A New Representation of Chinese Chess Board

Conference Paper · August 2009
DOI: 10.1109/CADCG.2009.5246835 · Source: DBLP

3 authors, including:

Yong-Jin Liu
Tsinghua University
101 PUBLICATIONS **1,174** CITATIONS
SEE PROFILE

A New Representation of Chinese Chess Board

En-Lin Yang
Fundamental Science of Mathematics and Physics
Tsinghua University
Beijing, P. R. China

Yong-Jin Liu, Ling-Xi Xie
Department of Computer Science and Technology
Tsinghua University
Beijing, P. R. China

Abstract

Computer-games research has opened a door to a multi-discipline domain across artificial intelligence, computer-aided design and computer graphics. Chinese chess is one of the most popular board games worldwide and many researches on move generation, position evaluation, searching, opening and endgame play, have been developed. However, few work has been done in the basic representation of Chinese chess board. Currently the representation of 10×9 *integer array is widely used in Chinese chess programming. In this paper, based on concrete theoretical analysis, a new presentation of Chinese chess board with 6 long integers is proposed. Compared to the traditional* 10×9 *int-array representation that is often redundant in endgames, the cost of storage with the new representation is greatly reduced by an order of magnitude.*

1. Introduction

Computer-games research has opened a door to a multi-discipline domain across artificial intelligence, computer-aided design and computer graphics. In 1950 Shannon published the first paper on computer chess [16]. Nowadays, game-playing programs have become one of major research areas in artificial intelligence [15].

In two-player and zero-sum games with perfect information, Checkers, Othello, Chess, Chinese chess and go are the most popular board games worldwide. State-of-the-art computer level of Checkers [14] and Othello [5] has outperform the world champion of human tournaments. Computer programming *Deep Blue* has the comparative ability to the world champion [13]. However, the developments of computer Chinese chess and Go are still at the immature stage in between of amateur and grandmaster [8].

The complexity of Chinese chess is between the chess and Go. After Deep Blue's victory over Kasparov, Chinese chess is predicted to be the next hope at which computer program will defeat the top players of human being. Towards this goal, many researches of Chinese chess on move generation [7], position evaluation [3], searching [1], [12], [12], opening [9] and endgame play [4], [6], [10], have been developed. However, to the best knowledge of the authors, there is few work done for the representation of Chinese chess board. The only available representation in most open source of computer engine of Chinese chess adopts 10×9 integer array [2]. The simplest form, although it is very intuitive and easy for programming, leads to a heavy redundance for chessboard representation and storage, especially in endgames.

In this paper, based on concrete theoretical analysis, a new presentation of Chinese chess board with 6 long integers is proposed. Compared to the traditional 10×9 int-array representation, the cost of storage with the new representation is greatly reduced by an order of magnitude. To be compatible with most state-of-the-art game engines of Chinese chess, we alo provide the source code that transforms between the new representation and the traditional 10×9-int-array representation.

2. Background

The set of Chinese chess includes a board and 32 pieces for two players. The two players in opposite sides are called Red and Black. Each side has one King, two Mandarins, two Elephant, two Rooks, two Horses, two Cannons and five Soldiers. As an intuitive way, Chinese chess is usually represented by 10×9 integer arrays on computers, which is convenient for programmers to handle problems such as judging whether a certain move is allowable. However, since there are many unoccupied positions in the chess board, the 10×9 int-array representation turns out to be a waste of storage in some sense, especially in the final stage of a chess game.

A new representation of Chinese chessboard is exploited in this paper, with the purpose that it can reduce the unnecessary large cost of storage as well as speeding up the process of match Chess endgame database, without sacrificing position-searching speed and the ease of programming.

3. Theoretical estimate

We denote chess states by $m_1, m_2, m_3, \cdots$, and denote the set of all possible chess states by M. Then we come to a rough estimate of M:

$$\#M \leq 9^2 \times 5^4 \times 7^4 \times 47^{10} \times 90^{12} \leq 2 \times 10^{48}$$

978-1-4244-3701-6/09/$25.00 ©2009 IEEE

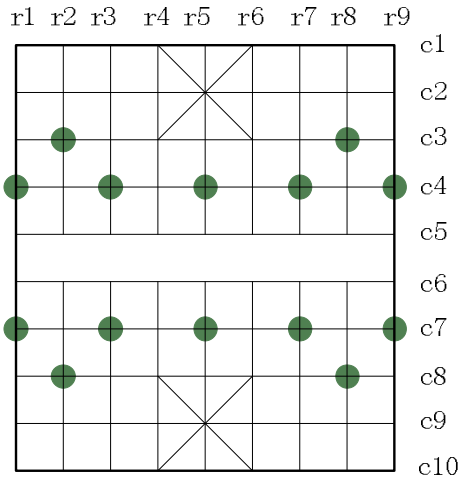

Figure 1. A Chinese chess board represented in a 10×9 integer array.

Obviously, if we wish to match the integers and chess states one to one, the largest among all used integers must have at least 48 digits. Now the question is: how many integers less than 2^{32} is needed to represent a 48-digit integer? Consider the following equation:

$$a = \min\{n|2^{32n} \geq 2 \times 10^{48}\}$$

It is easy to find out that $a \geq 5.0141$. So we can divide a 48-digit integer into 5 integers, each of which is no longer than 10 digits. It is true that some of these integers may turn out to be larger than 2^{32}, so we can only assert that every chess state can be uniquely matched to an integer array of length n, where n is no less than 5, and each integer in the array in no more than $2^{32} = 4294967296$ (10 digits).

4. A new representation

In this paper, we propose a new representation of Chinese chessboard using 6 long integers, which can greatly reduce the storage space to large amounts, especially in the endgames of chess play.

4.1. Basic idea

The position of King has only 10 possibilities (9 positions on the chessboard plus the one to be eaten). For Mandarins, if there are two Mandarins, we have 10 cases; if there is only one, we have 5 cases, and plus the situation with no Mandarins at all. Totally there are 16 cases. Similarly, there are 29 cases for the elephant. Each of Rooks, Horse and Cannons has 4096 cases. The situation is somewhat difficult when we come to the Soldiers, because there are much more Soldiers compared to other kinds of chess pieces and its law of movement is not consistent all the time. In consideration of the complexity cost of storage and transformation compatible with the traditional 10×9-int-array representation, we simply treat these Soldiers with 475 cases.

Second is about the idea of compression in storage. If the King has 10 possibilities and the Mandarins have 16 possibilities, we can use an integer (0-159) to represent them. The accurate description of the method is as follows: if there is an integer 84, then divide it by 10. The quotient is 8 and the residue is 4: this means that the bishop is now in the (8+1)th possible case, and the king is in the (4+1)th possible case. The method of encoding/decoding is in the similar spirit.

Finally we need some initialization. We need to decide exactly what a certain case for a chess piece is. For example, we can let the first case of the King been eaten, the second is on the position (1,4), and so on, and the 10th is on the position (3,6). In order to realize this, we need to have some large initial arrays, which are widely used in the process of compression. With proper initialization, we can speed up our program.

Having the above preparation at hand, efficient storage is now possible. Based on the above ideas, the King has 10 cases, the Mandarins has 16 cases and the Elephant has 29 cases. So there are totally 4640 cases and ultimately 21529600 cases for both sides. Then we can use a long integer to represent the King, the Mandarins and the Elephant for both sides. Similarly, we can use three long integers to represent Rooks, Horse and Cannons for both sides. At last, another 2 long integers are used to represent the Soldiers.

To summarize, we use 6 long integers to represent all the chess pieces in all states in Chinese chess.

4.2. Detailed parse process

In the beginning, it is necessary to explain some notations. The chess state is treated as a matrix. The first element in the top left corner $(0, 0)$ is denoted as 01 or 1, and $(x, y - 1)$th position is denoted as xy (we call these by coordinate number). We have special meaning for 00 or 0, which means the absence of some chess pieces. We may write two or more coordinates successively, for example 1234 means positions $(1, 1)$ and $(3, 3)$. Then we can follow the steps below to get the 6 integers which represent one unique chess state.

4.2.1. Chess pieces with limited cases. The pieces includes Elephants, Bishops and King. Since there are many cases of Soldiers, we treat them as "unlimited cases".

There will be 10 cases for the King, 16 cases for the Mandarins, 29 cases for the Elephant. So there are totally 4640 possibilities and ultimately 21529600 possibilities for both sides. We can use a long integer to represent the King, the Mandarin and the Elephant for both sides. This integer

is achieved in the following way, at first we arrange 6 arrays to mark the position of the chessmen:

Black King:

$$B\_KING[\,] = \{0, 04, 05, 06, 14, 15, 16, 24, 25, 26\}$$

$B\_KING[0]$ means that the black King does not exist.

Black Mandarins:

$$B\_Mandarin[\,] = \{0, 04, 06, 15, 24, 26, 0406, 0415, 0424, 0426, 0615, 0624, 0626, 1524, 1526, 2426\}$$

Black Elephant:

$$B\_ELEPHANT[\,] = \{0, 03, 07, 21, 25, 29, 43, 47, 0307, 0321, 0325, 0329, 0343, 0347, 0721, 0725, 0729, 0743, 0747, 2125, 2129, 2143, 2147, 2529, 2543, 2547, 2943, 2947, 4347\}$$

Red King:

$$R\_KING[\,] = \{0, 94, 95, 96, 84, 85, 86, 74, 75, 76\}$$

Red Mandarin:

$$R\_Mandarin[\,] = \{0, 94, 96, 85, 74, 76, 9496, 9485, 9474, 9476, 9685, 9674, 9676, 8574, 8576, 7476\}$$

Red Elephant:

$$R\_ELEPHANT[\,] = \{0, 93, 97, 71, 75, 79, 53, 57, 9397, 9371, 9375, 9379, 9353, 9357, 9771, 9775, 9779, 9753, 9757, 7175, 7179, 7153, 7157, 7579, 7553, 7557, 7953, 7957, 4357\}$$

We can easily decide the positions of the King, the Mandarin and the Elephant based on these arrays. For example, if the Black King is in the 7th case, we need only to check that $B\_KING[7-1] = 16$ in order to inquire that the position of the king is $(1, 5)$.

We use a 10-digit long integer to represent a typical case for the King, the Mandarin and the elephant for both sides at the same time. In order to explain it explicitly, we use "abcdefghij" to denote this 10-digit integer (including the case a=0).

Integer "ab" is used to note where the red Elephant is, and "cd" is used to note where the Black Elephant is. For example, if cd=09, it is to say that the Black Elephant is in the 9th case, that is, $B\_Elephant[09-1] = 307$, which means there are two Black Elephants, one in position$(0, 2)$ and the other in $(0, 6)$.

Similarly, integer "ef" is used to note where the red bishop is and "gh" is used to note where the black Mandarin is. Finally, i and j will be used to note the positions of the Kings, i for Red King, j for Black King.

4.2.2. Chess pieces with unlimited cases. The pieces includes Rooks, Horse, Cannons and Soldiers. The red side of the traveling Rooks has 4096 cases, so we use a 4096-unit-long array which can speed up our program significantly. To this end, we need only to use coordinates to represent them directly, there is no need to use additional arrays.

Here is an example: we use "abcdefgh" to note the coordinates of the Rooks for both sides. abcd is used to note where the red Rook are (abcd is a 4-digit integer which range from 0 to 9998). The coordinate of one red Rook is ab and that of the other is cd. The Horses and Cannons are treated in similar ways.

For Soldiers, we use two 10-digit integer to represent them. For example, one is abcdefghij, representing the case of red Soldiers: ab is the coordinate of the first Soldier, cd ,ef ,gh, ij are those of the remaining four Soldiers. The black side is handled in a same way.

4.2.3. Range adaptation. We use array $B[6]$ to note the six integers used in the new representation. The last five ones are used to represent the Rooks, Horse and Cannons, red and black's Soldiers,respectively. In order to restrict the values of these integers within the range of $[-2147483647, 2147483647]$, we need a transformation as follows (note that $B[1], B[2], B[3]$ each will have 8 digit at most):

$$B[1] = B[0]\%10 + B[1] \times 10$$
$$B[2] = B[4]\%10 + B[2] \times 10$$
$$B[3] = B[5]\%10 + B[3] \times 10$$

5. Experiments

We have the following clarification for our method. The corresponding array to the illustration in Fig. 2 is:

$$B[6] = \{90907073, 919901093, 929802089, 727822289, 616365676, 313335373\}$$

We only mention $B[0]$ and $B[1]$ here. Others can be handled in a similar way. $B[0] \times 10 + B[1]\%10 = 909070733$. The first number 9 in the integer 909070733 means that the red Elephant in on the 9th case. That is, $R\_ELEPHANT[9-1] = 9397$: since $9397 > 99$, it means that the red side has two Elephants at positions of $(9, 2)$ and $(9, 6)$. The second together with the third digit, 09, means that the Black Elephant in on the 9th case. The forth and fifth digit, 07, means that the red Mandarins is in the 7th case. Other digits in 909070733 can be read in a similar way. Since $B[1]/10 = 91990109$, the first 4 digits, 9199, indicate the coordinate of the red Rooks: since $9199 > 99$, it means that the red side has two Rooks, at the positions of $(9, 0)$ and $(9, 8)$. The latter 4 digits, 9199, indicate the coordinate of the black rook: since $9199 > 99$, it means the black side has two Rooks, at the positions of $(0, 0)$ and $(0, 8)$.

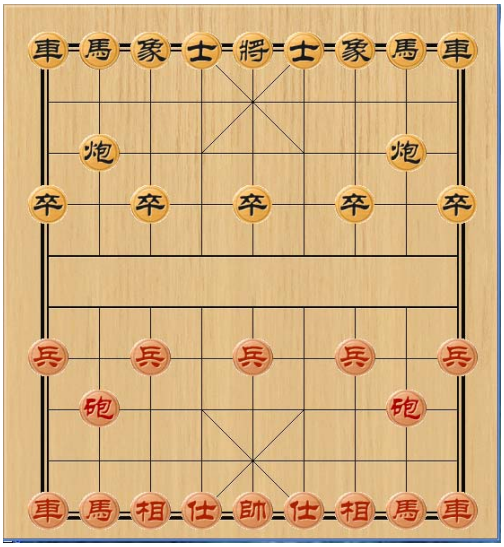

Figure 2. An illustration of a Chinese chess board.

6. Conclusions

In this paper, a new representation of Chinese chess board is presented. Based on concrete theoretical analysis, 6 long integer representation is used and shown to overmatch the traditional representation of 10×9 integer arrays. A prototype computer engine with the proposed representation is complete to demonstrate the advantage of the new chessboard representation. Finally, To be compatible with most state-of-the-art game engines of Chinese chess, the source code that transforms between the new representation and the 10×9-int-array representation is also generated.

Acknowledgement

The work was supported by the Tsinghua SRT Project (Project Number 082T0211) and Tsinghua Basic Research Foundation.

References

[1] L. Allis. Searching for solutions in games and artificial intelligence. PhD Thesis, University of Limburg, Netherlands, 1994.

[2] An Introduction to Chinese Chess. http://skookumpete.com/chess_intro.htm.

[3] I. Chang. Ten strategies for playing Chinese chess. Journal of Chinese Chess, 1985.

[4] B. Chen, P. Liu, S. Hsu, T. Hsu. Knowledge inferencing on Chinese chess endgames. CG 2008, LNCS 5131, pp. 180-191.

[5] S. Chong, M. Tan, J. White. Observing the evloution of neural networks learning to play the game of Othello. IEEE Transactions on Evolutionary Computation, 9(3):240-251, 2005.

[6] H. Fang, T. Hsu, S. Hsu. Construction of Chinese chess endgame databases by retrograde analysis. LNCS 2063, pp. 96-114, 2001.

[7] H. Fang, T. Hsu, S. Hsu. Indifinite sequence of moves in Chinese chess endgames. CG 2002, LNCS 2883, pp. 264-279, 2003.

[8] H. Herik, J. Uiterwijk, J. Rijswijck. Games solved: now and in the future. Artificial Intelligence, 134: 277-311, 2002.

[9] S. Hsu, K. Tsao. Design and implementation of an opening game knowledge-base system for computer Chinese chess. Bulletin of the College of Engineering, NTU, No. 53, pp. 75-86 (in Chinese).

[10] T. Hsu, P. Liu. Verification of endgame databases. ICGA Journal, 25(3):132-144, 2002.

[11] H. Kaindl, R. Shams, H. Horacek. Minimax search algorithms with and without aspiration windows. IEEE Trans. Pattern Analysis and Machine Intelligence, 13(12):1225-1235, 1991.

[12] D. Knuth, R. Moore. An analysis of alpha-beta pruning. Artificial Intellegence, Vol. 6, pp. 293-326.

[13] M. Newborn. Deep Blue's contribution to AI. Annals of Mathematics and Artificial Intelligence 28: 27-30, 2000.

[14] J. Schaeffer, N. Burch, Y. Bjornsson, et al. Checkers is solved. Science 317; 1518-1522, 2007.

[15] J. Schaeffer, H. Herik. Games, computers, and artificial intelligence. Artificial Intelligence, 134: 1-7, 2002.

[16] C. Shannon. Programming a computer for playing chess. Philosophical Magazine, 41: 256-275, 1950.