修 士 論 文

ターン制ストラテジーゲームにおける候補手の抽象化によるゲーム木探索の効率化

北陸先端科学技術大学院大学
情報科学研究科　情報科学専攻

村山 公志朗

2015 年 3 月

修 士 論 文

ターン制ストラテジーゲームにおける候補手の抽象化によるゲーム木探索の効率化

指導教員　池田 心 准教授

審査委員主査　池田 心 准教授
審査委員　飯田 弘之 教授
審査委員　長谷川 忍 准教授

北陸先端科学技術大学院大学
情報科学研究科　情報科学専攻

1210054　村山 公志朗

提出年月: 2015 年 2 月

概 要

これまで人工知能研究の一分野としてチェス，将棋，囲碁などを題材にして，探索や機械学習などの技術の研究が行われてきた．それらの進歩と計算機資源の増大に伴い 1997 年に IBM のチェスプログラム DEEPBLUE が人間のチャンピオンを破るほどの成果をもたらしている．また，コンピュータ将棋においても，ボナンザ法や実現確率探索などのブレイクスルーによりプロ棋士レベルにあり，これらのプログラムは強さにおいて一般的な人間プレイヤにとり十分な強さにある．しかし，チェスや将棋よりも複雑なルールを持つゲームの中には，古典的な着手決定アルゴリズムでは十分強いコンピュータプレイヤが作りにくいものも存在する．例えば，チェスや将棋などが属する二人交互離散ゼロサム完全情報ゲームであっても，本研究で取り上げる大戦略に代表される，同一手番において複数の駒を操作できる，ターン制ゲームのコンピュータプレイヤにおいては，人間の上級者程の強さには至っていない．

また，ターン制ストラテジーゲームでは，コンピュータ将棋におけるボナンザ，ぷよぷよにおける Poje，StarCraft での BWAPI のように研究用途に焦点を置いた，オープンソースのプラットフォームが少数である．さらに，研究用の統一ルールも確立されておらず，関連研究も特に国内においては，ほぼ皆無である．そこで本研究では，ターン制ストラテジーゲームの研究用統一ルールやプラットフォームが不在であること，およびコンピュータプレイヤの強さが十分な域に達していないこと，この二つの問題の解決を目的とした．

まず初めに，研究用途に適したルールを提案するため，既存の戦略ゲームからそれらを構成している 25 の特徴要素を列挙した．次に，統計処理ソフトを用いて代表的な 17 の既存の戦略ゲームを計 13 個からなる特徴量によりクラスタ分析を行うことで，グループ化を試みた．この分析結果から，チェスや将棋等の古典的なゲームのグループからは，まず大戦略など現代的な戦略ゲームのグループへと派生し，そこから RPG 要素を加えたグループ，不完全情報やリアルタイム制などの要素により，複雑なグループなどへ枝分かれしていくことが分かった．そこで，チェスや将棋の次に取り組む対象としては，大戦略等の戦略ゲームの中では比較的単純な，主要ルールのみを持つグループへと進む途上にあるようなルールを策定する方針をとった．

この方針に従った結果，既存の戦略ゲームとの親和性も踏まえ，シリーズを重ねてルールが洗練されている，ファミコンウォーズ DS2“Advance Wars Days Of Ruin”（FWDS2）を基本に，このゲームのルールを簡略化する方向性でルールを提案した．詳細なルールの策定は，健全で再現可能な論文作成に必要な学術研究プラットフォームに必要と思われる 6 つの方針に従って行った．

次に，提案ルールに基づいて，実際に研究用プラットフォームを構築した．本研究で開発したシステムは，あくまで研究用途に特化したものであり，遊ぶ上での面白さを過度に重視するものではない．そこでまず，ルール提案と同様にゲームＡＩ研究用のツールとし

てシステムが要求される機能を，いくつかの項目として洗い出した．要求項目のうち代表的なものには，システムを利用して人間同士，人間と AI，AI 同士の対戦比較が可能といったものから，対局の棋譜を保存できて，後から読み込み，再生してログの確認を行える機能など，プラットフォームに必須の機能を取り入れた．また，プラットフォームそのものを操作するための機能とは別に，思考ルーチンを開発する上で，ユーザーが開発に専念し易くするため，プラグラム設計に工夫を凝らし，ゲーム AI 開発専用のライブラリを用意するなどした．また，プラットフォームの名称は TUrn-Based-STrategy-Academic-Package の頭文字を取り TUBSTAP（タブスタップ）と命名した．構築したプラットフォームは，JAIST 池田研究室の web ページからダウンロードして利用することができ，既に国内の複数の大学や高等専門学校の研究室でゲーム AI の研究開発に利用されている．

また，ルールやプラットフォームの整備に加え，アルゴリズムの改良も行った．大戦略やファミコンウォーズ等の市販ゲームの思考ルーチンは，しばしば賢くない動作を取っているように思われる．これは，着手可能候補手数が膨大でチェス，将棋，囲碁のように $\alpha\beta$ 探索やモンテカルロ木探索を適用することが困難なのが一因である．しかしながら，提案ルールの特徴として，候補手の中に類似性が多く見受けられる．特に駒（ユニット）を操作させる手には類似手が多く含まれる．例えば，自分の駒を敵の駒から離すような手は，厳密には全て異なる手だが，戦況を大きく揺るがすような手は少ない．そこで，本研究では主に駒を移動させる手を抽象化するアプローチを取った．対象のゲームには TUBSTAP を用いて，具体的には一つの駒を移動させる手を，敵から離す手と敵に近づける手の二つに大きく抽象化した．また，敵駒を攻撃する手を抽象化しても性能に影響を及ぼすのか検証するため，4 通りの抽象化法を提案して対戦実験を行った．その結果，同一パラメタの実験条件のもと，駒を移動する手のみ抽象化を行った UCT 探索が，抽象化を行わない UCT 探索に対して 63.1%となり抽象化が有望であることが示せた．

目 次

第 1 章　はじめに **1**
1.1 背景 1
1.2 目的 1

第 2 章　関連研究 **2**
2.1 既存の戦略ゲーム・複数駒移動ゲームについて 2
2.1.1 Simulation Role Playing Game（SRPG） 2
2.1.2 Real Time Strategy 2
2.1.3 Arimaa 3
2.2 既存の研究用プラットフォーム 4
2.3 アルゴリズムに関する既存研究 6

第 3 章　既存の戦略ゲームとルール分析 **7**
3.1 戦略ゲームを構成する要素 7
3.2 戦略ゲームの構成要素のクラスタ分析 11

第 4 章　研究用基盤ルールの提案 **13**
4.1 ルールの設計目標 13
4.2 ルールの設計方針 15
4.3 具体的なルール 16

第 5 章　研究用プラットフォームの構築 **22**
5.1 プラットフォームに要求される機能 22
5.2 クラスの設計指針 24
5.2.1 思考ルーチン開発に適したクラス設計方針 24
5.2.2 クラスとその機能 24

第 6 章　提案ルールにおけるゲーム木探索の適用 **27**
6.1 ゲーム木探索適用のアプローチ 27
6.2 TUBSTAP へのゲーム木探索適用の困難さ 28
6.3 モンテカルロ木探索 29
6.3.1 UCT（UCB applied to Tree） 30

第 7 章　UCT 探索の TUBSTAP への適用 **31**
7.1　候補手の抽象化とは 31
7.2　TUBSTAP における候補手の抽象化の考察 33
7.3　移動行動の抽象化法 34
7.4　攻撃行動の抽象化法 36

第 8 章　抽象化の評価実験 **38**
8.1　有望な抽象度を探る実験 38
8.2　行動可能ユニット数に応じて抽象度を変更する実験 40
8.2.1　行動可能ユニットが少ない局面での調整 41
8.2.2　行動可能ユニットが多い局面での調整 42

第 9 章　まとめと今後の課題 **44**
9.1　まとめ 44
9.2　今後の展望 45

第1章　はじめに

1.1　背景

これまで人工知能研究の一分野としてチェス，将棋，囲碁などを題材にして，探索や機械学習などの技術の研究が行われてきた．それらの進歩と計算機資源の増大に伴い 1997 年に IBM のチェスプログラム DEEPBLUE が人間のチャンピオンを破るほどの成果をもたらしている．また，コンピュータ将棋においても，ボナンザ法や実現確率探索といった手法のブレイクスルーによりプロ棋士レベルにあり，これらのゲームのコンピュータプレイヤはその強さにおいて一般的な人間プレイヤの相手としてほぼ十分である．

だが一方で，本研究で取り上げる『大戦略』に代表される，同一手番において複数回着手可能なターン制ゲームのコンピュータプレイヤにおいては，人間の上級者に匹敵するには至っていない．その原因の一つが，自分の手番に全ての駒を自由な順に操作できるゲームの特性である．これにより，局面分岐因子数が膨れ上がり，MinMax 戦略に基づいたコンピュータプレイヤの開発が困難となる，そのため，チェスや将棋などで蓄積されてきた技術の適用がならず，依然弱いままであるのだと考えられる．

また，本研究で扱う“ ターン制ストラテジーゲーム ”とは，盤面上で多数の駒を操作するゲームである．ターン制ストラテジーでは，コンピュータ将棋におけるボナンザ，ぷよぷよにおける Poje，StarCraft での BWAPI のように研究用途に焦点を置いた，オープンソースのプラットフォームが少数であるといった課題がある．

1.2　目的

本研究では，ターン制ストラテジーゲームの研究用統一ルールやプラットフォームの不在，コンピュータプレイヤの強さが十分な域に達していないといった，二つの課題の解決を目的とする．まず既存の戦略ゲームのルール分析を行い，研究用途として妥当なルールとは何かを考察，提案し，コンピュータプレイヤ開発にあたって，有用なプラットフォームの開発を行う．さらに候補手の抽象化による解決法を用いて，局面分岐因子が膨大で，ナイーブな実装では MinMax 戦略をとることが困難な問題に対して，ゲーム木探索の効率化するにはどうすればよいかを研究する．そして，モンテカルロ木探索に代表する，これまで囲碁などの二人ゲームで成功を収めた手法を適用し，ターン制ストラテジーにおける強いコンピュータプレイヤの構築を目指す．

第2章　関連研究

本章では，大戦略とそこから派生した，SRPG，RTS などの既存の戦略ゲームおよび，同一手番において複数回着手可能なゲームである Arimaa 等について，その特徴と研究対象としての適性を考察し 2.1 節で述べる．次にこれに対して既存手法を適用しコンピュータプレイヤの性能向上を試みた関連研究を，2.2 節で紹介する．さらに，2.3 節では既存のコンピュータゲームの研究用プラットフォームを例として挙げ，研究用プラットフォームが必要とされる意義や，要件について述べる．

2.1　既存の戦略ゲーム・複数駒移動ゲームについて

大戦略 [16] とは，1985 年にシステムソフト社より発売された PC ゲームで，正規のシリーズだけでも国内累計売り上げ本数が 80 万本にも達し，PC ゲームとしてはかなり多い部類である．チェスや将棋のように，盤面（マップ）上に配置された駒（戦車や戦闘機などの兵器．ユニット）を手番に動かし，それを勝敗条件がみたされるまで繰り返すゲームであるが，複数着手性などさまざまなルールが追加されている．

大戦略はそこから派生したゲームも非常に多い．本節では，古典的戦略ゲームと言える将棋やチェスや大戦略から見た場合に，どのような派生が行われているのか，それらが研究対象として適しているのか述べる．

2.1.1　Simulation Role Playing Game（SRPG）

シミュレーションロールプレイングは戦略ゲームの要素と，RPG の要素（物語性・成長など）を併せ持ったゲームジャンルである．代表的なものとして，Final Fantasy Tactics[5] があり，国内 170 万本を売り上げているが，他にも人気シリーズは多い．ゲームの進化においては，このような特徴・要素の水平伝播は頻繁に生じるが，思考ゲームの学術的ベンチマークとしては瑣末な要素が多すぎるきらいがあり不適だと言える．

2.1.2　Real Time Strategy

StarCraft[10] や Age of Empire を代表とするリアルタイム戦略ゲーム（RTS）は，PC の性能向上に伴い比較的新しく出来たジャンルである．RTS では，手番という概念は存

在せず，プレイヤ達は駒にそれぞれ任意のタイミングで指示を出す．盤面にも離散化されたマスという概念はない．欧米では RTS は非常に人気であり，また学術研究も盛んに行われ，国際会議 IEEE-CIG[6] では論文の数割が RTS に対するものである．しかし，ルールが複雑・煩雑でまた思考時間がリアルタイム制のために限定されるなど，チェスや将棋の次に取り組むべきベンチマークとしては，一足とびになっている感が否めない．

2.1.3 Arimaa

Arimaa[1] は，チェスの盤，駒を使用して遊べる二人対戦用のボードゲームで，2002 年に元 NASA 職員の Omar Syed により提案されたボードゲームである．チェスと同様の二人零和確定完全情報ゲームであるが，“ 同一ターン内で 4 手まで自分の駒を動かすことが可能 ” という大きな特徴を持つ．合法手数が膨大になるため，ナイーブな実装では木探索アルゴリズムの適用は困難である．本ゲームは複数着手性のあるゲームの学術的ベンチマークとして利用可能だが，プレイヤ数が少なく棋譜収集や評価が困難といった問題もある．

2.2　既存の研究用プラットフォーム

個人，あるいはゲームＡＩの研究者がAIプログラムの開発に取り掛かろうとしたさいに，オープンソースの研究用プログラムが入手可能であるか否かは大きな関心事である．仮にプラットフォームが入手不可能なゲームの思考アルゴリズムを構築しようとしたとき，開発者はゲームのルール等，本質とは関係のない所のプログラミングから始める必要があり，ＡＩ開発の負担増加となる．さらに，プログラミングの経験は浅いがＡＩの開発を行いたい人にとっても，ＡＩ開発の敷居を高くする要因になっていると言える．また，オープンソースのプラットフォームを用いた競技会が国内外を問わず開催されている背景から，ゲームＡＩの研究を活性化するための相乗効果も期待でき，その必要性は明らかである．そこで，本節ではいくつかの種類のゲームについて，オープンソースの研究用プラットフォームを簡単に紹介する．

【StarCraft】

StarCraftは，ブリザード・エンターテインメント社が1988年に発売したRTSゲームである．ゲームＡＩの国際会議IEEE-CIG[6]でも頻繁に，ゲームAIの競技会が行われており，思考ルーチンの開発のためのBWAPI（The Broad War Application Programming interface）が公開されている．これは，C++言語で作られのプログラムである．これを使って開発者はプログラムを開発し競技会に臨み，その強さを競い合う．

【FPS（First Person Shooting Game)】

FPSでは，チューリングテストの派生形の一つであるBotPrizeというコンペティションが有名である．チューリングテストを行うことでゲームＡＩの人間らしさを評価するための大会が頻繁に開催されている．強さに焦点を置いた研究ではないが，2012年のThe2KBotPrize[11]において，大会史上初で人間よりも人間らしいと評価されるゲームAIの開発が成功している．

【Mario】

任天堂のスーパーマリオを基に開発された，MarioAI BencbMarkというJava環境のプラットフォームがある．これは，StarCraftと同様に，IEEE-CIGでMario AI Competision[8]が開催されている．コンペティションでは，いくつかのトラックがあり，作成したゲームAIのステージクリアまでの成績を競う．ゲームをクリアすることを目的としたトラックから，チューリングテストを行うもの等がある．

【囲碁】

オープンソースで開発されている囲碁の思考エンジンにFuego[3]がある．これはモンテカルロ木探索をベースに採用しており，現在，無償で手に入る囲碁プログラムの中では最高クラスの実力を持つ．国内外を問わず多くの論文でこのプログラムを対戦比較用のベンチマークとして利用するなど，囲碁ＡＩのプラットフォームとして有用である．

【将棋】

コンピュータ将棋のオープンソースプラグラムとして有名なソフトにボナンザ（Bonanza）[2] がある．保木によって開発されたプログラムで，ボナンザメソッドと呼ばれる機械学習による評価関数の自動学習をコンピュータ将棋に始めて適用した．その結果，2006 年に開催された，第 16 回世界コンピュータ将棋選手権大で歴戦の将棋ソフトが並ぶ中，初優勝を果たしている．昨今のコンピュータ将棋選手権に登場するプログラムの大半は Bonanza を基に開発されたものであり，このプログラムがコンピュータ将棋界にもたらした影響は計り知れない．

【ぷよぷよ】

研究用の代表的なプラットフォームに，JAIST 池田研究室から Web 上に Poje[17] というプラットフォームが公開されている．このプログラムはぷよぷよのルールをターン制にしたもので，これを土台に落下型パズルゲームの学術研究が行われている．

【格闘ゲーム】

格闘ゲームにおいても，コンペティションが開催されている．FightingICE[4] がその一例として挙げられる．これは，格闘ゲームの AI キャラクタを開発するためのプラットフォームである．Java 言語による開発環境が用意されている．

2.3　アルゴリズムに関する既存研究

複数回着手性を持つゲームに対して，モンテカルロ木探索を適用した研究について述べる．モンテカルロ木探索はコンピュータ囲碁において成功を収めた手法であり，ゲームの知識を使用して評価関数を作成せずともよく，低負荷で思考アルゴリズムの開発が可能なため，多くのゲームでこれに基づいたプログラムが開発されている．複数回駒移動ゲームに対してモンテカルロ木探索を適用した研究には次の二つが挙げられる．

一つは，Tomas らによる，モンテカルロ木探索の一手法である UCT 探索を用いて，Arimaa の強いコンピュータプレイヤ構築を目指した研究である [12]．この研究では行動評価関数やヒストリーヒューリスティックを用いて UCB 値に補正を加えることで，UCT 探索の効率化を試みている．その結果，$\alpha\,\beta$ 探索を土台にした既存の Arimaa プログラムと比較して有望な性能が得られた．これにより，分岐数の多い複数駒移動ゲームに対しても囲碁やチェス等で洗練されてきた手法の適用による強いコンピュータプレイヤ構築の可能性が示された．

また，加藤ら [15] は自分の手番に全ての盤上の駒を操作可能なゲームを対象にして，コンピュータプレイヤの構築を行った．アプローチには UCT 探索をベースに，駒を移動させる手の中で，無駄に思える手を予め枝刈りすることで探索の効率化を図っている．これによって，枝刈りを行わない UCT に対して有望な勝率が得られた．

第3章　既存の戦略ゲームとルール分析

戦略ゲームにはそれぞれルールが定まっており，重要なものからそうでないものまで，それぞれのゲームの特徴を形作っている．本章では次章以降で研究用途として妥当なターン制ストラテジーのルールを設計し，プラットフォームを構築するための足掛かりとして，まずは既存の戦略ゲームにどのような要素が存在するのかを列挙する．そのうえで，多様な戦略ゲームをいくつかのグループに分けることを試みる．

3.1　戦略ゲームを構成する要素

本節では，我々が標準的だと考える戦略ゲームに登場する概念・要素・システムを列挙する．これらが用いられるかどうかはゲームごとに異なる．前半はチェス等にも登場する概念であるが，後半はより発展的な要素である．以下に F1〜F25 までの要素について順に説明する．

F1. 盤面

将棋やチェスのように正方形のマス（図 3.1）のほか，図 3.2 に示すような蜂の巣状のマス，またはマスなしなどが用いられる．

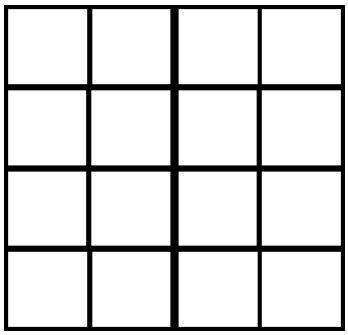

図 3.1: 正方形のマス

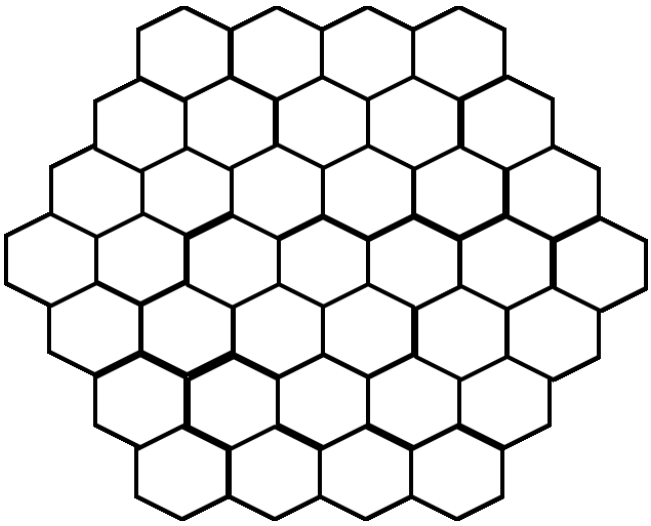

図 3.2: 蜂の巣形の盤面

F2. プレイヤ数

２人とは限らない．例えば他人数対戦ゲームとしてカタンの開拓者 [14] などがある．

F3. 駒（ユニット）

通常，複数種類の駒がある．SRPG などでは１ユニットが１つのキャラクタのことが多く，それぞれに個性がある．

F4. 着手順

交互に１ユニットずつ（図 3.3），交互複数ユニット（図 3.4），交互全ユニット（大戦略），リアルタイム（StarCraft）などがある．

図 3.3: 交互１駒の例（将棋盤）

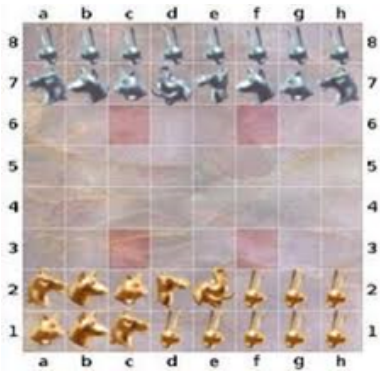

図 3.4: 交互複数駒の例（Arimaa）[1]

F5. 勝利条件

相手駒を全て破壊する，特定駒を取る，特定マス・拠点を占領するなど，様々．

F6. 駒の相性

じゃんけんのように，駒の間に得手不得手があることが多い．

F7. HP（ヒットポイント）

１つの駒の体力・健全度などを表す指標で，0 になると破壊される．下記ルールの前提になっている場合が多い．

F8. 攻撃方法

チェスなどでは相手駒のマスに移動することでその駒を倒せるが，隣接マスから攻撃して HP を減らすタイプのゲームが多い．遠距離攻撃や範囲攻撃が可能なことも多い．

F9. 反撃

隣接攻撃の場合，攻撃を受けた側も（後手あるいは同時に）攻撃できるシステム．

F10. 地形

マスには沼地・要塞など，移動や防御に影響を与える種類のものがあることが多い．通常は固定だが，橋をかけたり 要塞を構築できる工事のシステムがあることもある．また地上と空や海中などを多層的に持つものもある．

F11. 移動

ユニットは「移動力」を持ち，地形に対する「移動コスト」を払いながら１歩ずつ進むことが多い．自ユニットは通過できることが多く，敵ユニットはできない場合が多い．Zone of Control （ZoC）という，敵のそばをすり抜けることもできないルールもある．

F12. 非対称性

初期配置，地形等は必ずしも対称ではなく，有利不利がある場合もある．

F13. 占領

マスに都市・工場・飛行場などの機能を持つ種類のものがあり，歩兵など特定ユニットでそれを自軍のものにできるシステム．

F14. 生産

都市からの収入を用い，工場などから新規にユニットを登場させるシステム．

F15. 練度・経験値・レベル

経験を積むことでユニットの性能が向上するシステム．

F16. 弾数
　特定の攻撃手段は計何回までしか使えないといった限定を設けるシステム．

F17. 補給，補充
　都市等で，損耗した HP や弾数を回復させるシステム．

F18. 索敵
　自ユニットの周囲特定マスしか敵ユニットの存在が確認できないシステム．不完全情報ゲームになるため，アルゴリズムの作成難度に影響が大きい要素である．

F19. ランダム性
　攻撃が一定の確率で当たらなかったり，HP に与える損害が上下したりするシステム．状態遷移が不確定になるため，思考アルゴリズムの作成に影響が出うる．

F20. 指揮官
　複数の近隣ユニットに影響を与えたり，特殊能力を持つ存在のシステム．

F21. 合流，分散
　複数のユニットが同じマスに集まって HP や弾数を合算したり，その逆を行うシステム．

F22. 内政
　生産力を上げる，新種ユニットを開発するなどができるシステム．

F23. 陣形，包囲効果，支援効果
　戦闘時の周囲の状況により，攻撃力等に変化が出るシステム．

F24. 戦略爆撃
　敵の都市などの施設を爆撃し機能を低下させるシステム．

F25. 天候，季節，時刻
　夜は視界が落ちる，雨が続くと平地が沼地になるなどさまざまな状況が場合によって変化するシステム．

3.2　戦略ゲームの構成要素のクラスタ分析

この数十年で非常に多くの戦略ゲームが提案・発売されたが，それらを「X は Y の発展形」「X は Y と Z の合体」のように関係づけて進化の様相を探ることは困難である．本節では，代表的な戦略ゲームをクラスタ分析することで，それらがどのようにグループ化され，どのような近さにあるのかを見ることにする．

図 3.5 は,（恣意的に選んだ，ただしそれなりに著名な）17 の戦略ゲームと，4 章で提案する「研究用基盤ルール」を持つ戦略ゲームを，統計処理ソフトの R を用いてクラスタ分析し，2 次元上に表現したものである．分析に用いたのは，駒の個性固有（F3），複数着手性（F4），リアルタイム性（F4），相性（F6），HP（F7），地形（F10），地形の多層性（F10），ZoC（F11），占領（F13），生産（F14），レベル（F15），索敵（F18），内政（F22）からなる 13 個の特徴量である．この結果次のことが見て取れる．

- Arimaa 複数着手性があり，軍人将棋や軍棋には不完全情報性や相性があるため，将棋やチェスに比べ現代的なゲームに近い位置にある．
- 中心付近に大戦略・ファミコンウォーズなど，基本形に近いと思われるゲームが位置する．
- ファイアーエムブレム等の SRPG は基本形から一部要素（生産等）が単純化され古典的ゲームに近付く一方，ユニットの固有性が追加され，枝分かれした位置にある．
- アドバンスド大戦略・大戦略 EX などは基本形から索敵等高度な要素を加えた発展的位置にある．
- 三国志，StarCraft などは，内政やリアルタイム性など別の高度な要素を加えた発展的位置にある．

次章で述べる提案ルールは，図 3.5 の分析結果から，Arimaa やチェス・将棋等の古典的ゲームからみて，現代的な戦略ゲームの入り口にあり，次に取り組むべきベンチマークとして好ましい位置にあると言える．

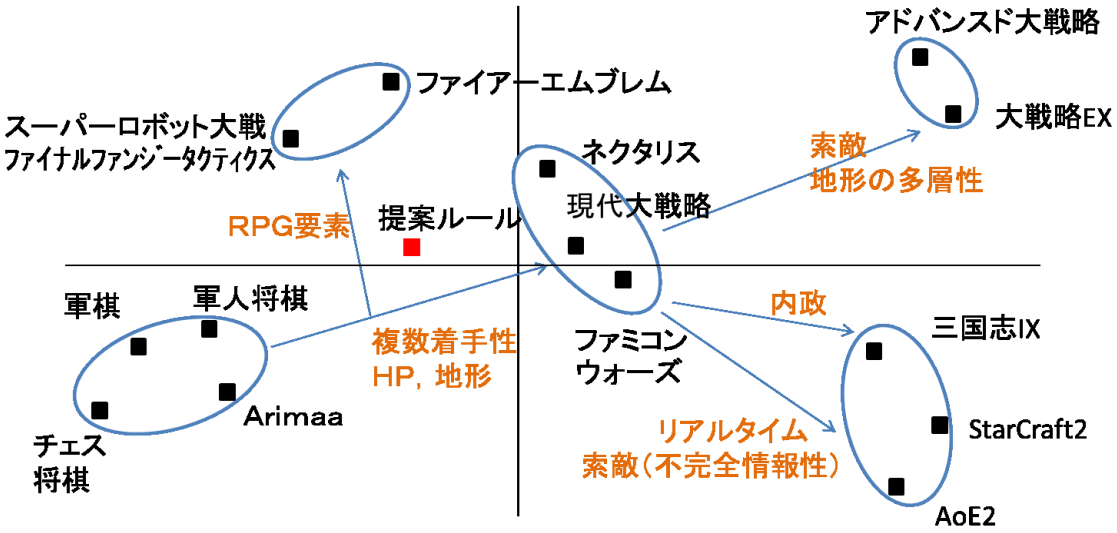

図 3.5: 戦略ゲームのクラスタ分析の結果

第4章　研究用基盤ルールの提案

前章では，多様な戦略ゲームとそれらを構成する要素について紹介し，チェスや将棋を源流とした関係性をみるためのクラスタ分析を行った．本章では学術的なベンチマークとして適したゲームとそのルールを提案するため，多くの戦略ゲームに共通する基盤的なルールを提案する．そのためにまず，基盤ルールに必要とされる要求項目を取り上げる．続いて前章でのクラスタ分析の結果を踏まえて，既存の戦略ゲームのルールを簡略化する方針でルールセットの構築を行う．そして 4.3 節では 3 章で紹介した戦略ゲームを構成する 25 つの要素から，提案ルールに含める具体的な 11 つの要素を示し詳細を述べる．

4.1　ルールの設計目標

ゲームを研究対象にアルゴリズムの性能評価実験など行おうとしたとき，そのルールが面白さを損なわずに学術用途に適度に簡略化されているのは大きな関心事である．健全で再現可能な論文作成のためにも重要である．本節では設計目標として，ルールに求められうる要求項目を挙げる．必須で無い項目も含むが，これらを意識してルール設計を行っていく．

R1. ルールが明確に定義されている．
　思考ルーチンを開発して性能評価実験を行おうとしたとき，ルールが再現できるか否かは比較の面で重要である．

R2. 多くのゲームに共通する要素を持ち，少ないゲームにしかない要素は持たずにコンパクトである．

R3. より高度な要素を含むルールセットへの拡張可能性を考慮している．
　思考ルーチンの性能が十分な域に達した場合，より複雑な対象ルールが求められうる．そのため，次の段階の複雑さを持つルールへの拡張が容易であることが望まれる．

R4. プレイして楽しく，人による評価や人同士の棋譜の収集が望める．
　プレイした上での面白さは，ゲームの愛好者を増やすためにも必要であると考える．それにより人間同士の対戦棋譜の収集が見込めれば，棋譜を用いた学習等のベンチマークとしての利用を期待できる．

R5. 既存のゲームに似ており，とっつきやすい．.

ターン制ストラテジーの愛好者は勿論のこと，チェスや将棋は熟知しているが，この手のゲームが初めてな人にとり，とっつきやすい内容である必要があると考える．これは，ゲームの愛好者や，競技人口を増やすために必要である．

R6. 既存の市販ゲームと対戦比較が可能である．

既存の市販ゲームと対戦して思考ルーチンの性能評価が可能であれば，アルゴリズムが市販ゲームに対して有意な性能を出せた場合，インパクトのある結果として示すことができる．

4.2　ルールの設計方針

ルール設計に関する中心的な要請は，前節 R2 と R4，面白さを損なわず本質を失わない程度に，できるだけ簡素化することである．我々は，前節 R5 と R6，既存ゲームとの親和性の要請を踏まえ，シリーズを重ねすでにルールがかなり洗練されている任天堂社の“ファミコンウォーズ DS2 Advance Wars Days Of Ruin”（FWDS2） を基本に，そこから高度あるいは瑣末な要素を削っていく方向でルールを設計することにする．なおこのソフトはオプション選択やマップ作成機能を持ち，提案するルールが一部を除いて再現可能である．FWDS2 は，3．1 節で述べた要素で言うと，四角形の盤（F1），2 人（F2），複数種類の駒（F3）という意味でチェス等と似ているが，複数着手性（F4）を初め，F5 から F21 までの全ての要素を含んでおり，大きく異なる．我々は 3．2 節のクラスタ分析に用いた 15 つ以外にも多くの戦略ゲームを分析し，FWDS2 の持つ要素を大まかに以下の 4 グループに分類した．

Group1. 殆どの戦略ゲームにある，必須のもの．複数着手性（F4），相性（F6），HP（F7），隣接と遠隔の攻撃（F8），移動（F11）．

Group2. 多くのゲームにあり，必須であるとは言えないが残しても良いと思われるもの．さまざまな勝利条件（F5），反撃（F9），地形 （F10），非対称性（F12）, 小さなランダム性（F19）.

Group3. 多くのゲームにあるが，高度であるので拡張性を考慮しつつ初期ルールセットに含めないもの．占領（F13），生産（F14），弾数 （F16），補給・補充（F17），索敵（F18）．

Group4. 一部のゲームのみにあるか，瑣末であり，考慮しないもの．レベル（F15），指揮官（F20），合流（F21）．

なおこれ以外に，FWDS2 が持たず，提案ルールにも含めないものとして，ZoC（F11），リアルタイム性（F4），分散（F21），内政（F22），陣形（F23）などがある．

G3 に含まれるうち「生産」「占領」は初代大戦略を初め多くの戦略ゲームに存在する，かつ思考ルーチンにも大きな影響を与える要素であるが，チェス等との類似性と，できるだけの単純さのため，ここでは廃した．一方 G2 は廃しても良いのだが，思考ルーチンに与える影響が小さいこと，多くの人間プレイヤがこれらに慣れていることから残すこととした．ルールの詳細は次節で述べるが，3．2 節で述べたように，チェス・将棋・Arimaa などとも比較的近く，複雑な戦略ゲームへの橋渡しになるような部分のルールセットになっていると考えている．

4.3　具体的なルール

これまで，我々が簡略化したルール内に大戦略の各要素を採用するか否かについて議論した．本節では具体的にユニットのパラメータ等のゲームを構成する項目について，F1〜11 の各要素ごとに示す．F12〜25 の要素は含めない．

F1.【マップ，盤面】将棋盤などと同様に，四角形マスからなる二次元の盤面を用いる．(図 4.1）縦横のサイズは特に指定しない．また，将棋等では常に同じ盤面・駒の初期配置を用いるのに比べ，戦略ゲームでは通常，特徴の異なる複数の設定を用い，プレイ時にそれを選ぶ．一つの盤面サイズ，それぞれのマスの地形，用いる駒の種類と数と初期配置の組合せを「マップ」と呼ぶ．これらには非対称性（F12）が導入されていることも多い．

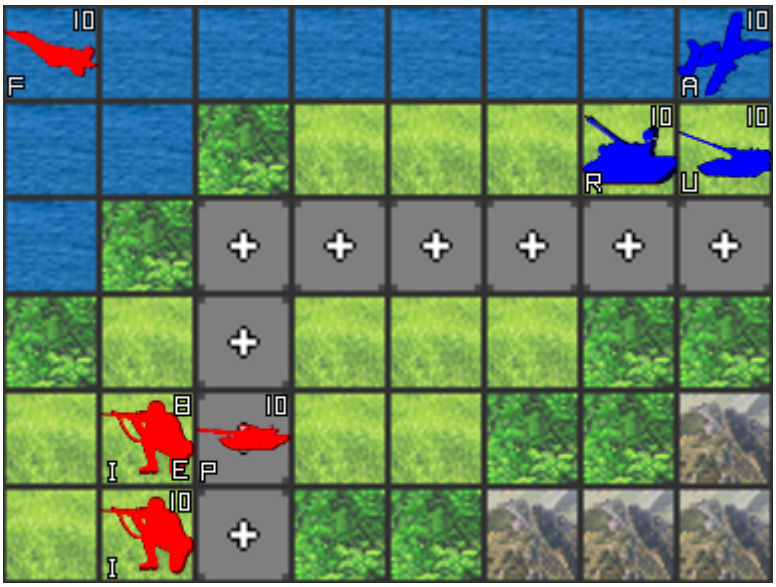

図 4.1: マップの一例

F2.【プレイヤ数】 チェス･将棋と同じでゲーム情報学では二人零和有限確定完全情報ゲームに分類される．将来的に多人数ゲームにルールを拡張することも可能であるが，今回のルールセットは二人ゲームとする．

F3.【駒，ユニット】 ゲームに用いる駒の種類は 6 種類からなる．戦闘機（記号 F で表す），攻撃機（A），戦車（P），対空戦車（R），歩兵（I），自走砲（U）の 6 種類の 駒（ユニット）を用いる．F と A は航空ユニット，他は地上ユニットである．各駒とその特徴と外観を図 4.2 に示す．

- 戦闘機（F）: 移動範囲の広い対空ユニット．A に強く，R に対して弱い．
- 攻撃機（A）: 対地上ユニットで，P・U に対して強く，F・R に対して弱い．
- 対空戦車（R）: F・A に対して強い唯一の対空ユニットで，さほど強くないが対地上ユニットでもある．
- 戦車（P）: 対地上ユニットである．R・U に対して強く，A に弱い．
- 歩兵（I）: 歩兵以外への攻撃力は微々たるものであるが，大事な自ユニット守るための壁として役立つ．
- 自走砲（U）: 唯一の遠距離攻撃ユニットで，範囲内にいる地上ユニットに対して一方的に攻撃可能である．

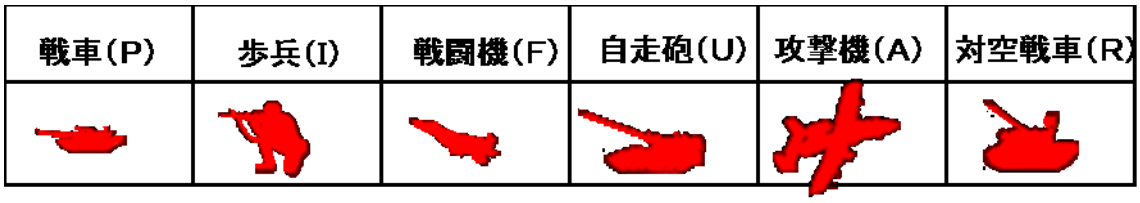

図 4.2: 各ユニットの外観

F4.【着手順】各手番では，プレイヤは全てのユニットを 1 回ずつ自由な順番で選んで行動させることができる．全てのユニットを行動させると，相手の手番となる．両者がそれぞれ手番を終えるまでを 1 ターンと呼ぶ．1 ユニットが 1 ターンに可能な行動は,「移動のみ行う」「移動して隣接攻撃する」「（隣接・遠距離）攻撃のみ行う」「何もしない」の 4 通りである．

F5.【勝利条件】いずれかのプレイヤのユニットが全滅して盤上から消滅した場合，ユニットが盤面に存在している方の勝利となる．また，ターン数に上限を設けて，その上限以内に全滅条件を満たさなかった場合の処理は，引き分けか残りユニット数等により別途条件を設けることで勝敗判定を行う．

F6.【ユニット相性】攻撃側ユニットと防御側ユニットの組合せにより，攻撃の効果が表 4.1 のように異なる．HP に与える具体的なダメージ計算式については後述する．F と R は航空ユニットを得意とし，A，T，U は地上ユニットを得意とする．U のみ間接攻撃が可能である．I は壁役のユニットである．

表 4.1: ユニット相性

	防御側A	F	R	I	T	U
攻撃側A	0	0	85	115	105	105
F	65	55	0	0	0	0
R	70	70	45	105	15	50
I	0	0	3	55	5	10
T	0	0	75	75	55	70
U	0	0	65	90	60	75

F7.【HP】各ユニットは 1 から 10 の HP を持つ．攻撃を受けることで HP は減少し，0 になるとその駒はマップから消滅する．

F8.【攻撃方法】・F9.【反撃】U 以外の各ユニットは，移動前あるいは移動後に，敵ユニットと隣接した状態でのみそのユニットに攻撃ができる（図 4.3 左）．攻撃を受けた側は，HP が 1 以上残っていれば，反撃ができる．U は，移動前のみ，マンハッタン距離で 2 以上 3 以内の敵ユニットに対して一方的に（反撃なしに）攻撃ができる（図 4.3）右．

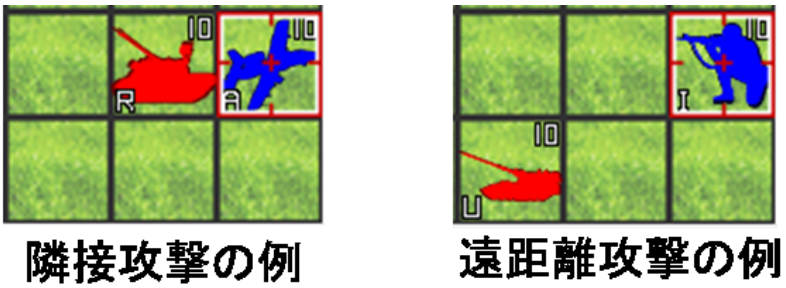

図 4.3: 隣接攻撃と遠距離攻撃がある

F10.【地形】今後拡張していくことを念頭に置きながらも，現時点のルールセットでは，山，海，森，草原，道路，要塞，禁止区域の 7 種類の地形を用いる．図 4.4．地形は，防御効果と移動コストに影響を与える．表 4.2 は各地形とユニットの組み合わせからなる，地形の防御効果である．また，各ユニットごとの地形の移動コストを表 4.3 に示す．

草原		**移動は容易で，防御には向かない地形**
森		**車両は進入困難だが，防御には比較的適している**
山		**地上ユニットは歩兵のみ進入でき防御に最適**
海		**航空ユニットのみ移動可能**
道路		**移動は容易だが防御には向かない地形**
陣地		**移動が容易で，かつ防御に適する**
禁止区域		**全種類のユニットが移動可能な地形**

図 4.4: 地形の概要

表 4.2: 防御効果

防御側	山	森	草原	道路	海	陣地
A，F	0	0	0	0	0	0
R，I，T，U	0.4	0.3	0.1	0	0	0.4

表 4.3: 移動コスト

防御側	山	森	草原	道路	海	陣地
A，F	1	1	1	1	1	1
R，T，U	∞	2	1	1	∞	1
I	2	1	1	1	∞	1

【攻撃の効果】攻撃の結果どれだけ相手の HP を減らせるか（ダメージ）は，攻撃側防御側の現 HP，表 4.1 の相性，表 4.2 の防御側地形の防御効果によって以下の式で定まる．ランダム性（F19）は導入しない．

$$\text{相性} = \frac{\text{相性×攻撃側 } HP}{10 + \text{防御効果×防御側 } HP}$$

F11.【移動】各ユニットは表に示す移動力を持つ．ユニットは上下左右に 1 マスずつ，地形に対する移動コストを消費しながら移動できる．移動力を使い切る必要はない．移動時に自ユニットのいるマスを一時的に通過することはできるが，敵ユニットがいるマスを通過することはできない．図 4.5 は戦車（P）の移動範囲を示した例である．また，表 4.4 は各ユニットの種類ごとの移動力である．

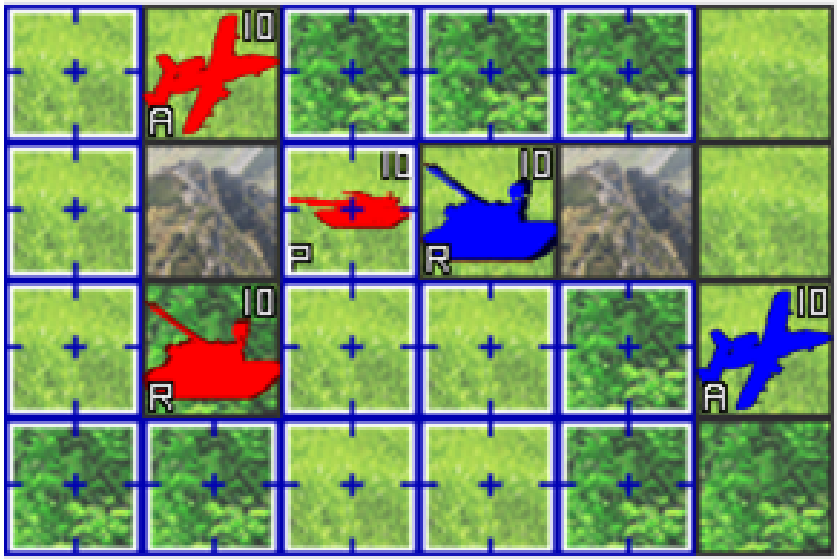

図 4.5: 移動の例，青枠で囲まれた個所が移動可能範囲

表 4.4: 移動力

	A	F	R	I	T	U
移動力	7	9	6	3	6	5

第5章　研究用プラットフォームの構築

前章では既存の戦略ゲームのルール分析を行った．ゲームＡＩ開発のためのベンチマークとして，遊ぶ上で面白さを損なわない範囲で簡略化されたルールを提案した．本章では提案ルールを基に研究用途として有用なプラットフォームの構築を目的とする．そのため，まずはプラットフォームに要求される機能を掲げ，開発環境として，VisualStudio C＃.Net を用いて GUI アプリケーションとしてシステムを構築する．

また，5.2 節では今回構築したシステムにおけるクラス間の設計方針を紹介し，ゲームＡＩを開発するユーザーにとり利便性の高い設計手法について考察する．

5.1　プラットフォームに要求される機能

本研究で構築するプラットフォームは，あくまで研究用途に特化したものであり，遊ぶ上での面白さを過度に重視するものではない．そこで，ゲームＡＩ研究用のツールとして要求される機能を P1〜P9 の項目として洗い出す．そしてこれに基づきシステムを構築した．

P1. 提案ルールの土台として選択したゲーム（FWDS2）のＡＩと上で開発した思考ルーチンとの対戦比較が可能である．

P2. ＧＵＩ環境で人間対人間，人間対プログラム，プログラム対プログラムの対戦が可能である．

P3. 開発した思考ルーチンの評価実験のための画面表示スキップ機能があるなど，長大な時間を要さない．

P4. ルールや GUI が実装されていて，連続対戦による数千回に及ぶ実験環境が整備されている．

P5. 対戦用のマップを選択できる．

P6. プログラム同士を対戦させるとき，評価実験用に複数のマップを自動で選択して高速に対戦させる機能がある．

P7. 対戦中の着手を棋譜として保存し，後から読み込み・再生することができる．ログを吐かせる機能がある．

P8. ユーザーが思考ルーチンの開発だけに専念できるよう，ゲームシステムと思考ルーチン開発部分とのプログラム設計上の分割が出来ている．また，開発をサポートするための便利なツールが揃っている．

P9. ネットワーク対戦機能や自動対戦サーバが実装されている．

図 5.1 に示すのが，本研究で構築した研究用プラットフォームである．本論文の執筆時点では R9 以外の機能は実装されており，研究用のツールとして利用できる機能は実装済みである．アプリケーションの名称は TUrn-Based-STrategy-Academic-Package の頭文字を取り TUBSTAP（タブスタップ）と命名した．また，プログラムは本学，池田研究室の web サイト [13] からダウンロードし利用可能であり，既に国内における複数の大学や高等専門学校の研究室で研究開発に利用されている．

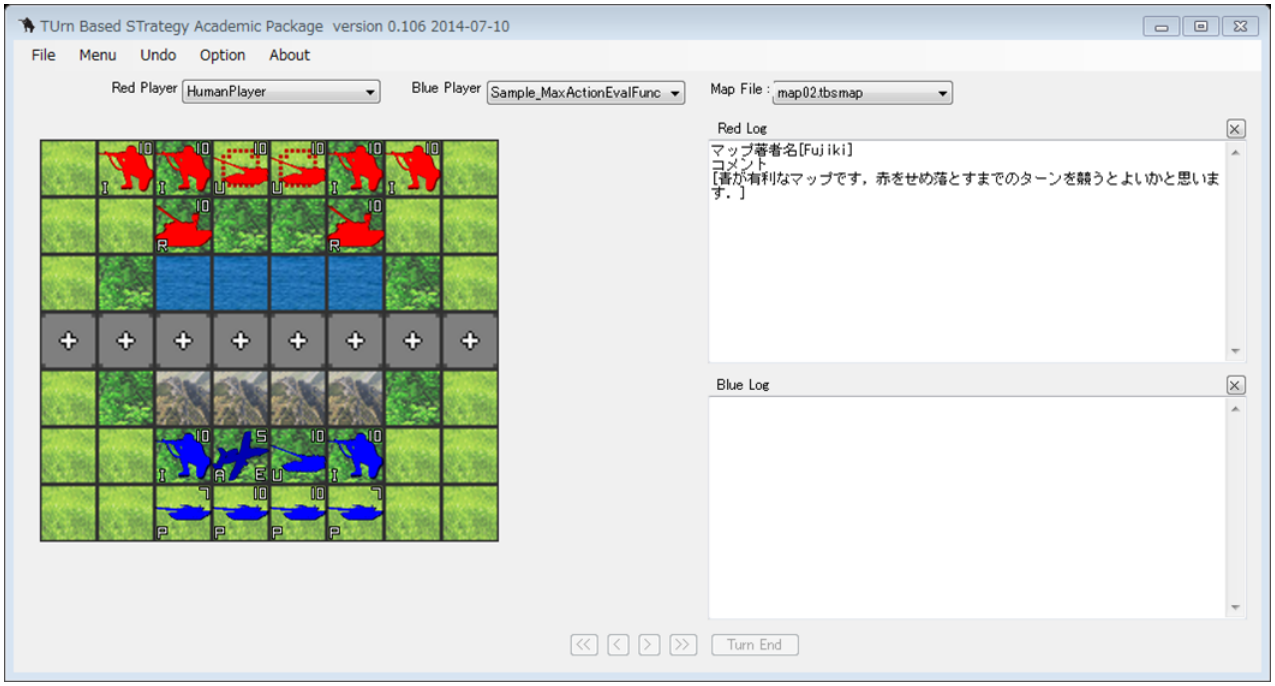

図 5.1: TUBSTAP の画面

5.2　クラスの設計指針

本節では，P8 の要求であるプラットフォームを利用するユーザーが，思考ルーチンの開発だけに専念できるようにするために施した，プログラム設計上の工夫について紹介する．これは，ここで紹介する方法が今後，研究用プラットフォームが存在しないゲームを対象に思考ルーチンを開発しようとする人にとって，クラス設計のための指針となることを目的としたものである．まず，TUBSTAP における，ゲームの思考ルーチン開発に適したクラス設計方法について 5.2.1 節で考察する．さらに，5.2.2 節では本研究で実際に構築した研究用基盤である，TUBSTAP において作成したクラスや関数と，その機能について具体例として紹介する．

5.2.1　思考ルーチン開発に適したクラス設計方針

ゲームＡＩと一口に言っても，捉え方は様々であるが，筆者は主に二つのアプローチがあると考える．一つは，ゲームそのものをプレイするＡＩである．これは，例えば一つの局面が与えられたとき，その局面において自分が取れる全ての行動から，何らかの方策に従って行動選択することを意味する．一方，もう一つのアプローチはゲームの局面上に存在する駒（キャラクター）をＡＩと見なす方法である．これは，一つのエージェント自らが存在するゲーム内の状態を観測して，そのエージェントが方策に従って，自らにとり最適な行動選択をとることを目的とするアプローチである．このような方法はロボカップサッカーのシミュレーションリーグや，StarCraft，その他国内における RPG などにおいて見受けられる．本研究で提案したゲームの TUBSTAP は，チェス，囲碁，将棋など古典的なゲームの発展系としての位置づけにあるため，前者のアプローチを採用する．そして以降，本論文内でゲームＡＩと表記した場合この意味とする．

このような方針でクラス設計を行う場合，ある局面を表すクラスとエージェント（ゲームＡＩ）を表すクラス，行動を表現するためのクラスが最低限必要になると考える．

5.2.2　クラスとその機能

本節では，TUBSTAP に作成した主要なクラスの概要と，主な機能を以下に示す．そして，それらが思考ルーチンを開発する利用者にとり，どのような位置にあるのか分類した．それを図 5.2 に示す．一般的なボードゲームのゲームＡＩ開発においても応用可能である．

- Map
  - 盤上の地形やユニットなど“状態”を保持しておくためのクラス．チェスや将棋における Board である．機能としては，盤上におけるユニットの状態や，あ

るマスにおける地形やユニットの情報を得る関数，ユニットの集合を得る関数が代表的なものとしてある．

- Unit
  - 敵味方を問わず，一つのユニットを表すクラス．そのユニットの情報（HP，ユニットの種類，ユニットを操作済みか否か）を得るための関数がある．
- Action
  - ゲームＡＩの取った行動を保持するクラス．1 ユニットを行動させた情報を表す．思考ルーチンは，Map のオブジェクトを受け取って Action のオブジェクトを返すように設計する必要がある．
- AiTools
  - AI クラスから利用されるべき，便利な関数をまとめるために用いるクラス．例えば，Map（状態）を入力として，その状態において可能となる合法手を列挙して返す関数等を揃えている．
- Spec
  - 各ユニットの性能性能など表すクラス.Unit はその種類ごとに Spec を持っている．攻撃力や，移動力なども得ることができ，Spec を参照することで可能である．
- Player
  - 思考ルーチンである AI クラスのインテーフェースとなるクラス．関数には Action を戻すためのもの，ＡＩのパラメータを返すためのものがある．AI クラスでは，これらの関数を必ず実装する必要がある．
- PlayerList
  - 開発した思考ルーチンを登録する役割をするクラス．このクラスに思考ルーチン（AI クラス）を登録するだけで，ＧＵＩ上で対戦用や高速自動対戦によって AI の性能評価に使用することが可能である．
- Logger
  - 情報の表示や保存を容易にするためのクラス．必須ではないが，デバッグやＡＩの思考過程を可視化したとき，ログとしてテンポラリにＧＵＩ上に保存できるため，便利な関数である．
- RangeController

- あるユニットの移動可能範囲，攻撃可能範囲，攻撃可能な敵ユニットとその数などを得るための関数を定義している．思考ルーチンを書く際に便利である．また，このクラスの関数で行っている，移動可能範囲や攻撃範囲の計算に工夫を凝らすことにより高速化の余地がある．

- GameManager
  - ゲームの進行を担当するクラス．1 ゲームをとり行ったり，指定回数だけ自動で対戦実験を行う処理も担当する．

- DrawManager
  - 画像ファイルの管理，GUI へのマップの描画などを担うクラス．

- SGFManager
  - 棋譜の出力や読み込み・再生を行うクラス．

- MainForm
  - 主画面のクラス．主にマウス等のイベントを処理する．

図 5.2: 設計したクラスの分類

第6章　提案ルールにおけるゲーム木探索の適用

前章までは，ターン制ストラテジーの統一ルールを提案し，そのルールを基に研究用途のシステム（以下 TUBSTAP と表記する）の構築を行った．本章以降では，提案ルールに対してゲーム木探索を適用しコンピュータプレイヤを実装する．その事前準備として，基盤となる，モンテカルロ木探索とその一手法である UCT 探索について解説する．

6.1　ゲーム木探索適用のアプローチ

囲碁や将棋などでは，1 手を 1 エッジとしたゲーム木として表現する．一方，多くのターン制ストラテジーゲームや TUBSTAP では，1 ターンに複数のユニットを操作できるため，ゲーム木の構成が一意には定まらない．アプローチの仕方には二通り考えられる．一つは 1 ユニットの行動を 1 エッジとする方法で，もう一方が 1 ターンの“全ての自分のユニットの行動”を 1 つのエッジとする方法である．図 6.1 に各方法により，構成されるゲーム木を示す．前者は分岐数が少ない代わりに深い探索が必要で，どちらかというと攻撃的な良い着手の発見に適している [15]．後者は分岐数が膨大となるが，防御的な良い着手の発見に適している．[18]

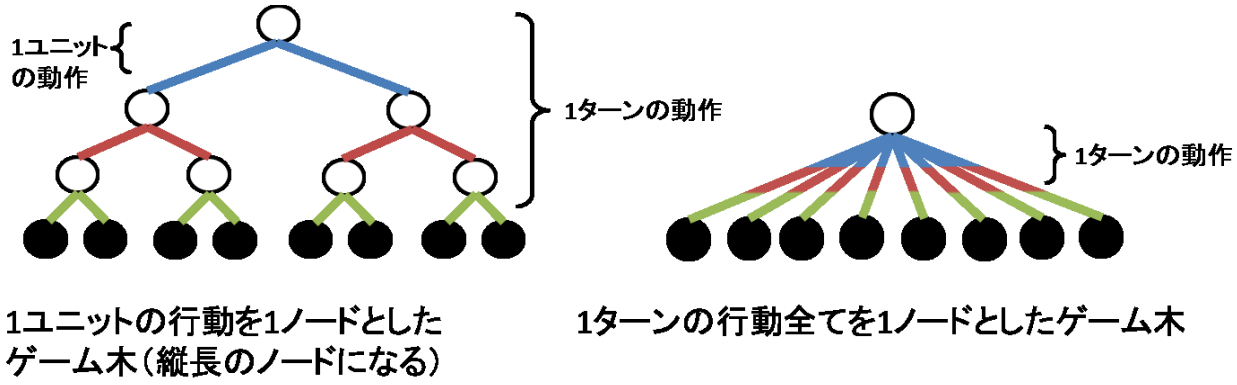

図 6.1: 二通りのゲーム木の構成法

6.2 TUBSTAP へのゲーム木探索適用の困難さ

提案ルールを取り入れたシステムである TUBSTAP では，MinMax 戦略に基づいたゲーム木探索手法の適用が困難である．その原因が局面における分岐因子数である．1 駒辺りの行動数を n，駒数を r とすると，1 ターンの行動の組み合わせは n の r 乗× r!通りにもなる．図 6.2 に示したマップは，マップのサイズが 6 × 6 とこの手のゲームの中では比較的小さいマップである．このような小規模マップにおいても，仮に $n = 10, r = 6$ とすれば，1 ターンの行動の組み合わせは 7 億 2000 通りにも及ぶ．6.1 節で解説したように，駒数の多い局面ではナイーブな実装では，$\alpha\beta$ 探索やモンテカルロ木探索など，既存手法の適用は困難である．なお，本研究では，以降の全ての対戦実験にこの対戦マップを用いることとする．

図 6.2: 6 × 6 のマップの例

6.3　モンテカルロ木探索

モンテカルロ法は候補手に対しランダムに終局までシミュレーションを繰り返し，有望な手を探索する手法である．最も単純なものが原始モンテカルロ法 [19] である．これは現在局面における各合法手に対して均等にシミュレーションを行い，その中で期待報酬が最大となる候補手を選択する手法である．だが，プレイアウト（単にシミュレーションとも呼ぶ）を繰り返しても木が成長しないため，相手がミスすることを期待して手が評価されることがあり，いくらプレイアウトを増やしても最善でない手を選択する可能性が生じてしまう．これに対処するために提案された手法がモンテカルロ木探索 [19] である．

モンテカルロ木探索は，コンピュータ囲碁において成功を収めた手法で様々な改良が成されるなど盛んに研究されている．その理由の一つに評価関数を必要としない点が挙げられる．これにより，知識表現が難しいゲームや関連研究の少ないゲームでも低負荷で AI を開発できる．以下に示すのが，モンテカルロ木探索のアルゴリズムで，図 6.3 はそれを図示したものである．

1. 選択
    - 方策に従って有望か十分調べてないノードの選択を行う．展開されてない末端ノードに至るまで，これを繰り返す．
2. 拡張
    - 末端ノードのプレイアウト回数が閾値を超えた場合，そのノードをもう一段展開する．そして，子ノードを 1 と同様の基準で選択する．
3. プレイアウト（シミュレーション）
    - 選択した末端ノードが展開条件を満たしていない場合，終局までプレイアウトを行う．
4. 更新
    - これまで辿ってきたノードにプレイアウト結果を反映する．

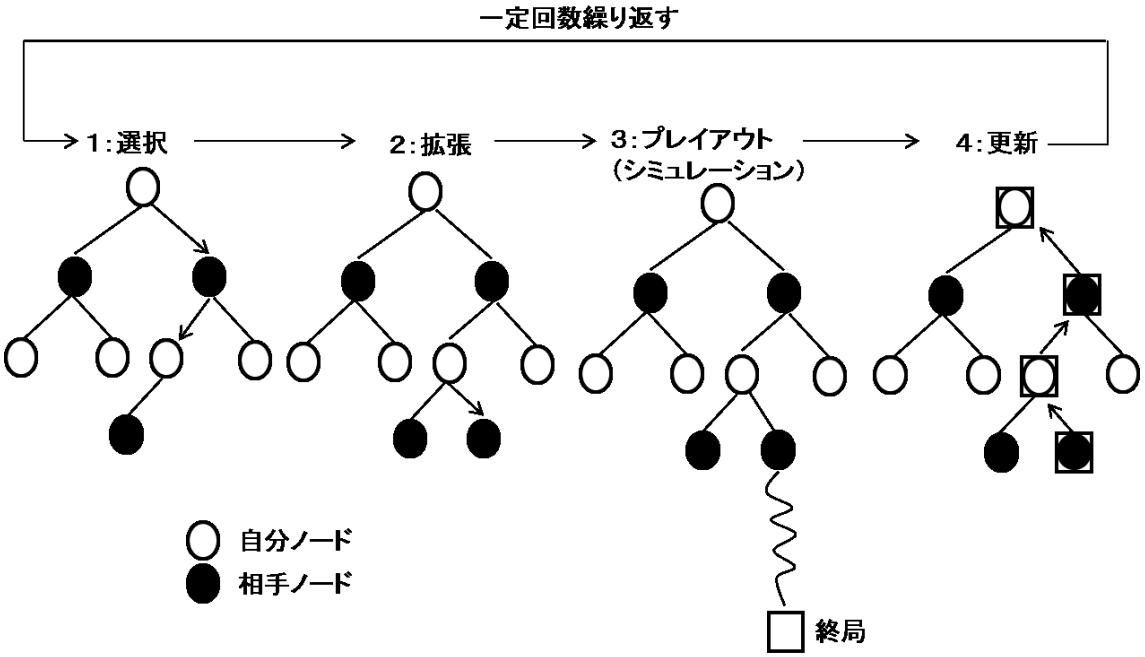

図 6.3: モンテカル木探索の概要図

6.3.1 UCT（UCB applied to Tree）

UCT（Upper Confidence Bound applied to Trees）はコンピュータ囲碁で発表されたモンテカルロ木探索の 1 手法である [7]．UCT では UCB（Upper Confidence Bound）という戦略に基づき，UCB 値 [9] の高いノードを根ノードから辿り末端ノードにまで至れば，そこからプレイアウトを行う．式 6.1 に示すのが，UCB 値の計算式である．

$$UCB = \frac{x_j}{n_{\,\mathrm{j}}} + c\sqrt{\frac{\log n}{n_j}} \tag{6.1}$$

x_j：j 番目の候補手の勝利回数
n：それまでにプレイアウトした回数
n_j：j 番目の候補手の合計プレイアウト回数
c：重みづけ定数

c はアルゴリズムの性格を決めるパラメタであり，これの適切な値は，実験により求める必要があるが本研究では，c=1 と定める．

UCT では，各候補手に均等にプレイアウトを行うのとは異なり，得られた報酬の期待値が高い手や，運悪くプレイアウトが少ない手に優先的にプレイアウトを行う．そのため，有望な手と有望である可能性がある手に満遍なく，プレイアウトを割り振ることができ，単純なモンテカルロ木探索よりも少ないプレイアウト回数で最善手を発見できる．

第7章　UCT探索のTUBSTAPへの適用

本章では，TUBSTAP に UCT 探索の適用を試みる．6.2 節で TUBSTAP では，候補手数が膨大なため，UCT 探索や $\alpha\beta$ 探索を直接適用するのみでは，強いゲームＡＩの開発は困難だと述べた．7.1 節では候補手の抽象化について解説する．続いて，7.2 節以降は，実際に TUBSTAP においてゲーム固有のヒューリスティックを用いて，考えうる候補手の抽象化法を提案する．

7.1　候補手の抽象化とは

本研究では，候補手の抽象化とは候補手をグループ化することであると定義する．図 7.1 は候補手をグループ化した抽象化の概念図である．同一グループに抽象化する候補手の選定には様々な方法が考えられるが，本研究では類似した候補手を一つのグループへと抽象化する．また，抽象化の利点は分岐因子数を減らせる点であり，UCT 探索を適用し，同一の投入計算資源で探索を行った場合，少ない候補手に対して重点的にプレイアウトを行うことが可能である．これにより，抽象化を行わない普通の UCT 探索に比べ，限られた候補手に対して，より多くの情報が得られる．

次に，UCT 探索に抽象化を適用し探索を行う方法には，加藤ら [15] によれば，二通りの方法がある．一つは同一グループの中から代表的な手を選出し UCT の一つのノードにする方法である．これは，直接的には枝刈りと同じ作業であるが，ProgressiveWidening のように探索の進展に伴い抽象化の度合いを落とし，今までの探索結果を似たノードで分配するなどの工夫が容易である．

もう一方が，UCT を辿る際に同じグループからランダムに候補手を選択する手法である．それぞれには利点と欠点がある．前者は実装が簡単である反面，代表手の選び方にアルゴリズムの性能が依存してしまう可能性が生じる欠点がある．一方，後者の手法はプレイアウトの際にグループの中から，ランダムに候補手を選択するため，どのグループが上手く行くのかどうかといった指針が得られる．だが，あるノードを選択した場合の合法手が別のノードを選択した場合には合法手にならないかもしれないといった課題もあり実現には困難が伴う．本研究では，前者のアプローチを採用することにした．この手法により構成されるゲーム木が図 7.2 である．

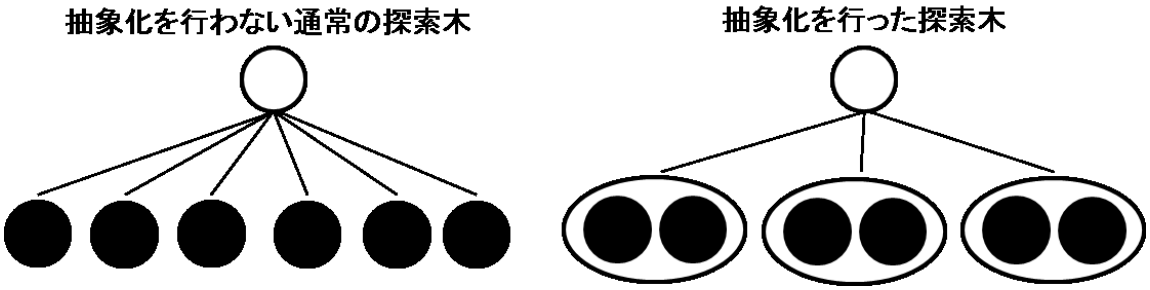

図 7.1: 抽象化により構成されるゲーム木

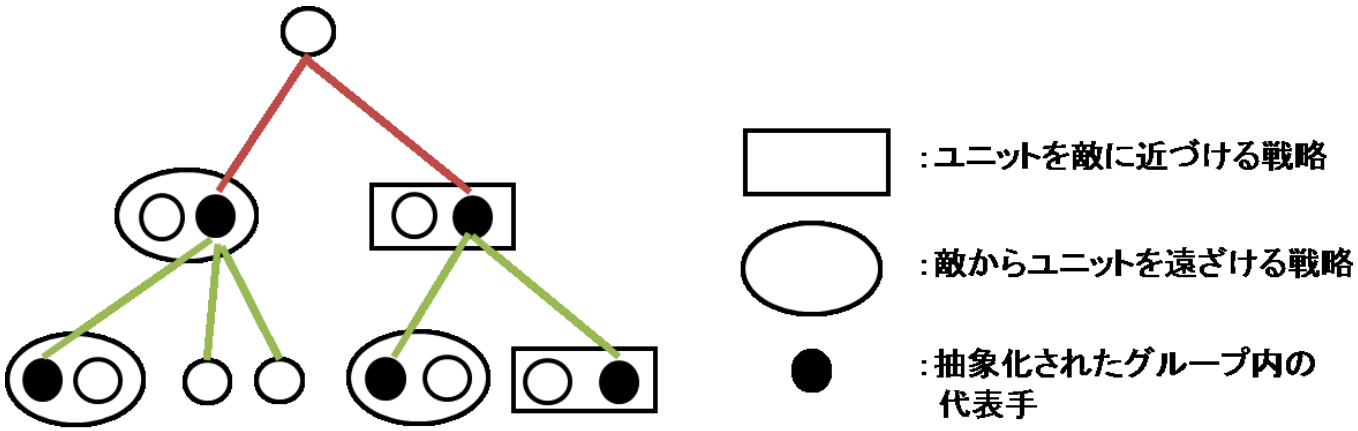

図 7.2: 候補手のグループから代表的な手を選択し，一つのノードにする

7.2 TUBSTAP における候補手の抽象化の考察

本節では実際に TUBSTAP を用いて候補手の抽象化を適用するにあたり，その対象とする候補手について解説する．1 つのユニットを操作する行動には，敵ユニットを攻撃する行動（攻撃行動）と移動させる行動（移動行動）がある．攻撃行動においては，選択によっては，その後の戦況に大きな違いが生じる．そのため，攻撃と移動それぞれに対し，抽象化の方法を分けることとする．

具体的には，攻撃に関しては 4 通り，移動に関しては 3 通りの抽象化法が考えられ，その組み合わせと各抽象化の度合いにより，Level1〜Level4 の抽象化法を提案する．また，本論文では抽象化したグループ内の候補手数の量を表す指標を“ 抽象度 ”と定義する．表 7.1 は各 Level における，攻撃行動と移動行動の抽象化法の概要をまとめたものである．Level0 は全く抽象化を行わないことを意味する．

7.3 節で移動行動の各抽象化の方法を説明し，7.4 節で 4 通りの攻撃行動の抽象化法を解説する．なお，攻撃行動に関しては，異なる意味を持つ行動が多いため，抽象化を行わない方が良い可能性があるが，これにより局面分岐数をさらに絞り込むことで，限定された候補手を重点的に探索できれば有望な結果が得られる可能性もあるため，抽象化を行うこととする．

また，これらの候補手の分類方法は，対象ゲームである TUBSTAP のみならず，その他の類似したターン制ストラテジーゲームでも応用可能であると考える．

表 7.1: 抽象度に基づいた，4 通りの抽象化法．

抽象度	攻撃行動	移動行動
Level 0	全て探索対象	全て探索対象
Level 1	全て探索対象	1 ユニットにつき 2 行動へと強く抽象化する．図 7.4
Level 2	攻撃場所抽象化．図 7.8	Level1 と同様
Level 3	攻撃対象ユニットの種類で抽象化． 例えば図 7.9 における歩兵の攻撃行動は 2 手に抽象化．（本来は 7 手ある）	Level1 と同様
Level 4	1 ユニットにつき 1 行動に大きく抽象化する． 図 7.9	攻撃と同様 1 ユニットにつき 1 行動．図 7.5

7.3　移動行動の抽象化法

移動行動の抽象化につき，3 通りの方法を解説する．まず一つ目が図 7.3 に示した，全く抽象化を行わない（つまりユニットの移動行動を全て探索対象にする）方法である．味方ユニット（赤）の移動行動は，合計で 9 行動存在する．二つ目は，図 7.5 のように全移動行動を，合計 9 行動から 1 行動へと大幅に抽象化する方法である．そして，三つ目の方法は，自分のユニット（赤色）の移動場所が，敵（青色）の攻撃範囲に入るか否かによって図 7.4 のように 2 つの行動へと抽象化する方法である．なお，移動の抽象化法については様々な方法が考えられるが，本研究で用いる対戦用マップ（図 6.2）が小規模であることから，この 2 通りの抽象化を行うことにした．

図 7.3: Level0 の抽象化．味方歩兵（赤色）の移動行動を全て探索対象とする．つまり移動は抽象化しない．

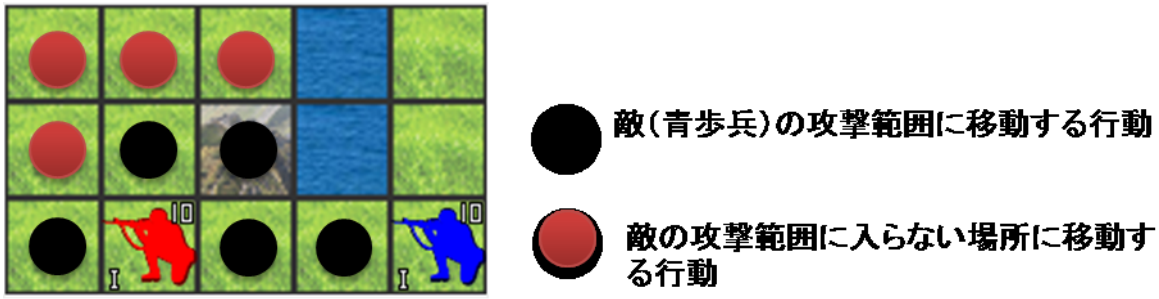

図 7.4: Level1〜3 の抽象化．味方歩兵（赤）の移動行動を 2 通りに抽象化する．敵の攻撃範囲に入るか否かで分類を行う．

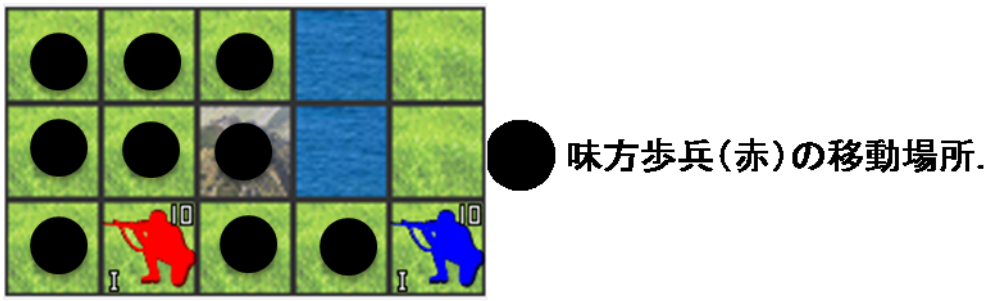

図 7.5: Level4 の抽象化．味方歩兵（赤）の全ての移動行動を，1 行動のみに抽象化する．

7.4　攻撃行動の抽象化法

本節では，4 通りの攻撃行動の抽象化法を 7.2 節と同様に図を用いて順に解説する．まず，図 7.6 は攻撃行動の抽象化を行わない Level0 の抽象化である．味方の歩兵（赤）から見て，7 つの攻撃行動が存在する．次に，図 7.7 は，敵ユニットを隣接攻撃するさいに，移動場所の抽象化の抽象化を示したものである．これにより，7 つあった行動を 3 つ（敵のユニットの数）までに減らすことが出来る．

三つめの Level3 の攻撃行動の抽象化では，敵ユニットの種類ごとに抽象化する．図 7.8 の例では，2 種類の敵ユニットが存在するため，7 つの行動を 2 つにまで抽象化することになる．最後に，Level4 の抽象化では図 7.9 のようにユニットの全攻撃行動を 1 つに抽象化する．

図 7.6: Level0・Level1 の攻撃行動の抽象化．味方歩兵（赤）の全攻撃行動を探索対象とする．

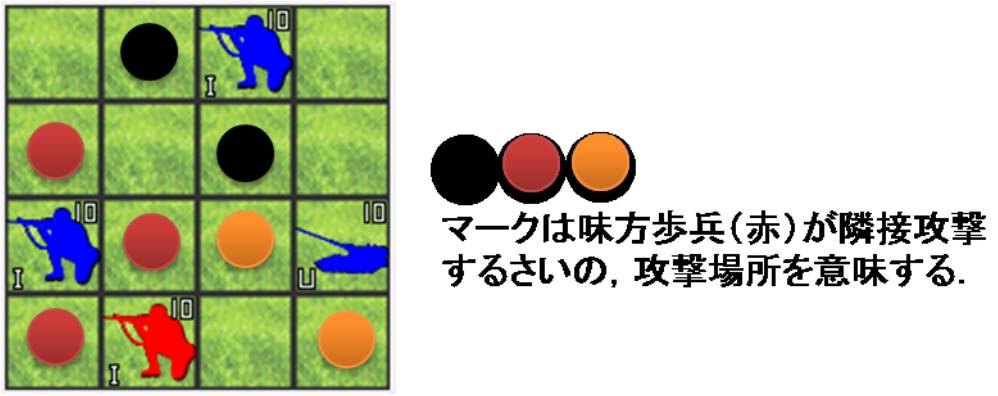

図 7.7: Level2 の攻撃行動の抽象化．味方歩兵（赤）の攻撃行動を，隣接攻撃時の場所ごとに抽象化する．

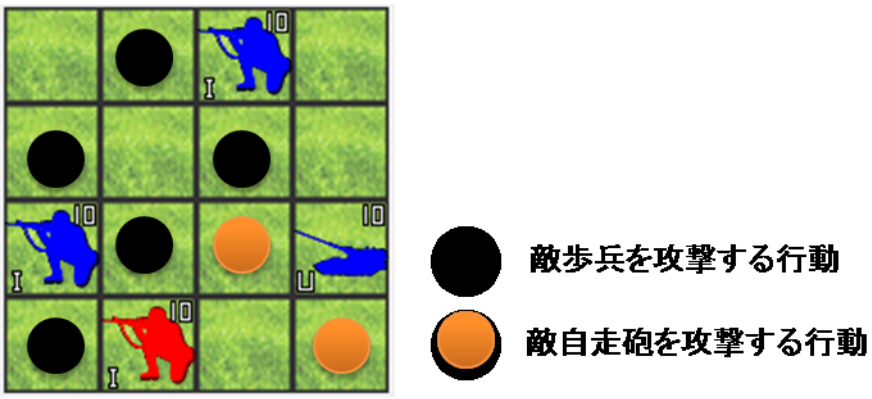

図 7.8: Level3 の攻撃行動の抽象化．味方歩兵（赤）の攻撃行動を，攻撃対象となる敵ユニットの種類により抽象化する．

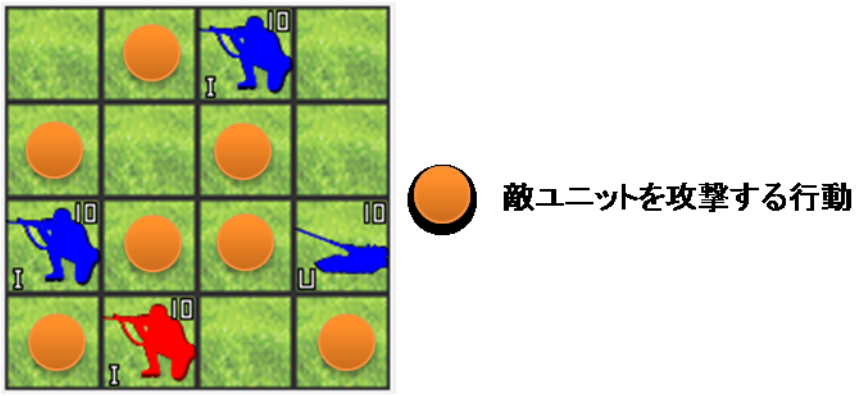

図 7.9: Level4 の攻撃行動の抽象化．味方歩兵（赤）の全攻撃行動を，1 行動だけに抽象化する．

第8章　抽象化の評価実験

本章では7.3節で提案した4通りの抽象化の方法に基づき，それらの性能評価実験を行うことで，抽象化が有望であるのか，実験から得られた結果が妥当であるのかの検証を目的とする．

8.1　有望な抽象度を探る実験

4通りの各抽象化法の性能を知るため，抽象化を行わないナイーブな実装の標準的なUCT探索（以下，標準UCT）を評価のために用いる．実験のパラメタは以下の通りである．

- 実験対象マップは図6.2に示したマップである．
- 合計対戦回数は1000回（先手，後手，各500戦ずつ，行う．また，引き分けは0.5勝利として扱う.）
- 1手あたりのプレイアウト回数は各（500，1000，2000，3000，4000）の5通りのプレイアウト回数で，抽象化UCT探索と標準UCTを同一のプレイアウト回数に設定し，実験を行う．
- LEVEL1〜4の抽象化のパラメタは，表7.1で解説した抽象度を意味する．
- UCTのシミュレーション中の方策には，ユニットの攻撃行動が存在するならば，必ず攻撃行動を選択するヒューリスティックを導入した．これは，囲碁やオセロとは違い，完全にランダムなプレイアウトを行った場合，プレイアウトが収束する保証が無いため，無限ループに陥ってしまうことへの対処である．このような収束のためのテクニックは，将棋で敵の駒を取る手を優先することに似ている．

表8.1に示した対戦実験の結果から，いずれのプレイアウト回数を見ても，Level1〜Level3の抽象化法が，標準UCTに有意に勝ち越していることが分かり，抽象化が上手く行ったと言える．対象ゲームでは，敵ユニットを攻撃する行動が重要となる性質があるため，移動行動を抽象化し，攻撃行動に探索が重点化されるLevel1〜3の抽象化法が有意な結果を出したのだと考えられる．さらに，Level1の抽象度が標準UCTに対する勝率が最大であることから，攻撃行動は抽象化を行わないのが良いと分かる．

また，図8.1のグラフを見ると，いずれの抽象化法でも一定のプレイアウト回数で抽象化UCTの勝率の上限が訪れているのが見て取れる．これは，プレイアウト回数が十分で

ない場合，標準 UCT では候補手に対して十分なプレイアウトを割り振ることが出来ていないのが一因だと思われる．

表 8.1: 抽象度ごとの標準 UCT に対する勝率．

プレイアウト回数	Level1	Level2	Level3	Level4
PO500	57.3（560-25-415）	51.7（505-24-471）	51.6（498-35-467）	41.9（391-55-554）
PO1000	58.8（581-14-405）	54.0（528-23-449）	50.9（497-23-480）	35.4（338-32-630）
PO2000	63.1（620-22-358）	60.0（589-21-390）	55.4（540-27-433）	32.7（321-11-667）
PO3000	61.7（609-15-376）	58.8（576-24-400）	55.5（543-24-433）	29.5（287-15-698）
PO4000	61.2（600-24-376）	57.4（566-16-418）	54.4（535-18-447）	25.1（244-13-743）

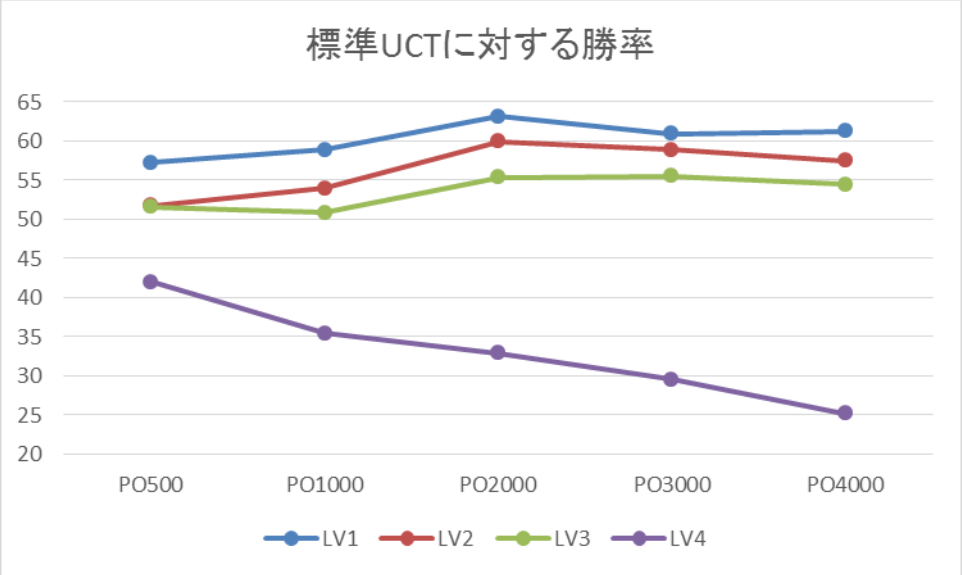

図 8.1: 各プレイアウト回数ごとの，標準 UCT 探索に対する勝率の推移

8.2　行動可能ユニット数に応じて抽象度を変更する実験

8.1 節の実験では，抽象化が有望化どうかを検証するための実験を行った．その結果，LEVEL1 の“ユニットの移動行動を抽象化して，攻撃行動は全て探索対象にする”手法が有効なのが分かった．また一方で，プレイアウト回数が高回数な実験条件では，候補手を削減した欠点が生じたためか，標準 UCT の勝率が上昇する結果となった．

TUBSTAP の候補手を多くしている要因は，“全てのユニットを自由な順に動かせる”点である．そのため，操作できる自分のユニット数が多い局面ほど，これに比例して候補手の数は増加する．そこで，“行動可能ユニットが多い局面では強い抽象化を行い，少ない局面では抽象度を弱くする”手法が有効に働く可能性があると考えた．対戦実験に用いるマップによって，この手法による実験結果は変わりうるが，本節では以下この仮説の検証を行っていく．

8.2.1　行動可能ユニットが少ない局面での調整

本節では，行動可能なユニットが少ない局面で抽象化を行わないことにより，性能にどう影響を与えるのか実験し考察する．対戦実験では，行動可能ユニットが 3〜5 以上のときのみ，抽象化しないように実験条件を設定した．図 8.2 は，行動可能なユニットが 3 以上の場合にのみ，抽象化を行うゲーム木である．

次に，詳細な実験条件を 8.1 節と同様に，プレイアウト回数を 1 手あたり 1000 回，基になる抽象度を Level1 に設定して対戦実験を行った．この結果が表 8.2 の通りである．これにより，調整を行わない Level1 の抽象化 UCT が勝率 58.8%であるのに対し，ユニット数 3 以上のときにのみ抽象化を行うパラメタが勝率 63.2%と，有意な差が見られた．

この手法が上手くいった理由として，ユニット数が少ない局面では全ての候補手を探索対象にしているため，“歩兵が味方のユニットを敵の攻撃から守る位置に移動する”といった戦略が発見できたのが，一因として考えられる．

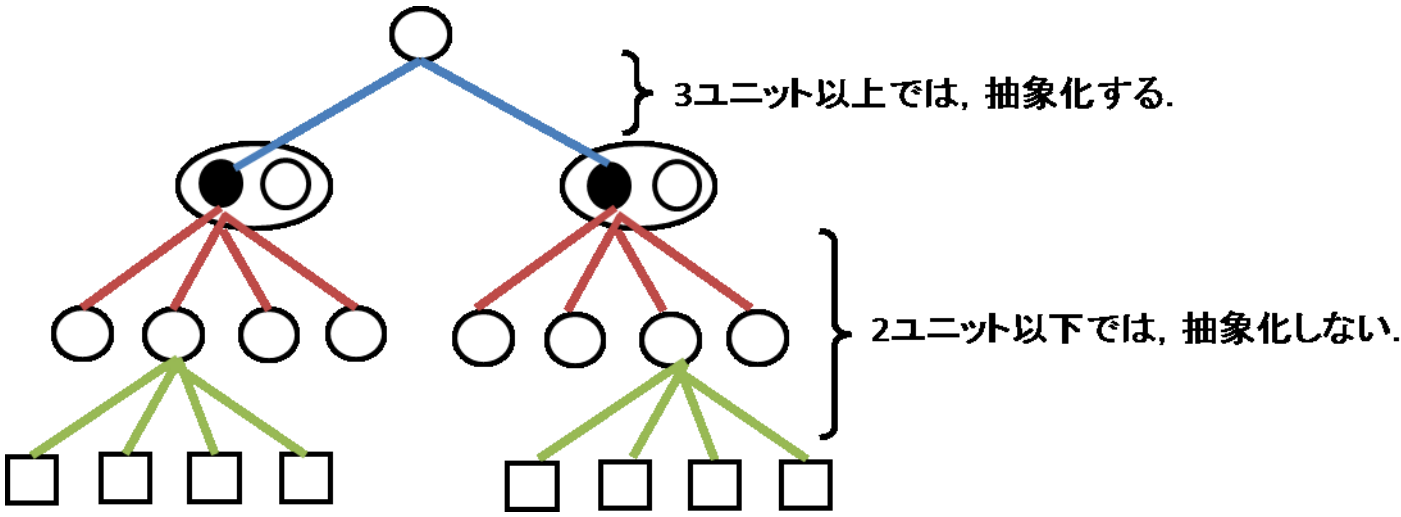

図 8.2: 行動可能ユニット数が少ない局面では，抽象化を行わない．

表 8.2: 行動可能ユニットが少ない場合，抽象化を行わない対戦結果．プレイアウト回数は 1000 回．

抽象化を行うユニット数の上限	標準 UCT 探索に対する勝率
調整を行わない（抽象度 Level1）	58.8（581-14-405）
3 ユニット以上	63.2（626-11-363）
4 ユニット以上	62.7（620-13-367）
5 ユニット以上	55.0（543-14-443）

8.2.2 行動可能ユニットが多い局面での調整

8.2.1 節の実験結果より，行動可能ユニット数が少ない局面では，抽象化を行わない手法が有効であることが分かった．このように行動可能ユニット数に応じて，段階的に抽象度を切り替えることは，人間も行っていると思われる．だとすれば，ユニットが少ない局面だけでなく，それが多い局面では抽象度を上げて，行動数をさらに絞りこむ必要があるかもしれない．そこで本節では，“行動可能ユニットが多い局面における抽象度の増加”を試みる．

実験には 7.3 節で提案した，Level1〜Level4 の抽象化法を用いる．行動可能ユニット数が 3 または 4 以上の局面で抽象化する 8.2.1 節の手法を基に，抽象度の調整を行う．具体的には，行動可能ユニット数（このマップでは 6 から始まり最後は 1）ごとにどのレベルの抽象化を行うか（最大で Level4, 最小で Level0 つまり抽象化を行わない）をパラメータとして持つことにする．本論文ではこの 6 つのパラメータを 441000 のような格好で表記するが，これは (Level4, Level4, Level1, Level0, Level0, Level0) を省略した表記であり，“残りユニット数が 6 か 5 の場合は強い抽象化，4 の場合は弱い抽象化，3,2,1 の場合は抽象化を行わない”ことを意味する．

8.2.1 節の実験と同様にプレイアウト回数を 1000 回として，対戦実験を行った．行動可能ユニット数が 3 以上と 4 以上で抽象化を行った実験結果を，それぞれ表 8.3 と表 8.4 に示す．8.2.1 節で用いたパラメタは“111100”で標準 UCT に対する勝率が 63.2%であるのに対し，抽象度を増やしたパラメタ（“442100”など）の他の実験では，特に際立った性能の向上は見られなかった．だが，中には“441000”の勝率が 67.3%と，僅かながら勝ち越しているパラメタも見受けられ，このような行動可能ユニット数に応じて抽象化の度合いを差別化する手法がうまく行ったと言える．

表 8.3: ユニット数 3 以上で，抽象度を変更した実験．

実験パラメタ	標準 UCT 探索に対する勝率
111100	63.2（626-11-363）
331100	58.3（574-17-409）
443200	63.5（619-32-349）
442100	65.1（642-17-341）
443100	66.2（648-27-325）
432100	62.7（611-32-377）
333100	59.4（585-17-398）
433100	64.3（631-24-345）
432100	62.7（611-32-377）

表 8.4: ユニット数 4 以上で，抽象度を変更する実験.

実験パラメタ	標準 UCT 探索に対する勝率
111000	62.7（620-13-367）
211000	55.1（510-81-407）
311000	54.8（544-8-448）
321000	56.5（561-8-431）
331000	54.8（541-13-446）
332000	58.9（576-26-398）
333000	56.1（549-24-427）
431000	61.5（609-11-380）
432000	63.3（625-15-360）
433000	62.3（611-23-366）
441000	67.3（635-75-290）
442000	63.1（621-20-359）
443000	64.6（635-22-343）
444000	61.2（599-25-376）

第9章　まとめと今後の課題

9.1　まとめ

本研究では，ターン制ストラテジーゲームの研究用統一ルールやプラットフォームが不在であること，およびコンピュータプレイヤの強さが十分な域に達していないこと，この二つの課題の解決を目的とした．まず初めに，研究用に適したルールを提案するため，既存の戦略ゲームからそれらを構成する特徴要素を抽出しクラスタ分析を行うことで，ルールの分類を試みた．そして，チェスや将棋の次に取り組むべきベンチマークとして妥当な位置づけにある，ルールを提案した．次に，研究用のプラットフォームの構築を行った．構築したシステムには，ゲームＡＩの研究用途して必要と考えられる要求項目を取り入れた．また，構築したプラットフォーム（TUBSTAP）は，JAIST 池田研究室の Web ページから公開しており，国内のいくつかの大学や高等専門学校で，研究用に利用されている．

また，実際にプラットフォームを用いてコンピュータプレイヤを構築も行った．アプローチには，モンテカルロ木探索の一手法である UCT 探索を適用した．さらに，ゲーム固有の知識を用いて類似した候補手の抽象化を行った．抽象化にあたり，4 通りの抽象化法を提案し，有望な抽象化の程度を知るための実験を行った．その結果，ユニットを移動させる行動のみ抽象化する手法が，抽象化を一切行わない UCT 探索に対して，最も高い勝率を得られることが分かった．また，操作可能な自分のユニットの数に応じて，抽象化の程度を適切に調整することで，若干の性能向上が見られた．

9.2　今後の展望

本研究では UCT 探索を TUBSTAP に適用し抽象化を行ってきたが，同一グループの中から代表的な手を選んで探索対象として固定する手法を，実験および検証するに留まった．今後の展望としては，プレイアウトの結果に応じて，Progressive Widening のように有望なグループから，動的に探索対象となる候補手を増加させていくなどの改良や，プレイアウト中に同一グループから探索対象にする候補手をランダムに選択する手法などの検証を行って行きたい．また，プラットフォームへの機能拡張に関しては，今後は，ネットワーク対戦機能の追加や，ユーザーインターフェースの改良，対戦用サーバーの構築を行って行くことが，TUBSTAP を広め，より多くの人に利用されるために必要であると考える．ゲームのルールの方も，提案ルールにおいて，十分に強いコンピュータプレイヤが構築できれば，生産や占領といったゲームをより複雑にしていくための要素を取りいれて行きたい．

謝辞

まず初めに，本研究を始めるにあたりご指導いただきました主指導教員の池田心准教授，副指導教員の飯田弘之教授に深く感謝致します．本戦略ゲームプロジェクトを共に進めてきた，池田研究室の藤木翼君にも感謝しております．また，TUBSTAP を利用して下さった，飯田研究室の石飛太一さんを初め，他大学の方々から多くの意見を頂き，大変感謝しております．他にも，池田研究室・飯田研究室の皆様にも様々なご協力を頂き，感謝いたします．最後に，私のようなもののために，大学・大学院と 7 年間も学生でいることを許容し，援助して下さった両親に深い感謝の意を表します．

参考文献

[1] Arimaa, http://arimaa.com/arimaa/.

[2] Bonanza - The Computer Shogi Program, http://www.geocities.jp/bonanzashogi/.

[3] Enzenberger,M.,Muller,M.,Arneson, B. and Segal, R Fuego-An Open-Source Framework for Board Games and Go Engine Based on Monte Carlo Tree Search, IEEE Trans. Computational Intelligence and AI in Games, Vol.2,No.4,pp.259-270 2010.

[4] Feiyu Lu, Kaito Yamamoto, Luis H. Nomura, Syunsuke Mizuno, YoungMin Lee, and Ruck Thawonmas, “Fighting Game Artificial Intelligence Competition Platform” Proc. of the 2013 IEEE 2nd Global Conference on Consumer Electronics, pp. 320-323, 2013.

[5] FINAL FANTASY TACTICS, http://dlgames.square-enix.com/fft/.

[6] IEEE-CIG , http://cig2015.nctu.edu.tw/.

[7] Kocsis,L and Szepesvari, C., Bandit Based Monte-Carlo Planning, 17th European Conference on Machine Learning, ECML 2006, LNCS, Vol4212, pp.282-293, 2006.

[8] Mario AI BenchMark, https://code.google.com/p/marioai/.

[9] P.Auer,N.Gesa-Bianchi,P.Fischer, Finite-time analysis of the multiarmed bandit problem, Machine Learning , Vol. 47, pp. 235-256, 2002.

[10] StarCraft2-homepage, http://us.battle.net/sc2/en/.

[11] The2K BotPrize : Home, http://botprize.org/.

[12] Tomas Kozelek, Method of MCTS and the game Arimaa, Master’s Thesis,2009.

[13] TUBSTAP-プロジェクトページ, http://www.jaist.ac.jp/is/labs/ikeda-lab/tbs/.

[14] カタンの開拓者, http://www.catan.jp/.

[15] 加藤千裕, 三輪誠, 鶴岡義雅, 近山隆, ターン制ストラテジーゲームにおける戦術決定のための UCT 探索とその効率化, 第 18 回ゲームプログラミングワークショップ 2013.

[16] 大戦略 -wikipedia, http://ja.wikipedia.org/wiki/大戦略シリーズ.

[17] 富沢大介, 池田心, 落下型パズルゲームの定石形配置法とぷよぷよへの適用, 情報処理学会論文誌,Vol.53, No.11, pp. 2560-2570, 2012-11.

[18] 藤木翼, 村山公志朗, ターン制ストラテジーのための状態評価型深さ限定モンテカルロ法, 第 8 回 Ｅ＆Ｃシンポジウム 2014.

[19] 美添一樹, 山下宏, 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,2011.