卒業研究報告書

題目

ヘクスリバーシの AI 作成

指導教員

石水　隆　講師

報告者

18-1-037-0119

清水 亮

近畿大学理工学部情報学科

令和 5 年 2 月 8 日提出

概要

リバーシは 2 人のプレイヤーが白と黒の石を交互に打ち，自石で敵石を挟むとひっくり返して自石にできるゲームであり，様々な年齢層にはば広く遊ばれている．リバーシには様々なヴァリエーションがあり，盤面の小さいミニリバーシや大きいグランドリバーシ，盤面が円形のニップや盤面が八角形のエイトスターズリバーシなどがある．これらはリバーシと同じルールでありながら，リバーシとは違った戦略が必要となる．

リバーシのヴァリエーションの一つに，ヘクスリバーシがある．ヘクスリバーシはマス目が六角形で，各マスは 6 個のマスと隣接する．通常のリバーシでは，4 つある角を取ることが戦略上重要となる．それに対してヘクスリバーシは，角が 6 つあるため，一つ一つの角の重要性は通常のリバーシよりも低い．また，辺の長さも短いため辺上のマスの重要性も低くなる．このため，ヘクスリバーシでは通常のリバーシとは異なった戦略が必要となる．

通常のリバーシは多くのＡＩが作られており，人間のトッププレイヤーを上回る棋力を持つＡＩも存在する．一方，ヘクスリバーシはマイナーなゲームであるため，既存のＡＩは存在しない．そこで本研究では，ヘクスリバーシで α β 法を用いて着手選択を行う AI を作成する．

目次

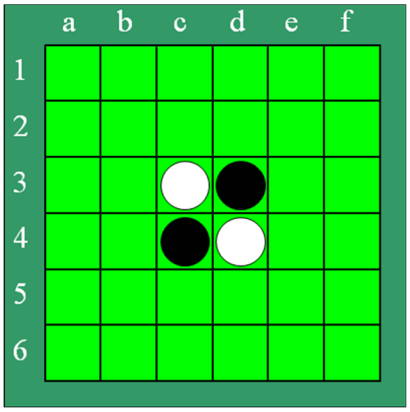
1. 序論

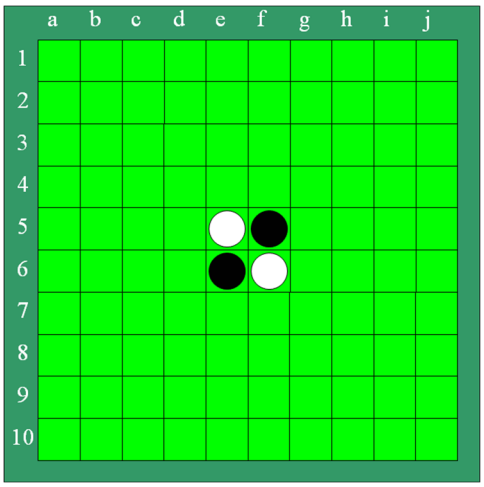
1.1 本研究の背景

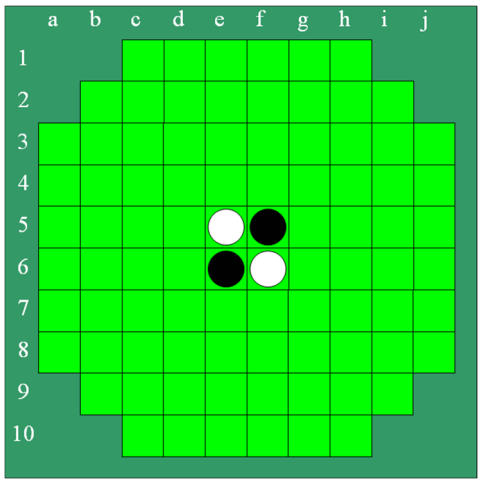
リバーシとは，2 人のプレイヤーが交互に白と黒の石を盤面に打ちながら，相手の石を自分の石で挟むことによって自分の石へと換えていき，最終的な盤上の石の個数を競う 2 人零和有限確定完全情報ゲームである．

オセロには様々なヴァリエーションが存在し，代表的なヴァリエーションとしては，盤面サイズの小さいミニオセロ，10 × 10 の大きな盤を用いるグランドオセロ，盤面が八角形で角が 8 個あるエイトスターズオセロ，盤面が円形で角が無いニップ等がある．図 1 に各ヴァリエーションの盤面を示す．

リバーシのヴァリエーションの一つにヘクスリバーシがある．







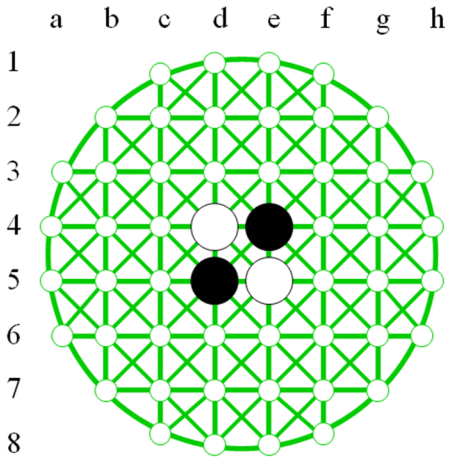


図 1 ミニオセロ，グランドオセロ，エイトスターズオセロ，ニップの盤面

リバーシは古くから存在するゲームであり，様々なプログラムが作られてきた．1997 年、コンピュータオセロソフト「Logistello」が当時の世界チャンピオン村上健 7 段と対戦し，6 戦全勝する[5]．また，リバーシは理論的な解析も行われてきた．6x6 のミニリバーシでは，完全解析により双方最善手を打つと 16 対 20 で後手が勝つことが判明している[6]．その他のサイズのミニリバーシでも解析が行われている[7]．表 1 にミニリバーシの既知の結果を示す．8x8 の通常のリバーシに関しては，2023 年 10 月に双方最善手を打つと引き分けとなるという論文が滝沢により公開された[8]．ただし現時点ではこの論文は追試および査読が行われていないため真偽は不明である．

表 1 ミニリバーシの完全解析結果[7]

初期配置	サイズ	勝敗	石数
○● ●○	4x4	後手勝ち	黒 3 白 11
	4x6	先手勝ち	黒 20 白 4
	4x8	先手勝ち	黒 26 白 0
	4x10	先手勝ち	黒 39 白 0
	6x6	後手勝ち	黒 16 白 20
●○ ●○	4x4	後手勝ち	黒 6 白 9
	4x6	先手勝ち	黒 21 白 3
	4x8	先手勝ち	黒 28 白 0
	4x10	先手勝ち	黒 32 白 0
	6x6	後手勝ち	黒 17 白 19

1.2 本研究の目的

ヘクスリバーシの AI やプログラムはあまり存在しないため，本研究でヘクスリバーシの強い AI を作成することを目指す．

1.3 本報告書の構成

本報告書の構成は以下の通りである.まず第 2 章において，本研究の対象であるヘクスリバーシのルールとその戦略について説明する.続く第 3 章で，本研究で作成したヘクスリバーシプログラムにつ いて説明する.第 4 章で本研究から得られた結果について述べ，最後に第 5 章でまとめと今後の課題について述べる．

2 ヘクスリバーシ

本章では，本研究で扱うヘクスリバーシのルールとその戦略について説明する．

2.1 ヘクスリバーシのルール

通常のリバーシでは正方形のマスと盤を用いるが，ヘクスリバーシは六角形のマスと盤を使用する．図 2 にヘクスリバーシのゲーム盤と石の初期配置を示す.通常のリバーシは,縦横斜めの 4 方向で挟むことが可能であるが,

ヘクスリバーシでは縦と斜めの 3 方向でしか挟むことが出来ない．また，角は 4 個から 6 個に増えているため，角の価値は下がっている．

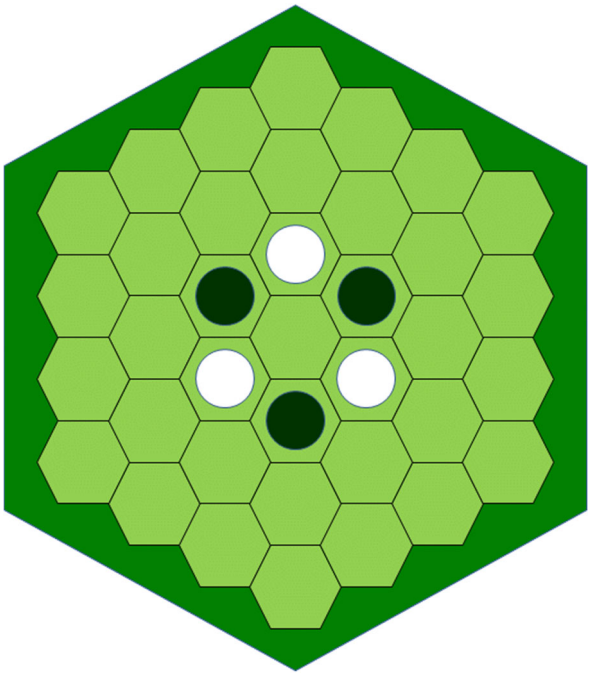


図 2　ヘクスリバーシのゲーム盤と石の初期配置

2.2 リバーシの戦略

本節では，リバーシの着手選択を行うために用いられている戦略について述べる．

リバーシは古くからプレイされているゲームであるため，定石が確立している．リバーシの序盤定石には鼠定石，牛定石，兎定石等があり，定石をデータベースとして持つことで序盤は定石通りに打つことができる．

リバーシでは角に置いた石はゲーム終了までひっくり返されることがないため，角に石を置くことは重要である．また，角から繋がって並んだ辺上の石もひっくり返されることが無い．このようなゲーム中に絶対にひっくり返されることが無い石を確定石と呼ぶ．リバーシではできるだけ確定石を増やすことが中盤の重要戦略とされる．通常のリバーシでは辺上のマスは角のマスから順に角マス，C マス，B マス，A マス，角から斜め方向に 1 マス内側の位置のマスは X マスと呼ばれる．図 3 に通常のリバーシの各マスの呼び名を示す．角に隣接する C マスと呼ばれるマスは，角が空いているときに C マスが自石だと相手に角を取られやすくなるため，C マスに打つのは危険とされる．一方角に自石が置かれている場合，C マスの自石もひっくり返されることが無い確定石となるため，C マスに打つのが有利になる．同様に X マスは通常は非常危険なマスとされできるだけ打たないようにするべきであるが，状況によっては打った方が有利となる．

リバーシの局面評価では，何手か先の局面を先読みし，その局面の評価値から mini-max 法やαβ法を用いて評価値を求める．mini-max 法とは，自分の手番で最も評価値の高い手を採用し，相手の手番で自分にとって不利である最も評価値の低い手を採用し探索する手法である．またαβ法とは，mini-max 法を改良し計算量を減らした手法であり，自分の手番で最大の評価値αが相手の手番で評価値がαより小さい場合に探索を打ち切り，相手の手番で最小の評価値βが自分の手番で評価値がβより大きい場合に探索を打ち切る．

ゲーム終盤では，完全読みおよび必勝読みが用いられる．完全読みとはゲーム終了まで読み，最も得点が高い勝つ手を選択することであり，必勝読みとは勝つ手を選択することである．まず必勝読みで勝利を確定し，さらに完全読みで得点の高い勝ちを目指すことができる．

リバーシで用いられるその他の着手選択のための手法としては，モンテカルロ法および機械学習がある．モンテ

カルロ法とは乱数を用いて試行を繰り返すことで近似解を求める手法である．また機械学習とは AI の働きの 1 つで，コンピューターに明示的な指示を与えなくても，大量のデータをもとに自動的に反復学習する技術である．

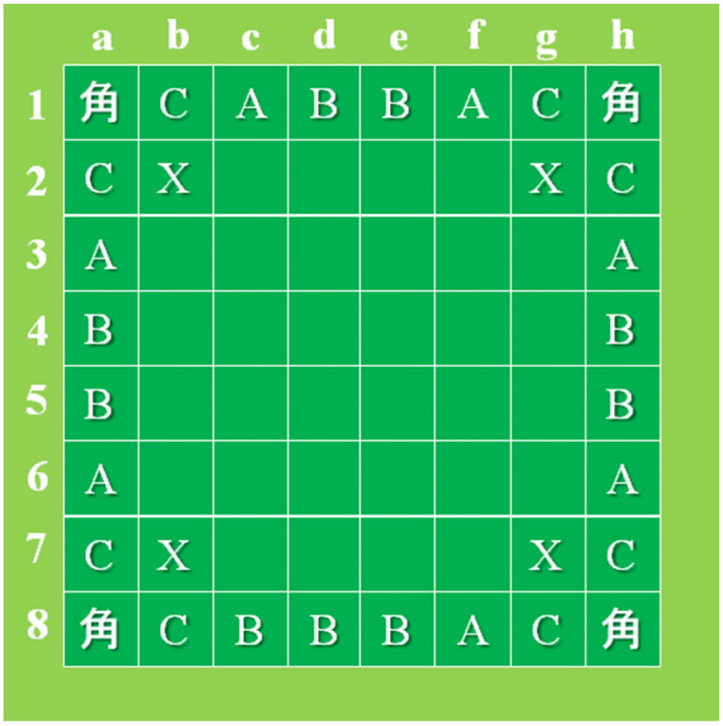


図 3 通常のリバーシの各マスの呼び名

2.3 ヘクスリバーシの戦略

前節で述べた通り，通常のリバーシは様々な戦略が確立している．一方，ヘクスリバーシはマイナーなゲームであるため，既存の戦略が存在せず，一から戦略を考える必要がある．ヘクスリバーシの着手選択では，通常のリバーシと同一の部分，異なる部分を考慮して戦術を考える必要がある．

通常のリバーシと同様に，ヘクスリバーシでも角ヘクスを取ることが重要であると考えられる．そこで角と角に隣接するヘクスに自石を置いた場合の評価値を考える．本研究では，通常のリバーシに準じて，角ヘクスに隣接する辺上のヘクスを C ヘクス，角ヘクスの一つ内側のヘクスを X ヘクスとする．図 4 にヘクスリバーシの各ヘクスの呼び名を示す．

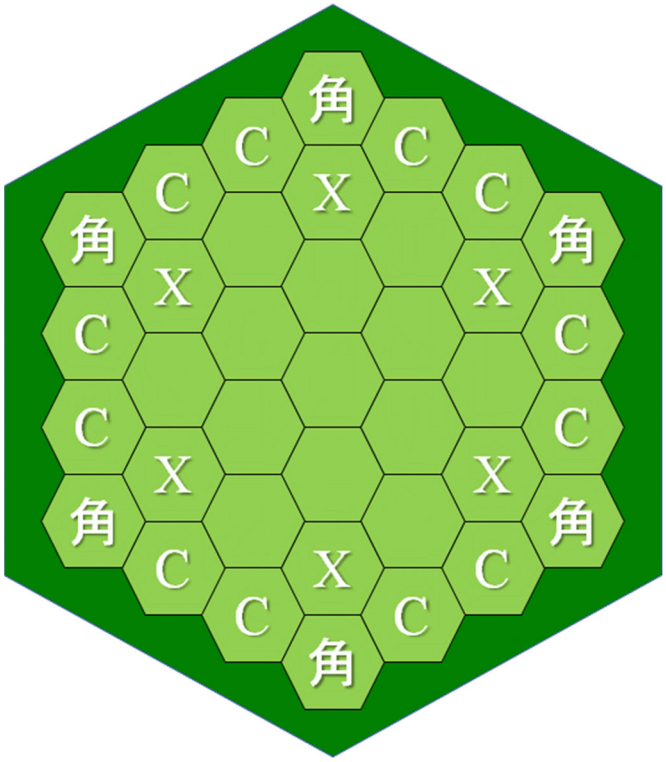


図 4　ヘクスリバーシの各ヘクスの呼び名

角ヘクスに置いた石はゲーム終了までひっくり返されることがない確定石となる．よって，角ヘクスに自石がある場合，その局面の評価値をプラスにする．一方，角ヘクスに隣接する C ヘクスおよび X ヘクスに自石がある場合，相手に角ヘクスを取られる危険が高くなる．よって，C ヘクスおよび X ヘクスに自石がある場合は局面の評価値をマイナスにする．

C ヘクス，X ヘクスは一般には打つと危険なヘクスであるが，状況によっては安全となる場合がある．隣接する角ヘクスに自石がある場合，C ヘクスに置いた石は確定石となる．また，隣接する C ヘクスおよびその先の角ヘクスに自石がある場合も，C ヘクスに置いた石は確定石となる．このため，C ヘクスに自石があるとき，上記の 2 つの条件のいずれかを満たす場合は，局面の評価値をプラスにする．図 5 に石を打っても安全に C ヘクスの状態を示す．同様に，X ヘクスは通常は危険なヘクスであるが，隣接する角ヘクスおよび 2 つの C ヘクスが全て自石がある場合，X ヘクスに置いた石は確定石となる．よって，X ヘクスに自石があるとき，上記の条件を満たす場合は，局面の評価値をプラスにする．図 6 に石を打っても安全な X ヘクスの状態を示す．

上記の評価法を用いることにより，ヘクスリバーシに適した局面の評価値を求めることができると考えられる．

局面の評価値を求めることができれば，通常のリバーシと同様に mini-max 法やα β法を用いて先読みするのは有効である．また，終盤での完全読み，必勝読みも通常のリバーシと同様に有効な手段と成りえる．

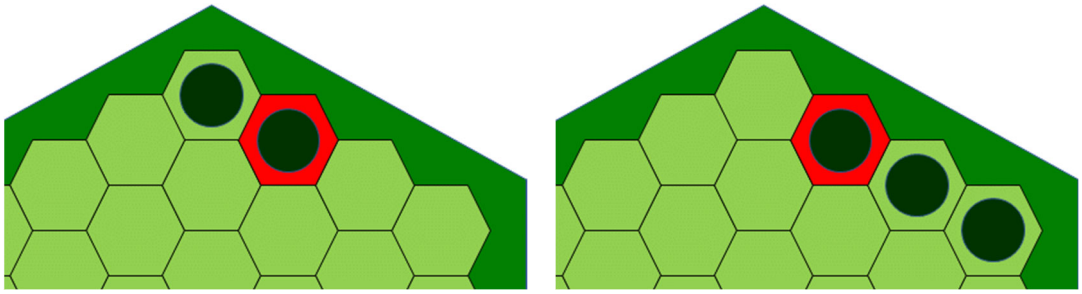


図 5 石を打っても安全な C ヘクスの状態

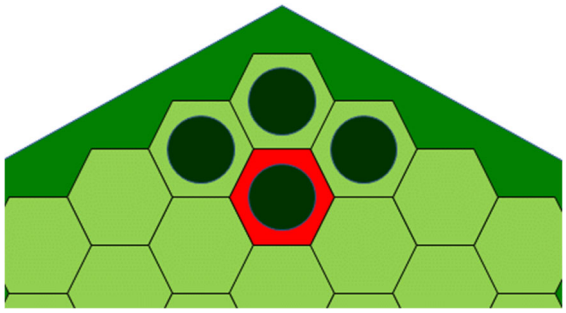


図 6 石を打っても安全な X ヘクスの状態

3 ヘクスリバーシプログラム

本研究では，Java を用いてヘクスリバーシプログラムを作成した．本研究で作成したプログラムのソースを付録に示す．本研究で作成したプログラムのソースを付録に示す.本研究で作成した Java プログラムは，AI.java, Board.java, BoardTest.java, BookManeger.java, BookTest.java, ColorStone.java, ConsoleBoard.java, Disc.java, Evalutor.java, GUIReversi.java, MidEcalutor.java, Point.java, ReversiGage.java の 13 個のクラスから成る．以下の各クラスについて説明する．

3.1 AI.java

AI.java はヘクスリバーシ AI の動作を記述する抽象クラスである．図 7 に AI.java のクラス図を示す．また，図 8 に AI.java の継承クラスである AlphaBetaAI.java のクラス図を示す.

AI.java は AI の動作を表す抽象クラスで有るので、このクラスを継承して AI の具体的な動作を表す具象クラスを作成する必要がある．本研究では，AI.java の具象クラスとして AlphaBetaAI.java を作成した．AlphaBetaAI.java は，α β 法を用いて数手先の局面を先読みし，先読みした局面から最も評価値に高くなる局面に到達する手を選ぶ AI である．

AI.java			# AI の動作を記述する抽象クラス
+	presearch_depth	: int	# 探索の前処理の深さ
+	normal_depth	: int	# 探索の深さ
+	wld_depth	: int	# 必勝読みの深さ
+	perfect_depth	: int	# 完全読みの深さ

図 7 AI.java のクラス図

AlphaBetaAI.java			# α β 法で探索する AI の動作を記述するクラス
-	Eval	: Evaluator	# 評価値計算を行うオブジェクト
+	move (board:Board)	: void	# 石を打つ
-	alphabeta (board:Board, limit:int, alpha:int, beta:int)	: int	# α β 法で探索を行う

-	sort (board:Board, movables:Vector, limit:int)	: int	# 評価値順に並べ替える
-	evaluate(board:Board)	: int	# 局面の評価値を返す

図 8 AlphaBetaAI.java のクラス図

3.2 Board.java

Board.java はヘクスリバーシの盤面を記述する抽象クラスである.図 9 に Board.java のクラス図を示す. Board.java では 2 次元配列 RawBoard を用いて局面を表現している.

通常のリバーシでは，各マスは縦横に配置されているため，2 次元配列で表すことができる．一方，ヘクスリバーシはヘクスの配置が縦横ではないため，表現の仕方に工夫せねばならない．ヘクスリバーシのヘクスの配置は、ヘクスをずらすことにより簡単に縦横にならんだ四角形の並びに変換することができる．図 10 にヘクスリバーシのヘクスから四角形への変換の仕方を示す．この変換を用いることでヘクスリバーシの盤面を通常のリバーシとほぼ同様に扱うことができる.

Board.java			# 盤面を記述するクラス
+	BOARD_SIZE	: int	# 盤面の大きさを表す定数
+	MAX_TURNS	: int	# 最大手数を表す定数
-	NONE	: int	# 無方向を表す定数
-	UPPER	: int	# 上方向を表す定数
-	UPPER_LEFT	: int	# 左上方向を表す定数
-	LEFT	: int	# 左方向を表す定数
-	LOWER_LEFT	: int	# 左下方向を表す定数
-	LOWER	: int	# 下方向を表す定数
-	LOWER_RIGHT	: int	# 右上方向を表す定数
-	RIGHT	: int	# 右方向を表す定数
-	UPPER_RIGHT	: int	# 右上方向を表す定数
-	RawBoard[][]	: int	# 盤面を表す行列
-	Turns	: int	# 手数
-	CurrentColor	: int	# 現在のプレイヤー
-	UpdateLog	: Vector	# ログの更新
-	MovablePos[]	: Vector	# 着手可能位置
-	MovableDir[][][]	: int	# 反転可能方向
-	Discs	: ColorStorage	# 打つ石の色
+	Board()		# コンストラクタ
+	init()	: void	# 初期配置
+	move(point:Point)	: boolean	# 石を打つ
+	undo()	: boolean	# 一手戻す
+	pass()	: boolean	# パス
+	getColor(point:Point)	: int	# 指定したマスの石の色を得る

+	getCurrentColor()	: int	# 手番の色を得る
+	getTurns()	: int	# 手数を得る
+	isGameOver()	: boolean	# ゲーム終了判定
+	countDisc(color:int)	: int	# 石の数を得る
+	getMovablesPos()	: Vector	# 着手可能マスを得る
+	getHistory()	: Vector	# 着手履歴を得る
+	getUpdate()	: Vector	# 着手履歴を更新する
+	getLiberty(p:Point)	: int	# 未使用
-	checkMobility(disc:Disc)	: int	# 反転可能方向を調べる
-	initMovable()	: void	# 着手可能マスの初期化
-	flipDiscs(point:Point)	: void	# 石の反転

図 9 Board.java のクラス図

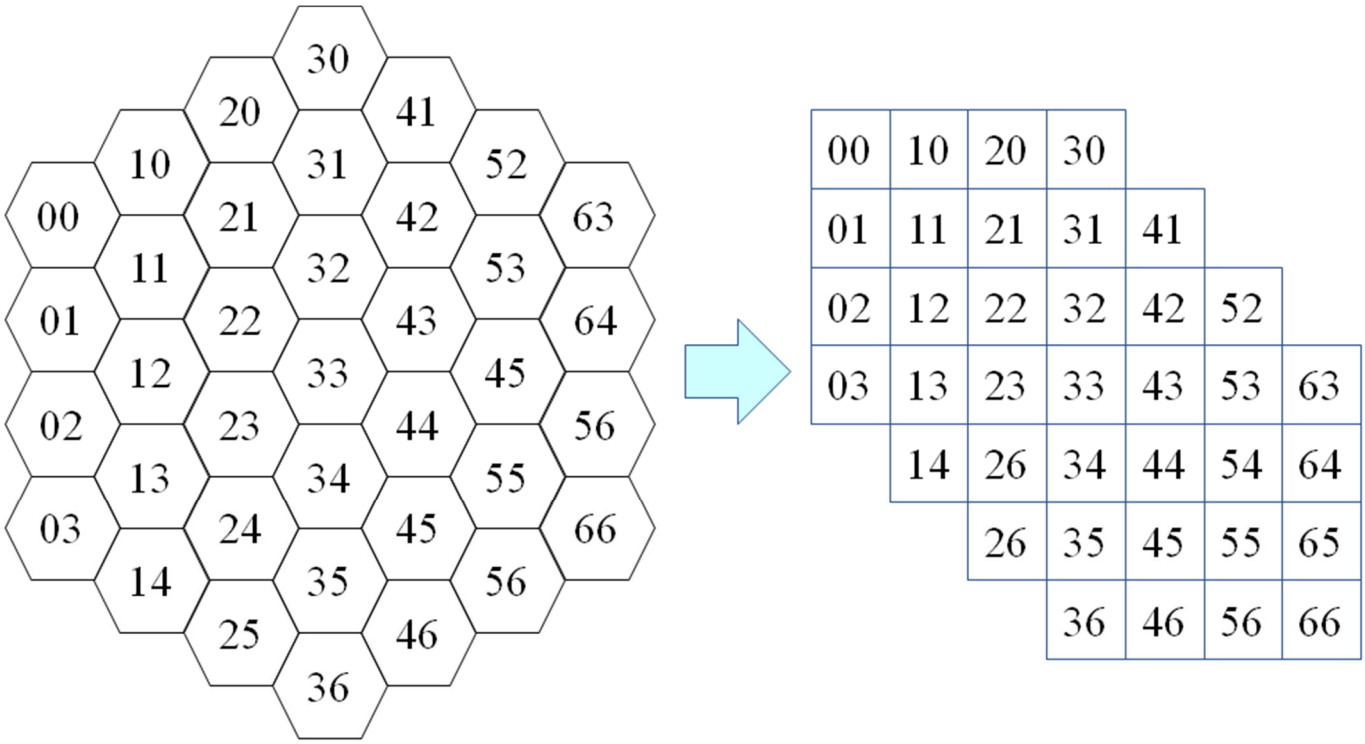


図 10 ヘクスから四角形への変換

3.3 BoardTest.java

BoardTest.java は AI と対戦中のシステムメッセージを表示するためのクラスである.

3.4 BookManager.java

BookManager.java はリバーシの定石をデータベース化し, 定石データベースの保管と探索をするクラスである.

3.5 BookTest.java

3.6 ColorStorage.java

ColorStorage.java			# 石数を格納するクラス
-	data[]	: int	# 石数を保存しておくオブジェクト
+	get(color:int)	: int	# 指定した色の石の数を得る
+	set(color:int, value:int)	: void	# 指定した色と石の数を設定する

図 11 ColorStorage.java のクラス図

3.7 ConsoleBoard.java

ConsoleBoard.java はヘクスリバーシの盤面を出力するクラスである.

3.8 Disc.java

Disc.java はリバーシの石を表現するクラスであり，Point クラスを継承している．図 12 に Disc.java のクラス図を示す.

Disc.java			# リバーシの石のクラス
+	BLACK	: int	# 黒石
+	EMPTY	: int	# 空
+	WHITE	: int	# 白石
+	WALL	: int	# 壁
+	Disc()		# コンストラクタ
+	Disc(x:int, y:int, color:int)		# 色を表す定数を設定する

図 12 Disc.java のクラス図

3.9 Evaluator.java

Evaluator.java は評価関数のインタフェースである．PerfectEvaluator クラスは完全読みの評価関数であり，WLDEvaluator クラスは必勝読みの評価関数である.

3.10 GUIReversi.java

GUIReversi.java はリバーシプログラムの GUI を表すクラスである.

3.11 MidEvaluator.java

MidEvaluator.java は中盤の評価関数を表すクラスである.

3.12 Point.java

Point.java は石の位置を指定するために座標を定義するクラスである.

3.13 ReversiGame.java

ReversiGame.java はヘクスリバーシをプレイするためのメインクラスである.

4　結果

本研究で作成したヘクスリバーシプログラムと対戦してみたところ，ルール通りには打ててはいるものの，あまり強く無かった．本研究で作成したヘクスリバーシプログラムはα β法で数手先の局面を生成しその局面の評価値から着手選択しているが，局面の評価値を求める部分がうまくできていないと考えられる．したがって評価値を求める部分については，プログラム実行時に，生成した局面とその評価値をログファイルに出力し適切な評価値となっているかの検証が必要であると考えられる．

また，現状では終始α β法を用いて着手選択しているが，終盤に関しては着手選択をα β法からゲーム終了まで読み切る完全読みに切り替えればより強くなると考えられる．

5　結論・今後の課題

本研究は人と AI が決着まで対戦できるヘクスリバーシのプログラムを作成することができた．

しかし，α β法で着手選択するヘクスリバーシ AI を作成したが，局面の評価値を計算する MidEvaluator.java クラスが未完成であり，ヘクスリバーシに適した評価値計算ができない．ヘクスリバーシに適した局面の評価値の求め方を考案し，そのプログラムを作ることが今後の課題である．

謝辞

本研究を行うにあたって，石水隆講師から卒研レジュメや卒業論文の推敲，資料の提供など様々なご指導を受けました．ここに感謝の意を表します．

参考文献

[1] Seal software：リバーシのアルゴリズム C++ & Java 対応, 工学社 (2003).

[2] 大筆豊 : オセロプログラムの評価関数の改善について, 研究報告ゲーム情報学 (GI), Vol.2003-GI-011, pp.15-20, 情報処理学会 (2004).

[3] 森田悠樹, 橋本剛, 小林康幸 : オセロ求解に向けた単純な縦型探索をベースにする探索方法の研究, ゲームプログラミングワークショップ 2010 論文集, Vol.2010, No.12, pp.36-41, 情報処理学会 (2010)

[4] 上田陽平, 池田心 : 遺伝的アルゴリズムによる人間のレベルに適応する多様なオセロ AI の生成研究報告ゲーム情報学(GI), Vol.2012-GI-27, No.5, pp.1-8, 情報処理学会 (2012)

[5] 五十嵐利幸, 滝沢雅樹：コンピュータ、人間をしのぐ, 新潟日報 教育モア(2010/12/23) https://www.niigata-nippo.co.jp/articles/-/14896

[6] Joel Feinstein：Amenor Wins World 6x6 Championships!, Forty billion noted under the tree, pp.6-8,British Othello Federation's newsletter. (1993)
http://www.britishothello.org.uk/fbnall.pdf

[7] 竹下拓輝, 池田論, 坂本眞人, 伊藤隆夫：縮小盤オセロにおける完全解析, 情報処理学会九州支部火の国情報シンポジウム, No.1A-2, pp.1-6 (2015)
https://www.ipsj-kyushu.jp/page/ronbun/hinokuni/1004/1A/1A-2.pdf

[8] Hiroshi Takizawa：Othello is Solved (2023) https://arxiv.org/abs/2310.19387

ソースプログラム

以下に本研究で作成したプログラムのソースを示す.