平成 22 年度 電気通信大学情報工学科 卒業論文

モンテカルロ碁におけるプレイアウトのための機械学習

指導教員　村松 正和 教授

平成 23 年 1 月 31 日

電気通信大学情報工学科

0711121
下川 和也

目 次

1　はじめに

1.1　研究背景

本研究の対象となるコンピュータ囲碁も含め思考型ゲームのプログラムは，人工知能研究の題材として研究されている [3]．人工知能研究の目的は「人間の知的活動をコンピュータ（プログラム）で実現する」ことにある．そのため，適度に難しくかつ目標の定義や結果の評価をしやすいゲームプログラミングは題材として適しているのである．

ゲームプログラミングの研究は様々なゲームにおいて行われているが，もっとも盛んに行われていたチェスは，20 世紀末にコンピュータが人間の世界チャンピオンに勝利するまでに至った．また，持駒の再利用ルールがあることから，チェスよりも複雑だとされている将棋でさえ，現在最強のプログラムはアマチュア高段者レベルにあり，人間の最強プレイヤに勝利するのは時間の問題だと考えられている．

一方，コンピュータ囲碁の研究は，現在トップクラスのプログラムでもアマチュア初段程度とされている．そのため，囲碁が次の目標となっている．囲碁は，チェスや将棋と比べてコンピュータにとってさらに複雑で難しい．要因としては，非常に探索空間が広いため十分に有効な先読みを行うことができないということと，局面の有利不利を評価する因子がはっきりしないので，評価関数[1]の作成が困難であるということが挙げられる．そのため，これまでにチェスなどで成功した手法の延長だけでは不十分であり，これまでとは異なる更なる工夫が必要である．

従来，囲碁プログラムは知識ベースによるアルゴリズムが主流であり，囲碁プログラムの開発者は多大な労力をかけ，それぞれが工夫を凝らしてより良い評価関数を得るために研究を続けてきた．しかし，数十年の研究にもかかわらずアマチュア級位者の実力を脱することがなく，人間の初段と互先[2]で戦って勝つのはほぼ不可能だと考えられていた．それでも，この手法の強さにはある程度定評があった．しかし，近年は技術的に停滞し始めていた．

そのような状況の中で，モンテカルロ碁が革命的な変化をもたらした [8]．モンテカルロ碁は，2006 年の Computer Olympiad の 9 路盤においてフランスの CrazyStone が優勝したことで一躍脚光を浴びることとなった．モンテカルロ碁は，モンテカルロ法に基づいた手法で，ランダムな着手で終局まで対局をシミュレーションしてその中で最も勝率の高い着手を選ぶというアルゴリズムである．知識ベースによるアルゴリズムと異なり，評価関数が不要で，囲碁の高度な知識がいらないことから，開発に多大な労力を必要としない非常に画期的なアルゴリズムだった．また，プログラムの棋力[3]はシミュレーションの試行回数により向上するため，並列化が容易であることやマシンパワーを十分に活用できることが注目される要因となった．現在，囲碁プログラムはモンテカルロ碁が主流となり，多くの研究者によって様々な研究が行われ，改善が進められている．

[1] 局面の状態を数値に変換する関数
[2] ハンディキャップのない対局
[3] 囲碁などの強さを指し示すもの

1.2 研究概要

本研究は，モンテカルロ碁におけるプレイアウトの「精度」の向上を目的とする．ここで「精度」とは，プレイアウトが実際の対局にどれだけ近いか，つまり，プレイアウト中に打たれる着手がもっともらしい手かどうかである．

コンピュータ囲碁の研究は，プログラムの棋力の向上が基本的な目的となるが，現在主流となっているモンテカルロ碁においては，UCT アルゴリズムの改善がその対象となる．UCT アルゴリズムは，木探索部分とプレイアウト部分で構成されるが，従来の研究により，プレイアウトの「精度」を向上させることでプログラムの棋力に良い影響を与えることが知られている．これを実現するためには，囲碁の知識を導入することが必要である．その手法としては，強豪囲碁プログラム Crazy Stone の作者 Rémi Coulom が，Elo レーティングの概念と Minorization-Maximization アルゴリズムを用いて，棋譜から着手パターンの教師あり学習を行うという手法を提案しており，その学習した結果を用いてプレイアウトを改良することでプログラムの棋力の向上に成功している [1].

一方，2010 年の Computer Olympiad の 19 路盤において優勝した Erica は，プログラムで用いる囲碁の知識として，大きなひし形のパターンを採用していることが知られている．このような大きな形のパターンの導入は，今後プログラムの棋力向上のためには重要な要素の一つとなると思われる．

本研究は，Rémi Coulom の手法を用いて，3x3 パターンとひし形パターンの 2 種類の形のパターンを複数の戦術的な特徴と共に機械学習し，その結果を用いて改良を加えたプレイアウトを自作の囲碁プログラムに実装し，その有効性を検証した．

2　囲碁

囲碁とは，二人で行うボードゲームで，交互に盤上に石を置いていき，自分の石で囲んだ領域の広さを争うというものである．

本節では，本研究に必要な用語 [7] とルールを簡単に説明する．

2.1　用語

- 碁盤
  用具の一つで石を置く板のこと．19 路盤や 9 路盤などがある．
- 対局
  囲碁で対戦すること．
- 着手
  石を盤上に置くこと．囲碁では「打つ」という．
- 局面
  盤上の状態のこと．
- 終局
  対局が終了すること．
- コミ
  黒が白に与えるハンディキャップのこと．
- 空点
  石が置かれていない点．
- 地
  どちらかの石にのみ囲まれた場所のこと．陣地．
- アゲハマ
  取り上げた相手の石のこと．
- 眼
  同色の石で区切られた地の一団のこと．
- コウ
  交互に相手の石一個を取り返しできる形．
- アタリ
  石をあと一手で取ることができる状態，もしくはそういう状態にすること．

- シチョウ
  石を階段状にアタリ，アタリと追いかけて取る方法.

- トリ
  石を取る手のこと.

- ノビ
  自分の石をつなげて伸ばす手のこと.

- ノゾキ
  次に打つと相手の石を分断できる手のこと.

2.2 ルール

- 自分の石と相手の石を一手ずつ交互に打つ
  先手が黒，後手が白をもち，図 1 のように一手ずつ交互に石を打つ．石は盤上の線の交点に打ち，一度打った石は動かすことができない.

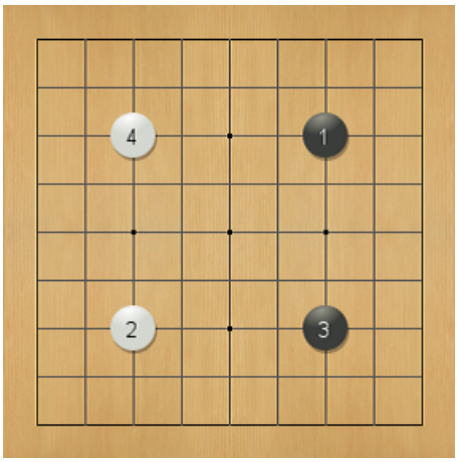

図 1: 着手の例

- 相手の石を囲めば取れる
  相手の石の四方を囲むと，囲まれた相手の石は盤上から取り除かれる.

- 地の多いほうが勝ち
  両者がパスをしたら終局となり，互いの地を数え多いほうが勝ちとなる．地の数え方には，日本ルール[4]と中国ルール[5]がある.

[4]自分の石に囲まれた空点の数－相手のアゲハマの数
[5]自分の石に囲まれた空点の数＋盤上の自分の石の数

- 着手禁止
  相手の石によって四方が囲まれているところには打てない．ただし，相手の石が取れるときは打つことができる．
  例を図 2，3 に示す．図 2 において，白は×に打つことはできない．しかし，図 3 で白が×に打つと，△の黒石を取ることができるので，打つことができる．

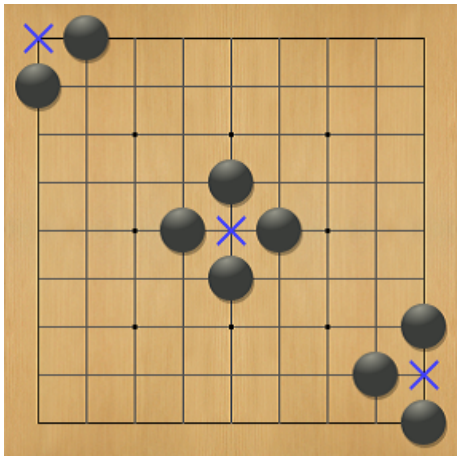

図 2: 着手禁止点

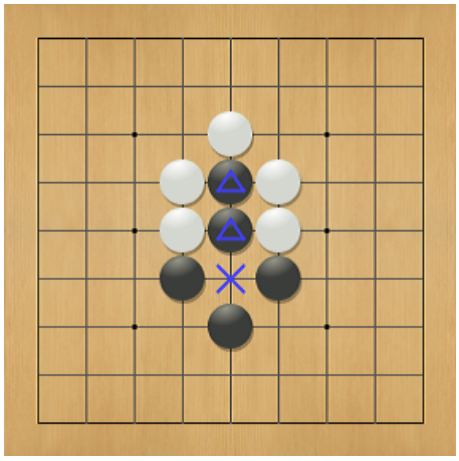

図 3: 例外

- 同形反復禁止
  コウを取られた方は次の着手でそのコウを取り返すことはできない．
  例を図 4，5 に示す．図 4 において，白が×に打って黒石△を取ると図 5 の形に遷移する．図 5 で，同様に黒が○に打つと再び図 4 の同一局面が現れる．この繰り返しを防ぐため，黒がすぐに○に打つことは禁止されている．

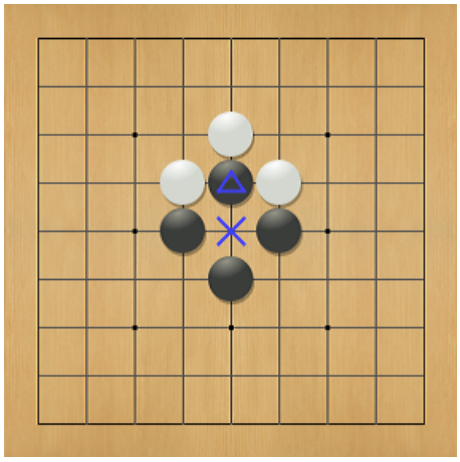

図 4: 白番

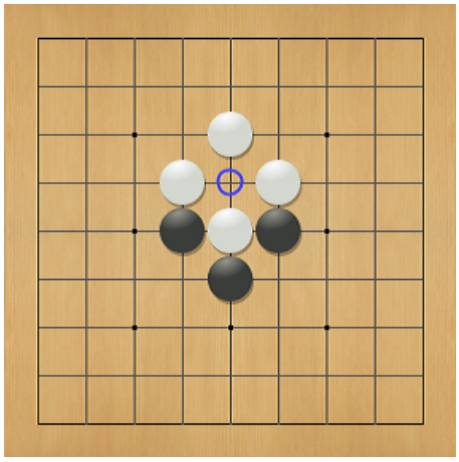

図 5: 黒番．コウの発生

3　コンピュータ囲碁

3.1　用語

本研究に必要な用語 [3] を説明する．

- 連
  縦横につながった同じ色の石の集合．

  「連」という概念はコンピュータ囲碁の研究の中から生まれたもので，石の繋がりを単位とするものである．モンテカルロ碁において必要な高速な計算を可能とするデータ構造である．例を図 6 に示す．×，○，□，△はそれぞれ別の連である．

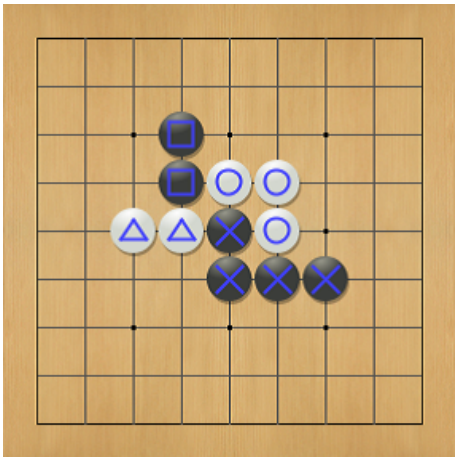

図 6: 連の例

- プレイアウト
  現在の局面からランダムに着手を行い，終局まで打つこと．

3.2　従来研究

3.2.1　モンテカル木探索

モンテカルロ木探索は，モンテカルロ法に基づいた手法で，ゲームのプレイアウトによって現在の局面の評価を行う．プレイアウトでは現在の局面からゲームの終局までランダムに合法手を選ぶことで局面を展開する．そして現在の局面におけるそれぞれの合法手に対して，プレイアウトによる勝敗を与え，このプレイアウトを繰り返すことで，現在の局面からそれぞれの合法手を選択した場合の勝率を得る．この勝率を合法手の評価値として用いる．

モンテカルロ木探索は，評価関数を用いず局面に評価値を与えることができるため，精度の高い評価関数の作成が困難なゲームであっても有効な手法とされている．

3.2.2 多腕バンディット問題

多腕バンディット問題とは，多くの腕を持つスロットマシンをプレイするギャンブラーに基づく機械学習問題である．

ギャンブラーがスロットマシンをプレイすると，各腕はその固有のある確率分布から定まる「報酬」を提供する．ギャンブラーはスロットマシンを繰り返しプレイし，その報酬の合計を最大化することを目的とする．ギャンブラーは最初は腕に関する知識を持たず，プレイを繰り返すことで報酬を高く得られる腕を見つけ出し，集中的にプレイすることができるようになるとされている．

多腕バンディット問題は，既に得られている知識に基づいて報酬を最大化することと，他の腕をプレイし，その腕に関する知識を増やすこととのバランスを取る問題であり，強化学習では「収穫と探検のジレンマ」として知られている．

モンテカルロ木探索では，それぞれの合法手を多腕バンディット問題におけるそれぞれの腕とし，プレイアウトの勝敗の結果が得られる報酬となる．モンテカルロ木探索は，多腕バンディット問題と類似している点が多いため，多腕バンディット問題のアルゴリズムを利用した手法が多く用いられている．

3.2.3 UCT

UCT(Upper Confidence Bound for Tree) は，多腕バンディット問題に対する解法として用いられる UCB1 アルゴリズムを適用することでモンテカルロ木探索を改良したものである．

UCT ではプレイアウトを行う合法手の選択に対して，勝率と探索された回数に依存する UCB 値によって制御を与えることにより，結果的に最善手として選択される可能性の高い合法手に対して多くのプレイアウトを行う．UCB 値の計算式はいくつか提案されているが，最も基本的なものは以下のように定義される．$\bar{X}_i$ を合法手 i の勝率，n_i は合法手 i が選択された回数，N はその局面における総プレイアウト回数である．

$$UCB(i) = \bar{X}_i + \sqrt{\frac{2\log N}{n_i}} \tag{3.1}$$

第一項により，勝率の高い合法手ほど選択されやすくなる．しかし，探索回数の少ない合法手の勝率は信頼性が低いので，より多くの探索をする必要がある．そのため第二項により，他の合法手よりも探索回数が少ない合法手が選択されやすくなる．

以下に，囲碁における UCT アルゴリズムを示す．また，動作イメージを図 7 に示す．

UCT アルゴリズム

1. 現局面 (ルート) を与える.
2. 候補手 (子ノード) を生成する.
3. UCB 値が最大の手 (ノード) を選択する．一度も選択されていない場合，UCB 値は∞として扱う.
4. ノードが子ノードを持ってる場合，3 に戻る.
5. ノードの訪問回数が閾値を超えていれば，2 に戻る.
6. プレイアウトを行い，結果を更新する.
7. 定められたプレイアウト回数を満たすまで，ルートに戻り，3 を繰り返す.
8. 現局面における候補手の中で，各々定めた条件 (最大プレイアウト回数など) により，選択された手を返す.

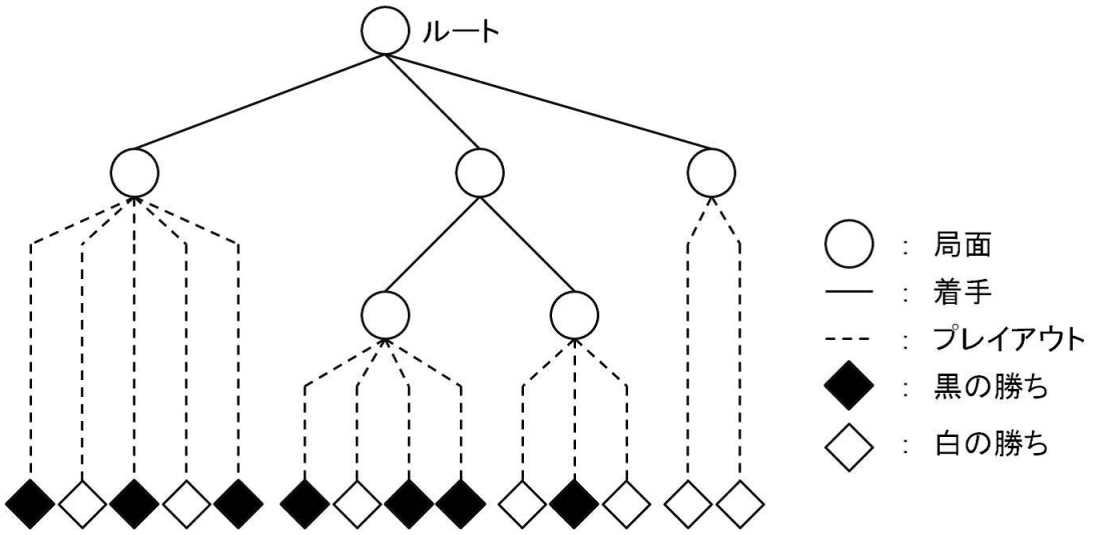

図 7: UCT の動作イメージ

4 基礎研究

本節では，本研究の基礎となる Rémi Coulom の研究 [1] について説明する．

4.1 概要

多くの囲碁プログラムは知識を導入しているが，着手パターンは囲碁プログラムに知識を導入する非常に重要な手法の一つである．

この研究では，Bradley-Terry モデルに基づいた着手パターンの教師あり学習のアルゴリズムを提案している．Bradley-Terry モデルは Elo レーティング・システムの理論的な基礎である．Elo レーティングに基づく方法は，相手の強さも考慮するため，着手パターンの強さをより正確に計算することができる．得られた着手パターンの Elo レーティングは，新しい局面に直面した時，それを用いて全合法手の確率分布を計算することができるということである．その確率分布をもとにプレイアウトの改良や探索木の枝刈りに利用する．

この手法は単純かつ効率的で，複数の特徴を組み合わせることができ，そして合法手の確率分布を得られるため，モンテカルロ碁に知識を導入するのに理想的である．

4.2 Elo レーティングと Bradley-Terry モデル

Bradley-Terry モデルは, 個人対個人の試合結果に関する予測を可能にするモデルであり，各個人 i の強さを正の数値 γ_i で評価する．γ_i は i が強いほど大きな値となる．予測は i が j に勝つ確率を推定する次式にしたがって行われる．

$$P(i \text{ が } j \text{ に勝つ}) = \frac{\gamma_i}{\gamma_i + \gamma_j} \tag{4.1}$$

個人 i の Elo レーティングは $r_i = 400 \log_{10}(\gamma_i)$ で定義される．

4.3 Bradley-Terry モデルの一般化

Bradley-Terry モデルは n 人の試合を扱うように一般化することができる．

$$\forall i \in \{1, ..., n\}, P(i \text{ が勝つ}) = \frac{\gamma_i}{\gamma_1 + \gamma_2 + ... + \gamma_n} \tag{4.2}$$

また，他に個人だけでなくチームを考慮するものがある．この一般化ではチームの γ はそのメンバーの γ の積で推定される. 例えば,

$$P(1-2-3 \text{ が } 4-2 \text{ と } 1-5-6-7 \text{ に勝つ}) = \frac{\gamma_1\gamma_2\gamma_3}{\gamma_1\gamma_2\gamma_3 + \gamma_4\gamma_2 + \gamma_1\gamma_5\gamma_6\gamma_7} \tag{4.3}$$

である．同じ γ が複数のチームに現れることはあるが, 同じチームに複数回現れることはない.

4.4 ベイズ推論

Bradley-Terry モデルは，参加する各個人の強さから将来の試合結果の確率分布を与えるが，ほとんどの場合パラメータ γ_i の正確な値は未知で，過去の試合結果から推定しなければならない．この推定はベイズ推定で可能である．

$\boldsymbol{\gamma}$ をパラメータのベクトル，$\boldsymbol{R}$ ををを過去の結果として，ベイズの公式は

$$P(\boldsymbol{\gamma}|\boldsymbol{R}) = \frac{P(\boldsymbol{R}|\boldsymbol{\gamma})P(\boldsymbol{\gamma})}{P(\boldsymbol{R})} \tag{4.4}$$

である．$\boldsymbol{\gamma}$ の事後分布は $P(\boldsymbol{R}|\boldsymbol{\gamma})$ から，Bradley-Terry モデルで得られる．ただし，$P(\boldsymbol{\gamma})$ はパラメータの事前分布，$P(\boldsymbol{R})$ は正規化定数，パラメータ $\boldsymbol{\gamma}$ は $P(\boldsymbol{\gamma}|\boldsymbol{R})$ が最大になる $\boldsymbol{\gamma}^*$ を見つけることで推定される．

この最適化は，Bradley-Terry モデルと同じ形の事前分布ををを選ぶことで，もっと容易にすることができる．つまり，仮想的な結果 $\boldsymbol{R'}$ が事前分布 $P(\boldsymbol{\gamma}) = P(\boldsymbol{R'}|\boldsymbol{\gamma})$ の役目を果たす．よって，モデルのパラメータの推定は $P(\boldsymbol{R}, \boldsymbol{R'}|\boldsymbol{\gamma})$ の最大化となる．

4.5 Minorization-Maximization アルゴリズム

$\gamma_1, ..., \gamma_n$ を n 人の強さパラメータ，各個人間の独立な N 回の試合結果 $R_1, ..., R_N$ を既知とすると，ある試合結果の確率は次式で書くことができる．

$$P(R_j) = \frac{A_{ij}\gamma_i + B_{ij}}{C_{ij}\gamma_i + D_{ij}} \tag{4.5}$$

ここで，A_{ij}，B_{ij}，C_{ij}，D_{ij} は γ_i と独立な係数．各 $P(R_j)$ をある特定の $\gamma_i 1$ 個の関数とすると，異なる n 通りの方法で書くことができる．例えば，式 4.3 は次のようになる．

$$R_1 = 1 - 2 - 3 \text{ が } 4 - 2 \text{ と } 1 - 5 - 6 - 7 \text{ に勝つ}$$

$$P(R_1) = \frac{\gamma_2\gamma_3 \cdot \gamma_1}{(\gamma_2\gamma_3 + \gamma_5\gamma_6\gamma_7) \cdot \gamma_1 + \gamma_4\gamma_2}$$

ここで，$A_{11} = \gamma_2\gamma_3$，$B_{11} = 0$，$C_{11} = \gamma_2\gamma_3 + \gamma_5\gamma_6\gamma_7$，$D_{11} = \gamma_4\gamma_2$．同様に $A_{21} = \gamma_3\gamma_1$，$A_{31} = \gamma_2\gamma_1$，$A_{41} = 0$，$etc$ となる．

E_j は $E_j = C_{ij}\gamma_i + D_{ij}$ と定義され，$W_i = |\{j|A_{ij} \neq 0\}|$ は個人 i の勝ち数である．目的は次式の最大化となる．

$$L = \prod_{j=1}^{N} P(R_j) \tag{4.6}$$

4.6 Minorization-Maximization 公式の導出

Minorization-Maximization は L を最大化する反復アルゴリズムである．γ の初期推定値 γ^0 から始め，γ^0 で L を少数化する関数 m をつくる．つまり，$m(\gamma^0) = L(\gamma^0)$ かつすべての γ について $m(\gamma) \leq L(\gamma)$．その後 m が最大値をとる γ^1 を求める．少数化によって γ^1 は γ^0 から改善されている．仕組みは，その最大化を閉じた形で計算できるように m をつくることにある．

この最適化のアルゴリズムは，しばしば古典的な勾配法より効率的である．

最大化すべき関数は次式である．

$$L = \prod_{j=1}^{N} \frac{A_{ij}\gamma_i + B_{ij}}{C_{ij}\gamma_i + D_{ij}}$$

L は γ_i の関数と考えることができ，その対数は

$$\log L(\gamma_i) = \sum_{j=1}^{N} \log(A_{ij}\gamma_i + B_{ij}) - \sum_{j=1}^{N} \log(C_{ij}\gamma_i + D_{ij})$$

となる．ここで，$B_{ij} = 0$(個人 i が勝ちチームである) または $A_{ij} = 0$(個人 i が勝ちチームでない) であるので，第 1 項は次のように書ける．

$$\sum_{j=1}^{N} \log(A_{ij}\gamma_i + B_{ij}) = \sum_{j, A_{ij}=0} \log(B_{ij}) + \sum_{j, B_{ij}=0} (\log A_{ij} + \log \gamma_i)$$

γ_i を含まない項を取り除くと最大化すべき関数は次のようになる．

$$f(x) = W_i \log x - \sum_{j=1}^{N} \log(C_{ij}x + D_{ij})$$

ここで，

$$g(x) = -\log(ax + b) \qquad a, b : \text{定数}$$

とすると，$x = \gamma_i$ における接線は

$$y = -\frac{a}{a\gamma_i + b}(x - \gamma_i) - \log(a\gamma_i + b)$$

であるので，

$$g(x) \geq -\frac{a}{a\gamma_i + b}(x - \gamma_i) - \log(a\gamma_i + b)$$

となる．したがって，x を含まない項を取り除くと，最大化する関数は

$$m(x) = W_i \log x - \sum_{j=1}^{N} \frac{C_{ij}x}{C_{ij}\gamma_i + D_{ij}}$$

となる．その微分は

$$m'(x) = \frac{W_i}{x} - \sum_{j=1}^{N} \frac{C_{ij}}{E_j}$$

である．$m(x)$ の最大値は $m'(x) = 0$ を解くことによって見つけられるので，次式で得られる．

$$x = \frac{W_i}{\sum_{j=1}^{N} \frac{C_{ij}}{E_j}}$$

4.7 Minorization-Maximization 公式

Minorization-Maximization は，次式にしたがってパラメータ γ_i を繰り返し更新する．

$$\gamma_i \leftarrow \frac{W_i}{\sum_{j=1}^{N} \frac{C_{ij}}{E_j}} \tag{4.7}$$

個人の強さが異なるとき，C_{ij} は試合 j での i のチームの仲間の強さ，E_j は参加する全員の強さである．

4.8 囲碁への適用

一般化 Bradley-Terry モデルを囲碁における機械学習に応用する．図 8 に示すように，棋譜において打たれた手を勝者，他の候補手が敗者の試合と考えることで適用できる．つまり，一般化 Bradley-Terry モデルにおける個人とは着手の特徴を表し，チームは着手を表すことになる．このように，着手を特徴のチームと考えることで，非常に低いコストで多数の特徴を組み合わせることができる．

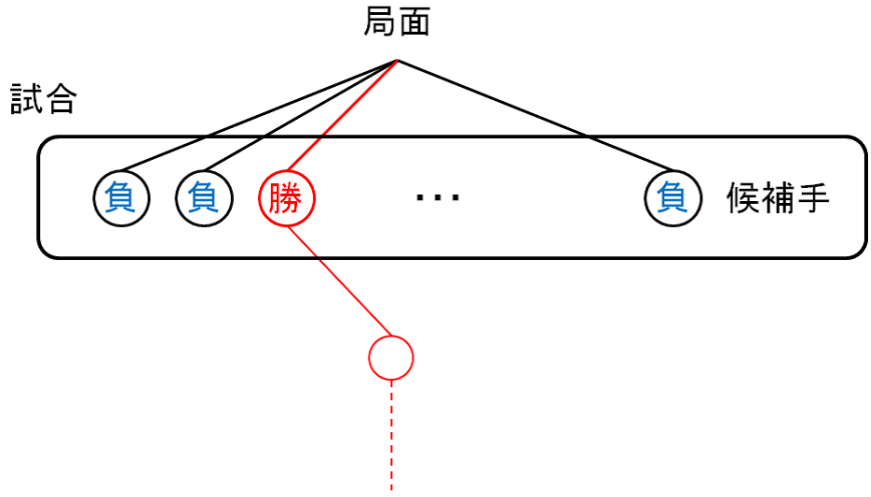

図 8: Bradley-Terry モデルにおける囲碁

5　実験 1

5.1　学習

4 章で説明した Rémi Coulom の手法に従い，プレイアウトで用いる着手の特徴を棋譜から学習する．棋譜は，自由に利用できるプロ棋士による対局のもので，ハンディキャップのない 10000 対局を用いる．本実験では，10000 対局 (2092156 局面) を用いる場合と，1000 対局 (210007 局面) を用いる場合の 2 種類を試みる．

学習する特徴は，3x3 パターン，ひし形パターンと 7 つの戦術的特徴，トリ，ノビ，自己アタリ，アタリ，盤端との距離，直前の手との距離，そして二手前の手との距離である．各特徴の詳しい定義は次節で述べる．これらの特徴が取り得る値を各々独立した個人とみなし，Bradley-Terry モデルにおけるパラメータ γ_i 1 個と関連付ける．

3x3 パターンとひし形パターンについては，各々の有効性を比較するため，組み合わせて学習するのではなく，それぞれ戦術的特徴と共に学習する．また，学習するパターンは棋譜に出現するすべての種類を対象とする．

5.1.1　特徴

学習する各特徴についてその定義を説明する．

本研究における 3x3 パターン，ひし形パターンとは，着手点の近傍の形のパターンを指す．例を図 9 に示す．3x3 パターンは，図 9 に示すような点 A の 8 近傍の形のパターンである．同様に，ひし形パターンは点 B の 12 近傍 (マンハッタン距離：2) の形のパターンである．

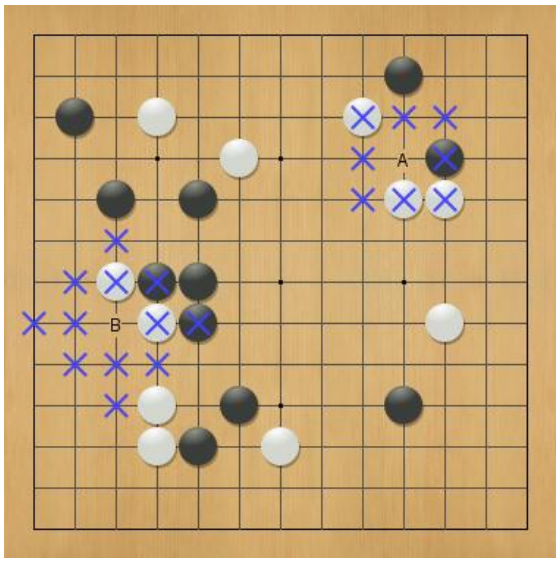

図 9: 形パターンの例

次に，各戦術的特徴について説明する．図 10 において，直前に黒が×に打つと，白の連△が，新しくアタリとなる．このとき白 A は，新しくアタリにされた連に隣接する黒石を取ることができるので，直前の手で新しくアタリにされた連に隣接するトリとなる．また，白 B は直前の手で新しくアタリにされた連に対するノビとなる．

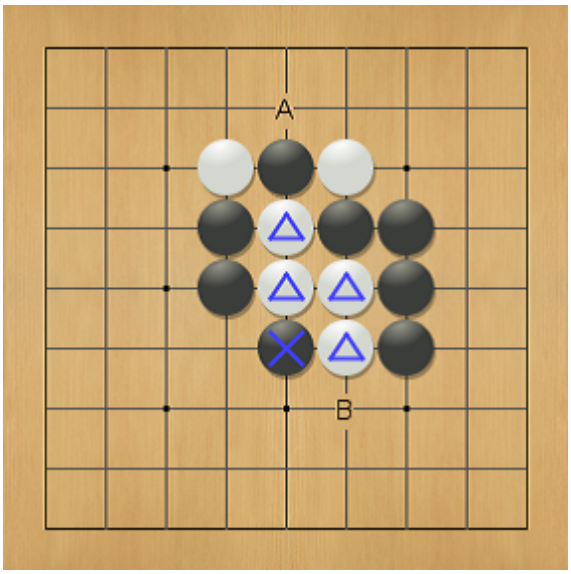

図 10: トリ・ノビの例

図 11 に自己アタリの例を示す．自己アタリとは，打った自分の石がアタリとなる白×のような手である．

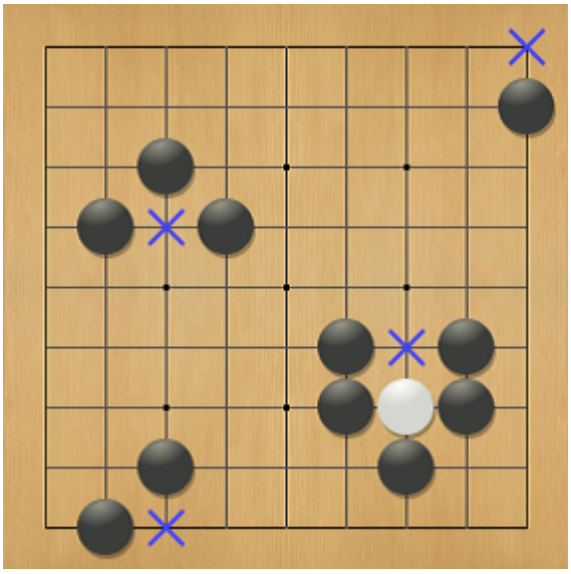

図 11: 自己アタリの例

次に図 12 に距離の例を示す．左側の数字は，盤の左端に対する距離を表している．黒石の周りの数字は着手に対する距離を表している．着手に対する距離は，式 $d(\delta x, \delta y) = |\delta x| + |\delta y| + \max(|\delta x|, |\delta y|)$ で計算される．

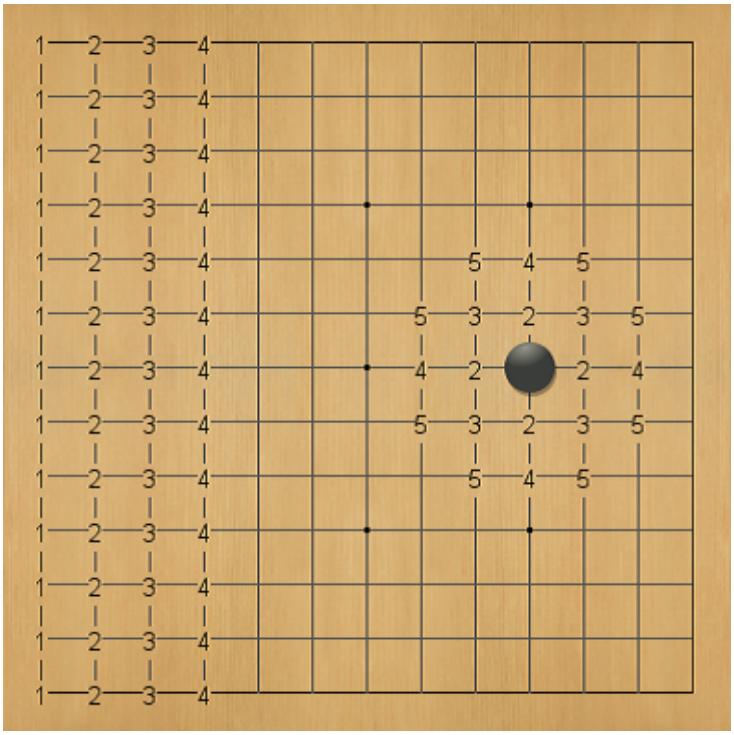

図 12: 距離の例

5.1.2　パターンの実装

本研究におけるプログラムでのパターンの実装方法について説明する．本研究では，パターンをビット列で表現している．パターンは，各点で黒，白，空点，盤外の 4 通りの値を取り得る．つまり，各点は 2 ビットで表すことができる．したがって，3x3 パターンは 8 点なので 16 ビット，ひし形パターンは 12 点なので 24 ビットを用いることで表現することができる．また，データの記録にはパターンを表す値を直接配列の添字として扱っている．

3x3 パターンにおける一例を図 13 に示す．

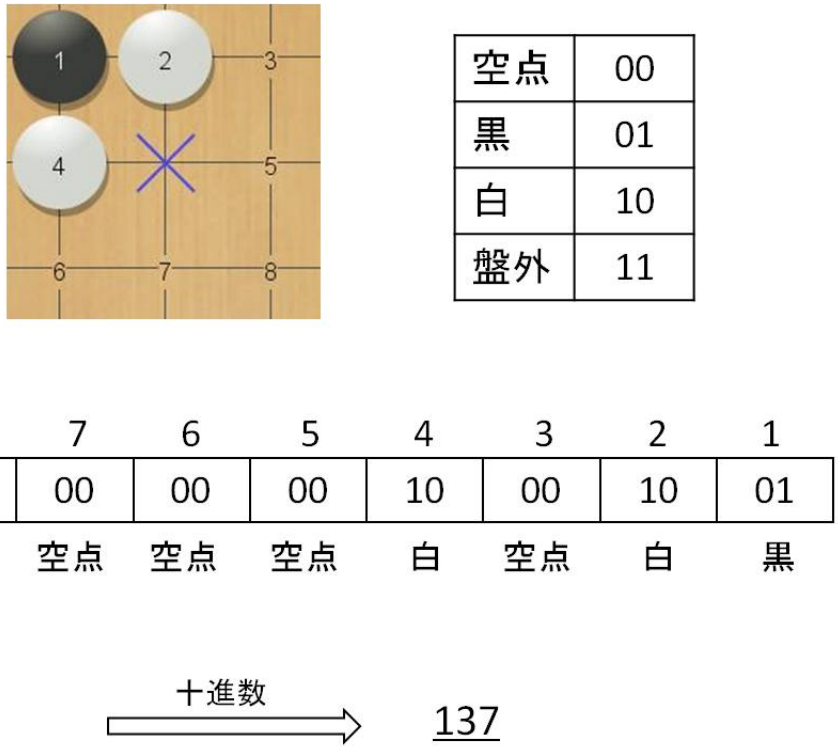

図 13: 3x3 パターンのビット列表現

5.2 対局実験

学習した結果を評価するために対局実験を行う．自作の囲碁プログラムを作成し，異なる「精度」のプレイアウトを実装し，GNU Go3.8[6] に対する対局と自己対局をさせてその棋力を評価する．自作の囲碁プログラムは，モンテカルロ碁で UCT アルゴリズムを実装している．プレイアウト回数は，単純にプレイアウトの「精度」による勝率の変化を比較するため固定している．

対局実験で用いる特徴のモデル・パラメータは，各々棋譜 10000 対局を用いて学習したもので，式 4.7 にしたがって繰り返し更新して得られたもので，十分に収束したと考えられる値である．値の詳細は 5.4 節に示す．

対局実験するプレイアウトは以下の通りである．

- ランダムプレイアウト (Random)
  着手を完全にランダムで選択する．
- MM プレイアウト 1 (MM 3x3)
  着手を学習した結果を用いて計算した確率分布に従って選択する．
  実装する特徴は，戦術的特徴と 3x3 パターン．
- MM プレイアウト 2 (MM Diamond)
  着手を学習した結果を用いて計算した確率分布に従って選択する．
  実装する特徴は，戦術的特徴とひし形パターン．

5.3 環境・条件

実験の環境と対局条件は以下の通りである．

- CPU : Intel Xeon E5430 2.66GHz
- 対局相手 : GNU Go3.8 Level10
- 9 路
- 中国ルール
- コミ 7.5 目

5.4 結果

5.4.1 学習結果

学習した形パターン以外の特徴のモデル・パラメータを表 1 に示す．この結果は，棋譜を 10000 対局 (2092156 局面) 用いて，3x3 パターンと共に学習したもので，100 回繰り返し更新して得られた値である．

表 1: 特徴のモデル・パラメータ

特徴		γ	説明
トリ	1	70.67	直前の手で新しくアタリにされた連に隣接するトリ
	2	1.30	その他
ノビ		4.04	直前の手で新しくアタリにされた連に対するノビ
自己アタリ		0.42	打った石がアタリとなる手
アタリ		1.35	アタリにする手
盤端との距離	1	0.49	
	2	0.99	
	3	1.66	
	4	1.43	
	5	1.07	
直前の手との距離	2	9.11	$d(\delta x, \delta y) = \lvert\delta x\rvert + \lvert\delta y\rvert + \max(\lvert\delta x\rvert, \lvert\delta y\rvert)$
	3	4.85	
	4	3.90	
	5	3.38	
	…	…	
二手前の手との距離	2	2.68	$d(\delta x, \delta y) = \lvert\delta x\rvert + \lvert\delta y\rvert + \max(\lvert\delta x\rvert, \lvert\delta y\rvert)$
	3	2.46	
	4	3.02	
	5	2.01	
	…	…	

表 1 より形パターン以外の特徴の中では，直前の手で新しくアタリにされた連に隣接するトリという特徴が最も高い値を示している．また，直前の手との距離という特徴も全体的に高い値を示している．一方，打った自分の石がアタリとなってしまう自己アタリという特徴は低い値を示している．

盤端との距離という特徴においては，盤の一番外側である 1 線が最も低い値を示し，実利線とも言われる 3 線が最も高い値を示している．

戦術的特徴 (トリ，ノビ，自己アタリ，アタリ) のパラメータ γ_i の値が学習過程で変化する様子を図 14，15 に示す．図 14，15 において，横軸は更新回数を表す．

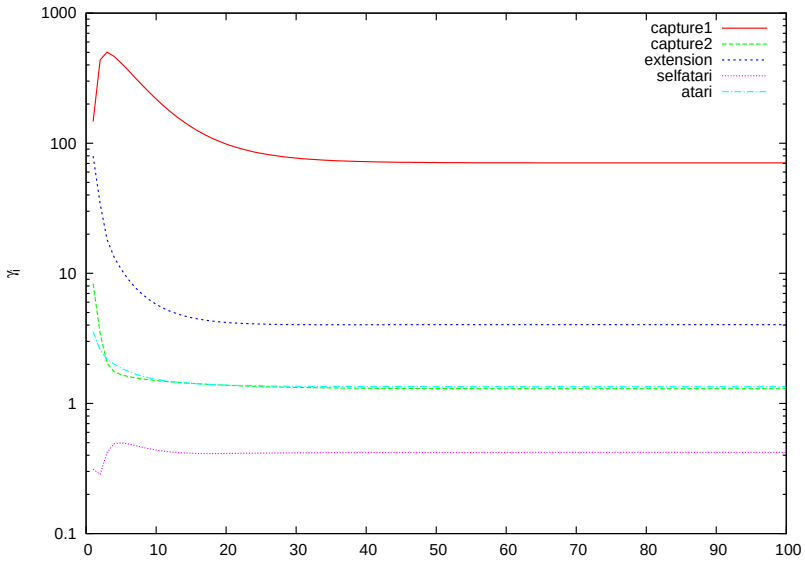

図 14: 学習過程 1 (10000 対局)

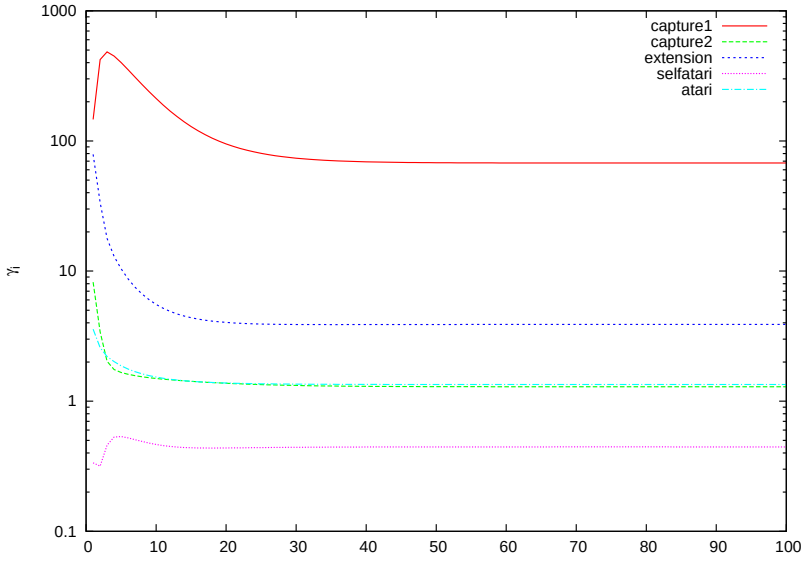

図 15: 学習過程 2 (1000 対局)

図 14，15 より，各戦術的特徴のパラメータ γ_i の値は，約 50 回程度の更新でほぼ一定の値に収束している．

また，学習に用いる棋譜の数 (局面数) によらず，ほぼ同様の変化をしている．

5.4.2 対局結果

対局の結果を以下に示す.

表 2: 対局結果 1 (GNU Go3.8 Level10，プレイアウト回数:10 万回/手)

プレイアウト	勝敗	勝率
Random	283 勝 717 敗 (黒:144 勝 356 敗, 白:139 勝 361 敗)	28.3%
MM 3x3	434 勝 566 敗 (黒:200 勝 300 敗, 白:234 勝 266 敗)	43.4%
MM Diamond	401 勝 599 敗 (黒:205 勝 295 敗, 白:196 勝 304 敗)	40.1%

表 3: 対局結果 2 (GNU Go3.8 Level10，プレイアウト回数:1 万回/手)

プレイアウト	勝敗	勝率
Random	83 勝 917 敗 (黒:45 勝 455 敗, 白:38 勝 462 敗)	8.3%
MM 3x3	171 勝 829 敗 (黒:69 勝 431 敗, 白:102 勝 398 敗)	17.1%
MM Diamond	139 勝 861 敗 (黒:62 勝 438 敗, 白:77 勝 423 敗)	13.9%

自己対局結果を表 4 に示す.

表 4: 自己対局結果 (対 Random，プレイアウト回数:10 万回/手)

プレイアウト	勝敗	勝率
MM 3x3	157 勝 43 敗 (黒:83 勝 17 敗, 白:74 勝 26 敗)	78.5%
MM Diamond	145 勝 55 敗 (黒:76 勝 24 敗, 白:69 勝 31 敗)	72.5%

5.5 考察

5.5.1 学習

表 1 より，形パターン以外の各特徴のパラメータが示す値については，その大小関係が人間の対局者の判断とほぼ一致することから，妥当な結果が得られていると考えられる．

次にパラメータ γ_i の更新回数についてだが，得られるパラメータの値が収束していることが望ましいので，ある程度繰り返し行う必要がある．図 14，15 が示すように，戦術的特徴 (トリ，ノビ，自己アタリ，アタリ) については約 50 回程度の更新でほぼ一定の値に収束している．しかし，パラメータの値の変化は各特徴によって様々なので，他のすべての特徴のパラメータの値が収束しているとは限らない．そのため対局実験で用いるのは，繰り返し更新して得られたもので，多くの特徴について十分に収束したと考えられる値である．

学習で用いる棋譜の数 (局面数) については，棋譜の数によって得られる値の大きさや学習過程に違いが現れることが予想されたが，図 14，15 が示すように，棋譜の数によらず，戦術的特徴のパラメータの値は，ほぼ同等の結果が得られた．これは，学習に用いた棋譜が各特徴を学習するために十分な数であったためだと考える．

5.5.2 対局実験

表 2，表 3 の対局結果より，戦術的特徴と 3x3 パターンを実装した MM プレイアウト 1(MM 3x3) と戦術的特徴とひし形パターンを実装した MM プレイアウト 2(MM Diamond) は，ランダムプレイアウト (Random) に比べて勝率が上昇している．また，表 4 の自己対局結果 1 より，自己対局でランダムプレイアウトに対して大きく勝ち越している．これより，学習結果を用いて改良を加えた各 MM プレイアウトはプレイアウトの「精度」が向上されていると考えられる．

しかし，MM プレイアウト 2 は，MM プレイアウト 1 に比べて勝率がやや低いという結果になった．ひし形パターンは，3x3 パターンと比べるとより広い範囲を考慮することになるので，形のパターンとしてはより正確な判断が可能になると思われるので，ひし形パターンが 3x3 パターンより単純に悪かったとは判断し兼ねる．そのため，ひし形パターンの学習で得られたパラメータの値が信頼できるものではなかったと考える．この学習で良い結果が得られなかったのは，ひし形パターンの種類が豊富であることと，棋譜での出現頻度にその原因があったのではないかと考えられる．この学習で用いた棋譜で出現したひし形パターンの種類は対称，色反転を考慮すると約 6 万通りであった．これは，3x3 パターンの場合では約 600 通りであるのに対して非常に多い．また，その中には棋譜に出現する頻度には偏りがあり，出現が稀なものも存在する．そのため，各特徴を学習するのに用いられるそれぞれのサンプル数に大きな差が生じ，妥当な値が得られなかったのではないかと考える．

本実験では，プレイアウトで用いるときに対応するパターンの数が多い方が良いと考えたため，棋譜に出現するすべてのパターンを学習の対象としたが，出現頻度が高いものや良い形のものだけを抽出して，学習することでより良い結果が得られるのではないかと考える．

6　実験2

6.1　学習・対局実験

前節の考察で述べたように，ひし形パターンの学習は，あまり良い結果が得られなかった．

そこで本実験では，出現頻度が高いパターンのみを抽出して学習する．また，前節と同様にその結果を対局実験で評価する．学習は，棋譜 10000 対局 (2092156 局面) に対して，1 万回以上出現するパターンを対象とした場合と 10 万回以上出現するパターンを対象とした場合を試みる．1 万回以上出現するパターンの種類は約 2600 通りで，10 万回以上出現するパターンの種類は約 300 通りである．

対局実験するプレイアウトは以下の通りである．

- MM プレイアウト 3 (MM Diamond 10000)
  着手を学習した結果を用いて計算した確率分布に従って選択する．
  実装する特徴は，戦術的特徴とひし形パターン (1 万回以上出現するパターン)．

- MM プレイアウト 4 (MM Diamond 100000)
  着手を学習した結果を用いて計算した確率分布に従って選択する．
  実装する特徴は，戦術的特徴とひし形パターン (10 万回以上出現するパターン)．

6.2　棋力の比較

各プレイアウトを実装したプログラムの棋力を比較するために，各プレイアウトの速度比較と思考時間を固定した場合の対局実験を行う．

6.3 結果

6.3.1 対局結果

対局の結果を以下に示す.

表 5: 対局結果 3 (GNU Go3.8 Level10，プレイアウト回数:10 万回/手)

プレイアウト	勝敗	勝率
MM Diamond 10000	500 勝 500 敗 (黒:238 勝 262 敗, 白:262 勝 238 敗)	50.0%
MM Diamond 100000	510 勝 490 敗 (黒:245 勝 255 敗, 白:265 勝 235 敗)	51.0%

表 6: 対局結果 4 (GNU Go3.8 Level10，プレイアウト回数:1 万回/手)

プレイアウト	勝敗	勝率
MM Diamond 10000	228 勝 772 敗 (黒:98 勝 402 敗, 白:130 勝 370 敗)	22.8%
MM Diamond 100000	248 勝 752 敗 (黒:112 勝 388 敗, 白:136 勝 364 敗)	24.8%

自己対局結果を表 7 に示す.

表 7: 自己対局結果 2 (対 Random，プレイアウト回数:10 万回/手)

プレイアウト	勝敗	勝率
MM Diamond 10000	162 勝 38 敗 (黒:81 勝 19 敗, 白:81 勝 19 敗)	81.0%
MM Diamond 100000	169 勝 31 敗 (黒:82 勝 18 敗, 白:87 勝 13 敗)	84.5%

6.3.2 棋力の比較結果

各プレイアウトの速度を表 8 に示す.

表 8: プレイアウトの速度比較

プレイアウト	回数/秒
Random	24000
MM 3x3	4600
MM Diamond	4200
MM Diamond 10000	4200
MM Diamond 100000	4200

思考時間を固定した場合の対局の結果を表 5 に示す.

表 9: 対局結果 5 (GNU Go3.8 Level10，思考時間:10 秒/手)

プレイアウト	勝敗	勝率
Random	208 勝 392 敗 (黒:114 勝 186 敗, 白:94 勝 206 敗)	34.7%
MM 3x3	235 勝 365 敗 (黒:106 勝 194 敗, 白:129 勝 171 敗)	39.2%
MM Diamond	235 勝 365 敗 (黒:121 勝 179 敗, 白:114 勝 186 敗)	39.2%
MM Diamond 10000	300 勝 300 敗 (黒:138 勝 162 敗, 白:162 勝 138 敗)	50.0%
MM Diamond 100000	298 勝 302 敗 (黒:147 勝 153 敗, 白:151 勝 149 敗)	49.7%

6.4 考察

6.4.1 学習・対局実験

対局結果より，MM プレイアウト 3(MM Diamond 10000) と MM プレイアウト 4(MM Diamond 100000) は，棋譜に出現するすべてのパターンを学習の対象とした MM プレイアウト 2(MM Diamond) と比べて勝率が上昇している．よって，出現頻度が高いパターンのみを抽出して学習する手法は有効であったと考えられる．

また，本実験では 10 万回以上出現するパターンを対象とした MM プレイアウト 4 が最も高い勝率となったが，これは，学習で対象とするパターンが少なくなったことで十分に収束した信頼できる値が得られていたということと，出現頻度が高いパターンを選択しているので，プレイアウトでも活用されたためだと考えられる．

6.4.2 棋力の比較

表 8 より，各 MM プレイアウトの速度はランダムプレイアウトと比較すると大幅に減少している．しかし，MM プレイアウトは着手の特徴を抽出したり，確率分布を計算しなければならないため，ランダムプレイアウトに対して，ある程度速度が遅くなるのはやむを得ないことである．また，プレイアウトの速度は実装方法に依存するので，プログラミングの技量に大きく関わる．本来プレイアウトの「精度」を向上させるのは，プログラムの棋力を向上させることが目的であるので，速度の低下を無視できるほど,「精度」を向上させることが求められる．

表 9 より，各 MM プレイアウトは，ランダムプレイアウト (Random) と比較して思考時間を固定した場合の対局の勝率が上昇している．つまり，僅かではあるがプログラムの棋力が向上されていると考えられる．しかし，5.4.2 節で示すプレイアウト回数を固定した場合の結果と比べて，勝率の上昇の程度は小さい．この要因としては，速度の大幅な減少がその一つとして考えられる．したがって，一概には言えないが MM プレイアウトの実装には改良の余地があると考える．

7　おわりに

7.1　まとめ

本研究は，Rémi Coulom の手法を用いた着手の特徴の学習と，その結果を用いたプレイアウトの改良を行った．

学習した戦術的特徴と形のパターンを実装した各プレイアウトについて，本研究の目的であるプレイアウトの「精度」の向上を示すことができた．さらに，プログラムの棋力が僅かではあるが向上したことから，改良したプレイアウトが有効であることを示すことができた．

7.2　今後の課題

今後の課題としては，本研究で用いた形のパターンよりさらに大きな形のパターンの学習とその実装方法を模索していきたい．

また，本研究で用いた着手の特徴の他にもプレイアウトの「精度」を向上させるのにより有効なものがあると予想されるので，特徴を吟味して実装していきたい．

謝辞

本研究を進めるにあたり，ご指導頂いた村松正和教授，脇隼人助教に感謝致します．また，ご協力頂いた村松研究室の皆様に感謝します．

参考文献

[1] Rémi Coulom，Computing Elo Ratings of Move Patterns in the Game of Go，ICGA journal Vol.30 No.4，(2008).

[2] 門脇聡広，モンテカルロ碁における統計的手法による特徴の学習，修士論文，電気通信大学,（2010）.

[3] 清 愼一，山下 宏，佐々木 宣介，コンピュータ囲碁の入門，共立出版株式会社,（2005）.

[4] 小谷善行，岸本章宏，柴原一友，鈴木豪，コンピュータ数学シリーズ　ゲーム計算メカニズム - 将棋・囲碁・オセロ・チェスのプログラムはどう動く -，コロナ社,（2011）.

[5] Crazy Stone
http://remi.coulom.free.fr/CrazyStone/,（2011）.

[6] GNU Go
http://www.gnu.org/software/gnugo/,（2011）.

[7] 囲碁用語辞典
http://www.godictionary.net/,（2011）.

[8] 美添一樹，コンピュータ囲碁におけるモンテカルロ法〜理論編〜，電気通信大学エンターテイメントと認知科学研究ステーション第 5 回招待講演会，(2008).