HICHAIN: カードの向きを指定して着手する二人完全情報ゲームの提案と，その AI への展望

伊藤滉貴[†1]　阿原一志[†2]

概要：本研究は第一著者が考案した HICHAIN(以下本ゲーム)という二人完全情報ゲームの提案，実装と，その AI 開発に関する研究である．本ゲームはアルファベットのカードを使ったゲームで，サイズに制限のない 2 次元の盤にカードの向きを指定して着手することで実質的に 3 次元の盤を実現している．また 1 ターンあたり平均約 1000 手と多く，探索空間は 10^{162} になる．このような盤を構成するため「相対参照アルゴリズム」とルールの実装に必要なアルゴリズムを考案し，本ゲームの AI の実装に向けて探索アルゴリズムや評価方法を考察した．

HICHAIN: A proposal of a new game for 2 people with perfect information by making moves specified a card direction and a prospect of an AI player of this game

KOKI ITO[†1]　KAZUSHI AHARA[†2]

Abstract: This research proposes a new game “HICHAIN” for 2 people with perfect information. The first author designs this game and implements the platform. We discuss algorithm of the implementation and survey an AI player of this game. This game uses alphabet cards and a 3-dimensional board substantially by making moves specified a card direction on 2-dimensional board without a limit of the size. The average of moves per turn is about 1,000 and search space is the 10^{162}. We devise “Relative Reference Algorithm” for the framework of board and other algorithm for programming the rule of the game. Thus we consider searching algorithm and evaluation method for developing the AI player.

1. はじめに

完全情報ゲームは古来より存在し，チェスや将棋，囲碁といった代表的なゲームは長年の研究の末 AI が人間を上回った．しかし日々ボードゲームは開発されており完全情報ゲームも例外ではない．HICHAIN(以下本ゲーム)は初期配置を中心にカードを周囲に置いていくことで盤のサイズに制限をかけず，それにより端のマスや座標に依存する評価が発生しない．またカードを向きによって異なる機能を持たせることで，多方向から盤を捉える視野が必要である．本論文ではこのような特徴のある完全情報ゲームを提案，実装し，AI への展望を考察した．

本ゲームは第一著者が 2015/10/29 に考案した二人用完全情報ゲームで，HIchain Project はゲームマーケットなどのボードゲーム頒布イベントで頒布した製品である．詳しいルールは 2 章で述べるが，アルファベットのカードを順番に繋げたり同じカードを繋げたりして連鎖させていくゲームである．ゲームの目標やルールが単純で実際にプレイしてみると初心者でも簡単に得点できるが，厳密に加点条件を定義すると複雑で，また加点に繋がらない悪手が多く探索空間も広い．このためコンピュータゲームの実装や手の評価が難解である．

2. ゲームのルール

本ゲームは図 1 に示すような得点制の 2 人用ゲーム[1]で，58 枚のカードを使用する．カードはアルファベットの A から Z と特殊記号の「*」(アスタリスク)がある．なお本ゲームの用語は[2]を参照すること．

2.1 ゲームの流れ

初めにプレイヤーは先手をどちらか決め，先手が黒，後手が赤のカードを場に置く．初めに盤には場のカードとは別に 4 枚置かれている．プレイヤーは 1 ターンにカードを場から 1 枚選び，着手ルールに従って盤の空所に置く．ゲームの目標は加点文字列と呼ばれるカードの並びを作り得点を得ることである．お互いカードを全て置き終わり，最終的に得点の高いプレイヤーが勝つ．

†1 明治大学 総合数理学部 先端メディアサイエンス学科
Department of Frontier Media Science, School of Interdisciplinary Mathematical Science, Meiji University
ev50532@meiji.ac.jp

†2 明治大学 総合数理学部 先端メディアサイエンス学科
Department of Frontier Media Science, School of Interdisciplinary Mathematical Science, Meiji University
ahara@meiji.ac.jp

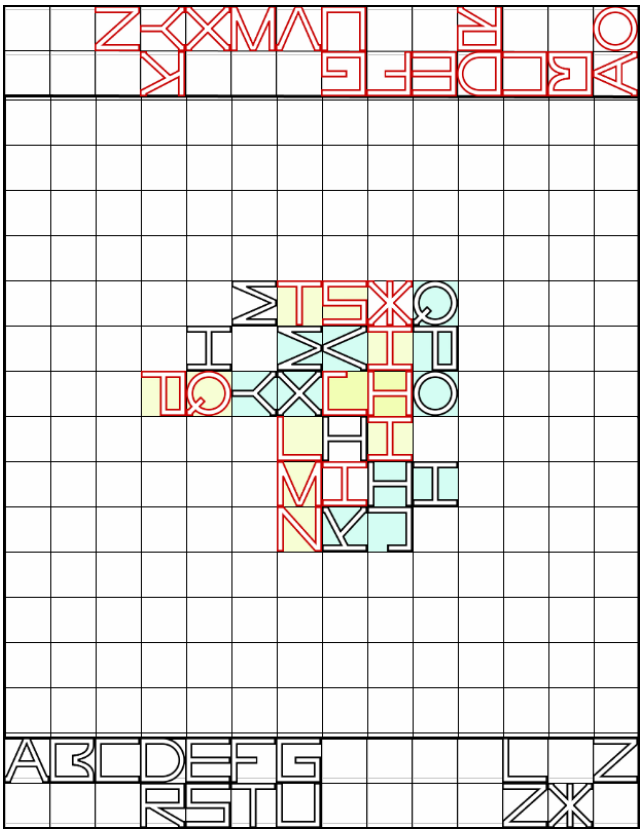

図 1　対局例

Figure 1　Example of the Game

2.2 着手ルール

図 2 のように本ゲームでは接しているカードの文字の形状によって着手ルールを定めている．着手ルールは以下の二点である．

(1)　相手のカードに 1 点以上接していること

(2)　色に関係なく辺と辺で接していないこと

文字には点と辺(合わせて PS と表す)という要素があり，文字によって異なる．また PS の配置は図 3 の通りであり 1 文字あたり点が 8 つ，辺が 8 つに分かれる．

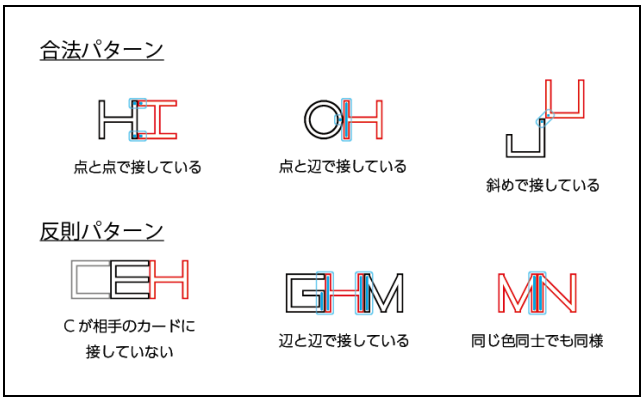

図 2　着手パターン

Figure 2　Available and Unavailable Patterns of Moves

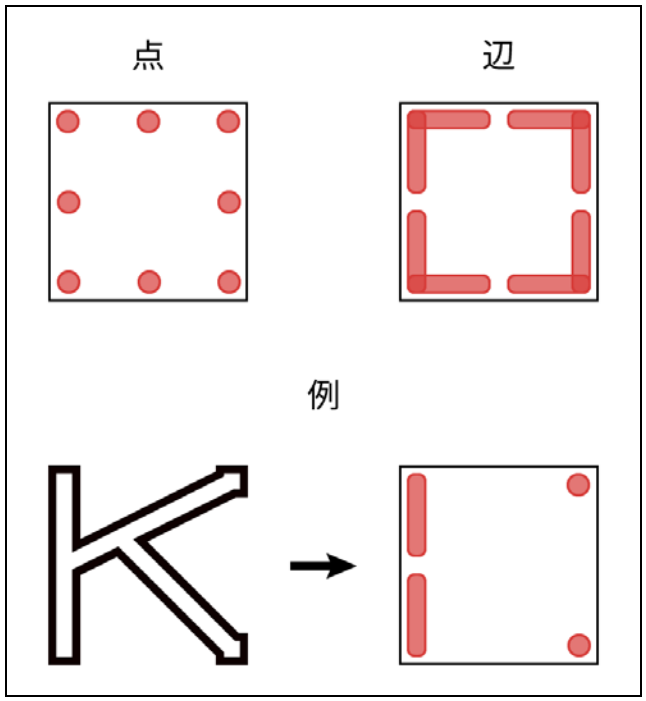

図 3　文字の点・辺配置と例

Figure 3　Layout of Sign Points and Sides, the Example of that

2.3 得点換算ルール

文字を繋げることを連鎖と呼び以下の 3 種類がある．

(1)　連番: アルファベット順に文字が並ぶこと

(2)　ぞろ目: 同じ文字が並ぶこと

(3)　ロイヤル:「CHAIN」が並ぶこと

連番において A の前は Z になる．またロイヤルが完成したらその時点でゲームは終了し完成させたプレイヤーが勝利する．このような連鎖は連番が 3 連鎖，ぞろ目が 2 連鎖以上から加点対象となり，加点対象の連鎖の文字列を加点文字列という．加点文字列は n 連鎖で n^2 点になる．連鎖は図 4 で示す通りの条件が必要である．向きによって複数の文字になりうるカードを兼用カードといい，N と Z の他に H と I，M と W などがある．「*」(アスタリスク)はどの文字にもなりうるカードで，「C*E」の場合は D の代わりとなるが，「CD*」や「*DE」のように連鎖の始点，終点にあると効果がない．なおアスタリスクはどの向きでも効果を発する．

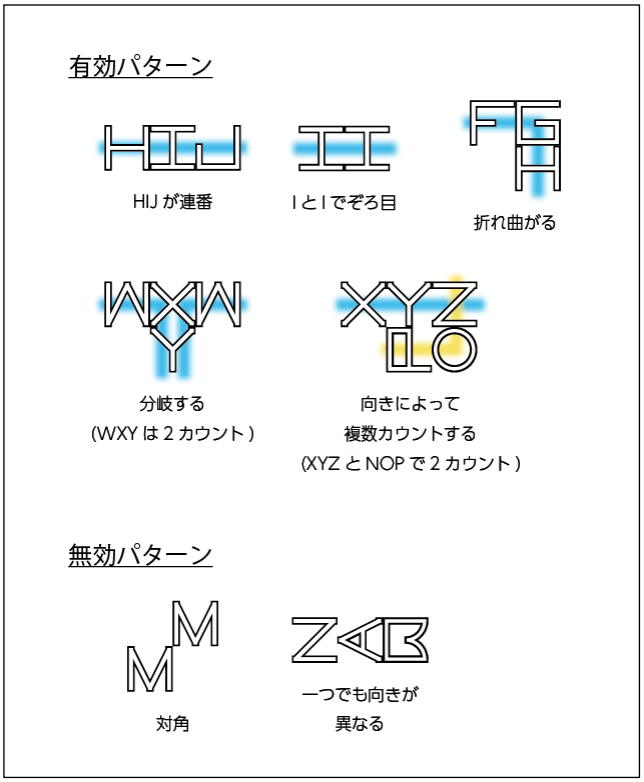

図 4　連鎖条件

Figure 4　Valid and Invalid Condition of Chain

3.　ゲームの特徴

3.1　ゲームの分類と研究

本ゲームは相手の手持ちのカードが見える状態でプレイするため完全情報である．また対局はカードが無くなれば終了するため有限で，偶然の要素はないので確定である．また得点制のためどちらが勝てばどちらかが負けるので零和である．

なお，場にカードが残った状態で合法手がなくなる状況が起きるかは証明されていない．他にも定跡や手番による有利不利も判明していない．

3.2　ゲームの特徴

本ゲームの特徴は大きく 3 つある．一つ目はカードの向きという概念が存在することである．即ちカードの向きにより 2 次元座標にもう 1 次元加わり実質的に 3 次元の盤で構成されているといえる．カードに向きを与えることでプレイヤーは常に多方向に盤を捉える必要がある．このことにより合法手が増加し探索空間が拡大する．兼用カードは同じ文字でも向きによって別の文字になれるためさらに探索の幅が広がる．その上兼用カードの存在により同じカードが複数生まれるため連鎖の分岐やぞろ目が実現できる．二つ目は盤のサイズに制限のないことである．即ちオセロや囲碁のように端を利用した戦術は存在しない．また着手ルールによりオセロ同様，必然的に盤上のカードの周囲にしか手を打てないのでマス目の書かれた盤を使用する必要はない．三つ目はアスタリスクという記号カードを用いることである．アスタリスクはどの文字にも使えることに加え向きもないので，連鎖の組み合わせが増加する．

3.3　探索空間

図 5 はお互いランダムに手を打った対局を 1000 回試行したときの着手可能な手数の遷移である．先手の場合初手が 738 手，後手は 943 手で最も手数が多くなるのは先手が 11 手目の約 1308 手，後手が 9 手目の 1360 手である．最終手には先手も後手も約 135 手に収束する．平均すると先手は約 960 手，後手は約 1014 手と後手の方が打てる手数が多い．またパスが起こらないと仮定すると 27 ターンで終局するので探索空間は平均すると約$(10^3)^{54} = 10^{162}$ となる．

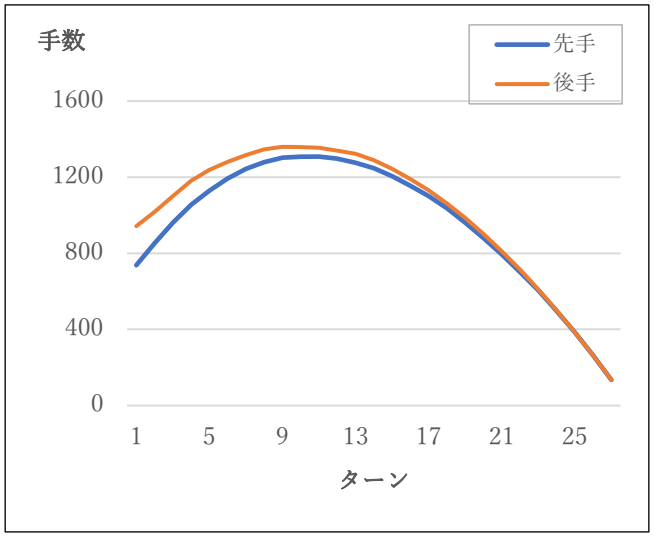

図 5　着手可能な手数の遷移

Figure 5　Transition of the number of available moves in one game

4.　他の完全情報ゲームとの比較

他の完全情報ゲームと比較すると，得点制や初期配置の形状，着手といったルールはオセロに近い．カードに特徴があることや，次の手で確実に加点される「詰み」の状況が起きることが将棋やチェスとの類似点といえる．また加点文字列の長さの 2 乗が得点となるので，深く探索しなければ最善手は得られない．このように局所的な考察が最善とは限らない点は囲碁に似ている．

4.1　TRAX との比較

本ゲームに似ているゲームとして TRAX[3]がある．TRAX は盤のサイズに制限がなくタイルに向きがある．また連鎖ルールというルールがありこれは本ゲームの連鎖とは意味が異なるが詰みのような状況を作り出すことができるとい

う意味で似ている．しかし使うカードの種類や着手ルールについて複雑さが大きく異なる．

5. 座標系と棋譜

5.1 座標系

本ゲームの盤はサイズを固定しないため初期配置を基点とする．座標系は図 6 の通りで先手が下として初期配置左上の黒の H を(0，0)とする．横は x 軸，縦は y 軸である．また文字の向きなどの方向を表す際は上を北として方角で表す．

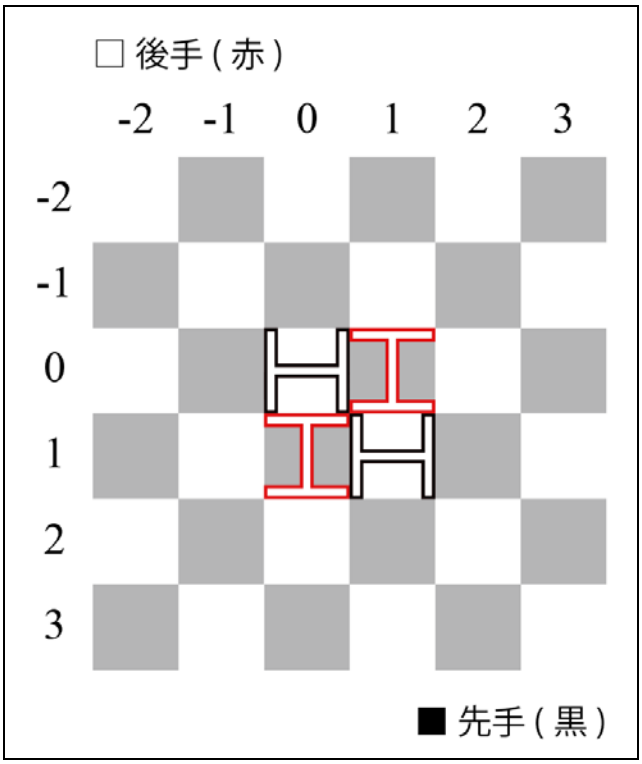

図 6　座標系

Figure 6　Coordinate System

5.2 棋譜の表記法

棋譜には記述式と代数式の 2 種類があり図 7 のように表記する．記述式は文字を実際に置いた向きに回転させて表記するのに対し，代数式は文字の向きを 1 文字で表記する．どちらの表記法でも y 座標の符号は必ず付加する．また手番を示す際は先手(黒)を■，後手(赤)を□と表記する．

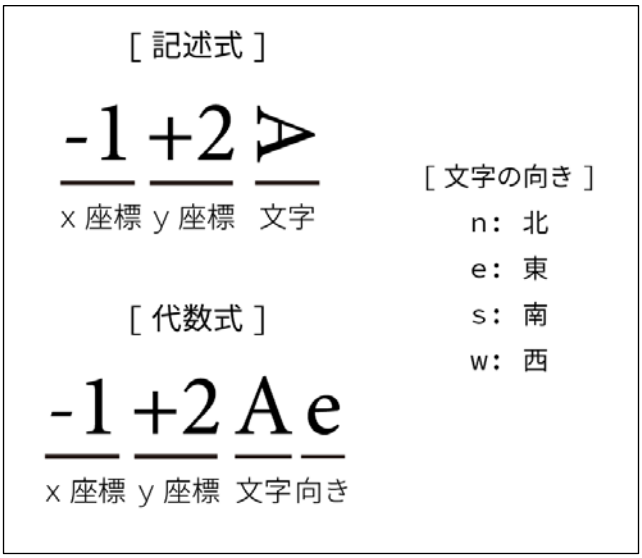

図 7　棋譜の表記法

Figure 7　Record Notation

6. ゲームの実装

本ゲームのプラットフォームは Prototype 版[4]と Java 版[5]があり，前者は Processing 3.1.2 で開発し基本ルールと GUI を実装した．後者は Prototype 版を元にアルゴリズムを改善しアスタリスクを除く基本ルールを Java 8 で実装した．今回は Java 版で使われているアルゴリズムを紹介する．

6.1 相対参照アルゴリズム

有限ゲームとは言え，盤のサイズに制限がないため有限の 2 次配列を用いて盤を実装することは適切ではない．また着手ルールや連鎖の判定ではある座標の周囲 8 マスしかアクセスしないことから，座標が周囲 8 方向の座標へのポインタを持つことで相対的に座標を参照できる「相対参照アルゴリズム」を考案した．なお周囲座標とは周囲 8 方向の座標を示す．

座標クラスでは，8 方向を持つ列挙型をキーとしたマップを使って周囲座標への参照を可能にした．最初に生成する座標は初期配置左上の(0, 0)であり，これがルート座標となる．ルート座標以外の座標は必ずある座標をソースとして生成し，ソース座標は新しく生成する座標の周囲座標でなければならない．座標は文字が置かれた座標の周囲にのみ動的に生成され，生成する座標は最小限になる．また座標クラスは絶対座標も持っていて，その値はソース座標を元に更新する．この絶対座標は周囲座標を更新するためのものでそれ以外の用途では使用していない．また盤上の全ての座標データベースは座標クラスが持っていて周囲座標の更新の際はこのデータベースを参照する．参考までに絶対座標を使用せずソース座標を元に周囲座標を探索するアルゴリズムも試案したが，効率的なアルゴリズムが発見で

きず実現に至らなかった.

6.2 着手ルール判定

着手ルールは周囲座標だけで判定できる．文字の PS(点と辺のこと)は図 3 の通り点 8 つ，辺 8 つの計 16 個で構成されているが，点と辺を同様に扱うため Prototype 版では図 8 のように点と点の間に新たな点を配置しこれを辺と対応させて全て点として扱う．これらの点は全て 1 ビットなので文字の PS は 16 ビットで表され，MSB から n 番目のビットが点 n に対応している. 例えば図 3 の例にある文字「K」は「1000110101010001」となる．また文字の PS の集合を SPS(SignPS)と呼ぶ.

着手ルールを満たすかどうかは主にビット演算を利用して判定している．辺と辺の判定だけはカードの色は関係ないためまずは辺と辺の判定を行う．2 枚のカードの向かう合う 2 つの辺 S_1 と S_2 は S_1 & S_2 = 1 のとき置けない判定となる．次に点と点の判定を行うが置くカードの色と同じ色の点は判定に関係ないため予め除外しておく. 点 P_1 と P_2 は P_1 & P_2 = 1 のとき置ける判定になる．ただ角の点の場合は角と接している 3 点のいずれか接していればいいので，置く文字の角の点を P_1，P_1 に接している点を P_2，P_3，P_4 とすると，P_1& (P_2| P_3| P_4) = 1 のとき置ける判定となる．これらの判定は置く文字の全ての PS と周囲の文字の PS を対象に行い，着手ルールを満たしているか判定している.

実際には効率化のため複数の PS を同時に判定しており，図 9 のように周囲の文字の PS から周囲 PS と呼ばれる PS の集合を作り出しそれと置く SPS を照合する．図 9 の矢印の先の PS にシフト演算を利用してビットをシフトする．角の場合は前述の通り 3 つの PS の OR をとり 1 つの PS に対応させる．また文字を回転すると PS も並び替えが起きる.

Prototype 版ではこの構造を採用しているが，点の配置順によってアルゴリズムが変わってしまい，また指定のビットを抽出する過程が分かりづらいことから，Java 版では点を 16 方向の列挙型と定義し順番によらない構造にした.

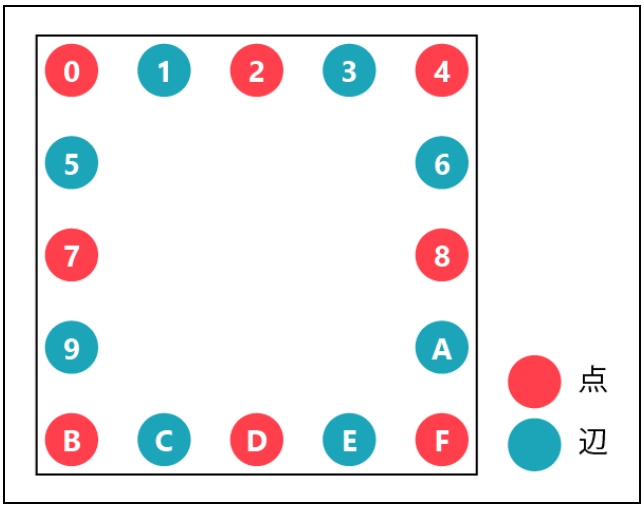

図 8　PS のデータ構造

Figure 8　Structure of Sign Points and Sides

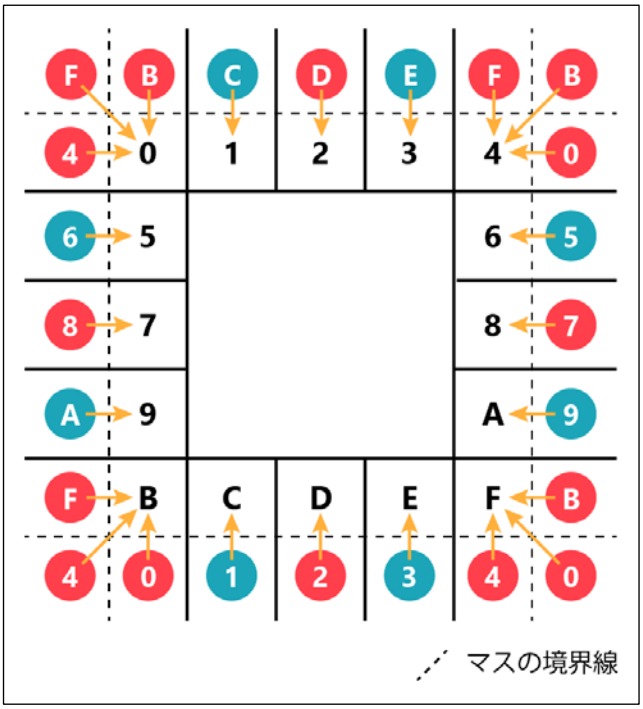

図 9　周囲 PS の生成方法

Figure 9　Generating Method of AroundPS

6.3 加点文字列探索

加点文字列を探索するには加点文字列の元となる連鎖を探索し，その連鎖が加点条件を満たすか判定する必要がある．連鎖は隣接する東西南北 4 方向にある同じ色の文字を対象とし，隣接座標とは隣接 4 方向の座標を示す.

6.3.1 連鎖探索

連鎖は文字の向き，連鎖の種類ごとに探索され，それらの組み合わせごとに親子を複数持つ木構造となっている．これを連鎖ツリーと呼び，連番の連鎖ツリー例を図 10 に示す．このとき H と I は兼用カードなので，隣り合うと全

方向で連鎖することに注意する．なお連鎖ツリーにおいて連番とロイヤルはルートからリーフにかけて昇順にノードを接続し，ぞろ目は親子どちらのノードに接続してもよいとする．連鎖ツリーは新しく置いた文字とその隣接文字を対象に更新する．文字クラスはそれぞれ連鎖の種類ごとに親子の文字を持っていて，そのデータを元に 2 つの文字の連鎖関係を判定する．このときぞろ目は親も子も同じ文字を持っていて，判定の時は親か子のどちらかのみ判定する．

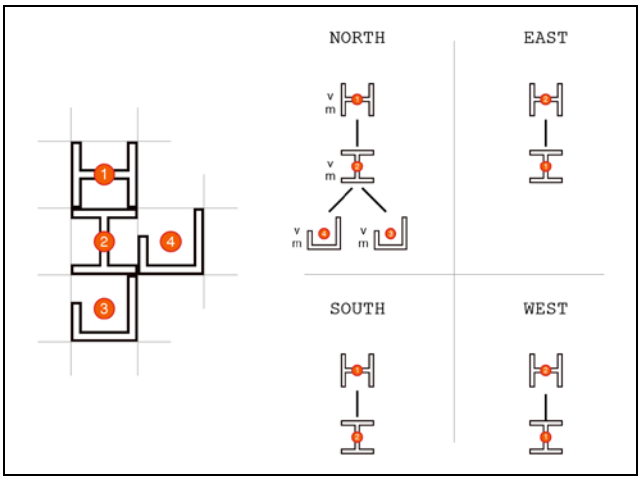

図 10　連番の連鎖ツリー例

Figure 10　An Example of Alphabetical-Chain Tree

6.3.2 連鎖の加点条件判定

次に連鎖が加点条件を満たしているかどうか判定する．加点条件には主に成熟と有効の 2 つがあり，どちらも満たしていないと加点対象にならない．成熟とは連鎖の長さに関する条件で，連鎖の種類ごとに定められている加点される連鎖の長さを満たしていれば成熟していることになる．有効とは連鎖の重複や分岐に関する条件である．連鎖はしばしば包含関係になることがあり，図 10 の盤には図 11 のような包含関係が存在する．このような関係において以下のルールがある．

(1)　$\alpha \subseteq \beta$ のとき β のみ有効 (包含択一の条件)

(2)　$\alpha \perp \beta$ のときどちらも有効 (分岐の条件)

これらを有効の条件という．図 11 に示した例以外にはぞろ目だと必ず全方向重複が生まれどれか一方向だけ有効となる．また図 10 の文字の左側の文字は v が valid(有効)，m が mature(成熟)を示している．連番ではほとんどの場合成熟していれば有効であることが多いが，「X*Z」のときは成熟しているが南北で重複しているため南北のどちらかが無効となる．このように成熟しているが無効である状態を共存という．

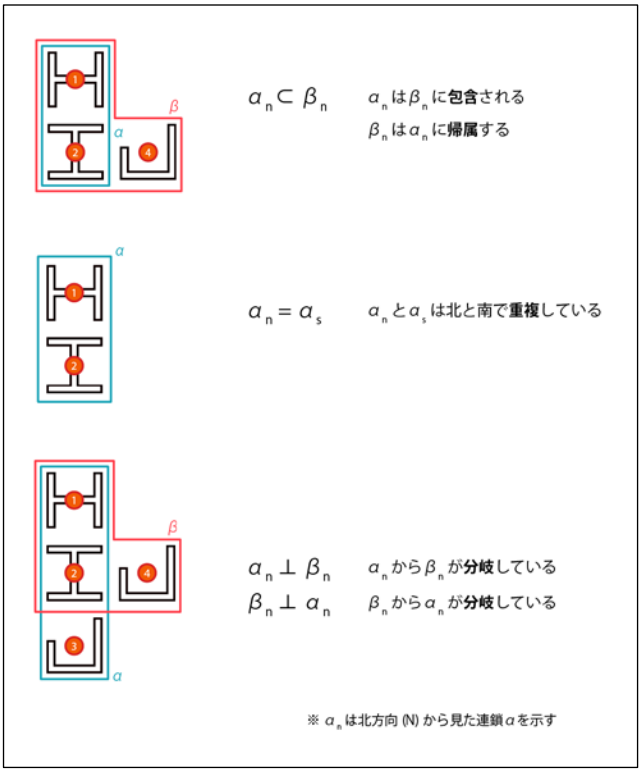

図 11　連鎖の部分集合

Figure 11　Subsets of Chain

6.4 得点換算

連鎖ツリーを探索して得点を換算する．探索は必ずルートノードから開始し全てのリーフノードまで再帰的に探索する．このとき通るノードは常に成熟，有効である必要がある．まず初めにロイヤルの連鎖を探索し，その次に連番、ぞろ目の連鎖を探索する．もしロイヤル文字列が完成していたらその時点で探索を中断しゲームは終了する．得点はリーフノードの高さの 2 乗となる．

7. AI の実装に向けて

本ゲームはプレイヤーが少ないため研究が進んでおらず，定跡や手筋といった勝ち方はまだ分かっていない．現時点では考案者であり最もプレイ回数の多い第 1 著者の手筋を元に，AI 開発を進めていくほかない．

7.1 手筋

本ゲームの手筋は主に以下の 4 つに分類できる．なお番号が若い方が簡単な手筋である．

(1)　連番，ぞろ目の連鎖を延ばす

(2)　相手の連鎖を妨害する

(3)　連鎖の新規開拓

(4)　ロイヤル文字列完成

(1)は得点を稼ぐ上で最も基本的な手筋で，既に盤にある文字から連鎖を組んだり，連鎖を延ばしたりすることであ

る．また得点は多項式関数で表されるので，常に長い連鎖を優先させる．

(2)の妨害は図 12 で例示している．この盤面で先手は P を使って NOP の連鎖を組もうとしているが，これを現在の手番である後手は妨害したい．例えば手 R_1 では I の辺を使って手 B_1 を妨害している．だが手 B_3 は防げていないので悪手である．手 R_2 の場合，元々着手可能だった手 2 つは防げるがその手によって B_2 は打ててしまう．このように本ゲームでは自分の手によって相手が着手可能な手が増えてしまうため，現時点で打てない手も考慮して手を打たなければならない．このとき最善手なのは R_3 である．Js の理由は O に接することと(-1, 2)の右下に点がないことである．また手筋(1)と(2)を両方達成することは難しく対象の連鎖は将来性があるか，現段階で優勢か劣勢かなど状況に応じて判断する必要がある．

(3)は加点に繋がる連鎖を組めない盤面のときに，一から連鎖を組む手筋である．場のカードから連鎖の組み合わせを探索し，最も長く連鎖を作れそうな配置を探す．なお基本的に辺の多い文字は連鎖が組みづらい．例えば A-G あたりは辺が多く連鎖が組みにくいが，T-Z あたりは辺が少ないので連鎖が組みやすい．そのため連鎖の組みにくいカードを積極的に妨害に使用し，組みやすいカードを新規開拓に使用するとよい．またできるだけ場のカードをバラバラに使用して虫食い状態になることは避けるべきである．特に終盤は妨害できるカードが減るため連鎖が組めるカードを残しておくのも手だ．他にも初手を兼用カードにすることで連鎖の組み合わせが増加し相手に精神的負荷をかけるという手筋もある．

(4)は最も難しく，「CHAIN」を揃えようとすれば相手に察知されるためほとんどの場合防がれる．

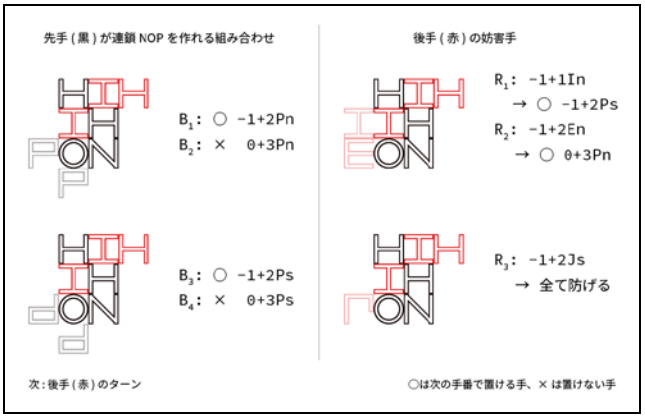

図 12　妨害の例

Figure 12　Examples of Blocking

7.2 詰み

本ゲームには相手の手に依らず必ず次の手番で加点される手を打てる「詰み」の状況がある．図 13 は 1 手詰，2 手詰の例示している．薄くなっている文字が次に打つ文字で，初手以外はどの手を選択しても黒が詰められる．1 手詰の盤面は初期配置だが，このように本ゲームでは先手の初手詰みが可能である．なお後手は妨害をせず同じような手を打てば先手に続いて詰むことができる．2 手詰は相当稀なケースで通常のゲームでは起きにくい．

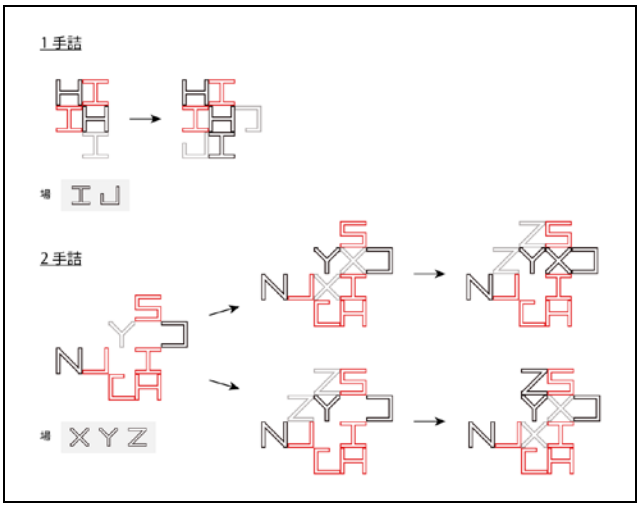

図 13　詰みの例

Figure 13　Examples of Local Checkmate

7.3 探索アルゴリズムと評価関数

オセロや将棋など多くの完全情報ゲームでは alpha-beta 法という探索アルゴリズムが有効である．探索空間が広いため α，β 値をどう設定するかが鍵となる．しかし 2 連鎖までは加点される手が少なく最低でも 3 連鎖は読まないと分からず，特に中盤は連鎖を含む合法手が多く手を刈りにくい．また最も難しい手筋が 3 番の新規開拓で，場にあるカードと盤面から連鎖がより続く最善の配置を選択する必要がある上，多項式関数的に得点が増加するためより深く探索する必要がある．また将棋やチェスと大きく異なり本ゲームは必ず一定のターンで終局するため，囲碁において有効なモンテカルロ木探索[6]を用いてプレイアウトを重ねて探索させることも検討している．

評価の対象となる要素は以下の通りである．

(1)　得点
(2)　連鎖数
(3)　文字の強さ
(4)　合法手の数

(3)の文字の強さとは点や辺の数や兼用できるかどうかなど連鎖の組みやすさを示す指標である．この指標により深く探索しなくても悪手を刈れることが期待できる．(4)は相手の合法手が少ない方が有利であるという憶測に基づいている．

7.4 ランダムに手を選択するプレイヤーの実装

本ゲームの AI 開発の前段階として，合法手を全探索するアルゴリズムを実装した. 指し手はランダムに選択する. 結果として 2 手先までは高速に処理できたが，3 手先ともなると平均して 10^9 手になるため処理が終わらず探索できなかった．2 連鎖までしか探索できないとなると得点の評価が困難なため全探索するアルゴリズムの実用性は低いことが分かった.

8. 課題と展望

本研究における課題は 3 点ある. 一つ目は本ゲームの AI 開発において手をどう評価し刈っていくかということである．二つ目は本ゲームのプラットフォームにおいてアスタリスクを導入することである．しかしアスタリスクはアルゴリズムの複雑化を招くため，まずはアスタリスクを考慮しない AI の開発が優先である．また現時点では考案者の手筋を元に開発するしかないので，リファレンスを含めてプラットフォームを公開し, AI 開発者を募ることを検討している．三つ目は人間同士で対局するネットワーク対戦の実装を検討している．これは本ゲームのプレイヤーを増やし，ゲーム自体の研究を加速させることが狙いである．また人間同士の対局の棋譜を収集することも視野に入れている.

他にも本ゲームで実装したデータ構造を元に，完全情報ゲームの汎用的プラットフォームの実装を検討している．これはオセロや将棋といった盤に配列する完全情報ゲームを対象として，ユーザがルールやコンポーネントを設定してプログラミングせずともゲームを実装できるような仕組みである．もしこれが実現すれば完全情報ゲームの汎用的な AI が実現できるのではないかと考えている．現段階では機械学習などを使って AI に学習させるには，プラットフォームを作りルールを与えなければならず，ゲームによって手筋や定跡の導入や調整が必要である．ただ完全情報ゲームは不完全や不確定のゲームよりは共通する性質が多いので，汎用化がまだ容易だと考えた．ルールを共通のプラットフォームで実装し，評価も統一化できれば，AI の統一化も図れると推察される．またルールの実装が難しければ機械学習を利用して棋譜からルールを推定することも視野に入れている．もしプラットフォームに与えるルールも AI が生み出せたら，AI がゲームを考案できる時代が来るかもしれない.

参考文献

[1] “HICHAIN ルール”. https://scrapbox.io/hichain/ルール, (参照 2017-10-09)
[2] “HICHAIN 用語集”. https://scrapbox.io/hichain/用語, (参照 2017-10-09)
[3] “Trax+FPGA JP”. http://lut.eee.u-ryukyu.ac.jp/traxwiki/index.php?Trax%2BFPGA%20JP, (参照 2017-10-09)
[4] “HIchain-Prototype ソースコード”. https://github.com/hichain/hichain-Prototype, (参照 2017-10-09)
[5] “HIchain-Java ソースコード”. https://github.com/hichain/hichain-Java, (参照 2017-10-09)
[6] 美添 一樹, 他. コンピュータ囲碁 ―モンテカルロ法の理論と実践―, 共立出版.