卒業論文

コンピュータ囲碁の思考部の作成[1]

氏　　名：三本木　啓
学籍番号：2260060052-8
指導教員：山崎 勝弘　教授
提 出 日：2010 年 2 月 19 日

立命館大学　理工学部　電子情報デザイン学科

内容梗概

本論文ではモンテカルロ法を取り入れた囲碁プログラムの作成を目的としているが、評価関数の作成の難しさなどの理由から今回はその最初の段階として“コンピュータ囲碁の入門”[1]に付属しているサンプルプログラムをベースにコンピュータ囲碁システムの思考部を作成した。作成したプログラムは「連、結線、群」、「地の認識」、「地を利用した候補手の生成」である。群は連と結線を組み合わせたものであり、地を認識する際に使用する。地の認識は候補手を生成するために使用する地と終局時に勝敗を決めるための地の二つを作成した。また、候補手の生成方法には様々あるが、本論文では地を利用して候補手を生成する手法のみを使用している。今回作成したプログラムとサンプルプログラムを対戦させた結果、今回作成したプログラムが全勝し、思考力の向上が確認できた。

目次

図目次

表目次

1. はじめに

コンピュータ囲碁に関する研究は 50 年近く行われてきている。その研究は囲碁を打つ対局プログラムだけでなく、詰碁やヨセの問題を解くプログラム、人間が上達するための囲碁講座や問題集のプログラム、インターネット碁会所など囲碁に関するたくさんのプログラムが存在する。

囲碁は東洋発祥のゲームだが、人工知能の研究対象の一つとしてとらえられていたため、コンピュータ囲碁の研究と開発は東洋ではなく、海外勢が先導してきた。日本では囲碁だけでなく、ゲームプログラミングをテーマにした研究自体がほとんど行われていなかったが、1980 年頃からコンピュータ囲碁、将棋などの論文が散見されるようになった。
下記の表 1 はコンピュータ囲碁の歴史を簡単にまとめたものである[1]。

表 1：コンピュータ囲碁の歴史

1962 年	Remus による初めてのコンピュータ囲碁の論文
1969 年	Zobrist による初めての対局囲碁プログラム（強さは約 38 級）
1979 年	Reitman と Wilcox のプログラム Interim.2（強さは約 15 級）
1984 年	初めてのコンピュータ囲碁の大会（USENIX）が開催される
1985 年	日本の国家プロジェクト「第五世代コンピュータプロジェクト」にてコンピュータ囲碁が研究テーマの一つに取り上げられる
1986 年	初めてのコンピュータ囲碁の世界大会（Ing Cup）が開催される
1995 年	日本棋院が初めて囲碁プログラムに級位を認定（5 級）
1996 年	日本で初めてコンピュータ囲碁の研究による工学博士号が授与される
2000 年	コンピュータによって四路盤の必勝法が解明される
2001 年	囲碁プログラムが初めて初段に認定される
2003 年	コンピュータによって五路盤の必勝法が解明される

本研究で数あるゲーム中から囲碁を選択した理由には「評価関数の作成の難しさ」「複雑さ（局面数）の多さ」などがあげられる。評価関数とは、局面の状況から有利不利を数値化する仕組みである。具体的には、どれだけの地を獲得しているか、アゲハマの数はいくつか、後に地になりそうな場所はあるかなどを対局中の盤面や棋譜等の石の配置から計算することが評価関数にあたる。また、「複雑さ（局面数）」とはゲームプログラミングの分野において、コンピュータによってゲーム終了まで選択可能なすべての手を調べつくすことに必要な手の数のことである。複雑なゲームであればあるほど先の局面を正確に認識することが難しくなり、それを見極めていくことはそれだけ人間の思考に近づくことにつながる。表 2 はチェス、将棋、囲碁の 3 つのコンピュータゲームの複雑さとレベルを比較したものである[1]。

表 2：ゲームの複雑さとレベルの比較

ゲーム	複雑さ（局面数）	レベル
チェス	10 の 120 乗	トップ 100 のレベル（人間の世界チャンピオンを破った）
将棋	10 の 220 乗	アマチュア高段者レベル
囲碁	10 の 360 乗	アマチュア 2,3 段レベル

表 2 から囲碁はチェスや将棋に比べて非常に複雑であることがわかる。複雑であるために先の局面を正確に認識することが難しくなり、評価関数の作成も難しくなる。2008 年には「Crazy Stone」というプログラムがプロ棋士の青葉四段に七子で勝って、アマチュア四、五段の実力があるかもしれないと言われるものもでてきたが、チェスや将棋に比べると低いレベルにある。つまり、まだまだ開発の余地があるということになる。これらの理由から囲碁を選択した。

また、ここ数年でコンピュータ囲碁はモンテカルロ法という、乱数を発生させ、終局までシミュレートし、評価するという方法を取り入れることで急激に進歩してきており、本研究でも将来的にはモンテカルロ法を取り入れた対局プログラムの作成を目的としている。しかし、対局囲碁プログラムの作成は非常に難しいため、今回はその最初の段階として、思考部の一部、石の生死の判定の部分に使用される『連』、『結線』、『群』や『群を使った地の認識』、『地を使った候補手の生成』を作成した。また、思考部以外の部分は“コンピュータ囲碁の入門”[1]に付属しているサンプルプログラムを利用した。

2. コンピュータ囲碁システム

2.1 囲碁のルール

ルール 1. 黒白交互に打つ

囲碁の対局に必要なものは碁盤と碁石である。

碁盤は主に 19 路盤、13 路盤、9 路盤などを使う。19 路盤で説明すると縦横それぞれ 19 の線が交わっており、合計で 361 の交差点ができ、その交差点に石を置いていくことになる（図 1)。19 路盤と 13 路盤、9 路盤の違いは縦横それぞれの線が（13×13)、(9×9）になるという点のみである。

石の種類は黒と白の 2 種類で先に打つ人（先番）が黒石を使用し、先番からうちはじめる。二人で交互に打つが、パスをすることもできる（終局間近で打つと損をする場合など)。一度打った石は他の場所へ動くことはない。

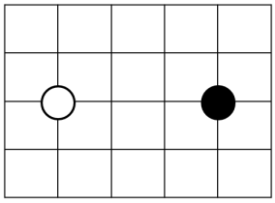
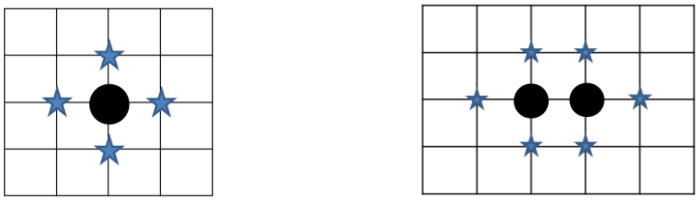


図 1：石の打つ場所

ルール 2. 上下左右の点を囲まれた石は取り除かれる

図 2 の黒石の上下左右の点（★）に白石を置かれると取られてしまう。



(a)囲まれた点 1　　(b)囲まれた点 2

図 2：上下左右を囲まれた点

図 3 のように★のある位置に相手の石（この場合白石)、図 3 左図で言えば、黒石は★のところに白石を置かれると上下左右の点を囲まれたことになる。このような状態を“あたり”といい、次にその最後の点（★）に白が打つとこの石を取ることができる。取った後は右図のようになる。

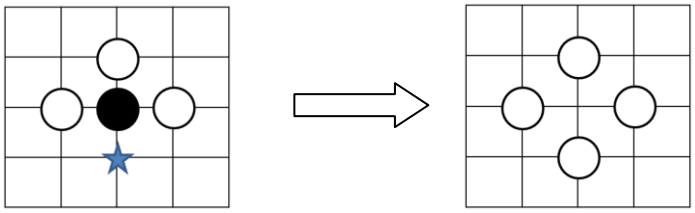


図 3：あたり 1

取った石のことをハマといい、対局終了時に使用する。また、石の数が増えても要領は同じで、上下左右の点をすべて塞げば取ることができる。図 4 で言えば、黒石は★に白石を打つと取ることができるということである。

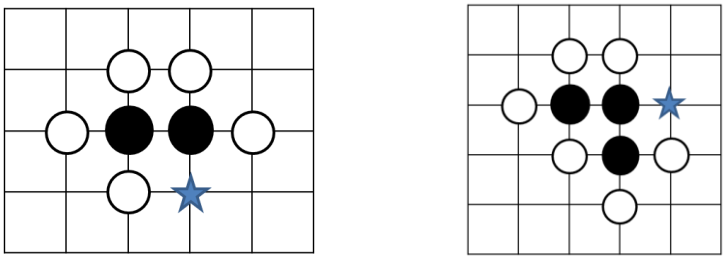


図 4：あたり 2

ルール 3. 4 方向を囲まれているところへは置けない

図 5 の×のあるところには打つことができない。なぜなら×のところはすでに上下左右の点が囲まれており、そこへは存在することができないからである。このような場所（×）のことを着手禁止点（眼）という。

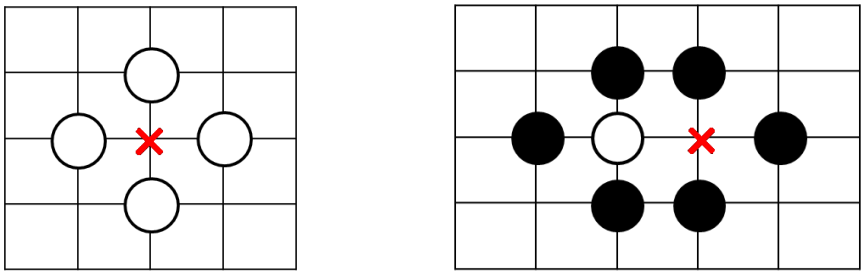


図 5：石を打てない

ただし、図 6 のような場合黒は打つことができる。なぜなら、打った黒石で相手の白石を取ることができ、石の上下左右を囲まれた形にならないからである。

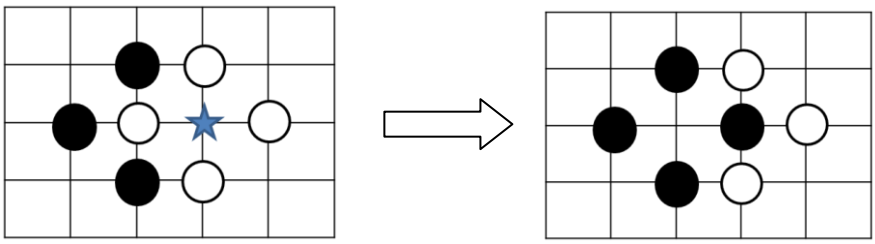


図 6：着手禁止点例外

ルール 4. 同型反復禁止（コウ）

図 6（右）で黒が取った後の場合を考える。同型反復禁止（コウ）というのはこのような場面ですぐ直後に白が黒を取り返せないというルールである。なぜなら取り返すとまた図 6（左）のような同じ場面になりまた黒が取り返して・・・・という繰り返しになり、終わらなくなってしまうからだ。

このような繰り返しを防ぐためのルールを「コウ」と呼び、コウになるとしかたなく他の場所に打つことになり、それに相手がつきあってくれると次には取り返すことができる。つきあうか、つきあわないかの判断などコウ争いは囲碁の中でも難解なものの一つとされている。

ルール 5. 勝敗は地の多少で決まる

陣地を取り合うことが碁の戦いだが、互いにパスを選択する（もう取り合う陣地が無くなる）終局となる。勝敗は「地」の多少で決まる。

地とは自分の石で囲まれた領土のようなもので、相手が侵入してきてもその石を取ることができる地域ことである。地については「3.2 地の認識」で詳しく説明する。例えば、図 7 の ■や□のところが地になる。■は黒の地、□は白の地となり、この四角一つを一目と数え、この目数が多いほうが勝者となる。また、取った石（アゲハマ）は最後に取った石の数だけ相手の地を減らすことができる。

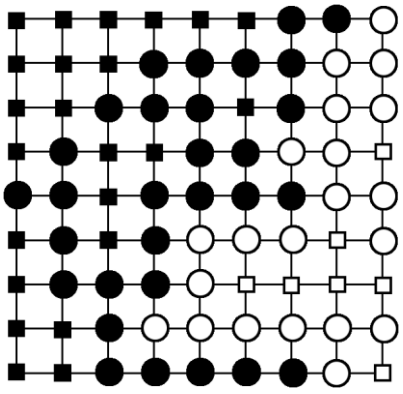


図 7：地の例

また、対局は黒から打ち始めるので先番の黒が常に有利となる。その有利さが何目くらいかということは難しいのだが５目から６目といわれてきた。したがって白を持つ人は一般的に五目半や六目半程度のハンデをもらう。このハンデのことを「コミ」という。

2.2　システム構成

図８にコンピュータ囲碁のシステム構成を示す。

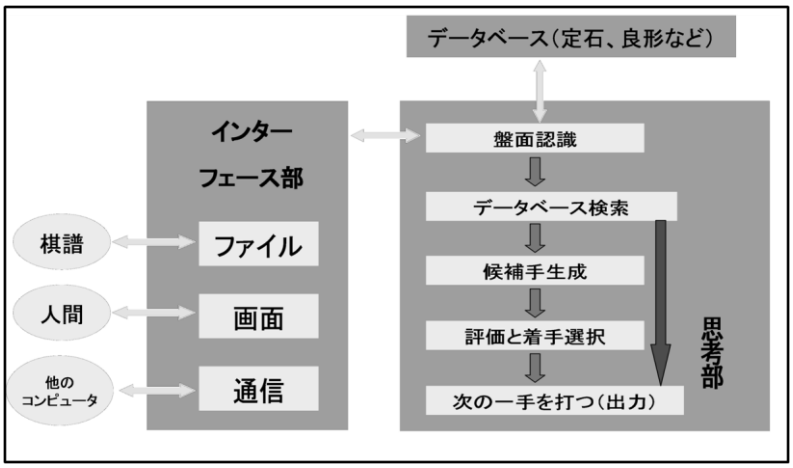


図 8：コンピュータ囲碁のシステム構成

まず、インターフェース部から棋譜や人間、他のコンピュータの打った碁盤の状況を読み取る。そして、思考部で処理をしていくという流れになる。思考部の中の流れは以下のようになっている。

(1) 盤面認識

まず、インターフェース部から碁盤を読み取り、その盤面から石の状況（群の有無、コウの有無など）を読み取る。

(2) データベース

データベースには石の結線のパターンや定石、好形等のパターンをあらかじめ用意しておく。

そして、盤面認識をした際にデータベース内にあるパターンと一致するものがあった場合、それに対応した処理を行う。例えば、ある石の配置を認識した場合、次の一手を打つ場所を定めたパターン（定石）を用意しておき、もし盤面にそのパターンと一致する場所を発見した場合、あらかじめ定めておいた場所に着手するようにする。

(3) 候補手生成

盤面にデータベースの中の次の一手を打つパターンと一致するものがなかった場合、次の手の候補を考える必要がある。

石の活き死にや地から次の一手の候補を選ぶ。

(4) 評価と着手選択

各候補手を評価し、その中で最良のものを選ぶ。

(5) 次の一手を打つ

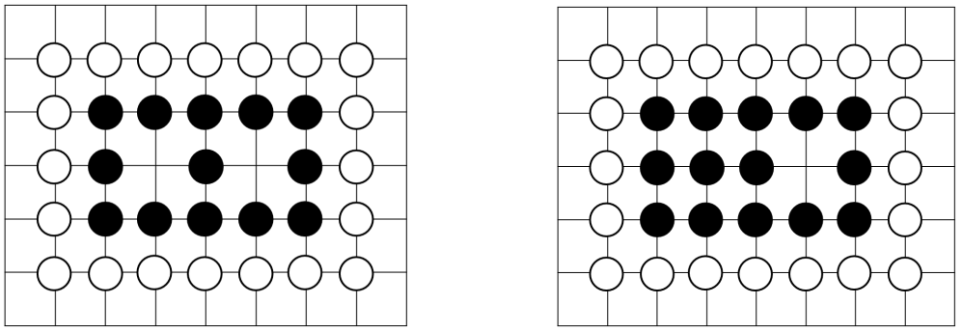
データベースのパターンや候補手の中から選ばれた手を次の一手として打つ。

3. 思考アルゴリズム

3.1 生死の判定

囲碁を打っていく際に石の生死というものがとても重要になってくる。

石の生死とは簡単にいえば、近くにある石の集合（群）が相手に取られてしまいそうか否かということである。例えば図 9(a)のように黒石の中に着手禁止点（眼）が２つ以上ある場合、その石の集合（群）は絶対に取られることはない。このように絶対に取られないような状態にあることを「生きている」と言う。逆に図 9(b)のように黒石が取られてしまうような状態にあることを「死んでいる」と言う。この生きているか死んでいるかの判断がプログラムを作成する際に非常に難しい点である。

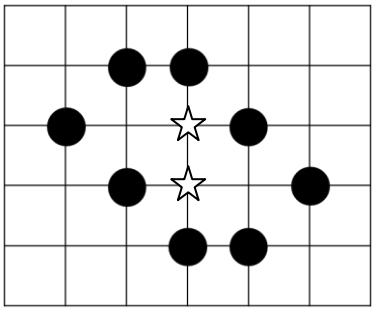


(a)石の生死１　　(b)石の生死２

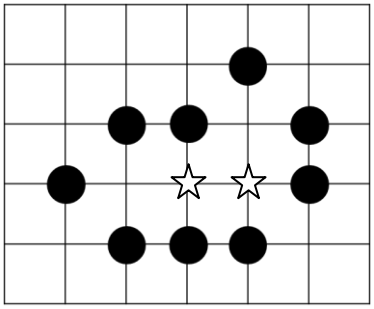
図 9：石の生死１

図 10 は両方とも眼が二つできる形である。つまり、石が「生きている」とされる形である。正確にはこの石が相手の石によって囲まれているかどうかによって「生きる」か「死ぬ」かが変わってくる。この石の集合が「死ぬ」場合というのは、周りを囲まれ、石を取られてしまい、この形が崩れてしまう時のことである。この形が崩されなければ、これらの形は「生きている」ということになる。

ここで、図 10(a)(b)が何故生きているかについて簡単に説明する。図 10 にはそれぞれ☆が二つ付いているが、黒はこのうちどちらか一つに打てば眼が二つできることになる。もしも次が白番で、☆に打ってきた場合、黒は残りの☆に打たなければならないが、☆が二つあるので、黒は必ず☆に打つことができる。つまり、この形になった時点でこの群は「生きている」ことになる。



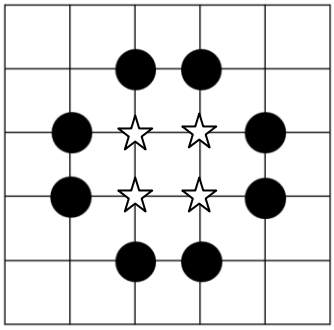
(a)2 眼の例 1



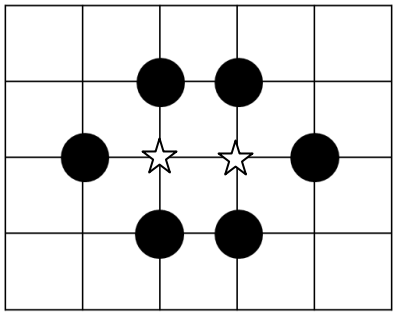
(b)2 眼の例 2

図 10：2 眼の例

次に、図 11 は両方共、眼が 1 つしかできない形である。つまり、このままでは「死んでいる」石になってしまう形である。この形から石が「生きる」為には、もう一つ眼ができるように石を置いていかなければならない。先ほどの図 10 とは逆に相手が☆のある点に打ってしまうと眼が一つしかできなくなってしまう。それぞれ☆が 2 つ以上あるため、黒番、白番に関係なく☆のある点に打たれてしまい、眼が一つだけとなってしまう。



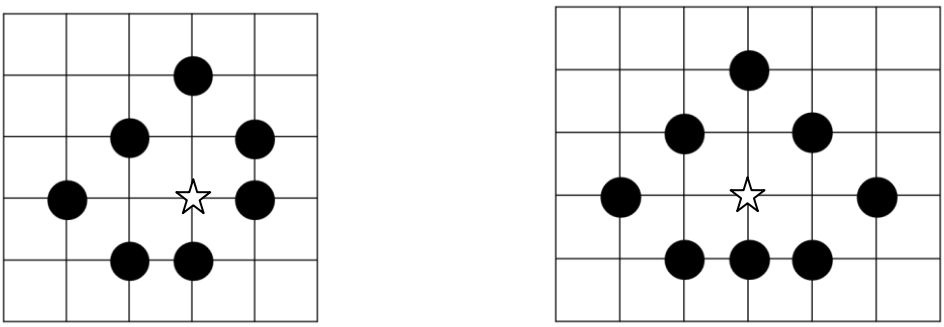
(a)1 眼の例 1



(b)1 眼の例 2

図 11：1 眼の例

最後に「生きている」か「死んでいる」かが分らない形を図 12 に示す。これらは眼が 1.5 眼という見方をする。つまり、次が黒番か白番によって生きるか死ぬか分らないということである。黒が☆に打つと 2 眼で「生き」、白が☆に打つと 1 眼で「死に」となる。



(a)1.5 眼の例 1　　(b)1.5 眼の例 2

図 12：1.5 眼の例

囲碁を打っていくとこのような石の生死が非常に分かりづらい状態が現れてくる。また、このような状態が複数現れ、どの群を生かすかなどの判断も必要になってくる。

このように、石の生死の判定が最も難しく、囲碁システムでも非常に重要である。

3.2　地の認識

2.1 で述べたように、囲碁の勝敗は地の多少によって決まる。地とは日本語規約[1]では「一方のみの活き石で囲んだ空点を『目』といい、『目』以外の空点を『駄目』という。駄目を有する活き石を『セキ石』といい、セキ石以外の活き石の目を『地』という」と定められているが、これは地の厳密な定義であり、一般的に使われている「地」の意味とは多少異なる。一般的に使われている地とは「今は確定的な地ではなくても、終局時には地になると予想される場所」のことも含まれている。なぜ、完全に地となっている場所以外も地と考えなければいけないのかというと、このように考えなければ対局中にどちらが何目勝っている、どちらが優勢か、という形勢判断ができない。

実際に地を認識させるには群の生き死になどを考えて、認識させていかなければならない。この地の認識は現在最もレベルの高いコンピュータ囲碁プログラムでも 100%の精度のものは存在していない。この精度を高めることによって、盤上の形勢判断が正確に行われるようになり、より最適な手を打つことができるようになる。

3.3　候補手の生成と評価

囲碁は、着手禁止点を除けば、盤面の空点は全て候補手となる。しかし、それでは候補手が多すぎて評価値を求めるのに時間がかかりすぎてしまう。したがって、明らかにふさわしくない手を評価する事をはぶき、良い手しか候補手としないようなプログラムを作り、評価値を求める時間を短縮させる必要がある。

人間の囲碁上級者は盤面を見ただけで、瞬時に打ちたい場所を数か所に絞れるといわれている。彼らは無意識のうちに相手の石を取る手や定石の手など、いろいろな知識や先読みの結果から候補手を生成しているそうだ。

しかし、プログラムでほんの数か所だけしか候補手を生成しないものを作るのは難しすぎるので、人間の囲碁上級者を倣い、盤上の状況を認識し、そこから得られる情報によって候補手を生成するプログラムを作っていけばよいと考えられる。

候補手を生成するアルゴリズムは 3 つに分類できる。[1]

1. 認識結果を利用する

   連（縦横につながった同じ色の石）、結線（石と石の間に引く仮想的につながっているとする線）、群の認識の段階で得られた情報を使って候補手を生成する。

   例えば、群を使い、地を認識し、その結果を利用して形勢を評価し、候補手を生成するようなこと。

2. 候補手専用のアルゴリズムを利用する

   人間は、布石の手や模様の手などは石の強さ、石と石の間の距離をもとに候補手の位置を決めている。このような手を求めるアルゴリズムを用意しておく。

   例えば、辺への割うちの手は、相手の石と石の間の座標を計算すれば求めることができる。割うちの手とは図 13 のように「相手が大きな模様を作るのを阻止するような手」のことである。

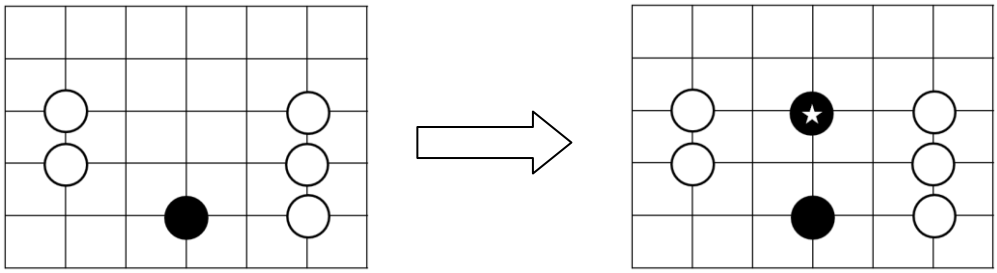
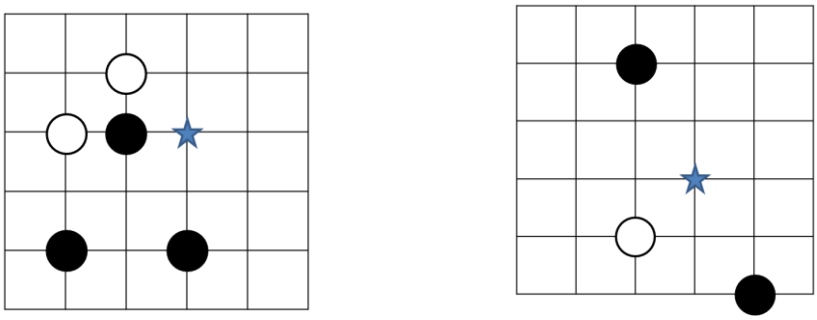


図 13：割り打ち

3. パターン（データベース）を利用する

   定石や手筋などを生成するために、石の配置と候補手の位置を記述したパターンを用意する。

   対局中、常に盤面とそのパターンを比較し、一致したものを検出した場合、その手を打つようにする。例えば、図 14 のようなパターンを用意しておくとする。このパターンは黒石の候補手を示したものである。もし盤上から図 14 のパターンと一致するものを検出した場合、★の位置に打つようにしておく。このようなパターンをいくつか用意することで、人間の打ち筋に近づけることができる。



(a)パターン例 1　　(b)パターン例 2

図 14：パターン例（黒番）

本研究ではパターンを利用し、結線（石と石の間に引く仮想的につながっているとする線）を認識している。また、候補手の生成には地の認識結果から形勢を判断し、評価している。

4. 思考部の作成

4.1 連について

強い囲碁プログラムを作るためには、どの石の集合が死んでいるか、生きているか、重要かなどといった、碁盤の石の並びだけではわからない盤面の状況を正確に認識する必要がある。盤面の状況を正確に認識することによって、どの手が有効かということを判別できる。本章では石の生死を考える際、最も基本的な単位である「連」の作成方法について説明する。

(1) 連とは

連とは縦横につながった同じ色の石の集合のことである。この「連」という表現は囲碁用語ではないが、多くの囲碁プログラムの作者は「連」と呼んでいるので、ここでも「連」と呼ぶことにする。具体的には図 15 の黒石、白石それぞれの集合のことをいう。

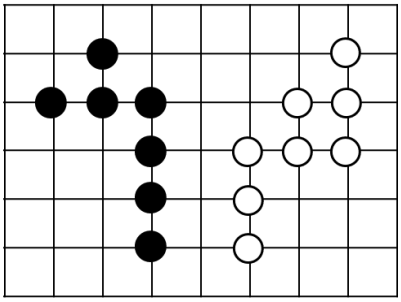


図 15：連

(2) 連のプログラムの作成方法

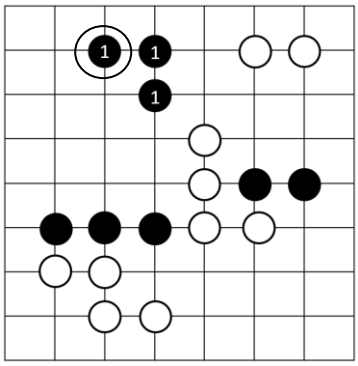
・すべての盤上の点を検索する。

・石がある場合、その石に番号が既につけられているかどうか調べる。

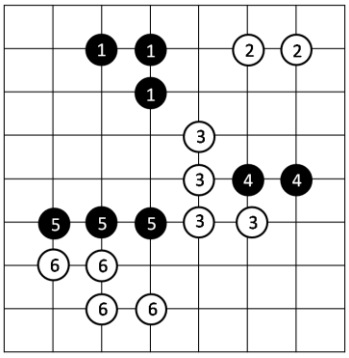
・番号が付いている場合、次の石を調べる。

・番号が付いていない場合、その石に番号をつけ、その石の上下左右に同じ色の石があるかどうかを調べ、石があった場合は同じ番号を付ける。（丸の付いた石に番号が付いていない場合、その石に番号を付ける。この場合１。そして、その石の上下左右を調べ、そこに同じ色の石があれば１を付け、さらにその石の上下左右を調べ同様にして、1 を付けていく。図 16(a)）

・上下左右に同じ色の石がない場合（連なる石に番号を付け終わった場合）、次の石を調べ、次の番号をつける。これを繰り返すと以下（図 16(b)）のようになる。



(a)連の作成 1



(b)連の作成 2

図 16：連の作成

4.2 結線について

(1) 結線とは

結線とは石と石の間に仮想的な線を引いて、石どうしを結んでいるとするデータ構造のこと。具体的には図 17 の丸で囲った石同士のように少し距離を置いた位置にある石をつながっているとするデータ構造である。

この認識ができるようになることで、近くにある石を一つの集合としてとらえることができる。

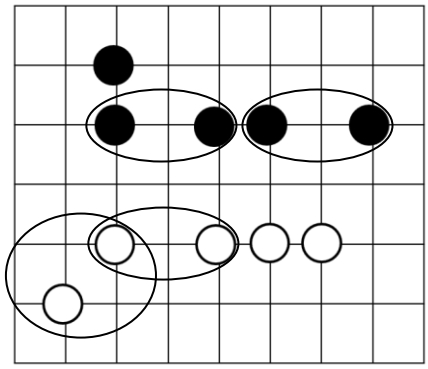


図 17：結線の例

(2) 結線のプログラムの作成方法

結線のパターンをいくつか用意しておき、そのパターンが盤面に見つかった場合、そのパターンで示してある位置を結線として認識させる。また、盤面には黒の結線が見つかった場合は&を白の結線が見つかった場合は＄を表示させる。パターンは必要に応じてそれぞれが回転したものを用意させておく。

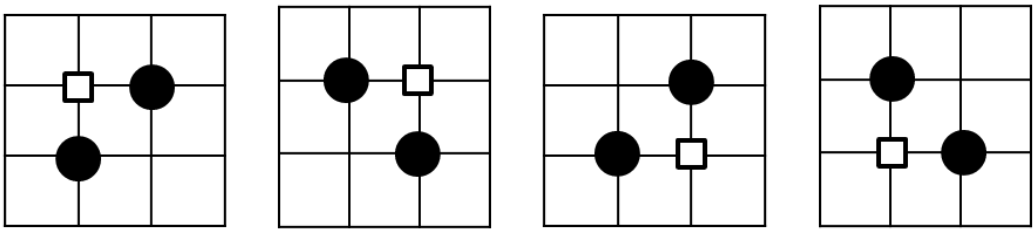
▲は空点、もしくは自分と同じ色の石。

□ は空点。

となっており、このパターンに一致した場合、□が結線として認識される。
結線のパターンは以下の 16 個である。図 18－図 22 に結線のパターンを示す。

二つの石が斜めに配置されているとき、両方の石が接する場所が一つでも空点の場合、その空点を結線としている（図 18）。

ケイマや大ゲイマのパターンを示す二つの石は、1 つまたは 2 つ間をあけて斜めに配置している。この場合、結線となる位置はケイマでは間の 2 つの石。大ゲイマでは間の 4 つの石とする。

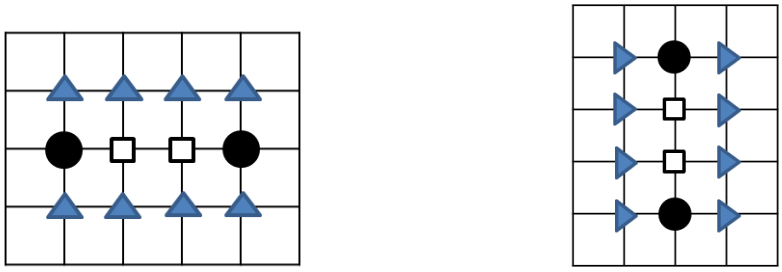


(a)斜め 1　(b)斜め 2　(c)斜め 3　(d)斜め 4

図 18：斜め

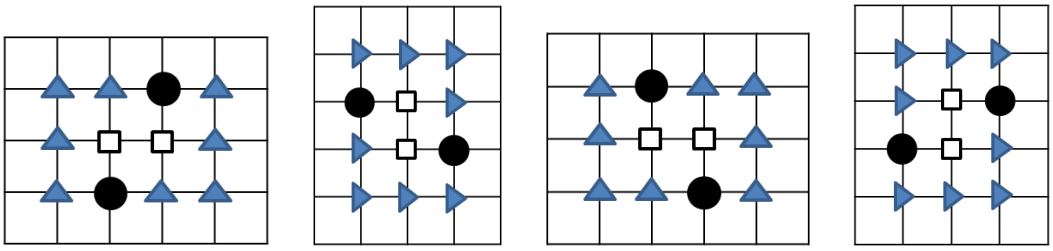
(a)一間 1　(b)一間 2

図 19：一間とびのパターン



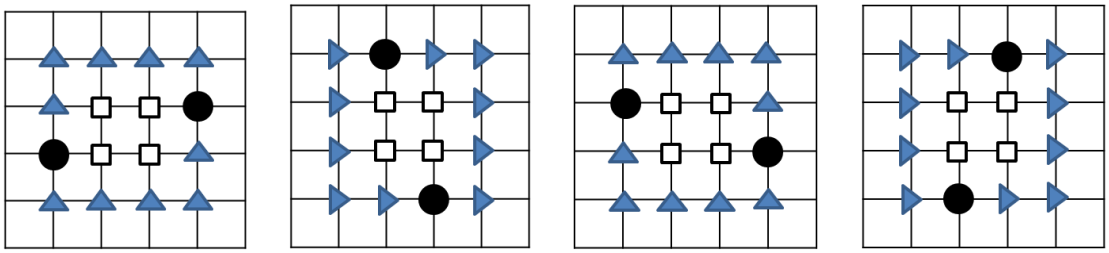
(a)二間 1　(b)二間 2

図 20：二間とびのパターン



(a)ケイマ 1　(b)ケイマ 2　(c)ケイマ 3　(d)ケイマ 4

図 21：ケイマのパターン



(a)大ゲイマ 1　(b)大ゲイマ 2　(c)大ゲイマ 3　(d)大ゲイマ 4

図 22：大ゲイマのパターン

4.3　群について

人間は囲碁を打っていく時、一つ一つの石の生死を考えているのではなく、近くにある複数の石をひとまとまりとした単位で生死を考えている。この単位で生死を正確に認識することで、局面をより正確に認識することができる。

ここでは石の生死を考える際にもっとも重要な単位である「群」の作成方法について説明する。

(1) 群とは

群とは近くにある複数の同じ色の石をひとまとまりにした単位のことである。石の生き死を判断するときはこの単位で考えていく。

(2) 群のプログラムの作成方法

群は「連」と「結線」を組み合わせて作成する。

① まず、盤上から結線を作る石の並びを認識する。

② 結線を作る石の並びを認識したとき、その石の連の番号を記憶しておく。

図 23 で言えば、1 と 2 を記憶しておく。(図 23 は石を連と結線で表したものである)

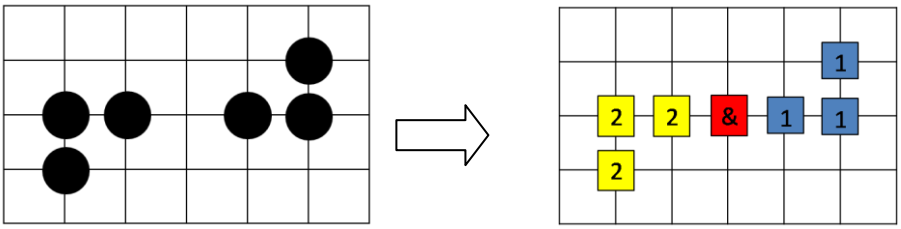


図 23：群の作成 1

③ 記憶した数字を比較して、小さいほうの数字に変換する。(1 のほうが 2 より小さいので 1 に統一する。) これで群が完成。図 24 は完成した群である。

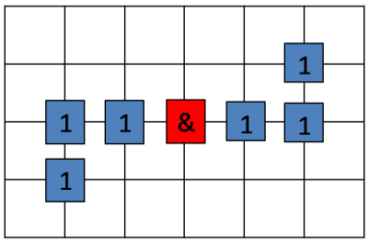


図 24：群の作成 2

4.4 地の認識

本研究で作成した地の認識プログラムは 2 種類ある。

一つ目は候補手生成時に使用する地である。二つ目は終局時に使用する地である。

まず、候補手生成時に使用する時から説明する。

(1) 候補手生成時に使用する地

各石から距離 2 以内をその石の地と考える。イメージとしては一つ一つの石がそれぞれ同じだけの力を持っていると考えてもらいたい。図 25 の■が地となり、石一つが最大 12 目分の力を持っていることになる。

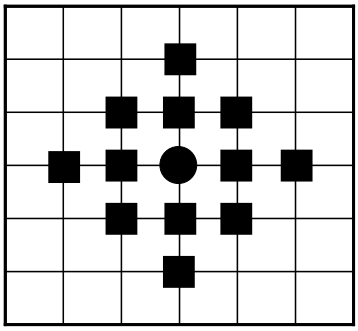
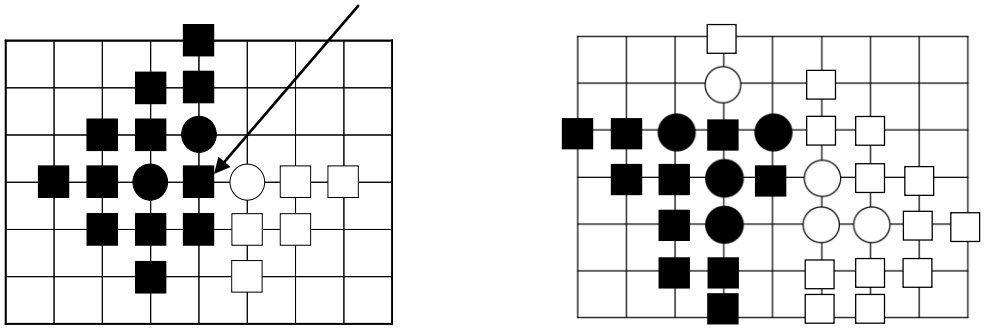


図 25：石一つの地

ここで、石が近くに複数ある場合を考える。ある場所で黒、白両方の地が重なった場合、影響力の強い方をその場所の地とし、力の強さが等しければ、打ち消しあい、そこはどちらの地でもないようにしている。例として図 26 を見てもらいたい。図 26(a)を使って説明していく。左図の矢印の指している地の部分は黒、白両方の力が及んでいる場所である。この場所は 2 つの黒石からそれぞれ力を受け、黒石の力が 2。白石からは石が一つなので力が 1。つまり、黒石の力が白石の力を上回っていることになるので、黒の地となっている。



(a)複数の石の地 1　　(b)複数の石の地 2

図 26：複数の石の地

(2) 終局時に使用する地

まず、対局が終了した時に碁盤をみて、弱い群を碁盤から削除する。また、削除された群の結線も削除する。すると以下のようになる。図 27 は弱い群を消去したものである。

＊現在の弱い群の定義は『石の数が 4 つ以下のもの』としている。

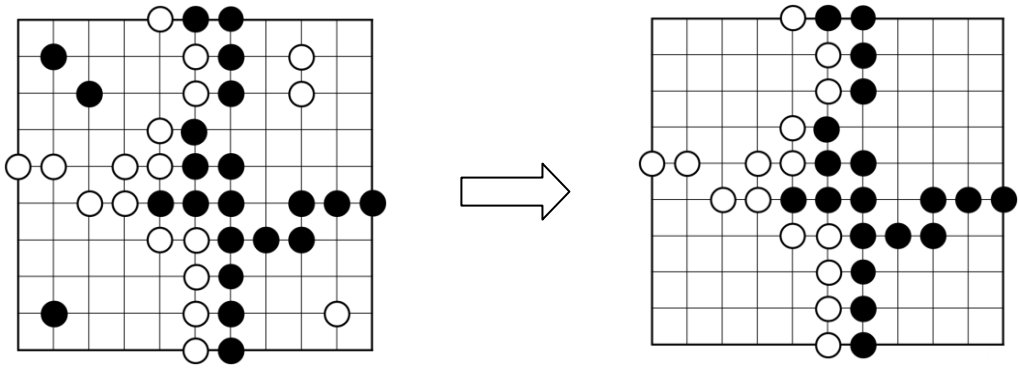


図 27：地の作成 1

次に結線をそれぞれの地とする。（&は黒の結線、$ は白の結線）

図 28 は結線を時に変換する様子である。

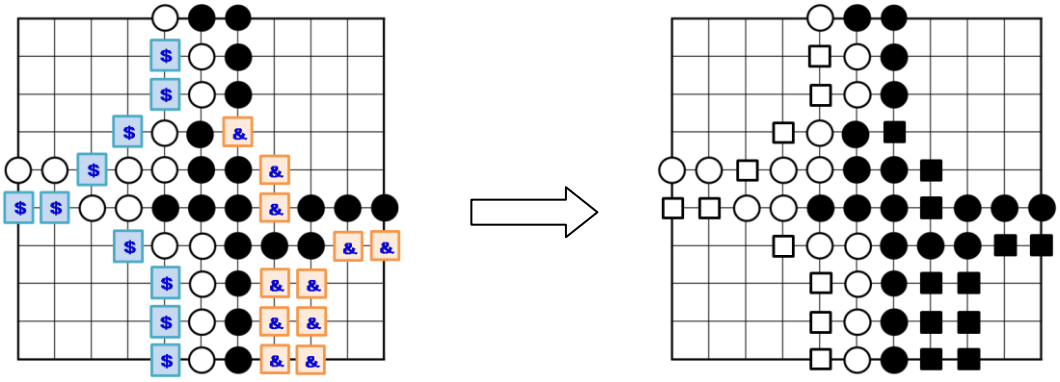
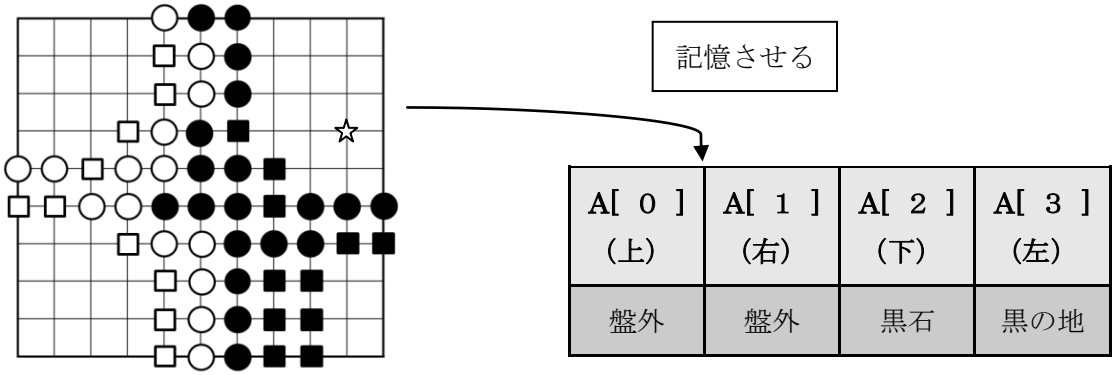


図 28：地の作成 2

ここからまずは黒の地を認識させる。方法としては碁盤の空点を検索し、空点をみつけた場合、その点から上下左右に手を伸ばして、『黒石、白石、黒の地、白の地、盤外』のどれに当たったかをそれぞれ記憶させる。この 4 点を調べ、記憶されているものが『盤外』が 2 つ以下かつ『白石、白の地』が 2 つ以下なら黒の地とする。例えば、図 29 の☆の点の場合、記憶されるのは以下のようになるため、黒の地となる。



A[0]（上）	A[1]（右）	A[2]（下）	A[3]（左）
盤外	盤外	黒石	黒の地

図 29：地の作成 3

白の地の認識も同様にして行うと、次のようになる。
図 30 は終局時の碁盤の認識結果の例である。

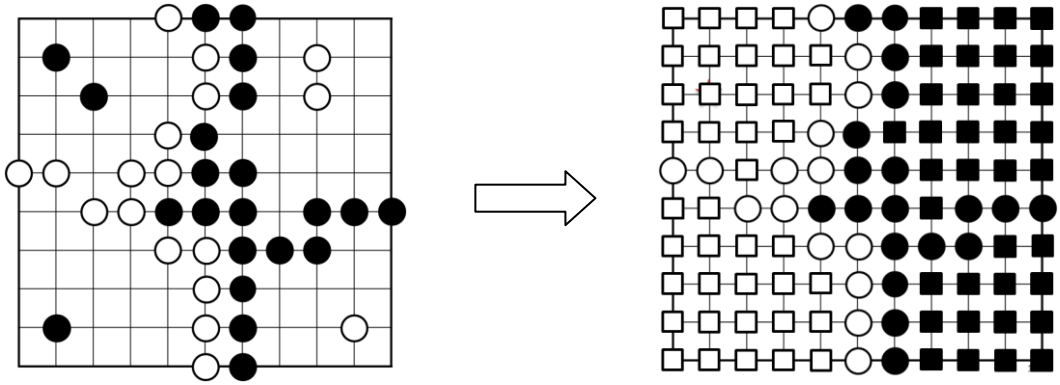


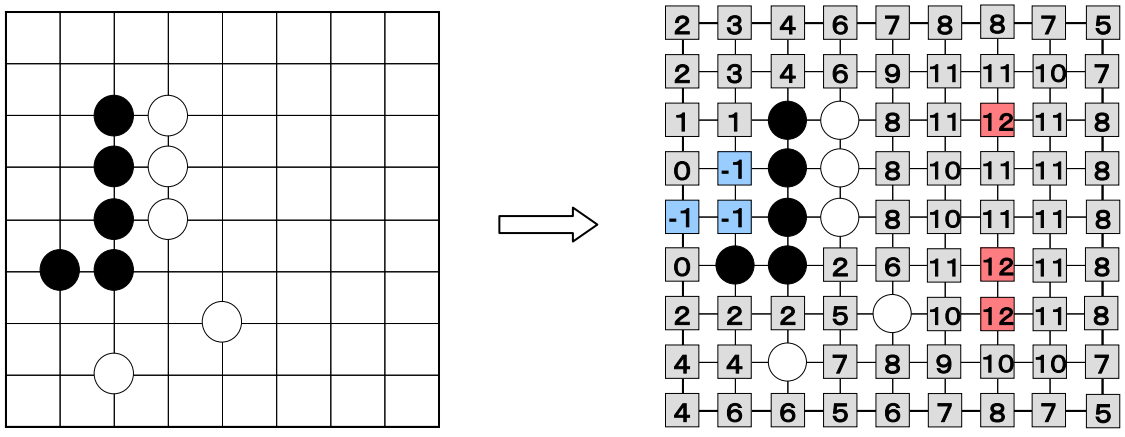
図 30：地の作成 4

4.5　候補手の生成方法

候補を生成する際には 4.4(1)の 候補手生成時に使用する地を利用して、次にどこに打てば自身の地を増やすことができるか考えるようにする。

方法としては、盤上の全ての着手可能な点に仮想的に石を打ち、現在の状況から何目地が増えたかを考える。

図 31(b)は碁盤に黒石をおいた時に、地が何目増えるかを表したものである。最高の評価値は(7,3), (7,6), (7,7)の 12 である。コンピュータはこの 3 点からランダムに 1 点を選び着手する。この処理を複数回繰り返すと、碁盤に 0 以下の数値しかでない局面が現れる。そのような局面になった場合はパスを選択させる。



(a)碁盤　　(b)地の増減

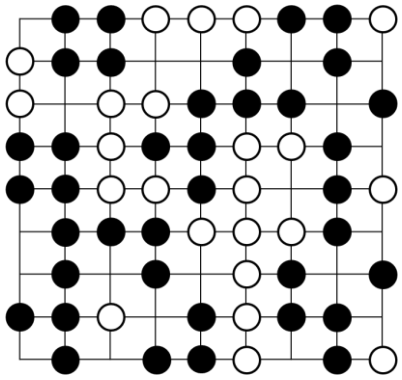
図 31：候補手作成

5. プログラムの評価

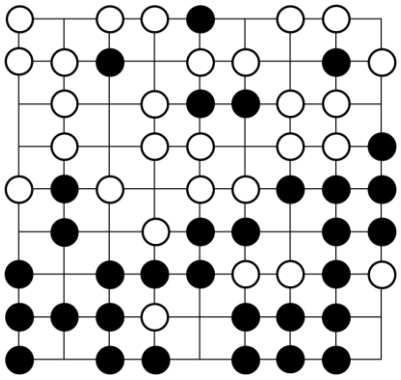
本章では今回作成したプログラムと作成する際に参考にしたサンプルプログラム[1]を対戦させ、プログラムの評価を行う。(黒は作成したもの。白はサンプル)

サンプルプログラムの着手方法は全ての着手可能な点をランダムに選ぶというものである。対戦結果は今回作成したプログラムの 10 戦 10 勝という結果になり、サンプルプログラムと比べて随分とレベルの高いものになっていることが分かる。しかし、相手が単純な乱数のみを使ったプログラムのため、打ち筋がでたらめで非常に弱いものなので、このプログラムの評価をするには十分ではない。したがって、今回作成したプログラム同士を対戦させた場合でも評価を行う。

この対戦では 10 戦中黒白それぞれが 5 勝ずつするという結果になり、黒番白番の力の差は見られなかった。図 32 は対戦結果であるが、作成したプログラムの対戦結果は碁盤を二分するような結果になることが多かった。これはそれぞれが候補を選ぶ際、地が最も増える場所、つまり、より広い場所に打つようになっているため自然と二分されているのだと考えられる。このままでは、盤上の様々な場所で陣地の取り合いをするような囲碁らしい形に中々ならない。そのため、データベースや候補手専用のアルゴリズムを作成するなどしてより囲碁らしい手を打たせる必要がある。また、終局時に使用する地の計算結果は群の生死の判断が不十分なため、正確に地を認識しておらず、改良していく必要がある。さらに、このプログラムは候補手を選ぶ際、評価値が 1 以上のものがあれば、次の一手を打つようにしているため、同じ場所の取り合いを繰り返し行ってしまうことがあり、アゲハマの数が 100 や 200 など囲碁としてはありえない多さになってしまっている。この点も改良していかなければならない。



(a)乱数との対戦結果



(b)作成したプログラム同士の対戦結果

図 32：対戦結果

6. おわりに

本研究の目的はモンテカルロ法を取り入れた囲碁プログラムの作成であるが、一から全てを作成するのは非常に難しいため、今回はその最初の段階として“コンピュータ囲碁の入門”[1]に付属しているサンプルプログラムをベースにコンピュータ囲碁システムの思考部を作成した。今回作成したプログラムは「連、結線、群」、「地の認識」、「地を利用した候補手の生成」である。「地の認識」では群の生き死の判断が不十分なため、群を用いた地の認識が正確に行われていない。今後、『群の目の数』、『群の包囲のされ具合い』、『群の大きさ』などからその群が生きるか死ぬか、つまり「群の強さ」を正確に認識させ必要がある。また、「地を利用した候補手の生成」では候補手を生成する際に **3.3 候補手の生成と評価**で述べた「2. 候補手専用のアルゴリズムを利用する」「3. パターン（データベース）を利用する」を利用して候補手の生成を行っていないため、定石や布石の手など囲碁らしい手を打つことができていない。したがって、これらの方法を取り入れ、より高度な思考をさせていく必要がある。そして、本研究の目的でもあるモンテカルロ法を取り入れ、終局までのシミュレーションを行い、候補手を評価することでより高度な思考をするプログラムになると考えられる。モンテカルロ法を取り入れると計算量が非常に多くなるため、計算時間も非常に長くなってくる。その問題を解決するためにプログラムの並列化を行い、計算速度の向上を計る必要がある。

謝辞

本研究の機会を与えて下さり、貴重な助言、ご指導を頂きました山崎勝弘教授に深く感謝いたします。 また, 本研究に関して様々な相談に乗って頂き、貴重なご意見を頂きました、高性能計算研究室の皆様に深く感謝いたします。

参考文献

[1]清愼一, 山下宏, 佐々木宜助：コンピュータ囲碁の入門, 共立出版社, 2005
[2]中村貞吾：“コンピュータ囲碁研究の現状と展望, FIT2008 コンピュータ囲碁最前線, 2008
[3]山下宏：囲碁プログラム　彩の仕組み, FIT2008, 2008
[4]村松正和：コンピュータ囲碁の現状, FIT2008, 2008
[5]飯田弘之：ゲーム情報学の中のコンピュータ囲碁, FIT2008, 2008
[6]清愼一：コンピュータ囲碁の歴史と将来の展望, FIT2008, 2008
[7]村松正和：プロ棋士対コンピュータ　FIT2008 における囲碁対局報告, 情報処理 Vol.50 No1 Jan.2009, p70-73, 2009
[8]山下宏：YSS と彩のページ, http://www32.ocn.ne.jp/~yss/index_j.html
[9]Fuego, http://www.perfectsky.net/fuego/index.html