五目並べプログラムの基礎研究

松田秀雄，澤柳一美，宮腰　隆，山淵龍雄，中嶋芳雄

A Basic Reserch for The Programming of Go-Moku

Hideo Matsuda, Kazumi Sawayanagi, Takashi Miyagoshi,
Tatsuo Yamabuchi, and Yoshio Nakajima

In go-moku, two players move in turn placing a stone on an intersection of Go boads, and then the player who first makes a line of five consecutive stones wins the game. In this paper we treat programs using on the UNIX X window for Go-moku game of man vs. machine. Especially, we lay much emphasis on the search method for the information judging whether the game is decided with this, and it is possible or not that a stone put on an intersection. A line of six intersections including the present move is classified into eight patterns by the shape of stone arrangement, and the combination of a pair of those patterns is used to estimate the move.

Kye Words : Go-moku, game, man vs. machine, strategic search

1．はじめに

五目並べとは，二人の対局者が碁盤の目に黒石と白石とを交互に置いていって，同じ色の石が縦，横，斜めのいずれかの方向に先に連続して五個並べば，“五”になったといって勝ちとするゲームである。子供にでもすぐできるこのゲームも機械にやらせるとなると，人間同士が行っている思考過程を一々模索して教え込まねばならず，人工知能の恰好の題材となる。例えば，通常五個まで石が並ばなくても，その先に，これは“四三である”といって，勝負がついたとして止めてしまう。五目並べに少し馴れた人間なら，これを並べられた石の形で見分けてしまう。又，“三三”のように禁止手もあり，これも，人間なら一目で分かる。しかし，機械にこれらのことを教え込むには，総当たり式に盤面を調べる方法しかない。効率的な方法というのは，如何に調べる範囲を狭められ，且つ，データ処理を能率的に行えるかにかかってくる。本研究では人間とコンピュータが五目並べをするプログラムについて，今まで行ってきた研究について述べる。特に，これで勝負がついたかどうか，とか，そこに石がおけるかどうかを判定するための情報を集める検索部分に重きを置いて記述する。

2．方　　法

プログラムはUNIX上のC言語で書かれ，Xウインドウ上のアプリケーションとして開発してい

る。まず，プログラムの構成要素は以下のようである。

2．1　プログラムの構成

2．1．1　メインプログラム部分　ゲームが開始されると，まず，この部分に実行権が渡される。ここで，グローバル変数（全部の関数から呼び出される変数）の初期化や，終了作業をする。ウインドウ上で動作するプログラムなので，gcの取得やウインドウの生成等もしている。

2．1．2　描画部分　ここでは，ウインドウに碁盤を描いたり，石を表示したりする部分である。Xウインドウにおいて'Expose'というイベントが発生する。これはウインドウが動いたり，他のウインドウに隠れて又一番上にきたときなどに発生する。このイベントが生じたら，ウインドウ内を再描画しないと，中のものがきちんと表示されなくなる。このため'Expose'イベントを監視し，発生したら再描画するようにする。又，マウスのイベントも監視し，押された位置を判定する。

2．1．3　制御部分　制御部分でユーザーとコンピュータの各順番を決めて，交代交代で手番を実行させるようにする。これは基本的には内部にカウンタを設けて，それを奇数か偶数かで判断させているだけである。

2．1．4　ユーザ（人間）部分　ユーザーの場合の処理をする。ゲームはマウスを用いて石を置くのでマウスのイベントを呼び出して，実際の位置を調べる。その位置を碁盤の座標に変換してそこに石が置けるかどうか，勝ったかどうかの判断をする。

2．1．5　コンピュータ部分　コンピュータの思考部分と制御部分に分かれる。五目並べのプログラムの神髄は，実に，この思考部分のルーチンの出来映えにかかるが，本報告ではまだ研究に着手したばかりなので，先読みの技法は入っていない。現在の盤面のみの情報から判断する方法をとっている。処理の優先順位として，（1）コンピュータが置ける五，四三などを打つ，（2）ユーザが置いた四，三などを止める。（3）コンピュータが置ける四，三などを打つ，（4）コンピュータにとって有利と思われる手を打つ，ということにする。

2．1．6　検索部分　ユーザー（あるいはコンピュータ）が石を置いた位置の周りの情報を読み取る部分である。

2．1．7　判定部分　判定部分は検索部分で読み取った情報からその手が何であるかを判定する。

2．2　検索方法

便宜上，黒石の番で説明する。今，一着手置いた石の周りの情報を検索するため，ここでは石の並びをあるパターン毎に分けて取り扱うことにする。

打った石を中心に盤の目の8つの方向に，図1のような0から7までの番号を割り当てて，それぞれの方向を表すことにする。検索時に収集する情報を表すため次の三つの変数を使う。

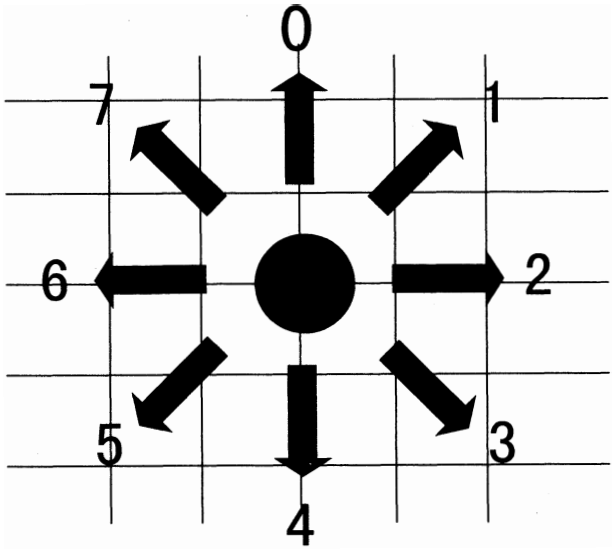

図1　検索時の方向の番号

0から7までの方向の石の並びの状態を表す
Sta［0～7］　：16進数4桁
置いた石を含む連続した石の並んだ数を表す変数
Row［0～7］　：自然数
一直線上に直したときの石の並んだ数を表す変数
Total［0～3］：自然数

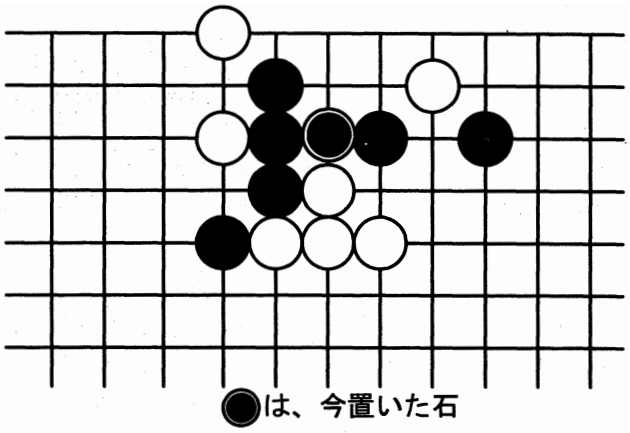

図2　石の並びの例

これらの中，Row［0～7］は，単に八つの方向のそれぞれの連続して並んだ石の数を数えるだけである。変数Totalは8方向に数えたRowをそれぞれ四つの方向の一直線上にあるもの同士を纏めたもので，m＝0，1，2，3として，

Total［m］＝Row［m］＋Row［m＋4］－1

である。

状態変数Sta［m］，但し，0≦m≦7，は初期値を0x0000としておいて，

（s1）　空白（目に石がない）のとき，2bit左にシフトし，0x0001を加える

（s2）　同じ色の石のとき，2bit左にシフトする

（s3）　違う色の石のとき，0x0400を加える

なお，これらの情報は打った石●（本文中は◆で表す）の隣から数えて，最大5個先の目の石の状況まで集めればよい。途中で，異色の石があるか，三つ空白が続いたら，その先は計算不用とする。

図2の石の並びについて，こ

表1　収集した情報（8方向）

n	Sta[n]	Row[n]	Patern ()
0	0x0405	1	O
1	0x0405	1	O
2	0x0045	2	B
3	0x0015	1	A
4	0x0400	1	X
5	0x0015	3	A
6	0x0400	2	X
7	0x0400	2	X

表2　4方向に直した情報

Total[m]=Row[m]+Row[m+4]-1

m	Total[m]	Patern (m)	Patern (m+4)
0	1	O	X
1	3	O	A
2	3	B	X
3	2	A	X

れらの変数の数値を計算してみる。

Row は各方向の連続した石の数を数えるだけであるから，例えば，２の方向の石の並びは◆●＋●＋＋　である故 Row［2］は2である。その他の方向も求めたのが，表１の Row［n］の欄である。又，本例の Total［m］は表２の，Total［m］の欄になる。次ぎに，［n］について計算したのが，図３である。０，及び１方向の場合でみると，

◆＋＋なし　になっている。

まず，Sta［n］（n＝０，１）の初期値0000-0000-0000-0000,

一つ先が＋なので

（s１）により，Sta［n］

＝ 0000-0000-0000-0001,

二つ先も＋なので，同じく（s１）0000-0000-0000-0101,

三つ先はなし（盤外）なので異色の石があるとみなして，

（s３）を適用0000-0100-0000-0101，四つ先，五つ先は，３つ先が異色の石のため検索する必要がない。図３，０，１，方向の表は以上を纏めたものである。表右端の列は Sta［n］を16進数表示している。２の方向の石の並びは

◆●＋●＋＋で，Sta［２］について，計算したのが図３中，２方向の表である。Sta［２］の初期値は１行目0000-0000－0000－0000，一つ先は黒なので，（s２）で２行目のように変わらない。二つ先は＋なので，（s１）により，同表３行目，三つ先は黒，（s２）により，Sta［２］

０、１方向　　●＋＋なし（○）

	状態	2進数表示	16進数表示
初期値	●	0000 0000 0000 0000	0x0000
1つ先	＋	0000 0000 0000 0001	0x0001
2つ先	＋	0000 0000 0000 0101	0x0005
3つ先	なし	0000 0100 0000 0101	0x0405
3つ目がない(異色)ため検索終了			

Patern 0

２方向　　●●＋●＋＋

	状態	2進数表示	16進数表示
初期値	●	0000 0000 0000 0000	0x0000
1つ先	●	0000 0000 0000 0000	0x0000
2つ先	＋	0000 0000 0000 0001	0x0001
3つ先	●	0000 0000 0000 0100	0x0004
4つ先	＋	0000 0000 0001 0001	0x0011
5つ先	＋	0000 0000 0100 0101	0x0045
五つ先まで見たため終了			

Patern B

３方向　　●＋＋＋

	状態	2進数表示	16進数表示
初期値	●	0000 0000 0000 0000	0x0000
1つ先	＋	0000 0000 0000 0001	0x0001
2つ先	＋	0000 0000 0000 0101	0x0005
3つ先	＋	0000 0000 0001 0101	0x0015
三つ空白が続いたため終了			

Patern A

４方向　　●○

	状態	2進数表示	16進数表示
初期値	●	0000 0000 0000 0000	0x0000
1つ先	○	0000 0100 0000 0000	0x0400
異色の石があったため終了			

Patern X

５方向　　●●●＋＋＋

	状態	2進数表示	16進数表示
初期値	●	0000 0000 0000 0000	0x0000
1つ先	●	0000 0000 0000 0000	0x0000
2つ先	●	0000 0000 0000 0000	0x0000
3つ先	＋	0000 0000 0000 0001	0x0001
4つ先	＋	0000 0000 0000 0101	0x0005
5つ先	＋	0000 0000 0001 0101	0x0015
五つ先まで見たため終了			

Patern A

６、７方向　　●●○

	状態	2進数表示	16進数表示
初期値	●	0000 0000 0000 0000	0x0000
1つ先	●	0000 0000 0000 0000	0x0000
2つ先	○	0000 0100 0000 0000	0x0400
異色の石があったため終了			

Patern X

図３　Sta［ ］の計算例

は四行目0000-0000-0000-0100となる。四つ先，五つ先は共に＋なので，（s１）を各適用して，Sta［２］は５行目，６行目のように計算される。５つ先までしらべたので，２の方向の探索はこれで終わる。３の方向の石の並びは◆＋＋＋ である。Sta［３］について，計算したのが図３中，３方向の表であるがこの場合は空白が三つ続くので，Sta［３］の計算は止める。図３にはその他の方向のSta［ｎ］の計算も載せている。各方向の最終的に求まったSta［ｎ］を表１のSta［ｎ］の欄に纏める。

２．３　判定方法

２．３．１　石の並び方とパターン化　　表３は一つの方向の予想される全ての石の並び方を48種類の石の並びに分類したもので，それぞれで，Sta［ ］の値は同じになる。又，次の一手を決める評価値が同じになるものを，A，B，C，D，E，F，O，Xの八つのパターンに分けている。但し，No．36のパターンの欄が特殊となっているのはこの並びでは打った石の５個隣まで，Sta［ ］の値を計算しても，初期値0x0000のまま変わらなかった場合で，これは黒石が連続６つあり，長連で負けで，実戦であり得ないことを表す。No.37からNo.48までは，このような石の並びでは，左から順にSta［ ］の値を計算していっても，途中で，もう計算しなくてもよいという条件が満たされて，実際上，

表３　石の並び方とSta［ ］，パターン対応表

No.	Sta[ ]	石の並び方		パターン
1	0x0000	●		特殊
2	0x0001	●＋		A
3	0x0004	●＋●		X
4	0x0005	●＋＋		A
5	0x0010	●＋●●		X
6	0x0011	●＋●＋		B
7	0x0014	●＋＋●		O
8	0x0015	●＋＋＋		A
9	0x0040	●＋●●●		X
10	0x0041	●＋●●＋		D
11	0x0044	●＋●＋●		E
12	0x0045	●＋●＋＋		B
13	0x0050	●＋＋●●		O
14	0x0051	●＋＋●＋		O
15	0x0054	●＋＋＋●	存在せず	
16	0x0055	●＋＋＋＋	存在せず	
17	0x0100	●＋●●●●		X
18	0x0101	●＋●●●＋		F
19	0x0104	●＋●●＋●		D
20	0x0105	●＋●●＋＋		C
21	0x0110	●＋●＋●●		E
22	0x0111	●＋●＋●＋		E
23	0x0114	●＋●＋＋●		B
24	0x0115	●＋●＋＋＋		B
25	0x0140	●＋＋●●●		O
26	0x0141	●＋＋●●＋		O
27	0x0144	●＋＋●＋●	存在せず	
28	0x0145	●＋＋●＋＋	存在せず	
29	0x0150	●＋＋＋●●	存在せず	
30	0x0151	●＋＋＋●＋	存在せず	
31	0x0154	●＋＋＋＋●	存在せず	
32	0x0155	●＋＋＋＋＋	存在せず	
33	0x0400	●○		X
34	0x0401	●＋○		O
35	0x0404	●＋●○		E
36	0x0405	●＋＋○		O
37	0x0410	●＋●●○		D
38	0x0411	●＋●＋○		B
39	0x0414	●＋＋●○		O
40	0x0415	●＋＋＋○	存在せず	
41	0x0440	●＋●●●○		F
42	0x0441	●＋●●＋○		C
43	0x0444	●＋●＋●○		E
44	0x0445	●＋●＋＋○		B
45	0x0450	●＋＋●●○		O
46	0x0451	●＋＋●＋○	存在せず	
47	0x0454	●＋＋＋●○	存在せず	
48	0x0455	●＋＋＋＋○	存在せず	

現れて来ない石の並びである。表中の“石の並び方”の欄を説明しよう。この欄の全行にわたって左端に黒石がある。この黒石はSta［ ］の値を計算する過程で，黒石がいくつも（一つの場合も含めて）連続して並んでいたとしたら，その最後の黒石を表している。Sta［ ］の値はそのあと，欄の石の並び方が●か＋か○かになっている間計算が続けられ，空白が表れるところで，最終値（表中，2列目のSta［ ］の欄の値）が決まる。例えば，図2の例題の0，1方向の石の並びは◆＋＋なし（白石○があるとみる）で，黒石の連続した並びは◆だけで，◆＋＋なし（○）でSta［ ］の値は決まる（図3の0，1方向の表）。表3ではこれは，36行目Sta［ ］の値0x0405の石の並びに当てはまる（パターンO）。例題の2方向の石の並びは◆●＋●＋＋で，連続した石◆●の右石から数えて●＋●＋＋でSta［ ］の値が決まった（図3の2方向の表）。これは表3の12行目Sta［ ］の値0x0045の石の並びに当てはまる（パターンB）。3方向の石の並びは◆＋＋＋で図3の3方向の表で計算したように，Sta［ ］の値0x0015はこの並びで決まるので，表3の8行目の石の並びに当てはまる（パターンA）。4方向の石の並びは◆○で，図3の4方向の表で計算したように，Sta［ ］の値0x0400はこの並びでだけで決まるので，表3の33行目の石の並びに当てはまる（パターンX）。5方向の石の並びは◆●●＋＋＋で，連続した石の右端からみた石の並びは●＋＋＋で，表3の8行目Sta［ ］の値0x0015の石の並びに当てはまる（パターンA）（図3の5方向の表でSta［ ］の値を計算している）。6，7方向の石の並びは◆●○で，連続した石の右端からみた石の並びは●○で，表3の33行目Sta［ ］の値0x0400の石の並びに当てはまる（パターンX）（図3の6，7方向の表でSta［ ］の値0x0400は計算済み）。　表1のPatern（ ）の欄はこのようにして得られた8つの方向のパターンを示したものである。表2は一直線上にある二つの方向のパターンをm（0，1，2，3の方向を示す数値）ごとに，纏めたものである。この2つずつのパターンの対を使って，図4でいま打とうとしている石の判定を下す。

図2の配石の例題で，更に，このことを説明する。この例題の4つの方向のTotal［ ］の値とパターンの対は表2に既に求まっている。まず，mが0の方向はTotal［ ］＝1，パターンの対（O，X）である。このとき，図4のTotal［ ］＝1の表で，O行X列をみる。”なし”となっている。これは三とか四とかといった直ぐに勝利に結びつく手ではないということをいっている。次に，mが1の方向はTotal［ ］＝3，パターンの対（O，A）である。同じようにして，図4のTotal［ ］＝3の表で，O行A列をみる。“三”になっている。これは三の手が出来たということをいっている。mが2の方向はTotal［ ］＝3，パターンの対（B，X）である。同じく，図4のTotal［ ］＝3の表で，B行X列をみる。”飛び四”になっている。これはこの方向に飛び四が出来たということをいっている。最後に，mが3の方向はTotal［ ］＝2，パターンの対（A，X）である。図4のTotal［ ］＝2の表で，A行X列をみる（“－”になっているときは行と列を入れ替える）。“なし”になっている。これもいま直ぐ三とか四とかといった直ぐに勝利に結びつく手ではないということをいっている。実際，図2をみると，◆の石に対して上の判定が正しく行われていることが分かる。

図4は各パターンの相互関係をみて，これとこれを組合わせたらどんな手が出来るかを調べて作った表で，比較的小さく判定が纏められた。

2．4　検索時間の測定例

目に石がない場合には空石があると考える。五目並べの（三になるとか，四になるとか，五になるとかいった評価の高い手を見分ける）判定アルゴリズムを作るには，置いた石◆の左右5個ずつ計11個の石の並びをみれば十分である。◆から6つ離れた目に黒石が置かれていても，◆と直接結びついて評価の高い手になることはない。そこで，11個の石の並びを数列パターンで表す方法について検討し別途報告した[1),2)]。又，11個の石の並びは，それぞれの目で黒石か白石か空石かであるから3^{11}通りある。これらの全てについて評価値を求めておいて表にし，判定にこれを引用する方法もプログラ

Total[ ]=1

Patern	A	X	B	C	D	E	F	O
A	なし	–	–	–	–	–	–	–
X	なし	なし	–	–	–	–	–	–
B	なし	なし	なし	–	–	–	–	–
C	飛び三	なし	なし	なし	–	–	–	–
D	なし	なし	なし	なし	なし	–	–	–
E	なし	なし	なし	なし	なし	なし	–	–
F	飛び四	飛び四	飛び四	飛び四	飛び四	飛び四	四々	–
O	なし	なし	なし	飛び三	なし	なし	飛び四	なし

Total[ ]=2

Patern	A	X	B	C	D	E	F	O
A	なし	–	–	–	–	–	–	–
X	なし	なし	–	–	–	–	–	–
B	飛び三	なし	なし	–	–	–	–	–
C	飛び四	飛び四	四三	四々	–	–	–	–
D	飛び四	飛び四	四三	四々	四々	–	–	–
E	なし	なし	なし	飛び四	飛び四	なし	–	–
F	なし	なし	なし	飛び四	飛び四	なし	なし	–
O	なし	なし	飛び三	飛び四	飛び四	なし	なし	なし

Total[ ]=3

Patern	A	X	B	C	D	E	F	O
A	三	–	–	–	–	–	–	–
X	なし	なし	–	–	–	–	–	–
B	飛び四	飛び四	四々	–	–	–	–	–
C	なし	なし	飛び四	なし	–	–	–	–
D	なし	なし	飛び四	なし	なし	–	–	–
E	飛び四	飛び四	四々	飛び四	飛び四	四々	–	–
F	なし	なし	飛び四	なし	なし	飛び四	なし	–
O	三	なし	飛び四	なし	なし	飛び四	なし	なし

Total[ ]=4

Patern	A	X	B	C	D	E	F	O
A	勝利四	–	–	–	–	–	–	–
X	四	なし	–	–	–	–	–	–
B	四	四	なし	–	–	–	–	–
C	四	四	なし	なし	–	–	–	–
D	四	四	なし	なし	なし	–	–	–
E	四	四	なし	なし	なし	なし	–	–
F	四	四	なし	なし	なし	なし	なし	–
O	勝利四	四	四	四	四	四	四	勝利四

Total[ ]=5

Patern	A	X	B	C	D	E	F	O
A	勝利五	–	–	–	–	–	–	–
X	勝利五	勝利五	–	–	–	–	–	–
B	勝利五	勝利五	勝利五	–	–	–	–	–
C	勝利五	勝利五	勝利五	勝利五	–	–	–	–
D	勝利五	勝利五	勝利五	勝利五	勝利五	–	–	–
E	勝利五	勝利五	勝利五	勝利五	勝利五	勝利五	–	–
F	勝利五	勝利五	勝利五	勝利五	勝利五	勝利五	勝利五	–
O	勝利五	勝利五	勝利五	勝利五	勝利五	勝利五	勝利五	勝利五

Total[ ]=6～9

Patern	A	X	B	C	D	E	F	O
A	長連	–	–	–	–	–	–	–
X	長連	長連	–	–	–	–	–	–
B	長連	長連	長連	–	–	–	–	–
C	長連	長連	長連	長連	–	–	–	–
D	長連	長連	長連	長連	長連	–	–	–
E	長連	長連	長連	長連	長連	長連	–	–
F	長連	長連	長連	長連	長連	長連	長連	–
O	長連	長連	長連	長連	長連	長連	長連	長連

図4　各パターンの相互関係

ム化している[2]。これら二つの方法と今回のパターン表を用いた方法の探索時間を図5の配石で比べてみた。探索時間とは図5の配石で碁盤の一つ一つの目に石が置ける所に石を置いていったと仮定して，それぞれの手で評価値を求め，その中で，最もよいと思われる手を決定するまでの時間である。

数列パターンにより判定する方法	: 304 秒	使用計算機
3^{11} 通りの表を用いる方法	: 103 秒	JCC（日本電算機）JS 5 /85
本法のパターン表を用いる方法	: 98 秒	OS : SunOS　4．1．4

3^{11} 通りの表を用いる方法はメモリを多く使うので，逐一その都度計算する数列パターンにより判定する方法より早くなるのは当然だが，本法のパターン表を用いる方法は 3^{11} より相当少ないメモリしか使わないのに，より小さい時間で探索ができた。

なお，ここでは，各手に次ぎのような評価値を仮定した。

“三のとき”　: 6937
“四のとき”　: 2325
“飛び三”　: 581
“飛び四”　: 145
“飛び五”　: 9
“長連になりそうなとき”
　: 2
“置けない手”　: 1
“勝ち”　: 1000
“通常の手”　: 36

どの手がどの手より評価値を高くするかは重要なことであるが数値そのものには余り意味がない。

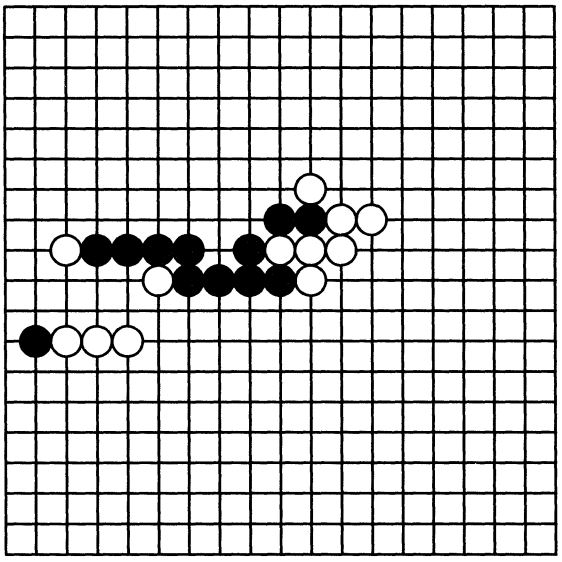

図5　検索プログラムの実行時間テスト用石の並び方

図6にプログラム xgomoku のゲーム画面を示す。

3．ま と め

五目並べのプログラムにおいては石を一手置くごとに，そこに石がおけるかどうか，又，四三になっているかどうかといった判定が必要で，そのために，周りの石の並びを探索して評価する効率的な方法を見つけることがゲーム全体を高速化するのに重要な課題の一つである。本稿では，すべての石の並びを同じ評価値を与える可能性のあるもの同志に分類して八つのパターンに分けた。打った石を中心にして，一方向を考え，そこの線上に連続して並んだ石の数と打った石の両方向のパターンの対からその手の評価値が判断できるように表を作った。この判定表に要するメモリ量は我々の既発表のものより小さく，一例についてではあるが，短い探索時間で判定できることも示した。目下のところ，思考ルーチンには，先読みするといった高度な技法が入っていないが，今後改善していきたい。

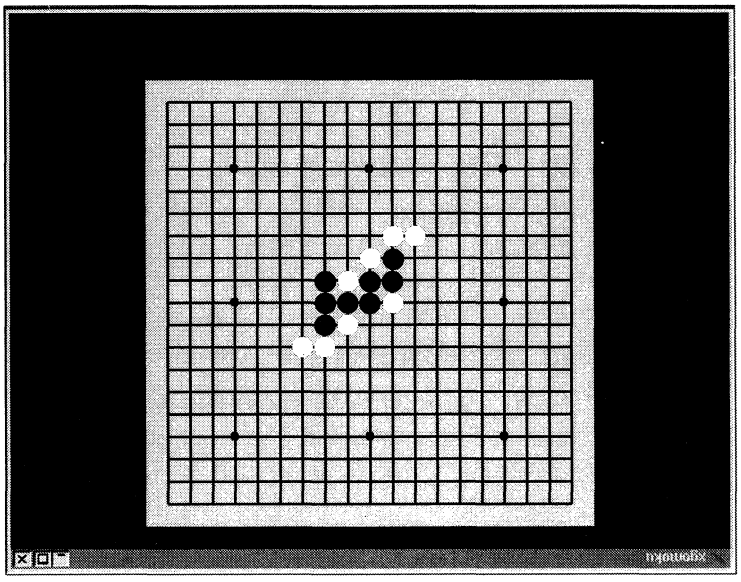

図6　xgomoku のゲーム画面

参考文献

1）奥原竜也，松田秀雄，宮腰　隆，中嶋芳雄：連珠（五目並べ）の勝負判定アルゴリズムについて，平成8年度電気関係学会北陸支部連合大会講演論文集　E-9（1996）.
2）奥原竜也：五目並べのプログラムの基礎研究1，平成8年度富山大学工学部卒業論文

平成9年度電気関係学会北陸支部連合大会で一部発表