Determination of Inner Areas of Moyo by Potential Values

TAJIMA Morihiko, SANECHIKA Noriaki

National Institute of Advanced Industrial Science and Technology
Tsukuba Central 2, 1-1-1, Umezono, Tsukuba-shi, 305-8568 JAPAN
tajima.m@aist.go.jp, sanetika@sepia.ocn.ne.jp

Abstract

It has been recognised that the position evaluation of the game of Go is an important theme, though it is very difficult. The authors have proposed a method that evaluates opening positions by utilizing the possible omission number (PON) and has shown its effectiveness. However, they have also shown that there exists problems the method cannot solve. One of the reasons is that the method considers only the neighboring points of groups. In order to solve the problem, a large framework of territory (*moyo*) should be dealt with. This paper presents an algorithm to determine inner areas of moyo by potential values.

Keywords: computer Go, fuseki, *moyo*, inner area, potential value

1 Introduction

In nowadays, the interest in computer games is being transfered from chess to shogi and go. Contrary to the case of chess, where brute force search is amazingly effective, it has been recognised that more intelligent methods are necessary for the games of shogi and go. Especially the game of go is regarded as the most challenging theme after chess because of the enormous size of the search space of the game tree and the difficulty in the evaluation of positions which have a lot of complexity and varieties in spite of the fact that the game uses only a single kind of piece, the stone.

Methods to make position evaluation has not been utilized effectively in playing programs yet. It is getting recognised, however, that position evaluation in the game of go is an important theme, though it is difficult (e.g. [1]).

In this paper, we present a method to determine the shape of *moyo* which is important for the position evaluation in the stage of fuseki or placing stones in the opening of the game of go. Moyo is the object that is very difficult to evaluate quantatively, though it is very important [2][3]. We have proposed a means to evaluate the enclosed equivalent size by the pairs of the cutoff numbers in [2]. However, it was not able to well express enclosures by diagonal lines. Fukui [3] can find straight weak connections by the use of Voronoi diagram. In this paper, we present a method to determine the shape of moyo by the use of a classical tool, potential value. The method can calculate diagonal enclusures by diagonal lines and enclusures by curved lines as well.

In general, evaluation methods of moyo tend to need enormous amount of calculation for accurate results, since they deal with global positional area. Practical algorithms with pretty good accuracy, however, should be as simple as possible satisfying the accuracy to apply for the evaluation function used in global search. In order to accomplish the purpose many improvements have been made so far for the methods utilizing potential values. This paper could be regarded as one of the improvements.

In this paper, we mainly utilize the position shown in Fig.1 as an example. This is the 34th position of the second game of Challenger Decision Match of Kiseisen in 1989 (Black: Chou Chikun 9 Dan, White: Takemiya Masaki 9 Dan, Result: White won by resignation after 168 moves.)[4]. It is an example where Black's strategy and White's strategy contrasts strik-

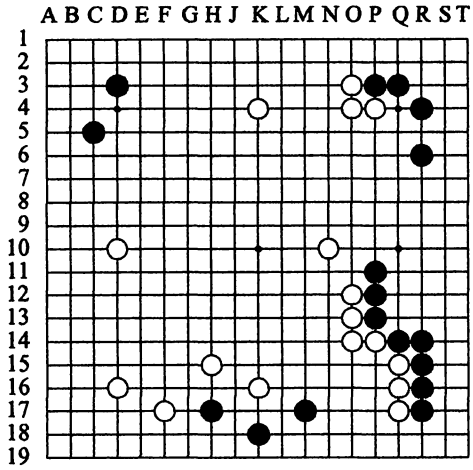

Figure 1: Example

ingly, i.e. while Black established territories on corners and an edge, White made a large moyo in the center of the board. In this position, how to evaluate the enclosure in the center of the board is the major problem. Our former method [2] was not able to recognize the long connection between K4 and D10.

We will show the relation between potential values and moyo in Section 2 and show the determining algorithm of inner areas and an example of the algorithm in Section 3. Then we will show the properties of the algorithm by the example of moyo whose inner area changes according as the progress of the game. Section 5 will show the cases where different influence functions are applied and shows that the effectiveness of an algorithm depends on the influence function. Section 6 will show especially useful applications. We will discuss several issues of the method in Section 7, and conclude this paper in Section 8.

2 Potential values and moyo

Potential value is one of the classical tools of computer go. It is the value that is given as the summation of the influences by life stones and is calculated by some influence function. Our influence function is based on [5] with some modifications. The function is not a simple one which is applied equally to all points of the board but a more practical one which is designed as fifteen sets of empirical values for the corresponding fifteen positions of the board (See Appendis A for details.). Those values were designed considering the specialty of corners and edges. The values were tuned by using the strength of connections (Here connections are considered as the boundaries of moyo.) and a special collection of opening books where the territories in the corners can be discriminated clearly from the outer area of influence. Each potential value can be converted to equivalent territory size.

Fig.2 shows the potential values which are calculated by our system. Positive numbers and negative numbers show the Black's advantage and White's advantage, respectively. The figure shows only the absolute values. It is expressed in Italic fonts that a number is negative. If the values are 0, they are not shown.

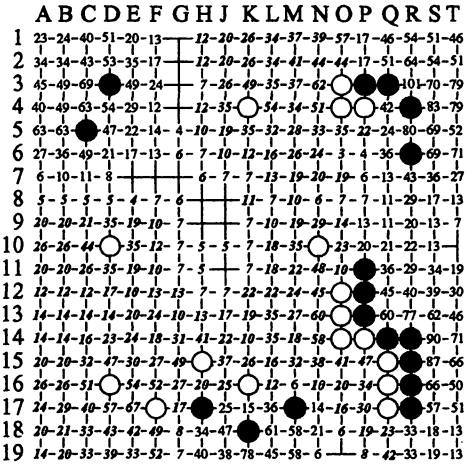

Figure 2: Potential value of each point

The closer a point is to a group (stone), the bigger the absolute value of the potential value is. And our influence function gives the influence to the point as far as distance 5. Therefore, the potential values themselves cannot be applied directly for the determination of enclosed shapes like moyo. [1]

[1] Gosedai [5] solved this problem to a limited extent by a makeshift named "reclamation of vacant points."

3 Algorithm

The algorithm which determines whether a vacant point p with a positive potential value is in the inner area of a Black's moyo or not is given as follows.

1. Let *Old* be the set of reached points, and let $p*$ be the focus.

   $Old \leftarrow \emptyset \qquad p* \leftarrow p$

2. If there are unreached adjacent points $p_i (i = 1, ..., n)$ (where n is the number of adjacent points and $p_i \notin Old$) of point $p*$, then find the point p' whose potential value $v(p_i)$ is minimum among them and goto 3. Else goto 4.

3. If $v(p*) \geq v(p')$, then $Old \leftarrow Old \cup p*$, $p* \leftarrow p'$, and goto 2. Else goto 4.

4. If $v(p*) < 0$, then p is an outer point. If $v(p*) \geq 0$, then $p*$ is an minimal point of Black and p is an inner point of Black's moyo.

5. End.

One can determine similarly whether a vancant point with a negative potential value is in the inner area of a White's moyo or not by reversing the sign.

Whether a point with the potential value of 0 is in an inner area or not can be determined by the following recursive procedure. Minimal points and maximal points are inner points.

- If one of the adjacent points of q is a minimal (maximal) point of colour c, then q is a minimal (maximal) point of colour c.

If there are more than one minimum points at the step 2 in the algorithm, more than one paths exist. In that case, we have two choices.

a If there is at least one path which determines that p is outer, let p be an outer point.

b If there is at least one path which determines that p is inner, let p be an inner point.

We adopt **b**.

Fig.3 shows the analysis of the board shown in Fig.2 by the algorithm. The points with a number is the points which are determined as inner points. If the minimal or maximal point finally reached is the same, the same number is assigned. Black's inner points and White's inner points are shown in Roman font and Italic font, respectively.

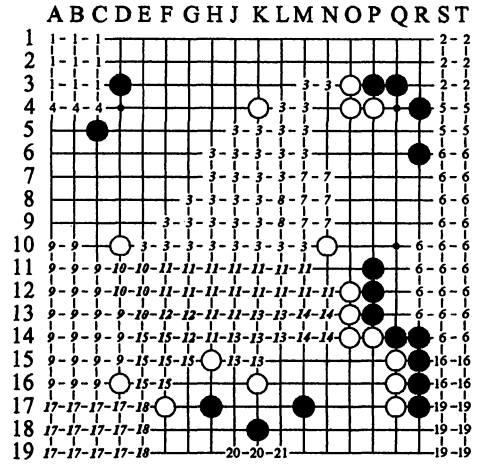

Figure 3: Inner areas

For example, the determination of point N12 is as follows: According to the algorithm, the path from N12 is N12-M12-M11(L12)-L11-K11-J11. Since the potential value of J11 is 0, N12 is an inner point of a White's moyo. Another example of point L3 is as follows: The path from L3 is L3-L2-K2-J2-H2-G2-F2$\cdots$. Since the path goes into a positive area, L3 is an outer point of White.

By the example, the algorithm can determine the inner area of a moyo easily and naturally. Especially the ability to form diagonal lines like the one formed by the White's moyo in the center of the board in the example was greatly improved.

4 Changes of inner area

Now we can see how the inner area changes according as the game proceeds. Fig.4 and Fig.5 show the inner areas of the positions two moves before and four moves before, respectively. From these figures one can observe that

each White move increases the inner area of the White's moyo and the moyo becomes established.

It might seem unnatural that the white stone D10 caused, for example, the point M3 to be in the White's inner area, since the right side of the moyo has no change. It is the effect of the wall from K4 to D10 caused by the 34th move.

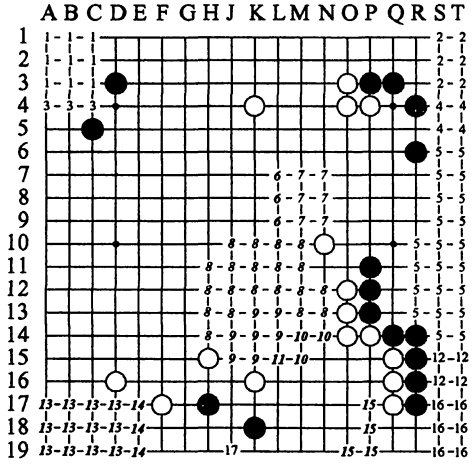

Figure 4: Inner area of the position two moves before

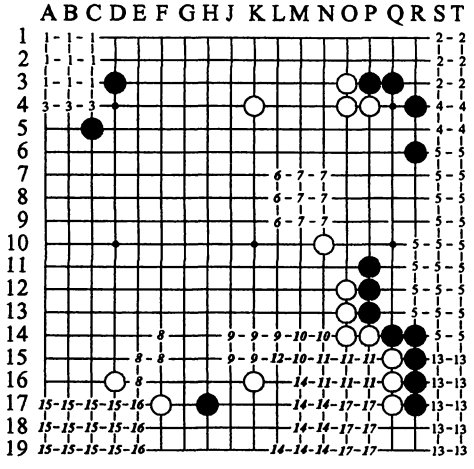

Figure 5: Inner area of the position four moves before

When one calculate an inner area of sequential positions incrementally, one can utilize the information of the previous position. For example, when a stone of the same colour as a moyo is put in the outer area, the inner area does not decrease. For example, the 33rd position, the position one move before the position shown in Fig.3, the White stone onto D10 does not decrease the White's inner area. Therefore, recalculation is not needed for the points which were already White's inner area. The situation is similar in the case where a stone of the same colour as a moyo is put into the inner area. However, some kind of attention should be paid, which will be described later.

5 Variations of the algorithm

The effectiveness of the method, which determines the area of moyo by potential values, depends on its influence function. While our influence function is very effective, influence functions which are popular and simple are not very useful. The functions $1/d$ and $1/2^d$ where d is distance are the popular ones. Though Manhattan distance is usually utilized as the distance, Euclid distance could be utilized. We will show such examples.

Fig.6 shows the inner areas when Manhattan distance and $1/d$ are utilized as the distance and the influence function, respectively. The boundaries of the areas are like straight lines and unnatural. It is not appropriate for the recognition of moyo. Fig.7 shows the inner areas when Euclid distance and $1/d$ are utilized as the distance and the influence function, respectively. The form is a little better than that when Manhattan distance is utilized, but it is still unnatural since the points like those from G9 to G14 are ignored. One should design and utilize empirically good influence functions which can generate natural potential values.

6 Useful application

1. Completion of moyo

   One can easily recognise by the method that a moyo will be completed by a next

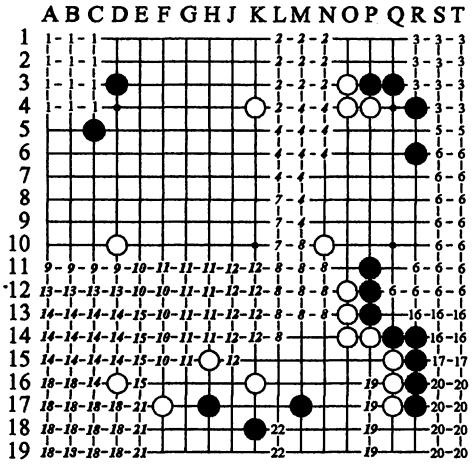

Figure 6: Inner areas determined by Manhattan distance

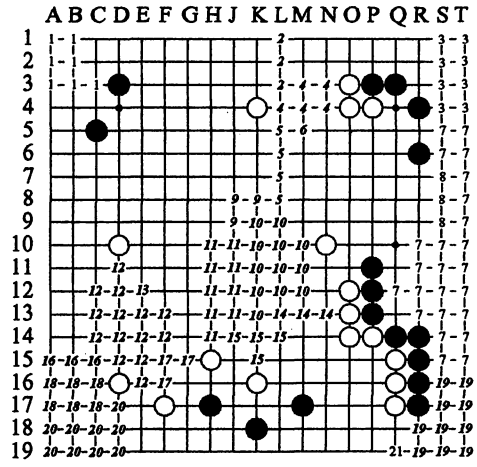

Figure 7: Inner areas determined by Euclid distance

single move. Fig.8 shows a moyo still incomplete which has an open side, and its inner area is still small. On the contrary, Fig.9 has a large inner area. Therefore the players can utilize the difference in the game.

2. Detection of weak points

   Since the strength of the connection between two remote stones is given by the potential value on the boundary point with the minimum absolute value of the potential value, one can detect the weak points of moyo by the information. In the example, the weakest points are H7 and G8 which have barely White potential value (-6). Therefore, the players should attack or defend such points in the game.

7 Problems

The method has two major problems. Connections generated by the algorithm are not necessarily straight because of the influences of opponent's stones. For example, the effect can be seen in the fact that the points N11 and L15 are classified as outer points in Fig.3. The fact is also a merit of the algorithm, but it sometimes differs from usual understanding. For example, the fact that the points R3 and Q13 are classified as outer points is unnatural determination. Those phenomena are often occured when the opponent's stones are very close to a moyo. Since the method is especially effective for a large moyo in the center of the board, it should be utilized for the evaluation of such moyo. General means by thickness or potential values should be used for the points where the opponent's stone are very close.

Another problem is the application of the algorithm to a moyo of concave shape. Fig.10 shows the result after applying the algorithm to the position after White's move J9 to Fig.3. Comparing with the position in Fig.3, one can observe an unnatural phenomenon that the inner area decreases in spite of the reinforcement with the stone of the same colour, since the peaks of the potential moved toward the inner area by the new stone. It is because that the target of the algorithm is essentially convex moyo. In order to avoid the unnnatural phenomena, the following operation is necessary when an additional stone is put.

1. When a stone of the same colour is put into the inner area, the influence of the new stone is ignored. But only the influences to the boundary points are counted, since the strength of the connection should be reinforced.

2. When a stone of the same colour is put in

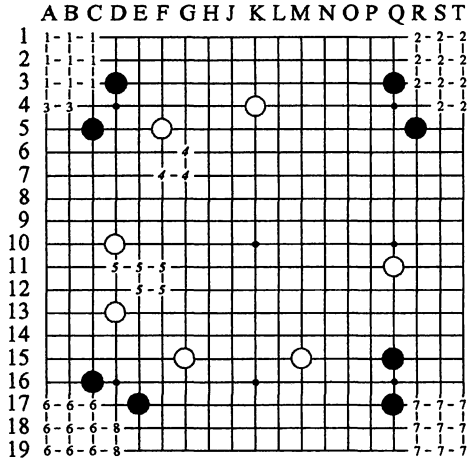

Figure 8: Incomplete enclusure

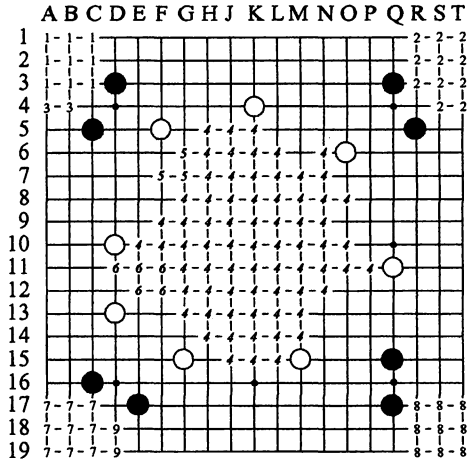

Figure 9: Complete enclosure

the outer area, above operation is made, if necessary, after experimentally reversing the order of putting stones, i.e. put the new stone first and put one of the surrounding old stone of the same colour next.

The necessity to handle such cases strictly is practically small from the viewpoint of the application of the method to position evaluation. Positions can be evaluated by usual means of thickness, since moyo is small in such cases. In other words, the assumption that moyo is convex is not very inappropriate in the cases where one should calculate by the algorithm large inner areas of moyo which is too large to handle as neighbouring points of groups.

Some phenomena which would seem to be unnatural may be necessary to indidate. The point M3 in Fig.4 was in outer area, but it changed to White's inner area by White's D10. It might be considered that the point was already in the White's inner area even before the move D10 was made. And we classified the enclosures shown in Fig.8 and Fig.9 into incomplete enclosure and complete enclosure, respectively. But such a great difference shown in the two figures might generally not be recognised between the two positions. By intuitive observation, the inner area in Fig.8 is too small. Further investigation should be needed.

Most of such problems can be practically solved by making unconditionally the points with the absolute value of the potential value over a certain level be the points of inner area.

8 Conclusion

Since a moyo is a large pattern with ambiguous shape, the effect is not clear and its evaluation is very difficult. The method presented in this paper could be called "Watershed Method". We consider that the calculation is easy and practical and that the method can naturally calculate the continuous elements of go such that weak connections are bended by opponent stones. Although there are some problems, we consider it is a useful and natural method. We are evaluating the effectiveness of the method on our playing program now.

References

[1] M. Müller. Position evaluation in computer Go, *ICGA Journal*, Vol.25, No.4, 219-228, 2002.

[2] Tajima M. and Sanechika N.. Equivalent Size of Moyo, *Proceedings of GPW2003*, 145-152 (2003).

[3] Fukui M., Takeuchi Y., Matumoto T., Kudo H., Yamamura T., and Ohnishi N.. A Method for Evaluating Go Positions,

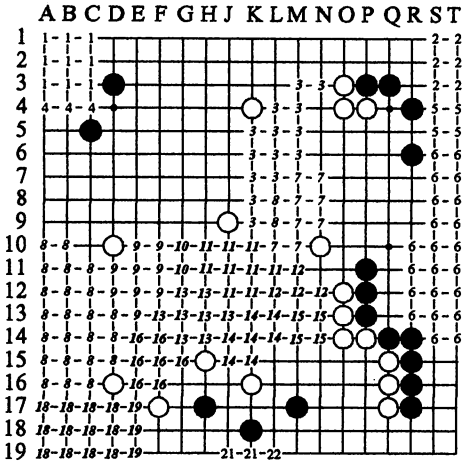

Figure 10: Self's stone to inner area

IPSJ SIG Technical Reports, 2004, 28, 21-26 (2004). (in Japanese)

[4] Takemiya M.. Uchuu-ryuu Kessaku-sen 3, Nihon Kiin, 1990. (in Japanese)

[5] Sanechika N. et al.. The specifications of "Go Generation", *Proceedings of the Game Playing System Workshop*, 73-155, Tokyo (1991).

A The influence function

```
/*************************************
** ポテンシャルパターンテーブ
*************************************/

LOCAL BYTE paternA[81] = {
  0,  0,  0,  0,  5,  0,  0,  0,  0,
  0,  0,  6,  7,  7,  7,  6,  0,  0,
  0,  6,  7, 10, 12, 10,  7,  6,  0,
  0,  7, 10, 19, 35, 19, 10,  7,  0,
  5,  7, 12, 35,100, 35, 12,  7,  5,
  0,  7, 10, 19, 35, 19, 10,  7,  0,
  0,  6,  7, 10, 12, 10,  7,  6,  0,
  0,  0,  6,  7,  7,  7,  6,  0,  0,
  0,  0,  0,  0,  5,  0,  0,  0,  0
}; /* 中原（五線以上）*/

LOCAL BYTE paternB[72] = {
  0,  7, 12, 20, 26, 20, 12,  7,  0,
  0,  7, 12, 20, 26, 20, 12,  7,  0,
  0,  7, 12, 26, 44, 26, 12,  7,  0,
  5,  7, 12, 35,100, 35, 12,  7,  5,
  0,  7, 10, 19, 35, 19, 10,  7,  0,
  0,  6,  7, 10, 12, 10,  7,  6,  0,
  0,  0,  6,  7,  7,  7,  6,  0,  0,
  0,  0,  0,  0,  5,  0,  0,  0,  0
}; /* 辺の四線 */

LOCAL BYTE paternB1[64] = {
 14, 20, 26, 26, 20, 14,  7,  0,
 20, 21, 26, 26, 20, 14,  7,  0,
 24, 24, 33, 44, 26, 16,  7,  0,
 26, 26, 44,100, 35, 12,  7,  5,
 20, 20, 26, 35, 19, 10,  7,  6,
 14, 14, 16, 12, 10,  7,  6,  0,
  7,  7,  7,  7,  7,  6,  0,  0,
  0,  0,  0,  5,  0,  0,  0,  0
}; /* 隅の星 */

LOCAL BYTE paternC[63] = {
  0,  7, 13, 20, 51, 20, 13,  7,  0,
  0,  7, 17, 29, 51, 29, 17,  7,  0,
  5,  7, 18, 42,100, 42, 18,  7,  5,
  0,  7, 10, 19, 35, 19, 10,  7,  0,
  0,  6,  7, 10, 12, 10,  7,  6,  0,
  0,  0,  6,  7,  7,  7,  6,  0,  0,
  0,  0,  0,  0,  5,  0,  0,  0,  0
}; /* 辺の三線 */

LOCAL BYTE paternC1[56] = {
 23, 24, 35, 51, 20, 13,  7,  0,
 27, 27, 36, 51, 29, 17,  7,  0,
 32, 32, 51,100, 42, 18,  7,  5,
 20, 20, 26, 35, 19, 10,  7,  0,
 12, 12, 12, 12, 10,  7,  6,  0,
  7,  7,  7,  7,  7,  6,  0,  0,
  0,  0,  0,  5,  0,  0,  0,  0
}; /* 隅の小目 */

LOCAL BYTE paternC2[49] = {
 27, 36, 57, 20, 13,  7,  0,
 36, 39, 58, 29, 17,  7,  0,
 57, 58,100, 42, 18,  7,  5,
 20, 29, 42, 19, 10,  7,  0,
 13, 17, 18, 10,  7,  6,  0,
  7,  7,  7,  7,  6,  0,  0,
  0,  0,  5,  0,  0,  0,  0
}; /* 隅の三々 */

LOCAL BYTE paternD[54] = {
  0,  7, 19, 38, 78, 38, 19,  7,  0,
```

```
  5, 13, 19, 45,100, 45, 19, 13,  5,
  0,  7, 10, 19, 35, 19, 10,  7,  0,
  0,  6,  7, 10, 12, 10,  7,  6,  0,
  0,  0,  6,  7,  7,  7,  6,  0,  0,
  0,  0,  0,  0,  5,  0,  0,  0,  0
}; /* 辺の二線 */

LOCAL BYTE paternD1[48] = {
 30, 30, 45, 78, 38, 19, 13,  0,
 33, 33, 54,100, 45, 19, 13,  5,
 20, 20, 26, 35, 19, 10,  7,  0,
 14, 14, 16, 12, 10,  7,  6,  0,
  7, 12,  7,  7,  7,  6,  0,  0,
  0,  0,  0,  5,  0,  0,  0,  0
}; /* 4の二 */

LOCAL BYTE paternD2[42] = {
 39, 48, 85, 38, 19, 13,  0,
 58, 61,100, 45, 19, 13,  5,
 29, 29, 42, 19, 10,  7,  0,
 17, 17, 18, 10,  7,  6,  0,
 13, 13,  7,  7,  6,  0,  0,
  0,  0,  5,  0,  0,  0,  0
}; /* 3の二 */

LOCAL BYTE paternD3[36] = {
 57, 88, 38, 19, 13,  0,
 88,100, 45, 19, 13,  5,
 38, 45, 19, 10,  7,  0,
 19, 19, 10,  7,  6,  0,
 13, 13,  7,  6,  0,  0,
  0,  5,  0,  0,  0,  0
}; /* 2の二 */

LOCAL BYTE paternE[45] = {
  5, 14, 22, 54,100, 54, 22, 14,  5,
  0,  7, 10, 19, 35, 19, 10,  7,  0,
  0,  6,  7, 10, 12, 10,  7,  6,  0,
  0,  0,  6,  7,  7,  7,  6,  0,  0,
  0,  0,  0,  0,  5,  0,  0,  0,  0
}; /* 辺の一線 */

LOCAL BYTE paternE1[40] = {
 51, 36, 63,100, 54, 22, 14,  5,
 20, 20, 26, 35, 19, 10,  7,  0,
 16, 14, 16, 12, 10,  7,  6,  0,
  7, 12,  7,  7,  7,  6,  0,  0,
  0,  0,  0,  5,  0,  0,  0,  0
}; /* 4の一 */

LOCAL BYTE paternE2[35] = {
 61, 70,100, 54, 22, 14,  5,
 29, 29, 42, 19, 10,  7,  0,
 17, 17, 18, 10,  7,  6,  0,
 13, 13,  7,  7,  6,  0,  0,
  0,  0,  5,  0,  0,  0,  0
}; /* 3の一 */

LOCAL BYTE paternE3[30] = {
 97,100, 53, 22, 14,  5,
 38, 45, 19, 10,  7,  0,
 19, 19, 10,  7,  6,  0,
 13, 13,  7,  6,  0,  0,
  0,  5,  0,  0,  0,  0
}; /* 2の一 */
LOCAL BYTE paternE4[25] = {
100, 54, 22, 14,  5,
 54, 19, 10,  7,  0,
 22, 10,  7,  6,  0,
 14,  7,  6,  0,  0,
  5,  0,  0,  0,  0
}; /* 1の一 */

/************************************
** 基準座標posから使用するポテンシャル
** パターンを決定するテーブル
** 実際に使用するパターンは
** paternTbl2[paternTbl1[pos.y-1][pos.x-1]]
となる。
************************************/
LOCAL SHORT paternTbl1[5][5] = {
  15, 14, 12,  9,  5,
   0, 13, 11,  8,  4,
   0,  0, 10,  7,  3,
   0,  0,  0,  6,  2,
   0,  0,  0,  0,  1
};

LOCAL PBYTE paternTbl2[16] = {
NULL,paternA,paternB,paternC,
paternD,paternE,paternB1,paternC1,
paternD1,paternE1,paternC2,paternD2,
paternE2,paternD3,paternE3,paternE4
};
```