously *without ending*, then he loses. In this case, a game without ending is formally defined as having the same position occurring three times during the game. This rule is called *checking indefinitely*. In real games, this means that a player loses if he cannot prevent his King from being captured without checking his opponent indefinitely.

As shown in Figure 1(a), the Black side can move either Pg1-g0 or Pg1-f1 to checkmate the Red King. The Red side cannot avoid his King being captured without checking the Black King by making the move Ra8-a9. The checking continues endlessly with the moves Kd9-d8, Ra9-a8, Kd8-d9, Ra8-a9, etc. In Chinese chess, the Red side loses the game because he violates the rule of checking indefinitely. Note that if one of the two Black Pawns is removed from the board, the Red side can play Ra8-a1 to capture the Black Pawn and then easily win the game thereafter.

The example in Figure 1(b) is most intriguing. The only logical move for the Red side that will not let him lose the game immediately is Rf0-e0. Then, if the Black side plays Ch0-e0 to capture the Red Rook, the game will end in a draw thereafter. However, if the Black side plays Ke8-f8 instead of Ch0-e0, the only move for the Red side is Re0-f0. The game continues cyclically with the moves Re0-f0, Kf8-e8, Rf0-e0, Ke8-f8, etc. The Black side wins because the Red side violates the rule of checking indefinitely. From this point of view, if one side can force his opponent to check him indefinitely, he wins.

In the databases constructed by retrograde analysis regardless of this rule, a position in which either side cannot checkmate or stalemate the other, resulting the game to continue endlessly, is regarded as a draw. Therefore, the positions in Figure 1 are incorrectly recorded as draws, but are treated as Black-to-win

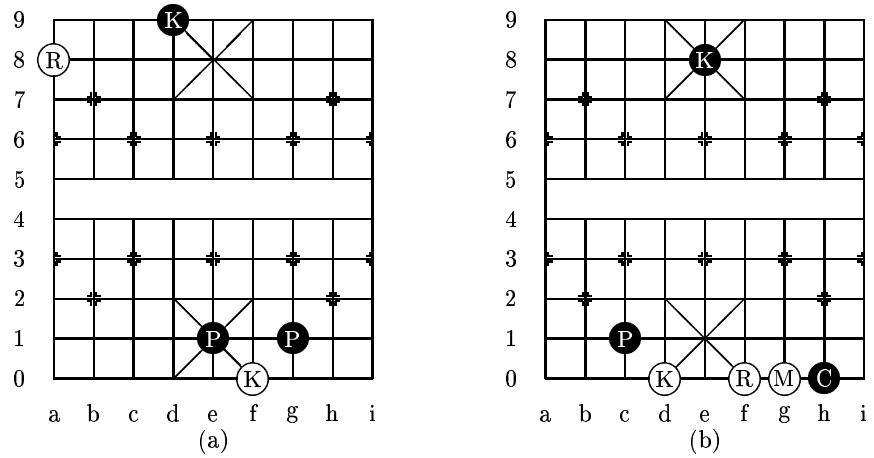

Fig. 1. Two examples of checking indefinitely.

according to the rule of checking indefinitely. Ignoring the special rules may therefore result in potential errors in the constructed endgame databases.

3.2 The Rule of Mutual Checking Indefinitely

The rule of checking indefinitely has an exception by another rule called *mutual checking indefinitely*, which means the two players check each other continuously without end. When this happens and the two players do not want to break the pattern, the game ends in a draw. As shown in Figure 2(a), if the game continues cyclically with the moves Re6-e7, Cf8-e8, Re7-f7, Ce8-f8 Rf7-e7, Cf8-e8, etc., the game is treated as a draw since both sides check each other indefinitely.

The example in Figure 2(b) is interesting as the game continues cyclically with the moves Pf5-e5, Pf3-e3, Pe5-f5, Pe3-f3, etc. The Black side only checks the Red side every two moves whereas the Red side checks all the time. This is not regarded as mutual checking indefinitely but treated as checking indefinitely. Thus, the Red side loses the game if he does not break the pattern.

3.3 Other Special Rules

In Chinese chess, there are rules of *chasing indefinitely*. The general concept is that one cannot chase his opponent's piece continuously without end. The term *chase* is defined similarly to the term check. The difference is that the prospective piece to be captured is not the King but another piece. The detailed rules of chasing indefinitely are very complicated. It is also possible that they may stain the endgame databases. In addition, there are special rules about other indefinite move patterns.

All these special rules can be classified into two categories. The first category is where, the game ends in a draw if a certain indefinite move pattern occurs and

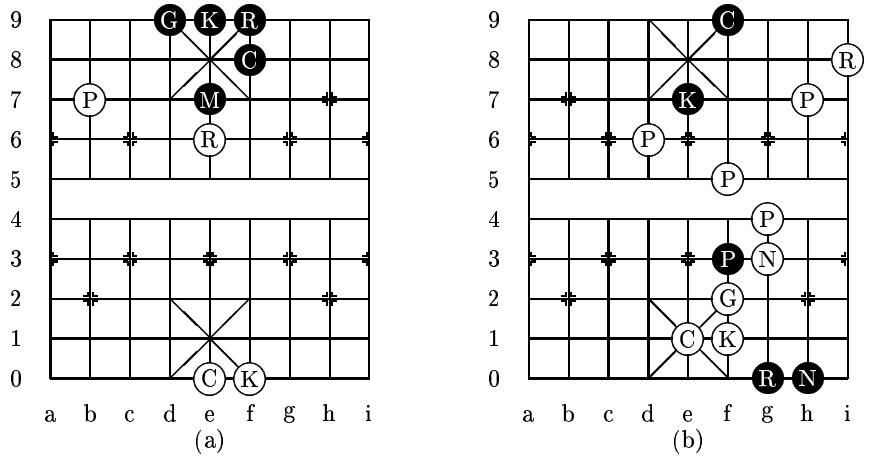

Fig. 2. An example of mutual checking indefinitely on the left and an example of non-mutual checking indefinitely on the right.

the two players do not want to break the pattern, e.g., the rule of mutual checking indefinitely. While constructing endgame databases by retrograde analysis, all the positions which are not propagated from checkmate or stalemate positions are eventually marked as draws. Therefore, the first category of special rules does not stain the endgame databases. The second category of special rules is when, if a certain indefinite move pattern occurs and the two players do not want to break the pattern, one side loses the game according to the rule. For example, the player checking indefinitely loses the game. The rules that may corrupt the databases will be dealt with in this paper.

3.4 Summary of Special Rules

In a sequence of moves with some position appearing 3 times, the sequence of all the plies of each side is classified as being *allowed* or *forbidden*. For example, checking indefinitely is is forbidden.[4] A sequence of plies is allowed if it is not forbidden.

According to [1], the special rules are summarized as follows and these are applied to an indefinite move pattern:

1. If only one side checks the other indefinitely, the side who checks loses the game.
2. Otherwise, if the sequence of plies of one side is forbidden and that of the other side is not all forbidden, the side with forbidden sequence of plies loses.
3. Otherwise, the game ends in a draw.

[4] In the Asian and Taiwanese versions of the rules, each forbidden sequence consists of only plies of checking or chasing. In the Chinese version of the rules, it may contain another ply called *threatening to checkmate*. It means a ply that leads to a win if the opponent gave up the right of the next move, i.e., made a null move.

With the above summary, both mutual checking indefinitely and mutual chasing indefinitely result in a draw. If one side chases his opponent all the time forming a forbidden sequence of plies and his opponent checks every other move with an allowed sequence of plies, the side which chases loses the game. In Chinese chess, the detailed special rules are very complicated. This summary is sufficient for the purpose of this paper.

4 Abstracting the Special Rules

A position is called a *state* in the graph representation. The state graph of Chinese chess is a finite, directed, bipartite and cyclic graph. The vertices are the states, and each directed edge indicates the corresponding ply (half-move) from one state to another, with the relationship of *parent* and *child* respectively. It is important to know the features of these indefinite move patterns in the state graph.

If we are given a series of moves, we can easily determine whether the rule of checking indefinitely is applied or not. However, to construct endgame databases, we assume both sides play perfectly and need to foresee which states must result in Red win or Black win because of the special rules. For this purpose, the term, indefinite move pattern for a special rule, needs to be defined rigorously.

Note that the algorithms of retrograde analysis construct an endgame database by first assigning initial values, or *seeds*, to terminal positions that are checkmate or stalemate. These values are then propagated to the rest of the positions. Finally, when no propagation can be done, the unknown positions are marked as draws. Given initial values for a subset of positions S in an endgame database expressed as a state graph G, let G_S be the state graph with values assigned using a typical retrograde analysis algorithm before the final phase. That is, it contains win/loss/draw information for positions that are reachable using propagation from initial values from positions in S. We call G_S the *propagated* state graph for G. A propagated database has the property that the children of every unknown vertex are either unknown, win or draw. Furthermore, an unknown vertex has at least one unknown child and has no loss child.

This paper addresses the special rule R^* of non-mutual checking indefinitely. For the convenience of discussion, we assume there is an attacking side and that the other side is defending. The attacking side tries to win the game by forcing his opponent to violate the special rules. An example of the special rule for checking indefinitely, is when the defending side always checks the attacking side forming a sequence of forbidden plies.

We define an R^* *primitive pattern* as a maximal (with respect to vertex addition) subgraph $G_S[R^*]$ of the propagated state graph G_S with the following properties.

1. The out-degree of any vertex in $G_S[R^*]$ is at least 1.
2. Each state s in $G_S[R^*]$ is marked unknown.

3. The edges from a vertex of the defending side to all of its unknown children are included in $G_S[R^*]$. A vertex of the defending side does not have children which are draws.
4. At least an edge from a vertex of the attacking side to one of its unknown children is included in $G_S[R^*]$.
5. In each vertex of the attacking side, the King of the attacking side is in check.

In an R^* primitive indefinite move pattern, the attacking side can always keep the game in the pattern, and the defending side cannot quit this pattern without losing the game. Therefore, all the defending plies are forbidden. That is, all the attacking vertices are win, and all the defending vertices are loss. Our goal is to find such patterns efficiently using an algorithm.

Note that an indefinite move pattern does not need to have a strongly connected component. That is, an indefinite move pattern may not be periodic. For example in Figure 1(a), we denote the sequence of moves Ra8-a9 Kd9-d8 Ra9-a8 Kd8-d9 by A, and Ra8-a9 Kd9-d8 Ra9-a8 Kd8-d7 Ra8-a7 Kd7-d8 Ra7-a8 Kd8-d9 by B. The game may continue as $ABAABAAAB\cdots$ without ending. The approach of periodicity does not meet our purpose. The defined primitive indefinite move patterns are the new seeds for finding the above patterns which may be non-periodic.

5 Constructing Complete Win-draw-loss Endgame Databases

We say some certain special rules are *harmless* to a certain database if the database maintains the correct win-draw-loss information when these special rules are ignored. An endgame database is called *complete* if it has correct information in concord with the special rules. In [2], a lemma is stated for the fact that special rules are harmless to the databases for endgames with exactly one armless side. Hence complete endgame databases are constructed using the classical retrograde analysis algorithm for 151 Chinese chess endgames, each of which has exactly one armless side.

In Chinese chess endgames, human experts believe the rule of checking indefinitely is much more influential to corrupt the databases than the other special rules. That is, special rules, other than checking indefinitely, need several more pieces on the board to corrupt the databases. In available literature, experts study the use of the special rules for chasing indefinitely with a total of at least three attacking pieces, four defending pieces and the two Kings. For example, in [1, page 82], the endgame with chasing indefinitely is KRGGKCPGG. Such an endgame contains at least $4.5{\cdot}10^{10}$ positions and is too large to be considered at this stage. It is possible that there are small games to demonstrate the effect of indefinite chasing. However, we cannot manually design any position to be stained by special rules other than checking indefinitely in our current study of computer endgames. Furthermore, in available literature, a meaningful

game relevant to mutual checking indefinitely contains at least five attacking pieces, two defending pieces and the two Kings. For example in [1, page 67], the endgame with mutual checking indefinitely is KRCPKRCGM. Such an endgame contains more than $2 \cdot 10^{13}$ positions and is certainly too large to be considered at this stage. This is confirmed by our experimental results, in which no instance of mutual checking indefinitely is reported during our construction of the 43 important endgames.

In contrast, the rule of non-mutual checking indefinitely can be applied to many positions with only two attacking pieces and even no defending pieces. Therefore, we focus on solving the problems caused by non-mutual checking indefinitely in this paper.

Note that the classical retrograde analysis algorithm for constructing the win-draw-loss endgame databases, without considering the special rules for Chinese chess, consists of three phases: initialization, propagation, and the final phase. For details, see [2]. Our algorithm differs from the classical algorithm through the revision of the propagation phase to the following phase:

- {∗ The revised propagation phase. ∗}
  the original propagation phase
- **repeat**
  - find primitive non-mutual checking indefinite positions
  - the propagation phase of checking indefinitely

  until there is no unknown position being changed in the last iteration.

In the following sections, we describe our algorithm for finding primitive non-mutual checking indefinitely positions.

5.1 Finding Direct Checking Indefinitely Positions

In Chinese chess, each player can win the game in two ways. One is to checkmate or stalemate the opponent. The other is to force the opponent to violate the special rules, e.g., forcing his opponent to check or to chase indefinitely. In real games of Chinese chess, a player will break the special rules only when he knows that all the other moves will lead him to lose the game. A master can foresee the conditions a great number of moves ahead whereas an amateur may only be aware of what might happen in a few moves. Therefore, a master knows more about which move will lead him to win or lose a game than an amateur does. In real games, the occurrence of checking or chasing indefinitely depends on the knowledge of the two players or the information of the chess playing system.

We call a win or loss position *relevant* to special rules if the winning side cannot win the game without these rules, i.e., he cannot checkmate or stalemate his opponent but can win the game by forcing his opponent to violate a special rule. This kind of position is special, since it is not noticed during the construction of databases by retrograde analysis ignoring the special rules. Note that in some positions, the winning side can win the game by either checkmate, stalemate his opponent, or by forcing his opponent violate the special rules.

Similarly, a position is called *relevant* to checking indefinitely if the losing side can avoid his King being captured or stalemated by checking his opponent indefinitely. Only this kind of position is relevant to special rules in our current endgame databases with the assumption that the special rules, other than checking indefinitely, are harmless to our current computer endgame study.

With a set of propagated endgame databases, our algorithm to detect checking indefinitely consists of three phases: initializing candidates, pruning candidates, and the final phase. The first phase is to find prospective states relevant to checking indefinitely. The second phase is to prune unqualified candidates by an iterative process. The final phase is to update the win-draw-loss information of the resulting states.

In the phase of initializing candidates, the database is traversed twice. The first is to initialize the *win candidates* and the second is to initialize the *loss candidates*. In the first traversal, each unknown state is marked as a win candidate if the King of the next mover is in check. Note that all the moves/edges leading the win candidates are checking moves. In the second traversal, each unknown state is marked as a loss candidate if its parent is a win candidate. The next mover of each win candidate is potential to win the game because he may be able to force the opponent to check himself in the next move. All loss candidates are potential to be loss states if its all children are win states or win candidates. Note that a win candidate may also be a loss candidate at the same time. These candidates with dual status are the ones possibly relevant to mutual checking indefinitely.

The phase of pruning candidates is composed of multiple traversals of the marked candidates until no further update on the candidates is possible. In the odd-numbered traversals, a loss candidate is pruned, i.e., no longer marked as a loss candidate, if it has one child that is neither a win candidate nor a win state. In the even traversals, a win candidate is pruned if it has no child as a loss candidate. The traversals end when no candidates can be pruned.

In the final phase, if there are candidates with dual status, they are the positions relevant to mutual checking indefinitely. However, we do not find any of them in our current experiments. Each win candidate has at least one loss candidate to keep the game in the pattern, and each loss candidate cannot quit the pattern without losing the game. Therefore, we mark the loss candidates loss and the win candidates win.

We call these newly marked win and loss states *direct* checking indefinitely since they immediately lead to checking indefinitely. They are new seeds for propagation. Thus, we have the propagation phase of checking indefinitely after that. The newly found win and loss states are called *indirect* checking indefinitely, because they may lead to checking indefinitely some plies later. For win-draw-loss information, the propagation of checking indefinitely are the same as the original propagation.

5.2 Splitting of State Graphs

In the design of endgame systems of western chess and Chinese chess, the state graph is usually split according to the total number of pieces on the board into multiple databases. In the graph representation, the vertices are the databases and the directed edges indicate the conversions of pieces. The graph representing the relations of the databases is non-cyclic; therefore, the bottom-up construction order can be clearly defined.

With the bottom-up construction order, each database has all its supporting databases before being constructed. This guarantees that the above algorithms work. In addition, the propagation might be from the states relevant to checking indefinitely in the supporting databases.

6 Infallible and Complete Endgame Databases

The Chinese chess endgame databases constructed by the algorithms in Section 5 contains complete win-draw-loss information. However, with only this win-draw-loss information available, one player may rove in the win states but never win the game by checkmate, stalemate, or forcing his opponent to check indefinitely. In this section, we show our approach to construct the infallible and complete endgame databases.

6.1 Position Values

Our algorithm for constructing endgame databases in distance-to-mate and distance-to-conversion metrics without taking care of special rules are discussed in [2]. The algorithm follows more or less a standard retrograde analysis algorithm devised for western chess. We use 0 to represent draw. We use an odd number i to denote win in i plies, and an even number j to denote loss in j plies. We now describe our scheme using positions that are marked with win or loss using the rule of non-mutual checking indefinitely.

To achieve infallibility, we set the position value of the win states of direct checking indefinitely to a special value $WinByDirectCheckingIndefinitely$. It is 255 in our implementation. And we set the position value of the loss states of direct checking indefinitely to a special value $LossByDirectCheckingIndefinitely$. It is 254 in our implementation.

The propagation phase of checking indefinitely starts after the phase of detecting checking indefinitely. However, it is a little different from the propagation of the states in distance-to-mate or distance-to-conversion metrics. Here we introduce the concept of *distance-to-check*, which is the distance to the win states by direct checking indefinitely. In our implementation, the position value in the distance-to-check metric is 255 minus the distance to win states of direct checking indefinitely. Each position in distance-to-check metric propagates its value minus 1 to all its parents, instead of plus 1 in distance-to-mate metric. For instance, the position value 250 in distance-to-check metric means 5 plies to the win state

of direct checking indefinitely. Each loss state of direct checking indefinitely has position value 254. Note that in our implementation, all positive even position values indicate loss states and all odd position values represent win states in our indexing scheme in any metric. The only exception is 0 for draw positions.

In real games of Chinese chess, there is no preference in winning the game by using checkmate or special rules. In the design of Chinese chess endgame databases in concord with checking indefinitely, we assume that loss by checkmate or stalemate is worse than loss by breaking the special rules. That is, if one can win the game without using special rules, he will prefer this way of ending the game, i.e., checkmate or stalemate. Similarly, if one is losing the game, then losing by special rules is preferred against losing by checkmate. The assumption used here is reasonable since the use of special rules requires more knowledge.

The current position values are updated if the propagated values are better. By the zero-sum property, our assumption implies that win by checkmate or stalemate is better than win by forcing the opponent to check or chase indefinitely. Therefore, win position values of distance-to-mate are better than those of distance-to-check. Loss position values of distance-to-mate are better than those of distance-to-check. This assumption helps for propagation to update the current position values of their parents.

As mentioned in Section 5.1, some new states of direct checking may be detected when more information regarding the states is gained. Therefore, as shown in the algorithm in Section 5, the phases of detecting and propagation of checking indefinitely are repeated as multiple iterations until no position information can be gained. We call the positions with information found in the first iteration, *first* order checking indefinitely. Similarly, those with information found in the kth iteration, are called kth order checking indefinitely. In our implementation, the position value of the win states of the second order direct checking indefinitely is the largest odd integer less than minimum position value of the first order checking indefinitely, and the position value of the loss states of the second order direct checking indefinitely is 1 less than that of the win states. The propagation of second order checking indefinitely remains the same without changing. Similarly, third order checking indefinitely and its index scheme can be clearly defined, and so are the fourth order, fifth order, etc. The process continues until no more states relevant to checking or mutual checking indefinitely can be found. Because of technical problems in encoding the position values for multiple order checking indefinitely in one byte, we currently do not consider second or higher order checking indefinitely.

As state space is spilt into multiple databases, each resulting database has its own indices of checking indefinitely of different orders, except for the first order. Our data structure has a separated table to record these indices.

6.2 Infallibility of Our Databases

With the rule of checking indefinitely, the concept of shortest win can hardly be defined. Hence we concentrate on infallibility. With our databases, the infallibility can be achieved as follows:

1. Starting from a draw state, we may move to any child draw state. Note that this strategy never results in loss because of checking indefinitely.
2. Starting from a win state with position value in distance-to-mate metric, we can win the game by leading the game to a checkmate or stalemate state by playing the best possible next move.
3. Starting from a win state with position value in distance-to-check metric, we can lead the game to one of the following states and remain in win status:
   - a state of indirect checking indefinitely with the same order and shorter distance-to-check.
   - a state of direct checking indefinitely, with the same order.
   - a state relevant to checking indefinitely with higher order, i.e., smaller order index.
   - a state with position value in distance-to-mate, i.e. irrelevant to checking indefinitely.

   Because the order and the distances cannot be endlessly larger, we can eventually win the game by checkmate, stalemate, or the rule of checking indefinitely.

6.3 Modifications for Distance-to-conversion

When checking indefinitely is taken into account, a database may have some positions in distance-to-check metric and the others in distance-to-conversion metric. We call it a database in *distance-to-mate/check* metric. When the position values in distance-to-mate and distance-to-check metrics are both encoded in 1 byte, the chance of overflow in representing the values is greater than those in pure distance-to-mate. Distance-to-conversion is introduced to reduce this potential problem. In this case, the propagated value of any win state in supporting databases is the smallest loss position value, and that of a loss state in supporting databases is the smallest win position value. The propagated value of a draw state remains a draw. Similarly, if a database has some positions in distance-to-conversion metric and the others in distance-to-check metric, we call it *distance-to-conversion/check* metric.

Note that the databases in distance-to-conversion/check metrics are still infallible. Starting from a win state in distance-to-conversion metric, we can always lead the game to some supporting database, or win by checkmate or stalemate. Starting from a win state in distance-to-check metric, we can always lead the game to the four types of states listed in Section 6.2 or a state in some supporting database. Because the order and the distances cannot be endlessly larger and the graph representation of databases are finite and acyclic, we can eventually win the game.

6.4 Verifying the Databases

We introduced new position values, other than those propagated from checkmate or stalemate, in our databases. We will assign numerical values to these position values such that they obey the following rules:

1. Win position values are better than draw position values, which is better than loss position values.
2. Win position values in distance-to-mate or distance-to-conversion metric are better than those in distance-to-check metric, whereas loss position values in distance-to-check metric are better than those in distance-to-mate or distance-to-conversion metric.
3. For win position values in distance-to-mate (respectively, distance-to-conversion) metric, the smaller the value, the shorter the distance to checkmate or stalemate (respectively, to reaching the supporting databases), and hence the better. For loss position values in distance-to-mate (respectively, distance-to-conversion) metric, the larger the value, the longer the distance to checkmate or stalemate (respectively, to reaching the supporting databases), and hence the better.
4. For the win states in distance-to-check metric, the higher the order, the better. For the loss states in distance-to-check metric, the lower the order, the better.
5. For the win states in distance-to-check metric with the same order, the larger the position value, the shorter the distance to the direct checking indefinitely, and hence the better. For the loss states in distance-to-check metric with the same order, the smaller the position value, the longer the distance to direct checking indefinitely, and hence the better.

To verify the correctness of a database with correct supporting databases, we just have to check if the position value is the best of the propagated values from all the children. The only exception is the win states of direct checking indefinitely. Each win state of direct checking indefinitely has at least one loss child of direct checking indefinitely and has no loss children in distance-to-check metric in higher order or in distance-to-mate metric. With the assumption that everything goes well in the phases of finding direct checking indefinitely, the verification is sufficient to prove the correctness and uniqueness of our databases. The verification program can detect program bugs or hardware errors which occur during the propagation.

7 Concluding Remarks

Retrograde analysis has been widely used and successfully applied in western chess. In Chinese chess, its application is confined because of these special rules. We successfully developed an algorithm to construct the endgame databases in concord with the rules of checking indefinitely. We list the statistics of 43 important endgame databases in the appendix. Our current code uses a large amount of main memory. To save memory usage, we need to encode the position values in one byte. Thus we currently do not consider second or higher order checking indefinitely, though our algorithm can handle them easily. We are working now in extending our code to deal with second and higher order checking indefinitely.

In the future, we plan to use knowledgeable encoding and querying of endgame databases discussed in [3] as an approach to extract win-draw-loss information

and condense it in physical memory to improve the ability of present Chinese chess programs. When this has been realized, the level of Chinese chess computer systems might be close to that of human champions. The process to construct endgame databases which are too large to be constructed in present physical memory, is another subject. With successful development and implementation, some open problems in Chinese chess endgames, such as KRNMKRMM endgame, may be able to be solved. It will fertilize the field of Chinese chess endgame studies.

References

1. China Xiangqi Association. *The Playing Rules of Chinese Chess*. Shanghai Lexicon Publishing Company, 1999. In Chinese.
2. H.-r. Fang, T.-s. Hsu, and S.-c. Hsu. Construction of Chinese chess endgame databases by retrograde analysis. In T. Marsland and I. Frank, editors, *Lecture Notes in Computer Science 2063: Proceedings of the 2nd International Conference on Computers and Games*, pages 96–114. Springer-Verlag, New York, NY, 2000.
3. E.A. Heinz. Knowledgeable encoding and querying of endgame databases. *ICCA Journal*, 22(2):81–97, 1999.
4. K. Thompson. 6-piece endgames. *ICCA Journal*, 19(4):215–226, 1996.
5. H.J. van den Herik, Jos W.H.M. Uiterwijk, and J. van Rijswijck. Games solved: Now and in the future. *Artificial Intelligence*, 134:277–311, 2002.
6. R. Wu and D.F. Beal. Fast, memory-efficient retrograde algorithms. *ICCA Journal*, 24(3):147–159, 2001.

Appendix: Statistics of 43 Endgame Databases

In this paper, we focus on the databases which are potentially stained by the special rules of indefinite move patterns. Thus, we list only the statistics for the databases satisfying the following requirements.

1. Both sides have attacking pieces.
2. There is an attacking piece other than the pawn.
3. If there are only two attacking pieces, then neither of them can be a pawn.

As we mentioned in [2], if only one side has attacking pieces, the special rules of indefinite move patterns do not stain the databases. Therefore, in requirement 1, we list only the statistics for databases in which both sides have attacking pieces. The rule of checking indefinitely stains the positions in which the defending side cannot avoid his King being captured without checking the attacking side. This means the defending side must be powerful enough to check the attacking side indefinitely. In general, Pawn is considered as the weakest attacking piece in Chinese chess. Therefore, we add requirements 2 and 3. Experts believe that endgames not satisfying these requirements can hardly be stained by special rules.

Table 1 gives statistics for 43 endgame databases satisfying the above requirements. For each database, we report its number of legal positions, number

of draw positions, maximal distances measured in plies, and distance-to-check rate for each database. The distance-to-check rate is the number of positions in distance-to-check metric divided by the number of legal positions. If no positions relevant to checking indefinitely exist, the maximal distance-to-check and distance-to-check rate are marked as *N/A*.

So far, we do not find any database with the positions of first order mutual checking indefinitely. Currently we only experiment on first order checking indefinitely. Therefore, the maximal distance-to-check value is the maximal distance measured in plies to the win states of first order direct checking indefinitely.

We construct a database in distance-to-conversion/check metric only if the position value cannot be encoded in 1 byte or it has some supporting databases in distance-to-conversion/check metric. Otherwise, the database is constructed in distance-to-mate/check metric. In our current experiments, the only two databases which have all supporting databases in distance-to-mate/check metric and cannot be indexed in 1 byte are KRKNGMM and KRKNCG. The databases above them are also constructed in distance-to-conversion/check metric as well, e.g., KRKNGGMM. In Table 1, if a database is in distance-to-conversion/check, the database name and the maximal distance-to-mate or -conversion has an ending character '*'. The only two databases in distance-to-conversion/check metric listed in the table are KRKNGMM and KRKNGGMM.

In distance-to-conversion/check metric, all win positions in the supporting databases propagate the same value to the positions in the constructed database, no matter they are distance-to-mate, distance-to-conversion, or distance-to-check. The loss positions perform the same propagation. Therefore, the maximal distance-to-check value of a database may be different in distance-to-mate/check and distance-to-conversion/check metrics.

If the special rules are ignored during construction, the database may be stained by these rules. One thing we are interested in is the *stained rate*, which is defined as the number of positions relevant to checking indefinitely divided by the number of legal positions. For databases in distance-to-mate/check, it is the same as distance-to-check rate. The statistics in Table 1 excludes second or higher order checking indefinitely. For endgames with positions of second or higher order checking indefinitely, the listed distance-to-check rate is lower than the stained rate. For databases in distance-to-conversion/check metric, some positions in distance-to-conversion may be relevant to checking indefinitely, because the positions in distance-to-check metric in the supporting databases do not propagate the checking indefinitely information during the construction. Thus, the distance-to-check rate is usually lower than the stained rate.

Table 1. Statistics of 43 endgame databases.

Database Name	*# of Legal Positions*	*# of Draw Positions*	*Maximal Distance-to-*		*Distance-to-Check Rate*
			Mate/Conv.	*Check*	
KRNKRGG	251181481	140966299	77	58	2.59%
KRNKRG	138660209	62141914	79	62	7.22%
KRNKRM	204263530	96645975	70	61	5.18%
KRNKR	30118362	9279051	79	59	13.05%
KRPKRG	84330363	48354000	32	24	1.17%
KRPKRM	124050578	68111972	36	33	1.48%
KRPKR	18443469	9383254	30	21	2.48%
KRKNN	16300026	4672873	88	53	1.59%
KRKNC	33568194	757635	116	73	1.84%
KRKCC	17300976	1409308	82	N/A	N/A
KRKNPGG	168307887	2433079	152	81	1.02%
KRKNPG	92456806	115809	60	14	0.64%
KRKNPM	136200539	470847	72	19	0.66%
KRKNP	20011890	20324	37	14	0.73%
KRKPPP	96366510	1278141	54	15	0.46%
KRKPPGG	52598998	947673	86	10	0.04%
KRKPPMM	122221940	2214611	94	12	0.06%
KRKPPG	28498574	53302	44	10	0.06%
KRKPPM	41658907	107788	44	10	0.08%
KRKPP	6084903	8238	34	10	0.10%
KNPKN	21682338	5008874	154	43	2.01%
KNPKCG	100076040	40958639	138	19	0.16%
KNPKCM	149719630	67594106	154	36	0.39%
KNPKC	22364304	6387677	46	N/A	N/A
KCPKC	22956705	20094132	41	N/A	N/A
KRKRGG	2997932	1840672	30	11	0.09%
KRKRG	1628603	966498	19	11	0.19%
KRKRM	2389472	1370793	19	N/A	N/A
KRKR	348210	193950	10	N/A	N/A
KRKNGGMM*	63684381	33638427	64*	23	0.24%
KRKNGGM	21879507	720797	104	6	0.33%
KRKNGMM*	35195142	4726833	82*	39	0.52%
KRKNGG	3221138	6578	74	3	0.37%
KRKNGM	12032732	27464	50	3	0.35%
KRKNMM	7654095	135941	111	75	17.31%
KRKNG	1762807	3275	40	1	0.39%
KRKNM	2605497	5200	34	1	0.37%
KRKN	380325	609	26	1	0.42%
KRKCGG	3322727	1379102	48	N/A	N/A
KRKCMM	7913097	2969808	64	N/A	N/A
KRKCG	1820350	2462	36	3	0.01%
KRKCM	2694400	91748	26	N/A	N/A
KRKC	393327	7479	16	N/A	N/A