麻雀AIの戦略的行動決定に関する研究

修士学位論文

麻雀AIの戦略的行動決定に関する研究

東北大学大学院 情報科学研究科

システム情報科学専攻 篠原・吉仲研究室

博士課程前期二年の課程

海津 純平

2017年2月17日

目次

第1章

序論

1.1 背景

近年，様々なゲーム AI の研究が行われている．そのゲームの中のひとつに，零和多人数不完全情報ゲームである麻雀がある [1, 2]．麻雀は，相手の手牌や自分と相手の点数状況など，考慮しなければならない要素が多い．そのため，麻雀 AI の研究は，AI に必要な要素についてそれぞれ研究を行うことが重要である．

麻雀を対象にした関連研究は，様々な要素について行われている．行動選択の研究としては，水上らが平均パーセプトロンを用いた捨て牌選択と SVM による鳴き局面の認識を行った [3]．学習にはオンライン麻雀ゲームの天鳳 [1] の牌譜を用いている．状況に応じた戦略の研究としては，田中らが麻雀初級者のための状況に応じた着手モデルを提案した [4]．天鳳の牌譜を用いた学習により，状況に応じた戦略を導出できる決定木を構築している．他にも，水上らは多クラスロジスティック回帰モデルによる期待最終順位の予測や，強化学習を用いて手牌から上がり点の予測を行い，その予測結果から状況に応じて戦略を変える手法を提案した [5, 6]．降りの研究としては，水上らが SVM を用いた降り局面の認識を行った [3]．また，我妻らは半強師あり学習の一種である Self-Training を用いて降り局面の判別を行った [7]．他プレイヤの不完全情報推定の研究としては，我妻らが SVR（Support Vector Regression）を用いて捨て牌の危険度推定を行った [8]．他にも，水上らはロジスティック回帰モデルによる他プレイヤの聴牌，待ち牌予測と，重回帰モデルによる上がり点予測を行った [9]．このように様々な研究が行われている中，本研

[1] http://tenhou.net

究では行動選択と状況に応じた戦略を対象とする．

本研究では，ゲーム中の大局的な状況を考慮し，大局的な状況に応じて打ち方の変更を行う麻雀 AI を提案する．多人数の麻雀では，トップを狙うべきである．そのため，自分の手牌や他人の捨て牌だけでなく，自分の現在順位，点差，何局目かなど様々な状況を考慮した打ち方の戦略を選ぶ必要がある．例えば，自分が最下位のときは上がり点を高くする打ち方にすべきであり，自分がトップのときはゲームを早く終了させるために早上がりをする打ち方にすべきである．また，自分の役割に応じて打ち方を変える場合もある．

まずは，打ち方の変更が可能な麻雀 AI を提案する．既存手法として，小松らが提案した，モンテカルロ法 [10] を基にした手法がある．麻雀におけるモンテカルロ法は，ほとんどのシミュレーションで報酬が得られない．これを改善するため，小松らの手法 [11] では，効率的なシミュレーションを行うことで性能を向上させている．また，原田らの手法 [12] では，決めた時間の中で小松らの手法のシミュレーションを繰り返す．このことによって，状況に応じた適切なシミュレーション回数を実現している．しかし，どちらの手法も麻雀において，上がり点が高くなることを重視した打ち方をしているため，上がるスピードが遅くなったり，上がれないことも多くなるという問題がある．本研究では，小松らの手法を基にし，一つのパラメータを変更することで打ち方を変えることが可能な手法を提案する．上がり点を高くするための評価指標と早上がりをするための評価指標の二つを報酬として用いている．

打ち方の変更が可能な手法を基に，大局的な状況に応じて打ち方の変更を行う麻雀 AI を提案する．ゲームの途中で，大局的な状況から適切な打ち方に変更する．大局的な状況としてゲームの進み具合と現在順位，自分の役割を考慮する．

計算機実験では，打ち方の変更が可能な手法のパラメータを変更することで，打ち方がどう変わるかを確かめる．また，市川が開発した麻雀 AI 対戦サーバ Mjai [2] を用いて，打ち方を変えない AI と，大局的な状況に応じて打ち方を変更する AI を対戦させる．これによって，大局的な状況に応じた打ち方の変更が有用であるかを確かめている．

[2]https://github.com/gimite/mjai

1.2 構成

第 2 章では，麻雀のルールついて説明する．第 3 章では，既存手法として，麻雀におけるモンテカルロ法，小松らの手法を説明する．第 4 章では提案手法を説明する．第 5 章では実験により打ち方が変わっているか確かめ，対戦実験を行う．第 6 章では，まとめと今後の課題について述べる．

第2章

準備

2.1 麻雀のルール

この節では，この論文において用いられるルールについてのみ説明する．麻雀は図 2.1 に示す 34 種類の牌がそれぞれ 4 枚，計 136 枚の牌を用いて 4 人で行うゲームである．牌は**萬子**(まんず)，**筒子**(ぴんず)，**索子**(そうず)と呼ばれる 1 から 9 の数字のいずれかを表す**数牌**と，文字が書かれた**字牌**がある．

ゲーム開始時は各プレイヤ 25000 点を持ち，ゲームが終了時に持っている点数で勝敗を競う．ゲームは，**局**と呼ばれる単位に分割されている．局を特定の回数繰り返すことでゲームが終了する．一般的に 1 ゲームは 8 局行われ，これを**半荘**(はんちゃん)と呼ぶ．それぞれの局で，プレイヤの 1 人が**親**という役割を担当し，その他のプレイヤが**子**という役割になる．局の開始時には，**山**と呼ばれる伏せられた牌の集合から，各プレイヤは 13 枚の牌を取る．山は伏せられているためプレイヤはどの牌を持ってくるかはわからない．プレイヤが所持する牌を**手牌**と呼び，他プレイヤには公開されない．

局が始まるとプレイヤは親から順番に，山から 1 枚の牌を取り，14 枚になった手牌の中から 1 枚の牌を場に捨てる行為を，各プレイヤが繰り返してゲームを進行する．牌を捨て，また牌を取る番が周ってくるまでの流れの単位を**巡**と呼ぶ．手牌が 13 枚のとき，山から取った牌または他プレイヤが捨てた牌を加えた 14 枚の牌の組合せが特定の条件を満たしていればプレイヤは**上がり**[1]，その組合せに応じた点数を得ることができる．山から取った牌で上がった場合は，他のプレイヤ全員が上がったプレイヤに点数を払う．他

---

[1]本来，和了（ほーら）と言うがこの論文では単に上がるという表現を用いる．

プレイヤが捨てた牌で上がった場合は，その牌を捨てたプレイヤが上がったプレイヤに点数を払う．また，上がったプレイヤが親の場合，点数は 1.5 倍となる．局は，山の残り牌数が特定数になるか，プレイヤの一人が上がったときに終了する．子であるプレイヤが上がった場合，親の役割が別のプレイヤへ移り，次の局が始まる．親であるプレイヤが上がった場合，親の役割は移らず，局数が変わらないまま局を繰り返す．これを**連荘**と呼ぶ．

同種の数牌が 3 連続している組を**順子**，同一の牌が 3 枚の組を**刻子**，同一の牌が 2 枚の組を**雀頭**と呼ぶ．順子，刻子，雀頭の例を図 2.2， 2.3， 2.4 に示す．原則として，順子と刻子を合わせて 4 つ，雀頭を 1 つの 14 枚の牌が上がりに必要な組合せである．図 2.5 に上がりに必要な 14 枚の牌の組合せ例を示す．この上がりに必要な 14 枚の組合せに**役**と呼ばれる特定のパターンがあることが上がり条件となる．複数の特定のパターンが組合せにある場合，複数の役があることになる．さらに，特定のパターンではない例外である役の一つに**リーチ**という役がある．リーチは行動名でもあり，あと 1 枚の牌を加えたら上がりに必要な組合せができることを宣言する．リーチをしたあとは，上がりに必要な組合せにならない限り，取ってきた牌をそのまま捨てなければならない．このとき，山から取った牌または他プレイヤが捨てた牌を加えた 14 枚の牌が，上がりに必要な組合せとなった場合，役のリーチが成立し上がることができる．役には，上がり点の要素となる**飜数**が割り当てられている．役は複数存在し，役の種類により飜数が異なる [13]．役による飜数，手牌の構成や上がり時の状況に応じて得られる上がり点が変動する [14]．

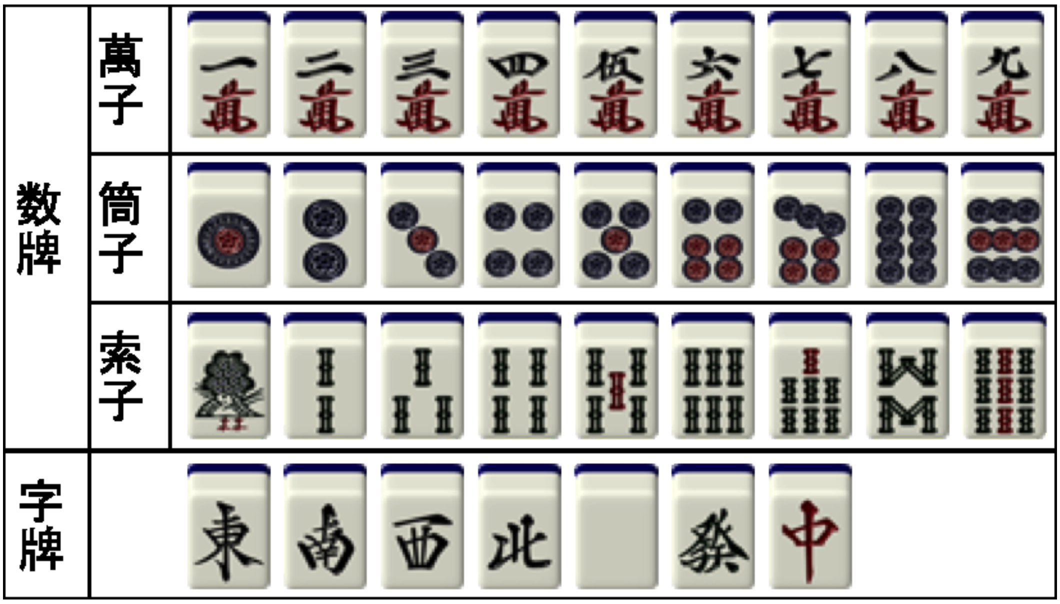

図 2.1: 牌の種類

図 2.2: 順子の例

図 2.3: 刻子の例

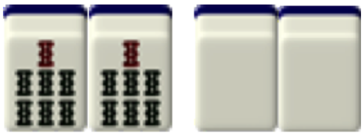

図 2.4: 雀頭の例

図 2.5: 上がりに必要な牌の組合せ例

第3章

既存手法

3.1 モンテカルロ法

モンテカルロ法 [10] では，ゲームを最後まで行うシミュレーションを繰り返し，シミュレーションの結果による**報酬**を行動に与える．また，報酬の合計を**価値**と呼ぶ．シミュレーションを複数回繰り返し，最も価値が高い行動を選択する．シミュレーション時の不確定情報や，プレイヤの行動はランダムに決める．

モンテカルロ法を麻雀に適用したものを Algorithm 1 に示す．モンテカルロ法を麻雀に適用すると，シミュレーション中の行動選択をランダムに行うため，ほとんどのシミュレーションで報酬を得られないことを小松らは実験的に示している [11]．行動の良さをうまく評価できず，良い行動選択を行うことができないため，単純なモンテカルロ法は麻雀に対して効率の悪い手法であると言える．

3.2 小松らの手法

前節で述べたように，単純なモンテカルロ法では，上がりの可能性が非常に低いことから，良い行動選択を行うことが難しい．そこで，小松らは麻雀の上がり可能性を考慮した手法として，Algorithm 2 に示すようなモンテカルロ法を応用した手法を提案した [11]．

Algorithm 2 では，3, 4 行目に示すように，手牌 14 枚 [1] と残り巡数 K 枚分のランダムに仮定した牌の中から，報酬 $R_{komatsu}$ が最大となる組合せを求める．その後，手牌の中で，求めた組合せに含まれていない牌 t_j に報酬を与える．これらの操作が 1 回のシミュ

[1] ここでは，もともと持っている 13 牌ととってきた牌を合わせた 14 枚を手牌と呼ぶ．

レーションとなる．シミュレーションを N 回繰り返したのち，最も報酬が高い牌を，捨てる牌として選択する．

手牌を $T = \{t_1, t_2, \cdots, t_{14}\}$ とする．残り K 巡でとってくるであろう牌を $T' = \{t'_1, t'_2, \cdots, t'_K\}$ とする．また，14 枚の牌の多重集合 S によって得られる点数を $P(S)$ とする．このとき，小松らの手法では，報酬として以下の値を用いている．

$$R_{komatsu} = \max_{S \subseteq T \cup T'} P(S) \tag{3.1}$$

つまり，小松らの手法では，得られる点数が最も高くなるような牌の組合せを求め，その得点を報酬としている．

しかし，報酬として $R_{komatsu}$ を計算する場合，$P(S)$ が最大となるような組合せ S が複数ある場合がある．ところが，小松らの手法ではその中でどの組合せを選択するかについては考慮されておらず，任意の組合せが選ばれる．表 3.1 は，小松らの手法で 10000 回のシミュレーションにおいて，最大得点となる組合せが複数ある場合の回数および，そのときの組合せの種類数に関する予備実験の結果である．また，最大得点となる組合せの交換回数とは，1 回のシミュレーションで求めた組合せと，元の手牌で変化した枚数のことである．1 回のシミュレーションにおいて求めた，最大得点となる組合せの中で，交換回数の最大値と最小値を求め，10000 回における平均を示す．

この結果からもわかるように，多くのシミュレーションにおいて，特に残り巡数が多い場合に，得点が最大となる組合せが複数存在していることがわかる．また，得点が最大となる組合せが複数ある場合，組合せによって上がるまでに必要な牌の交換回数が異なっていることがわかる．つまり，組合せの選び方によっては同じ点数を得るとしても，上がるまでに多くの巡数が必要になってしまうことになる．

Algorithm 1: 麻雀におけるモンテカルロ法

Input: 手牌 $T = \{t_1, t_2, \cdots, t_{14}\}$，シミュレーション回数 N，残り巡数 K

Output: 捨てる牌 t_s

すべての $i \in \{1, \cdots, 14\}$ について価値を格納する V_i を 0 にする;
for $i = 1$ **to** 14 **do**
    for $j = 1$ **to** N **do**
        牌 t_i を捨てる;
        for $k = 1$ **to** K **do**
            ランダムに 1 枚の牌を手牌に加える;
            if 手牌が上がり条件を満たしている then
                $V_i \leftarrow V_i +$ 報酬;   /*報酬は上がり点*/
                break;
            end
            ランダムに 1 枚の牌を手牌から捨てる;
        end
        手牌 T を入力時の手牌に戻す;
    end
end
$s \leftarrow \underset{1 \le i \le 14}{\arg\max} V_i$;
output t_s;

Algorithm 2: 小松らの手法

Input: 手牌 $T = \{t_1, t_2, \cdots, t_{14}\}$，シミュレーション回数 N，残り巡数 K

Output: 捨てる牌 t_s

すべての $j \in \{1, \cdots, 14\}$ について価値を格納する V_j を 0 にする;

for $i = 1$ **to** N **do**

  K 枚の牌をランダムに選ぶ;

  手牌 T と K 枚の牌の中から報酬 $R_{komatsu}$ が最大となる組合せ T' を求める;

  for $j = 1$ **to** 14 **do**

    if **牌** t_j **が組合せ** T' **に含まれない** **then**

      $V_j \leftarrow V_j + R_{komatsu}$;

    end

  end

end

$s \leftarrow \arg\max_{1 \leq j \leq 14} V_j$;

output t_s;

表 3.1: 小松らの手法で 10000 回のシミュレーションにおける得点が最大となる組合せの詳細

残り巡数	最大得点となる組合せ		最大得点となる組合せの交換回数	
	複数ある回数	種類数の平均	最大値の平均	最小値の平均
17	7101	19.0	8.8	6.6
16	6584	16.6	8.6	6.5
15	6538	17.4	7.8	5.6
14	5404	14.6	7.6	5.5
13	4208	13.1	7.3	5.4
12	3612	10.2	7.0	5.3
11	2597	7.9	6.8	5.3
10	1837	4.9	6.0	4.8
9	2031	5.4	5.2	3.9
8	1081	4.6	4.9	3.8
7	665	4.0	4.6	3.6
6	282	3.5	4.2	3.4
5	119	3.2	3.9	3.4
4	36	2.3	3.5	3.2
3	6	2.0	3.0	3.0
2	27	2.0	2.0	2.0
1	0	-	-	-

第4章

提案手法

4.1 打ち方の変更が可能な手法

小松らの手法は，上がりの可能性を考慮したモンテカルロ法であるが，高得点を得ることだけを目的としており，早く上がることを考慮していない．ここでは，小松らの手法を基に，早上がりについても考慮できるような手法を提案する．

4.1.1 牌の交換回数

早上がりとは，言い換えれば現在の手牌と牌の交換回数が少ない上がり形を目指すことである．そこで，牌の交換回数に関連した評価指標として，以下の式を定義する．

$$U(T, S) = |T \cap S|$$

ここで，T と S はそれぞれ 14 枚の牌からなる牌の多重集合であり，$T = \{t_1, t_2, \cdots, t_{14}\}$，$S = \{s_1, s_2, \cdots, s_{14}\}$ である．

シミュレーションにおいて，T を現在の手牌，S を求めた上がり形の牌とすることで，$U(T, S)$ は手牌の中で上がるために交換の必要がない牌の枚数となる．$U(T, S) = 14$ となるならば，すでに上がりであるため，捨て牌を選ぶ必要がなくなる．そのため，$S \neq T$ のときの $U(T, S)$ の最大値は 13 である．$U(T, S)$ の値が大きいということは，シミュレーションで求めた目指す手牌と，元の手牌で変化した枚数が少ない，つまり，少ない交換回数で上がることができる可能性が高いということである．

4.1.2　早上がりを考慮した報酬

ここでは，4.1.1 節で提案した交換回数に関する評価指標を基にすることで，早上がりを考慮したシミュレーションの報酬を提案する．提案する報酬は，単に早上がりだけを目指すのではなく，小松らの手法同様に高い点数を得られることをユーザが選択できるようにする．そこで，以下のような報酬を定義する．

$$R = \max_{S \subseteq T \cup T'} \left(\alpha \frac{P(S)}{M} + (1-\alpha) \frac{U(T,S)}{13} \right) \tag{4.1}$$

ここで，$0 \le \alpha \le 1$ はバランスパラメータである．$\alpha = 1$ のときは，小松らの手法と同じ報酬となる．式の第 1 項は上がり点の最大値 M(親は 48000，子は 32000) で，第 2 項では $U(T,S)$ に対して手牌の交換が不必要な枚数の最大値 13 で，それぞれ正規化している．式 4.1 を用いた打ち方の変更が可能な手法を Algorithm 3 に示す．

小松らの手法と同様に，$P(S)$ が高いほど，高い点数を得ることができる．また，これらの 2 つの値を，パラメータ α を用いて調整することで，得点または早上がりを優先した打ち方を選択することができる．

4.2　リーチのアルゴリズム

麻雀において，リーチは重要であるため，打ち方の変更が可能な手法に組み込めるリーチのアルゴリズムを提案する．捨てる牌を決め，その牌を捨てることでリーチができる場合，リーチをするようにしている．リーチのアルゴリズムを Algorithm 4 に示す．

4.3　大局的な状況に応じた打ち方の変更

大局的な状況に応じて打ち方を変更する AI を実現するために，打ち方の変更が可能な手法の α を局の開始時に変更する．局の開始時に，現在順位と局数，また自分の役割から α を決定する．現在順位が 1 位や 2 位であるときは，早くゲームを終了させるために α を小さくし早上がりを目指す．序盤の局では，1 位であっても点差をつけるために，$\alpha = 0.2$ などにして点数を取れるようにしている．終盤の局になるほど，他プレイヤに逆転をされないよう点数より早上がりを優先し勝ち逃げを狙う方が良いため，α をだんだ

Algorithm 3: 打ち方の変更が可能な手法

```
Input: 手牌 T = {t_1, t_2, ···, t_14}，シミュレーション回数 N，残り巡目 K
Output: 捨てる牌 t_s
すべての i ∈ {1, ···, 14} について価値を格納する V_i を 0 にする;
for i = 1 to N do
    K 枚の牌をランダムに選ぶ;
    手牌 T と K 枚の牌の中から報酬 R が最大となる組合せ T' を求める;
    for j = 1 to 14 do
        if 牌 t_j が組合せ T' に含まれない then
            V_j ← V_j + R;
        end
    end
end
s ← arg max_{1≤j≤14} V_j;
output t_s;
```

ん小さくしている．現在順位が 3 位や 4 位であるときは，高得点を目指すために α を大きくする．終盤の局になるほど，ゲームが終了してしまう前に大きく逆転するように α をだんだん大きくしている．また，親の役割のときは連荘があり，点数を取るチャンスであるため，現在順位が低い場合は連荘を狙うべきである．よって，上がる必要があるため，現在順位が低いほど α を小さくし早上がりを目指す．連荘となった場合，局数は変わらないが，繰り返された局の開始時には親であっても子であっても α の変更は行う．現在順位，局数，自分の役割と α の関係を表 4.1 に示す．

Algorithm 4: リーチのアルゴリズム

Input: 手牌 $T = \{t_1, t_2, \cdots, t_{14}\}$

Output: 捨てる牌 t_s，リーチをするか否か

手牌 T から捨てる牌 t_s を決める;

if t_s **を捨てることでリーチができる** **then**

  output t_s，リーチをする;

end

else

  output t_s，リーチをしない;

end

表 4.1: 現在順位と局数，自分の役割に対する α の値

		1位	2位	3位	4位
子	1局目	0.0	0.0	0.0	0.0
	2局目	0.2	0.5	0.6	0.9
	3局目	0.2	0.5	0.6	0.9
	4局目	0.2	0.5	0.6	0.9
	5局目	0.1	0.4	0.7	1.0
	6局目	0.1	0.4	0.7	1.0
	7局目	0.0	0.3	0.8	1.0
	8局目	0.0	0.3	0.8	1.0
親		0.6	0.4	0.2	0.0

第 5 章

実験

5.1 計算時間

他プレイヤを設定しない麻雀である一人麻雀を用いて実験を行う．一人麻雀とは以下のようなゲームである．

1. プレイヤは山から 13 枚の牌を取り，手牌とする．
2. プレイヤは山から 1 枚の牌を取り，手牌に加える．
3. 手牌が上がる条件を満たすならば，プレイヤは上がり点を得て，ゲームを終了する．
4. 手牌から牌を 1 枚捨てる．
5. 捨てた牌の枚数が 18 枚ならゲームを終了する．そうでないならば，2 に戻る．

それぞれの残り巡数において，牌を一枚取ってきて捨てる牌を決めるまでの計算時間を比較する．比較する手法は，モンテカルロ法，小松らの手法（$\alpha = 1$），報酬が U のみの手法（$\alpha = 0$），打ち方の変更が可能な手法（$\alpha = 0.5$）である．小松らの手法の報酬が P のみであるため，報酬が U のみの手法を含めている．ゲーム回数は 100 回，シミュレーション回数は 1000 回とする．

結果を表 5.1 に示す．それぞれの残り巡数における，計算時間（秒）の平均である．麻雀のオンラインゲーム（天鳳，真・雀龍門 [1] など）において，捨てる牌を決めるまでの制限時間は 5 秒である．打ち方の変更が可能な手法は，最も長い計算時間でも 0.11343 秒で

[1] http://www.ncsoft.jp/janryumon

あるため実用的な手法だと言える．モンテカルロ法は手牌一つずつに対してシミュレーションを行うため計算時間が遅く，残り巡数 17 のときの計算時間は 0.27589 秒である．そのため，シミュレーションを増やすと制限時間の 5 秒を超えてしまうと考えられる．これに対し，打ち方の変更が可能な手法はモンテカルロ法よりも計算時間が速く，残り巡数 17 のときでも，計算時間は 0.11343 秒である．よって，シミュレーションを回数を増やしても，十分実用的であると考えられる．

5.2 一人麻雀における性能比較

一人麻雀を用いた実験により，モンテカルロ法，小松らの手法との性能比較を行う．1 回のゲームでは各手法に対して同じ山を用いて行う．ゲームは 10000 回行い，平均結果を求める．小松らの手法，および打ち方の変更が可能な手法のシミュレーションは 1000 回とする．打ち方の変更が可能な手法における α の値の設定は 0 から 1 まで 0.1 刻みで行う．

結果を表 5.2 に示す．平均点数は得られた点数の合計をゲーム回数で除した値である．平均上がり得点は得られた点数の合計を，平均上がり巡数は上がったときの巡数の合計を上がった回数で除した値である．上がり率はどの α の値でも，モンテカルロ法よりも良い結果となった．平均上がり得点から，α を大きくすると上がり得点が高くなることがわかる．また，平均上がり巡数から，α を小さくすると早上がりができていることがわかる．α の値を変えることで，麻雀に必要な打ち方の変更が可能できていると言える．また，高得点と早上がりのバランスがとれているため，α によってはゲーム回数による平均点数が小松らの手法を上回っている．四人麻雀においては最終的に最も高い点数を持っているプレイヤの勝ちなので，打ち方の変更が可能な手法の方が，優位な手法と言える．

5.3 予備実験

麻雀 AI 対戦サーバ Mjai を用いて，打ち方の変更が可能な手法を，取ってきた牌をそのまま捨てるだけの AI 3 つと対戦させる．打ち方の変更が可能な手法では，リーチを行

う．1 回のゲームは半荘とし，1000 回行う．シミュレーション回数は 1000 回とする．α の値は 0 から 1 まで 0.1 刻みで行う．

結果を表 5.3 に示す．平均点数はゲーム終了時に持っていた点数の平均である．平均上がり得点と平均上がり巡数は上がった回数による平均である．平均上がり巡数から，α が小さいときは早上がりができており，平均上がり得点から，α が大きいときは上がり得点が高くなっていることがわかる．四人麻雀においても，α を変えることで，打ち方の変更ができている．また，平均点数は $\alpha = 0.9$ のとき最も高くなっている．前節の実験においても，$\alpha = 0.9$ の平均点数が最も高いため，$\alpha = 0.9$ が四人麻雀において最も良い α の値だと考えられる．

5.4　対戦実験

大局的な状況に応じて α を変更する AI の有用性を確かめる．打ち方の変更が可能な手法の $\alpha = 0.0$，$\alpha = 0.9$，$\alpha = 1.0$，つまり，打ち方を固定する AI と，大局的な状況に応じて打ち方を変更する AI を対戦させる．$\alpha = 0.9$ と対戦させたのは，前節の実験において，このパラメータが四人麻雀で強いと考察されたためである．リーチ無しの場合とリーチ有りの場合に分けて対戦させる．リーチ有りの場合は 4 つの AI すべてにリーチのアルゴリズム（Algorithm 4）を組み込む．1 位〜4 位率，平均順位，平均得点について比較する．平均点数はゲーム終了時に持っていた点数の平均である．1 回のゲームは半荘とし，1000 回行う．シミュレーション回数は 1000 回とする．

リーチ無しの場合の結果を表 5.4，リーチ有りの場合の結果を表 5.5 に示す．平均点数はゲーム終了時に持っていた点数の平均である．リーチ無しの場合では，1 位率，平均順位，平均点数において，大局的な状況に応じて打ち方を変更する AI は，打ち方を固定する AI の $\alpha = 0.9$ よりも悪い結果となってしまった．しかし，リーチ有りの場合では，これらの点において打ち方を固定する AI よりも良い結果となった．リーチをすることで，手牌に特定のパターンがなくても上がることができる．すなわち，他の役がなくてもリーチを宣言することで，役としてリーチがあるため上がることができる．リーチを組み込むことで上がりやすくなるため，α が小さいときはより早上がりになる．よって，リーチ有りはリーチ無しよりも早く上がれ，ゲームをより早く終了することができるため，勝

ち逃げをしやすい．このことから，リーチをしていないときよりも打ち方を変更することが有効に働いたと考えられる．これらの結果から，大局的な状況に応じて打ち方の変更を行う麻雀 AI が有用であると言える．

麻雀は運の要素があり毎回良い手が入るとは限らず，降りることも重要であるため，単純な降りを含めた場合の対戦実験を行う．リーチをしたプレイヤは他プレイヤが捨てた牌で上がる場合，自分が捨てた牌では上がることができない．また，リーチ後に捨てられリーチをしたプレイヤが上がらなかった牌は，上がり牌ではないということがわかる．これらの牌は安全牌と呼ばれる．他プレイヤがリーチをした場合に降りを行い，安全牌がある場合はそれを捨て，ない場合は提案手法で捨てる牌を決める．4 つの AI すべてがこの単純な降りを行う．

結果を表 5.6 に示す．大局的な状況に応じて打ち方を変更する AI の，1 位率や平均順位が表 5.5 の場合より上がり，降りの効果が見られる．しかし，平均順位や平均点数が打ち方を固定する AI の $\alpha = 0.0$ よりも悪い結果となってしまった．他プレイヤがリーチをした場合に降りを行っているため，早上がりを行う AI が有利になってしまったためと考えられる．よって，単純な降りではなく状況に応じた降りを行うのが重要であると考えられる．

表 5.1: ゲーム回数 100 回における 1 回の捨て牌選択の計算時間（秒）の平均の比較

残り巡数	モンテカルロ法	P のみ（小松らの手法）	U のみ	$\alpha = 0.5$
17	0.27589	0.11343	0.09934	0.12034
16	0.25882	0.09449	0.08379	0.10170
15	0.24190	0.07623	0.06845	0.08258
14	0.22525	0.06009	0.05412	0.06429
13	0.20935	0.04771	0.04403	0.05210
12	0.19363	0.03806	0.03495	0.04172
11	0.17849	0.03028	0.02870	0.03345
10	0.16327	0.02437	0.02277	0.02627
9	0.14810	0.01911	0.01802	0.02047
8	0.13257	0.01578	0.01480	0.01664
7	0.11731	0.01256	0.01211	0.01343
6	0.10216	0.01065	0.01010	0.01091
5	0.08682	0.00909	0.00863	0.00921
4	0.07144	0.00783	0.00749	0.00788
3	0.05465	0.00661	0.00631	0.00668
2	0.03801	0.00584	0.00550	0.00585
1	0.01974	0.00523	0.00489	0.00514
0	0.00034	0.00468	0.00421	0.00442

表 5.2: 10000 回の一人麻雀に対するモンテカルロ法，小松らの手法，打ち方の変更が可能な手法による結果の比較

	上がり率（%）	平均点数	平均上がり得点	平均上がり巡数
モンテカルロ法	7.0	352	5016	14.50
$\alpha = 0.0$	17.5	693	3958	14.02
$\alpha = 0.1$	17.0	829	4877	**13.95**
$\alpha = 0.2$	17.3	865	4999	14.02
$\alpha = 0.3$	16.9	863	5112	14.09
$\alpha = 0.4$	16.3	851	5210	14.14
$\alpha = 0.5$	15.7	850	5410	14.21
$\alpha = 0.6$	15.7	904	5776	14.36
$\alpha = 0.7$	14.7	905	6160	14.39
$\alpha = 0.8$	14.0	911	6510	14.58
$\alpha = 0.9$	13.2	**932**	7078	14.68
$\alpha = 1.0$（小松らの手法）	11.3	904	**8022**	14.90

表 5.3: 1000 回の半荘に対する打ち方の変更が可能な手法の α 変更の結果

	平均点数	平均上がり得点	平均上がり巡数
$\alpha = 0.0$	65624	7373	12.83
$\alpha = 0.1$	66034	7872	**12.74**
$\alpha = 0.2$	65131	7784	12.79
$\alpha = 0.3$	66289	7962	12.88
$\alpha = 0.4$	66209	8017	12.82
$\alpha = 0.5$	66001	8184	12.93
$\alpha = 0.6$	65926	8444	13.07
$\alpha = 0.7$	66373	8637	13.08
$\alpha = 0.8$	66124	8865	13.28
$\alpha = 0.9$	**66485**	9402	13.42
$\alpha = 1.0$ （小松らの手法）	63193	**9867**	13.81

表 5.4: 対戦実験の結果（リーチ無し）

	α 変更	$\alpha = 0.0$	$\alpha = 0.9$	$\alpha = 1.0$
1 位率 (%)	27.3	16.8	**32.5**	23.4
2 位率 (%)	24.9	25.8	24.9	24.4
3 位率 (%)	24.2	29.1	21.5	25.2
4 位率 (%)	23.6	28.3	21.1	27.0
平均順位	2.44	2.69	**2.31**	2.56
平均点数	25483	22705	**27344**	24468

表 5.5: 対戦実験の結果（リーチ有り）

	α 変更	$\alpha = 0.0$	$\alpha = 0.9$	$\alpha = 1.0$
1 位率 (%)	**28.2**	25.3	24.7	21.8
2 位率 (%)	25.2	25.8	26.3	22.7
3 位率 (%)	22.5	24.8	24.1	28.6
4 位率 (%)	24.1	24.1	24.9	26.9
平均順位	**2.43**	2.48	2.49	2.61
平均点数	**25998**	25813	25097	23093

表 5.6: 単純な降りを行う場合での対戦実験の結果

	α 変更	$\alpha = 0.0$	$\alpha = 0.9$	$\alpha = 1.0$
1 位率 (%)	**30.2**	30.1	22.6	17.1
2 位率 (%)	24.5	30.0	24.1	21.4
3 位率 (%)	21.8	22.5	27.6	28.1
4 位率 (%)	23.5	17.4	25.7	33.4
平均順位	2.39	**2.27**	2.56	2.78
平均点数	26752	**27894**	24647	20708

第6章

まとめ

本論文では，小松らの手法の報酬を変え，一つのパラメータを変更するだけで打ち方の変更が可能な手法を提案した．また，打ち方の変更が可能な手法を基に，大局的な状況に応じて打ち方の変更を行う麻雀AIを提案した．実験において，打ち方の変更が可能な手法が確かに打ち方を変えることができる手法であることを示した．対戦実験では，大局的な状況に応じて打ち方の変更を行う麻雀AIの有用性を示した．

現在，打ち方のパラメータの設定は，人手で決めた値となっている．良いパラメータ設定でないためか，リーチ無しの場合では大局的な状況に応じて打ち方を変更するAIは，打ち方を固定するAIよりも悪い結果となってしまった．今後の課題として，大局的な状況に応じた良いパラメータの自動設定を試みる．大局的な状況に応じて，パラメータを自動的に調整する関数を作成することを考えている．何回もゲームを行い，ゲームの結果から関数の調整をするような学習を行い，良い戦略を取れる関数を作成したい．また，今回の手法はリーチの選択が単純なアルゴリズムになっている．リーチをするか否かを，シミュレーションから決定する手法にしたいと考えている．麻雀において，自分の捨てる牌で他プレイヤが上がらないように行動選択すること，すなわち降りることは重要である．単純な降りを行う対戦実験では，大局的な状況に応じて打ち方を変更するAIは，打ち方を固定するAIよりも平均順位が悪い結果となった．よって，状況に応じた降りを行う手法や，他プレイヤが上がれそうな牌を考慮する手法を作成する必要がある．

参考文献

[1] 三木理斗. 多人数不完全情報ゲームにおける最適行動決定に関する研究. Master's thesis, 東京大学大学院工学系研究科, 2010.

[2] 北川竜平, 三輪誠, 近山隆. 麻雀の牌譜からの打ち手評価関数の学習. ゲームプログラミングワークショップ 2007 論文集, Vol. 2007, pp. 76–83, 2007.

[3] 水上直紀, 中張遼太郎, 浦晃, 三輪誠, 鶴岡慶雅, 近山隆. 多人数性を分割した教師付き学習による 4 人麻雀プログラムの実現. 情報処理学会論文誌, Vol. 55, No. 11, pp. 2410–2420, 2014.

[4] 田中悠, 池田心. 麻雀初級者のための状況に応じた着手モデル選択. 情報処理学会研究報告. ゲーム情報学（GI）, Vol. 2014-GI-31, No. 10, pp. 1–8, 2014.

[5] 水上直紀, 鶴岡慶雅. 期待最終順位に基づくコンピュータ麻雀プレイヤの構築. ゲームプログラミングワークショップ 2015 論文集, Vol. 2015, pp. 179–186, 2015.

[6] 水上直紀, 鶴岡慶雅. 強化学習を用いた効率的な和了を行う麻雀プレイヤ. ゲームプログラミングワークショップ 2016 論文集, Vol. 2016, pp. 81–88, 2016.

[7] 我妻敦, 田中哲朗. 半教師あり学習による麻雀の降り局面の判別. ゲームプログラミングワークショップ 2015 論文集, Vol. 2015, pp. 143–147, 2015.

[8] 我妻敦, 原田将旗, 森田一, 古宮嘉那子, 小谷善行. SVR を用いた麻雀における捨て牌の危険度の推定. 情報処理学会研究報告. ゲーム情報学（GI）, Vol. 2014-GI-31, No. 12, pp. 1–3, 2014.

[9] Naoki Mizukami and Yoshimasa Tsuruoka. Building a computer Mahjong player

based on Monte Carlo simulation and opponent models. In *2015 IEEE Conference on Computational Intelligence and Games (CIG)*, pp. 275–283. IEEE, 2015.

[10] 玉木久夫. 乱択アルゴリズム. 共立出版, 2008.

[11] 小松智希, 成澤和志, 篠原歩. 役を構成するゲームに対する効率的な行動決定アルゴリズムの提案. 情報処理学会研究報告. ゲーム情報学（GI）, Vol. 2012-GI-28, No. 8, pp. 1–8, 2012.

[12] 原田将旗, 古宮嘉那子, 小谷善行. 麻雀における手牌と残り牌からの上がり探索による着手決定アルゴリズム CHE. 情報処理学会研究報告. ゲーム情報学（GI）, Vol. 2014-GI-31, No. 13, pp. 1–4, 2014.

[13] 麻雀の役一覧. Wikipedia. `https://ja.wikipedia.org/wiki/`麻雀の役一覧.

[14] 麻雀の得点計算. Wikipedia. `http://ja.wikipedia.org/wiki/`麻雀の得点計算.

発表論文

- <u>海津 純平</u>, 成澤 和志, 篠原 歩. 一人麻雀における打ち方を考慮した評価指標に関する研究. ゲームプログラミングワークショップ 2015 論文集, Vol. 2015, pp. 172–178, 2015.
- <u>海津 純平</u>, 吉仲 亮, 篠原 歩. 大局的な状況に応じて打ち方の変更を行う麻雀 AI. ゲームプログラミングワークショップ 2016 論文集, Vol. 2016, pp. 154–157, 2016.

謝辞

素晴らしい研究の場を与えていただくとともに，多くのご教示ご鞭撻を賜りました東北大学大学院情報科学研究科 篠原 歩 教授，ならび吉仲 亮 准教授の両名に厚く御礼申し上げます.

また，本論文の副審査員を務めていただきました，東北大学大学院情報科学研究科 木下 哲男 教授，ならびに田中 和之 教授には，御専門の立場から的確なご助言や貴重なご意見を賜りましたことを，心より御礼申し上げます.

そして，日頃の研究室での活動において共に研究をした篠原・吉仲研究室の方々に感謝申し上げます.

最後に，この場を借りて，長い大学生活を温かく見守り，支えてくださった両親に深く感謝いたします．ありがとうございました.