

16.Java 言語を使用したオセロの作成と評価

11 番 佐々木 巧

指導教員 小笠原祐治

1. はじめに

私がこの研究テーマを選んだ理由は、将棋などのボードゲームの CPU 対戦をしてきたので、ゲームの思考ルーチンのアルゴリズムに興味を持ち、自分も作ってみたいと思いましたので、このテーマにしました。

2. 研究概要

2.1 目的

今までの授業で習ってきた Java 言語とネットワーク接続を使って、オセロの思考ルーチンを作り評価を行う。

ネットワーク接続をして、思考ルーチン同士で対戦させ、勝率等を求める。

2.2 開発環境

OS	Windows7
言語	java
ソフトウェア	Eclipse
	Notepad++

3. 思考ルーチンの評価について

3.1 評価方法

図の様に、クライアント（思考ルーチン含む）とサーバーで通信を行いクライアント同士で対戦させる。

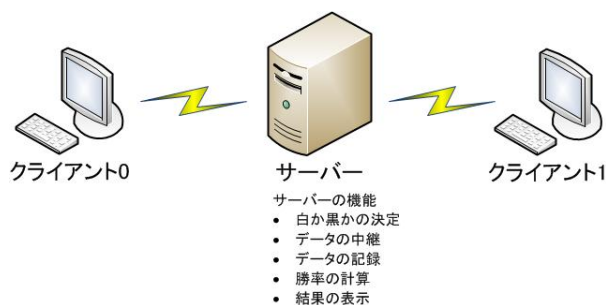


図 1. クライアントとサーバーの動作

それぞれの思考ルーチンの評価を行うために、複数回対戦を行い、以下に示す結果を求める。

- ・自分の石の数,思考時間,評価値（平均）
- ・勝った時の石の数,思考時間,評価値（平均）
- ・負けた時の石の数,思考時間,評価値（平均）

3.2 実行手順

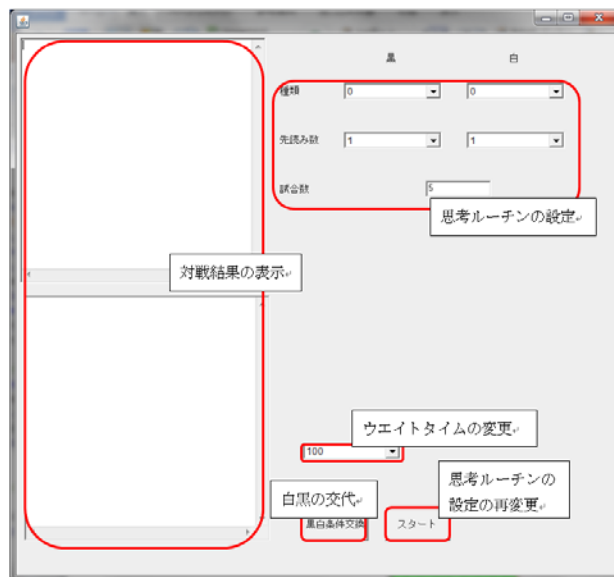


図 2. サーバーの画面

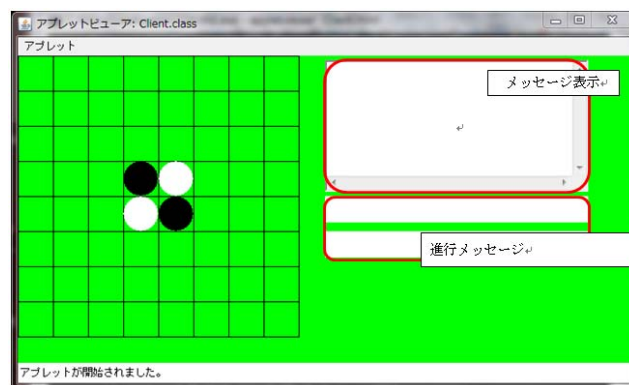


図 3. クライアントの画面(アプレット)

サーバーから思考ルーチンの詳細設定を行う設定項目は、以下の通りである。

- ・思考ルーチンの選択（1 または 2）
- ・先読み数（1~4）
- ・対戦数(1~)

最初に起動（接続）したクライアントが黒になる。次に起動（接続）したクライアントが白になる。両方のクライアントが起動された時点で対戦が開始される。

設定回数分対戦が行われた後,結果が表示される.

白黒交換ボタンを押した際は,黒の思考ルーチンと白の思考ルーチンの設定値が入れ替えられる.

スタートボタンを押した際は,設定されている条件で対戦が行なわれる.

先読み数が少ない場合,思考ルーチンの処理時間が短く応答が速いので,思考ルーチンが正しい応答しているか確認することが難しい. そこで,1手ごとにウェイト時間を入れ,見て確認できるようにした.

ウェイト時間は,サーバーから設定することができる.

思考ルーチンについては4章で説明する.

3.3 メッセージのフォーマット

サーバー,クライアント間で通信するメッセージの形式を以下に示す.

送信元 : メッセージの種類 : データ

① 送信元

S0 : サーバー

C0 : クライアント 0

C1 : クライアント 1

② サーバーが送信するメッセージの種類一覧

メッセージ種類	データ
ゲーム開始	駒の色
初期設定	思考ルーチンの種類, 先読み数
終了	“終了しました”

クライアントが送るメッセージ一覧

メッセージ種類	データ
接続要求	クライアントの名前
パス	“パスです”
座標データ	X 座標、y 座標、色、黒の総数、白の総数、評価値、時間
終了	黒の総数、白の総数

通信を行うために,通信用のクラス (PortCon,PortCom) を作成し,リスナー形式で受信イベントを処理する. 受信待ち中の画面の再描画を可能としている.

データのシーケンス

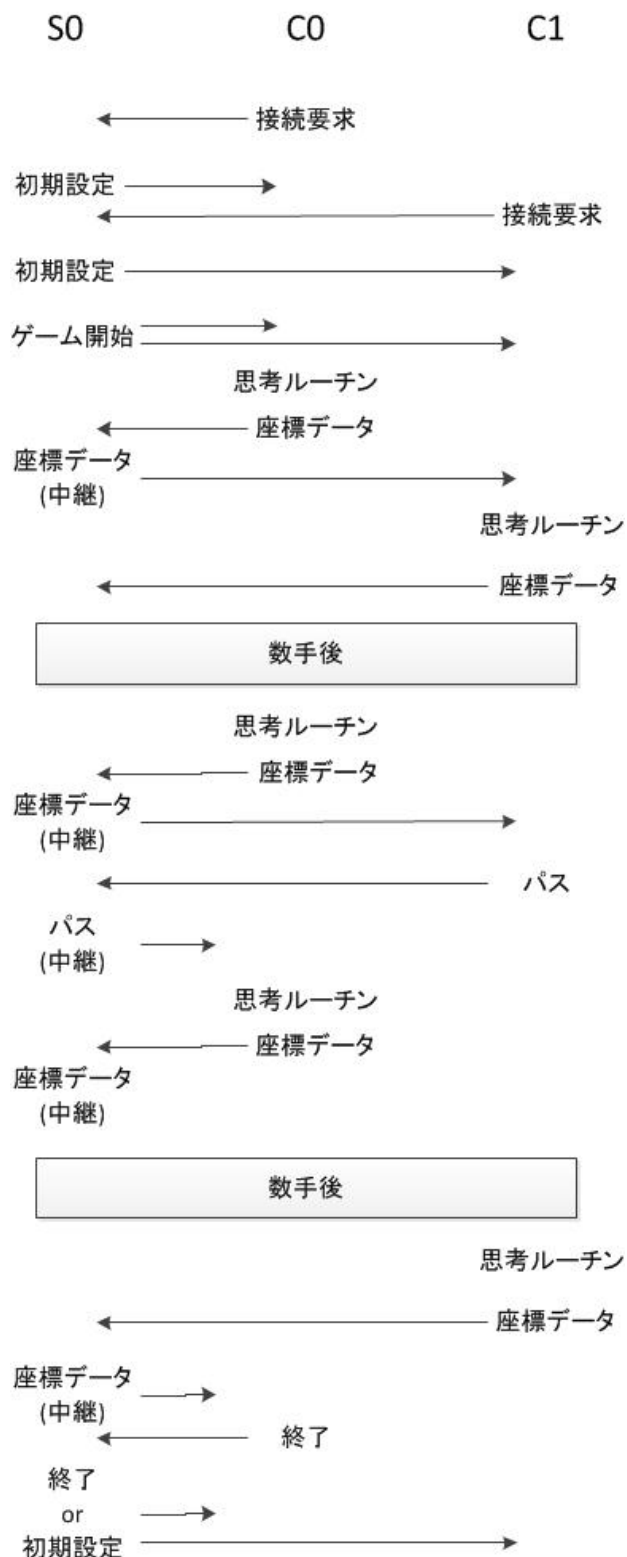


図 4. データの流れ

4. 思考ルーチンの詳細

相手の打った手に対して応答する思考ルーチンのアルゴリズムを説明する。

4.1 次の手の決定

自分が石を置ける場所の中で、自分にとって最も有利と思われる場所に石を置く。そのために、盤面の状況を数値で示す評価値という尺度を使用する。評価値を計算する評価関数を使用する。

4.2 評価関数について

現在の盤面が有利か不利かを示す評価値として、次のものを用いる。

- ・ 自分の石の数－相手の石の数
- ・ 重みづけした自分の石の合計
- ・ 重みづけした相手の石の合計

120	-20	20	5	5	20	-20	120
-20	-40	-5	-5	-5	-5	-40	-20
20	-5	15	3	3	15	-5	20
5	-5	3	3	3	3	-5	5
5	-5	3	3	3	3	-5	5
20	-5	15	3	3	15	-5	20
-20	-40	-5	-5	-5	-5	-40	-20
120	-20	20	5	5	20	-20	120

図 5. 盤面の重みの図

重みは場所によって異なり、重要な位置では大きい。

上記の重みの置き方では、盤面の 4 つ角と辺の中央部分が最も重みが大きい。上記の置き方のほかに、以下のような重みの置き方がある。

- ・ 定石を考慮したもの
- ・ 角と中央部分に重みを大きく置いたもの
- ・ 全体的に負の数にしたもの

序盤、中盤、終盤の重みを変えることによってより強くすることができる。

4.3 先読みの方

そこから 1 手、2 手先を読むために、Min-max

探索法を使用した。

Min-max 探索法とは…

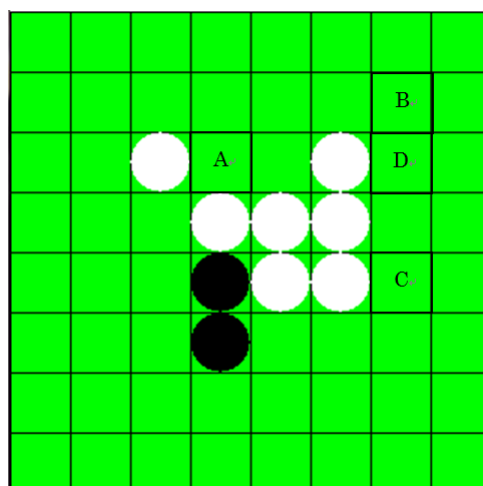


図 6. 盤面の例

上のような盤面があった場合、

1 手読む場合に、置ける場所は A~D の 4 か所あり、石を置いた時の評価値(重みなし)は、以下の通りとなる。

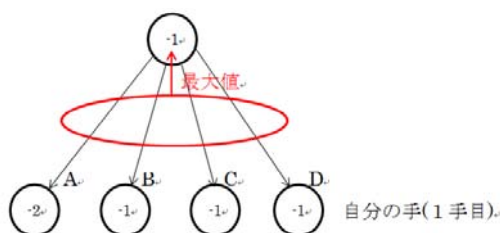


図 7. 1 手目の探索

このようになる。この場合、自分に有利になるように置くので、置く場所は、B、C、D の 3 箇所である。3 箇所とも評価値が同じなので乱数で置く場所を決める。

2 手読む場合は、

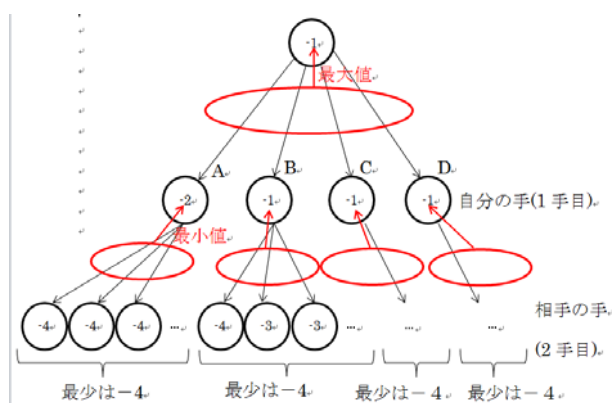


図 8. 2 手目の探索

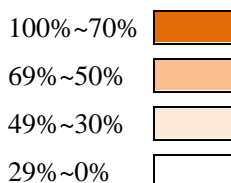
2 手目の枝分かれごとに最小値を選び、その中

(1 手目)の最大値を選ぶ。同じ評価値が複数の場合は、乱数で決める。

1 手先読みの場合は、自分の評価値が最も高い手を選択する。2 手先読みする場合は、2 手目の相手の評価値が最大になる手を選び、その中で最少になる 1 手目を選択する。3 手先読みする場合は、3 手目から自分の評価値が最大のものを選び、2 手目でその中から最少のものを選び、1 手目で最大のものを選ぶ。そのため、最大→最少を繰り返すことで N 手先読みすることができる。これを行うために再起呼び出しを行っている。

5. 対戦結果

各条件で 100 回対戦させた結果(先手の勝率)が以下のとおりである。



① 両クライアントとも重みなし

先手 (黒)	後手 (白)				
	先読み数	1	2	3	4
1		55%	43%	34%	33%
2		64%	47%	48%	44%
3		55%	55%	43%	44%
4		55%	54%	44%	56%

先読み数が多い方が勝率が高い。しかし、勝ち越した時と負け越した時の勝率の差は、約 30% 程しか差がない。

② 両クライアントとも重みあり

先手 (黒)	後手 (白)				
	先読み数	1	2	3	4
1		40%	20%	20%	19%
2		75%	42%	52%	48%
3		68%	52%	45%	45%
4		78%	49%	46%	45%

先読み数が多い方が勝率は高い。しかし、①と比べると勝率の差が最大 60% 程である。また、引き分けになる率が最大 7% もあり、勝率が 50% を切っても白の勝率より高いことがあった。

③ 後手が重みあり

先手 (黒)	後手 (白)				
	先読み数	1	2	3	4
1		68%	37%	35%	49%
2		71%	53%	47%	46%
3		72%	42%	49%	41%
4		74%	51%	46%	45%

後手の先読み数が大きくなるにつれて先手の勝率が下がっていつていることがわかる。しかし、②のような大きな差はない。

④ 先手が重みあり

先手 (黒)	後手 (白)				
	先読み数	1	2	3	4
1		36%	24%	18%	17%
2		53%	44%	49%	43%
3		56%	46%	42%	56%
4		50%	42%	56%	45%

③に対して全体的に勝率が低い。参考資料では、重みありの方が強いことになっており、今回の結果と異なる。次の原因が考えられる。

- ・プログラムにミスがある。
- ・重みの設定が適切ではない

6. まとめ

今回の卒業研究で Java 言語に関する知識がさらに深まった。しかし、前提条件であるネットワークを使用したオセロの完成や、思考ルーチンの完成に時間がかかってしまった。思考ルーチンの検討が不十分であり、対戦結果に対して、評価値、思考時間の検討ができなかった。

しかし、思考ルーチン同士の対戦を行うことによって、思考アルゴリズムの客観的な評価を行うことができた。

参考資料

オセロ・リバーシプログラミング講座

～勝ち方・考え方～

URL: <http://uguisu.skr.jp/othello/>