The Bitboard Design and Bitwise Computing in Conect6

Shi-Jim Yen[1], Jung-Kuei Yang[2]
[1]Department of Computer Science and Information Engineering, National Dong Hwa University, Hualien, Taiwan. sjyen@mail.ndhu.edu.tw
[2]Department of Applied Foreign Languages, Lan Yang Institute of Technology, I Lan, Taiwan. kueiy@mail.fit.edu.tw

Abstract: *Connect6 is becoming more and more populous in those years. It possesses three characteristics that is a fair and highly complex game with simple rules. Searching is an important method that is used in AI to reveal human wisdom. Efficiency is the key point to Connect6 except for searching theorems. Most related researches use bits to code the board states and relative bitwise computing to model the real problems. In this paper, we use the concept, bitboard, to parse the data structure in Connect6 and analysis it's bitwise computing to accelerate the change of board states and to gain information. The result shows that those methods are efficient and can be used to improve the search performance of a Connect6 program.*

Keywords: Bitboard, Bitwise Computing, Connect6

1. Introduction to Connect6

1.1. The progress of Connect6

Since Wu (Wu, 2005) offered online board games like k-in-a-row or Connect(m,n,k,p,q) in 2005, Connect(19,19,6,2,1) or Connect6 which is derived from Komoku has been a very popular topic to research. Connect6 is a fair and highly complex game with simple rules. It is not a new kind of game, but its characteristics offer a new way for research in the future.

1.2. The game rules

The rule of Connect6 is very simple. It is a game played by two people, one holds black stones and the other white stones. Both of them take turns putting their stones on the intersections formed by the lines of grids on the 19×19 board. Usually, first move will be played by black side to put one stone on the empty intersection, and then two sides take turns putting two stones on the empty intersections.

There isn't the design that stones will be killed and removed, so the state of an intersection will not be changed after stones are put on it. That means, during the whole game, the state of the board won't return to any state happened before. The one who first gets six or more consecutive stones (horizontally, vertically or diagonally) of her/his own wins. When all intersections on the board are placed without connecting six, the game draws.

2. Basic analysis of Board

Bitboard is a way to model states by binary coding. It transforms states into bits, and uses bitwise computing to accelerate the speed of getting information and improve the efficiency of searching (Pablo, 2006). The most widely used Bitborad is the design of Chess. It uses 64 bits to represent 64 positions on the board, and employs all kinds of search by using computers' great function of bitwise operators. For the efficiency of search, it has nice performance.

This research analyzes the board based on the notion above, and use Connect6 as a sample to develop a data structure based on Bitboard and programs of other related algorithms. The result of the analysis is suitable for any boards of any sizes, so a m×n board is the object to be analyzed.

2.1. Analyzing the board

The number of vertical grids is m, and the horizontal is n. the squares which are able to be put with stones are named Intersection or Cell. The number of the intersection depends on the number of grids of m and n, so the number of intersection is m×n. Ways to label intersections can be divided into coordinate one and the one of index, and their illustrations are as follows:

2.1.1. The coordinates of intersections

The coordinates of intersections are constructed by horizontal axis X and vertical axis Y; both of them start from the upper corner on the slash west of the board. The coordinates of the intersections will be represented by $I_{(x,y)}$ where x=0..m-1; y=0..n-1, and figure 1 shows the way the coordinates are numbered in a 9×9 board.

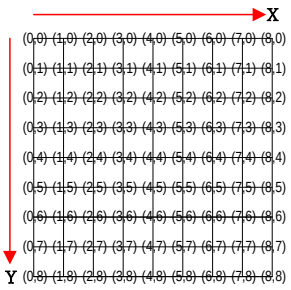

Figure 1, the numbering of coordinates in 9×9 board

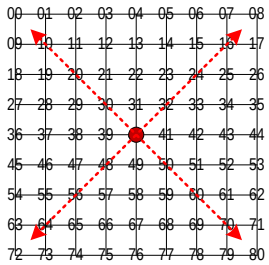

Figure 2, the direction and index of 9×9 board

2.1.2. The indexes of intersections

Starting from 0, the upper corner on the slash west of the board, and numbering from up to down, the horizontal line is prior to the vertical line. Figure 2 shows the way intersections are numbered by index.

2.2. Directional relation

There is interrelation between intersections when lining. We name this interrelation as the directional relation of intersections, R_D to represent it. This relation depends on the arrangement of position forming four directional relations: Horizontal, Vertical, Slash West, and Slash East.

Connected Cell can be formed by the four directional relations between each intersection like figure 2. I_{40} is able to connect with Connected Cell I_{20} and I_{60} in the same direction. Because of the relation between Connected Cell and directional position, $R_{D=V,X}$ is used to represent the Connected Cell which has the same x on the X axis vertically and x is 0..m-1.

Therefore, the relations between Connected Cells in four directions are displayed as follows:

- Vertical： $R_{D=V,x}=\{I_{(x,y)}|same\ x\}$
- Horizontal： $R_{D=H,y}=\{I_{(x,y)}|same\ y\}$
- Slash West： $R_{D=SW,x-y}=\{I_{(x,y)}|same\ (x-y)\}$
- Slash East： $R_{D=SE,x+y}=\{I_{(x,y)}|same\ (x+y)\}$

We can thus assume that in the vertical directional intersections with the same x coordinates are Connected Cells, and in the same way the notion of other Connected Cells can be proved.

2.3. Line

According to the four directional relations, all intersections of Connected Cells will form a collection, called Line (L for short). Because Line is related with directions, we use $L_{D=V,x}$ to describe the Lines with the same x coordinates in axis X in the vertical direction. Other Lines in other directions will be proved in the same way.

2.3.1. The definition of Line in four directions on the board

- Vertical： $L_{D=V,x}=\{R_{D=V,x}|x=0..m-1,\forall y\}$
- Horizontal： $L_{D=H,y}=\{R_{D=H,y}|y=0..n-1,\forall x\}$
- Slash West： $L_{D=SW,x-y}=\{R_{D=SW,x-y}|x-y=-(n-1)..(m-1),\forall x,y\}$
- Slash East： $L_{D=SE,x+y}=\{R_{D=SE,x+y}|x+y=0..m+n-2,\forall x,y\}$

Every Line has two end-points; we call the intersection with smaller index the Start of the Line, and, on the contrary, the bigger index will be the End.

2.3.2. The length and the amount of Line on the board

The length of a Line includes the amount of intersections form Start to End (including the two end-points). We use $|L_{D=V}|=n$ to show the length of a vertical Line is n. Because the vertical Line and the horizontal Line have the same length, we can ignore the number of axis x. The number and length of Lines in every direction on the board can be obtained through the number of grids m, n. The formula is as follows:

- Vertical: Numbers of lines is m, the length is $|L_{D=V}|=n$
- Horizontal : Numbers of lines is n, the length is $|L_{D=H}|=m$
- Slash West: Numbers of lines is m+n-1, the length is

$$|L_{D=SW}|=\begin{cases}Min\{m,n\},\ if\ m\neq n\wedge Max\{m,n\}-|x-y|>Min\{m,n\}\\ m-|x-y|,\ \text{others}\end{cases}$$

- Slash East: Numbers of lines is m+n-1, the length is

$$|L_{D=SE}|=\begin{cases}Min\{m,n\},\ if\ m\neq n\wedge\\ \qquad Max\{m,n\}-|(Max\{m,n\}-1)-(x-y)|>Min\{m,n\}\\ m-|(m-1)-(x+y)|,\ others\end{cases}$$

The numbers of Lines depend on the numbers of grids m, n, and k of the game. k means the least consecutive stones needed to win the game in Connect-k. Thus, according to the former notion of calculating the Line, the numbers of Lines can be divided into Lines without or with length smaller than k. The first formula shows the amount of Lines with length smaller than k, and the second formula shows the amount of Lines without length smaller than k.

- $(m+n)+2(m+n-1)=3m+3n-2$ .....(1)
- $3m+3n-2-4\cdot(k-1)$ .....(2)

2.3.3. The distance between two points on the same line

Through the feature of the length of Line mentioned above, we can find that the distance between Cells can be figured out through the coordinates of intersections. If the coordinates of two Cells are: $P_1(x_1, y_1)$、$P_2(x_2, y_2)$, and their index: $P_2>P_1$, the distance will be like that: $Max(x_2-x_1, y_2-y_1)+1$.

2.4. Connection

In a Line, the max range the same stones can develop is named Connection (C for short). Through this we can find that C⊆Line can be assumed. The connection in a Line will exist by the way of interlacing, like black white black..., or white black white..., and so on.

If we put k into consideration, the C in a Line won't interlace definitely. If the length of the C is smaller than k, we will exclude it.The way to represent C is similar to Line. The direction, end-point, length of Line are stable, but all of them in C are unstable. Besides, we use $|C|$ to tell the length of C.

2.5. The mode of connection

2.5.1. The shape and its binary bitwise coding

The Shape of connection is the arrangement of stones' related position in C. $\|C\|$ is used to represent the shape of C. We use bitwise coding to show the related arrangement of the stones in the C, and will be changed into integer. We take Start in C as the lower binary-weighted bits string. From Start to End, number intersections in turns in which those with stones are 1 and those without are 0. Take figure 3 for example. The Shape is coded as 000111100, so its integer is 60.

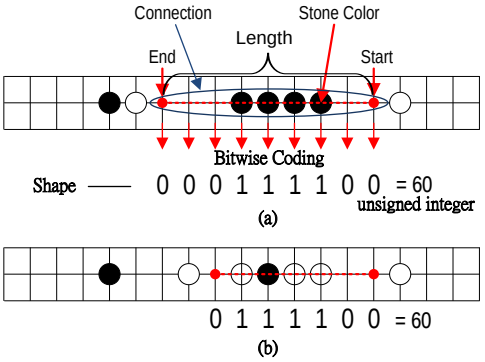

Figure 3, the coding of the Shape

Though the two different shapes with different length, 000111100 and 0111100, have the same integer, 60, we can observe that what it means has great differences because of the different length. Thus, when we take the arrangement of connected stones into consideration, we also have to take the length of a C into consideration in order not to make mistakes.

2.5.2. Pattern

According to the coding of the discussion and the shape above, we can code all the Shapes with different length based on the possible length of the connection in a line on the board. We name the shape with its length Pattern. Pattern is a kind of integer coding method after taking length of C's shape into consideration. About the addressing method, we precisely describe it in 3.3. Connection Set Array.

3. The design of Bitboard

3.1. The needed information of Connect6

There are two kinds of acts in the Connect-k game, offense and defense. When the attack side can win by attacking, the strategy is offense; if not, the strategy is defense.

3.1.1. The game states

According the number of grids m, n, the intersections on the board are $m\times n$. There are three possible situations of intersections: empty, black, and white. The collection of all these

situations of intersections is game state. We use this foundation to analyze the Connect-k game.

3.1.2. The needed information

At present, most of the searching method used as Connect6 program is Threat Space Search, TSS. The approach is offered by Professor Wu, and, about the specific illustration of TSS, please refer to (Wu, 2005).

Threat Space Search is an improvement of Dependency-based Search from Allis in 1994. The main idea of both of them is to control the move the defense side can play while threat happening to the attack side and the defense side has to do defense. In this situation, the attack side searches possible moves to achieve a victorious move. Based on this notion, the needed information is as follows when doing Threat Space Search:

- The Connection and it's Pattern

The goal of Connect-k is to form the connection of consecutive k stones, and each side at most can play p stones in a move. In this situation, by doing serial offense, TSS will cause a situation in which threats is bigger than p. When threats is bigger than p, the defense side can only play p stones to do defense; under this circumstance, the defense side will lose because it can't defend the attack of the offense side.

Thus, the search will depend on the information of the shape in this kind of game. The connected information of the board is the most dependent message because it can help understand whether there is threat in the game, and offer possible moves to do TSS.

- Empty Cell

After some connections, there will be some Empty Cells in the connections, so we will have an Empty Cell Pattern. Empty Cell Pattern refers to what Empty Cells mean to some connection, so connections will decide what Empty Cell Pattren will be. The information of Empty Cells depends on what the importance of Empty Cell to black or white stones is. Thus, in the same situation, the importance of Empty Cells to black or white stones is different.

We calculate the scores of Empty Cells from the Empty Cell Pattern. We choose candidate Empty Cells according to their scores from high to low when there are candidate Empty Cells.

3.2. The pattern of the shape

3.2.1. The coding of the shape

If the length of some connection is $|C|$=L, the number of this connection's shape is 2^L. Therefore, to do the coding of the shape in Connect-k Game, we start the coding from the length k of the shape, and the range of the coding is 0..2^k-1. In this way, we start from length k, and then k+1…, and so on.

We can thus get the coding transforming function of Connect-k's shape: $\Phi(L,\|C_L\|)=2^L-2^k+\|C_L\|$. The function supposes the length of C is $|C|$=L , and its shape is $\|C_L\|$. If the length of some connection is 10, and shape is 128, the Pattern will be $\Phi(10,128)=2^{10}-2^6+128=1088$.

3.2.2. Addressing

After coding the pattern of the shape, we can use this shape to compute the Pattern of all connections in order to figure out the meaning of the Pattern. About the discussion of the shape, professor Wu has illustrated clearly in his thesis, so we won't discuss it here.

The coding of the shape and using the coding to do addressing can let us get the Pattern of connection in a short time instead of spending time calculating the Pattern. In our program, we first figure out the meanings of all kinds of shape, and record them into the list of the arrangement. By searching, we can quickly get the information of a connection. This concept is illustrated in figure 4.

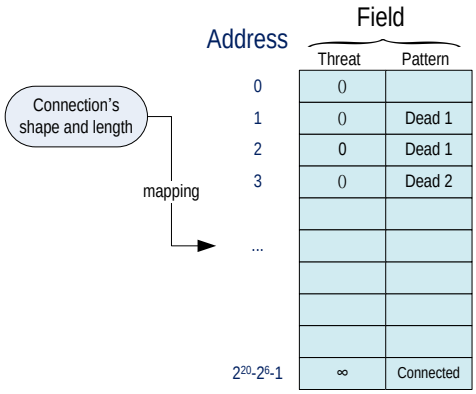

Figure 4, the shape list of 19×19 and its addressing

3.3. Connection Set Array

As to the storage of Connection in the board, we use Line to be the array storage of addressing. The Line of board is stable, so after making sure of the grids' number m, n, we can compute the number of Lines in every direction based on the discussion above. Numbering from 0, the way to number the Line of the board is Vertical, Slash West, horizontal, Slash East based on the direction.

3.3.1. The index of Line

The number of Connection in the Line is unstable. It may be 0 or many Connections, so we use connection set to store Connections in the Line. Because the length in the slash direction is unstable, the Line in the corner cannot produce connection (cause length is smaller than k). thus, we exclude the Line in these directions.

We can in this way correspond it to the Line on the board according to the coordinates (x, y) of intersections. Figure 5 illustrates our concept of addressing method from cell coordinate to connection set address. The mapping formula and its range are as follows: $(x,y)\xrightarrow{f(x,y)}\{L_V,L_{SW},L_H,L_{SE}\}$

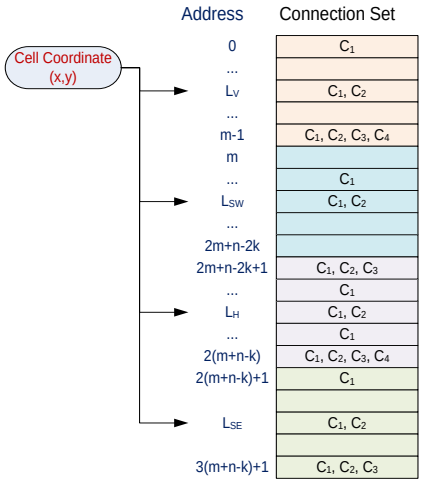

Figure 5, illustration of connection set

Vertical	$L_{D=V}=x$ Range：0..m-1
Slash West	$L_{D=SW}=m+(m-x+y)-1-(k-1)=2m-x+y-1$ Range：m..2m+n-2-2(k-1)

Horizontal	$L_{D=H} = 2m + n - y$ Range：2m+n-1..2(m+n)-2
Slash East	$L_{D=SE} = x + y$ Range：2(m+n)-1..3(m+n))-1

4. Bitwise computing

Most of the program languages offer bitwise computing nowadays. Bitwise computing has very high efficiency among computer calculation. After coding the informative structure of Connect6, we thus will design related bitwise computing. Only when coding and bitwise computing work together can we have improvement of computing efficiency.

4.1 Checking whether the game is over

There are many ways to check whether the game is over. The most used way is to check from 8 orientations of the intersections. If there is consecutive k stones, the game is over.

If we use bitwise computing to proceed the checking process, we can start checking from some intersection's end-point whose position is (k-1). Use the vectors of Check Mask of length k, and take turns using bitwise operator, "OR", to do bitwise computing. If the number is the same with Check Mask, we can learn that the check length is k. The algorithm and its example are in figure 6:

Algorithm- to check whether the game is over:

- At most 6 times
  - if((Check Mask & C Shape) == Check Mask)
    - ◆ Game over;
  - Check Mask <<= 1;

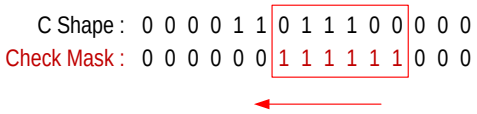

Figure 6, check whether the game is over

4.2 The Algorithm of calculating Threats

4.2.1 Professor Wu's algorithm (Wu, 2005)

Professor Wu's algorithm is based on Connection, scanning from any point of Connection to another point, checking Sliding Window (SW for short) a time. Calculate the stones in a SW; when the sum of stones are bigger than or equal to 4, accumulate the threats. The concept of the algorithm is as follows:

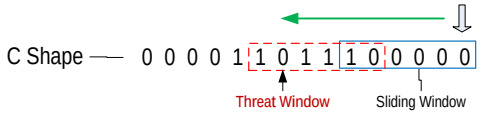

Figure 7, the concept to count threats

4.2.2. The approach to accelerate

Professor Wu's algorithm can correctly figure out the numbers of threats in a Connection. We use two ways to improve the algorithm.

- First, we can jump over the cells in a sliding window according to stones checked.
  - The number of jumping equals to the numbers to form threat window minus the numbers of stones in a sliding window.
  - For example, if there is no piece in a sliding window, we can jump over four cells to continue checking another sliding window. The reason is that it's impossible to count threat window inside four steps checking of a sliding window.
- Secondly, if it meets the threat window, we go on to check behind the farthest empty cell.

4.3. Occupy Cells

Occupying cells are the most important operation in state transition in Connect6. After occupying a cell, the related connection must be changed.

4.3.1. Occupy Cells into same connection

When we insert the same stone into a connection, the only change of the connection is the shape. At that time, all we need to do is to change the shape by using bitwise operator, "OR", to compute the new shape.

Figure 8, Occupy Cells in the same Connection

4.3.2. Occupy Cells into different connection

It's not so easy to insert different stone into a connection because the connection will be separated into two or three connections. Operating in this situation is more complicated because the whole Start, End, and Shape of Connection will be influenced. The following figure shows how to turn the concept into operation. (We only explain how to get the Shape in the two end-points and in the middle.)

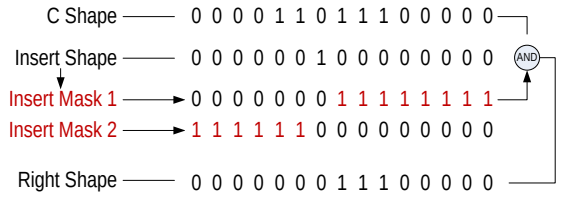

Figure 9, Occupy Cells in different Connection

- Use Insert Mask 1 and C shape to compute OR, we get the Shape of C on the right side: 11100000。
- Use Insert Mask 2 and C shape to compute OR, we get the Shape of C on the left side: 000011。
- At last, scan from the middle point to its left and right side, the C of the middle point can be sure.

5. Conclusion

Connect6 is a very interesting puzzle game. This research collects and analyzes Connect6's design of bitboard and the algorithm of its basic operation. We design a program of Connect6 according to the concept this research is discussing about. The program has nice performance so far. This is the first step for us to develop the higher A.I. of Connect6. Hope our study is helpful for those who are interested in this field.

6. Reference

Allis, L. V. (1994). Searching for solutions in games and artificial intelligence. Ph.D. Thesis, University of Limburg, Maastricht.

I-Chen Wu, and Dei-Yen Huang. (2005). A New Family of k-in-a-row Games. the 11th Advances in Computer Games Conference (ACG'11), Taipei, Taiwan.

I-Chen Wu, Dei-Yen Huang and Hsiu-Chen Chang. (2005). Connect6. ICGA Journal, Vol. 28, No. 4, pp. 234-241.

P. San Segundo, R. Galan, D. Rodriguez-Losada, F. Matia, A. Jimenez, (2006). Efficient search using bitboard models. Proceedings XVIII Int. Conf. Conference on Tools for AI (ICTAI 06), Washington, pp: 132-138.