Vol.2019-GI-41 No.26
2019/3/9

# コンピュータ大貧民における提出手の影響に関する研究

三石　亮[1]　大久保　誠也[2]　若月　光夫[1]　西野　哲朗[1]

**概要**：本研究の目的は，コンピュータ大貧民における，特定の相手に狙った影響を与える手法の検討である．そのため，モンテカルロ法を用いた手法を提案する．通常のモンテカルロ法は自らの得点期待値をもとにした評価値を参照するのに対し，提案アルゴリズムは対象のプレイヤの得点期待値をもとにした評価値を参照する．提案アルゴリズムは 3 つあり，それぞれ，対象のプレイヤの得点をできるかぎり下げるアルゴリズム，対象のプレイヤの得点をできるかぎり上げるアルゴリズム，自身の得点も考慮しつつ対象のプレイヤの得点を下げるアルゴリズムである．提案アルゴリズムの有効性を検証するため，2010 年 UEC コンピュータ大貧民大会優勝プログラムである snowl をベースとして各提案アルゴリズムを実装し，幾つかの計算機実験を行った．その結果，提案手法の有効性が示された．

## 1. はじめに

ゲーム情報学では，1 対 1 で行うゲームであるチェスや将棋，囲碁などの二人完全情報ゲームが多く研究されてきた．これらのゲームについては，現在ではコンピュータが人間のプロに勝つほど研究が進んでいる [1]．一方近年では，3 人以上で行う麻雀や人狼などの多人数不完全情報ゲームも注目を集めている．多くの多人数ゲームでは、他のプレイヤに直接影響を与えられるルールがある．例えば人狼では，ゲームから脱落させる人を投票によって決めるルールがある．そして，このようなルールを利用した戦略が存在する．例えば麻雀では，ポンやカンなど鳴きと呼ばれるルールが存在しており，特定のプレイヤの番を飛ばすなどの戦略が考えられる．

近年，コンピュータ大貧民の研究が行われている [2]．多くの研究では，UEC コンピュータ大貧民大会 [3] で使用されている UEC 標準ルールが用いられている．この UEC 標準ルールでは，他プレイヤに直接影響を与えるルールがほとんどない．そのため，特定のプレイヤに対して，狙った影響を与えることが可能であるか否かは明らかでない．また，特定のプレイヤに影響を与えることで試合を有利にする戦略も，ほとんど知られていない．したがって，コンピュータ大貧民においては，まず狙った影響を与えることが可能であるか否かを明らかにする必要がある．

本研究の目的は，コンピュータ大貧民における，特定のプレイヤに狙った影響を与える手法の検討である．特に，特定のプレイヤの得点を上げたり下げたりすることに注目する．また，試合を有利に進める戦略についても検討する．併せて，勝利する以外の目的に対する，モンテカルロ法の有効性も検証する．そこで，モンテカルロ法に基づくアルゴリズムを提案し，計算機実験によりその評価を行う．

[1] 電気通信大学大学院情報理工学研究科
[2] 静岡県立大学経営情報学部

## 2. コンピュータ大貧民

コンピュータ大貧民は，トランプゲームである大富豪もしくは大貧民を計算機上で行うゲームである．多人数不完全情報ゲームに分類され，3 人以上で行い, 相手の情報 (手札など) がすべて分かっていないことが特徴である．

コンピュータ大貧民については，UEC コンピュータ大貧民大会が 2006 年から電気通信大学で開催されている．人が直接プレイするのではなく，プレイするコンピュータプログラムを作成して持ち寄り，対戦させる大会である．プレイヤは 5 人で対戦し，各ゲームで上がった順に大富豪，富豪，平民，貧民，大貧民の階級が与えられ，各階級ごとに 5 点から 1 点の報酬値が与えられる．そして 1 試合数千ゲームを行い，合計得点で勝敗を競う．UEC 標準ルールは，UEC コンピュータ大貧民大会のために用意されたルールである．基本的なルールは，一般に遊ばれている大貧民とほぼ同じである．一方で，数多くあるローカルの多くを採用しておらず，特に特定の対象に対して影響を与えるルールがない．基本的な UEC 標準ルールは以下のようなものである．

**カード交換**

ゲーム開始前に，直前のゲームの順位をもとに (1 位から 5 位の順に) 大富豪・富豪・平民・貧民・大貧民の階級が与えられる．大富豪は大貧民から 2 枚のカー

ドをもらい，富豪は貧民から 1 枚のカードをもらう．その後，大富豪は 2 枚，富豪は 1 枚カードを選択し，カードを受け取ったプレイヤに渡す．大富豪・富豪のカードの選択は任意である．逆に，大貧民・貧民は強い順にカードを選択し，献上する．

**ゲームの開始**

ゲームはカード交換後，ダイヤの 3 を持っているプレイヤから開始される．そのプレイヤは必ずしもダイヤの 3 を出さなくてもよい．

**カードの出し方**

順番が回ってきたプレイヤは，カードを場に出すか，パスを選択することができる．場にカードがある場合は，場にあるカードと同じ役で同じ枚数のより強いカードを出すことができる．また役が階段の場合は，出すカードすべてが場にある階段のカードすべてより強くなければならない．場が空の場合は，好きな役のカードを出せる．

**場の流れ**

全員がパスをする，または特定のカードが出された場合に場が流れる．場が流れた時，最後にカードを出したプレイヤが，場が空の状態からカードを出すことができる．

**8 切り**

8 を含んだ手が提出されると，場が流れる．

**スペードの 3**

ジョーカーが単騎で場に出された場合，スペードの 3 を出すことができる．その場合，場が流れる．

**ジョーカー**

一般的なルールと同様，最強のカードである．ペアや階段ではどのカードにも代用することができ，代用したカードによる役が適用される．またジョーカーを単騎で出した場合，他のカードとして代用するのではなく，2 より強い単騎出しとなる．

## 3. モンテカルロ法

モンテカルロ法とは，乱数によるシミュレーションを複数回行うことで，近似的に解を求めるアルゴリズムのことである．特に，ゲーム情報学においては，各選択肢の評価値を決定するのに利用されている．素朴なモンテカルロ法は，以下のように行われる．

(1) ある局面において提出可能な手をすべて列挙する
(2) 1 の手の中から一つの手をランダムに選択する
(3) 選んだ手 $i$ を提出した次の局面から，乱数を用いて，ゲームを終局までシミュレートする
(4) 終局時の順位から手 $i$ の評価値 $\overline{X_i}$ を更新する
(5) 2-4 を複数回繰り返し，もっとも評価値の高い手を提出する

ここで，手 $i$ が選択された回数を $n_i$，終局時の順位から定まる報酬値を $V$，総報酬値を $X_i$ とすると，評価値 $\overline{X_i}$ は以下のように表される．

$$X_i \leftarrow X_i + V$$
$$n_i \leftarrow n_i + 1$$
$$\overline{X_i} \leftarrow X_i / n_i$$

また，限られた回数のシミュレーションでより良い結果を得るためには，自らが勝てそうな手を多くシミュレーションすることが有効である．どの手をシミュレーションするともっとも高い期待値が得られるかという問題は，多腕バンディット問題として定式化できる [5]．この問題を効率よく解くアルゴリズムが複数提案されている．UCB1[6] では，有望さとその評価の不確かさを組み合わせた評価基準である UCB 値を定義し，それが最大となる手を選ぶ．具体的には，合計のシミュレーション回数を $s$ とし，提出可能な手の集合を $A$，手 $i$ を選んだ場合の評価値を $\overline{X_i}$，その手が選択された回数を $n_i$ とすると

$$arg\max_{i \in A}\left(\overline{X_i} + \sqrt{\frac{2\log s}{n_i}}\right)$$

という手を選択する．

UCB1-Tuned[4] では UCB1 を少し改変し，分散も考慮している．

$$V_i = \overline{\sigma}_i^2 + \sqrt{\frac{2\log s}{n_i}}$$

として

$$arg\max_{i \in A}\left(\overline{X_i} + \sqrt{\frac{2\log s}{n_i} min(\frac{1}{4}, V_i)}\right)$$

の手を選択する手法である．

UEC コンピュータ大貧民大会に出場しているモンテカルロ法を用いたプログラムでは UCB1-Tuned 使用されていることが多い．代表的なプログラムとして，2010 年 UEC コンピュータ大貧民大会（UECda-2010）の優勝プログラムである snowl[7] がある．UECda-2010 では，部門が分かれていなかったが，無差別級に相当する計算時間が必要である．必勝手の探索を行い，見つかった場合はその手を提出するが，基本的にモンテカルロ法によるシミュレーションによって提出手を決定する．シミュレーションは数千回行い，すべてのシミュレーション結果から，自らの報酬値が最大となる手を選択する．シミュレーションの効率を上げるため，UCB1-tuned を用いている．以降の大会でこのプログラムをもとにしたプログラムが多数出場しており，また，多くの研究で用いられている．

## 4. 提案手法

モンテカルロ法のシミュレーション結果からは，各手に対する自らの評価値のみでなく，他のプレイヤの評価値も得られる．しかしながら，既存のモンテカルロ法を用いた

プログラムは，自らの得点を最大にするために，自らの評価値のみを参照して自らの評価値が最大である手を提出するため，他のプレイヤの評価値は全く考慮していない．

本研究では，他のプレイヤに影響を及ぼす手法の一つとして，他のプレイヤの得点を上げ下げすることを考える．そこで，モンテカルロ法のシミュレーション結果から他のプレイヤの評価値を参照し，その値が高い手，または低い手を提出するアルゴリズムを提案する．それにより，対象のプレイヤの得点に狙った影響を与える．

本研究では，3 つのアルゴリズムを提案する．一つ目は対象のプレイヤの点数をできるだけ下げるアルゴリズム，二つ目は対象のプレイヤの点数をできるだけ上げるアルゴリズム，三つ目は自らの点数も取りつつ，対象のプレイヤの点数を下げるアルゴリズムである．提案アルゴリズムは，モンテカルロ法を実行する際，対象プレイヤの順位に応じた値を $V$ とした評価値 $\overline{X_i'}$ も計算し，その値も考慮して提出手を選択する．各アルゴリズムの概要は以下のとおりである．

**提案アルゴリズム 1**

対象のプレイヤの得点をできるだけ下げるアルゴリズムである．その際，自らの得点は全く考慮しない．

(1) ある局面において提出可能な手をすべて列挙する
(2) 1 の手の中から一つの手をランダムに選択する
(3) 選んだ手 $i$ を提出した次の局面から，乱数を用いて，ゲームを終局までシミュレートする
(4) 終局時の各プログラムの順位から，2 で選択した手 $i$ の対象プログラムの評価値 $\overline{X_i'}$ を更新する
(5) 2-4 を複数回繰り返し，対象のプログラムの評価値の一番低い手を提出する

一般的なモンテカルロアルゴリズムに対して，自らの得点は下がり対象のプレイヤの得点も下がることが期待される．

**提案アルゴリズム 2**

対象のプレイヤの得点をできるだけ上げるアルゴリズムである．その際，自らの得点は全く考慮しない．

(1) ある局面において提出可能な手をすべて列挙する
(2) 1 の手の中から一つの手をランダムに選択する
(3) 選んだ手 $i$ を提出した次の局面から，乱数を用いて，ゲームを終局までシミュレートする
(4) 終局時の各プログラムの順位から，2 で選択した手 $i$ の対象プログラムの評価値 $\overline{X_i'}$ を更新する
(5) 2-4 を複数回繰り返し，対象のプログラムの評価値の一番高い手を提出する

一般的なモンテカルロアルゴリズムに対して，自らの得点は下がり対象のプレイヤの得点は上がることが期待される．

**提案アルゴリズム 3**

自らの点数も取りつつ，対象のプレイヤの点数を下げるアルゴリズムである．

(1) ある局面において提出可能な手をすべて列挙する
(2) 1 の手の中から一つの手をランダムに選択する
(3) 選んだ手 $i$ を提出した次の局面から，乱数を用いて，ゲームを終局までシミュレートする
(4) 終局時の各プログラムの順位から，2 で選択した手 $i$ の自身の評価値 $\overline{X_i}$ と，対象プログラムの評価値 $\overline{X_i'}$ を更新する
(5) 2-4 を複数回繰り返し，自身の評価値の高い手 2 つのうち，対象のプログラムの評価値が低い手を提出する

対象のプレイヤの評価値が低い手を選択する際，選択肢が 2 つのみである．そのため，提案アルゴリズム 1 と比較すると対象のプレイヤの得点は高いが，一般的なモンテカルロアルゴリズムと比較すると低いと推測される．また，自身の得点も考慮しているため，提案アルゴリズム 1 と比較すると高い得点が期待される．しかし，一般的なモンテカルロアルゴリズムと比較すると，最善手を必ず提出するとは限らないため，得点は低いと推測される．

## 5. 計算機実験

### 5.1 計算機実験の概要

提案手法の有効性を計算機実験により評価した．snowl に提案アルゴリズム 1 を実装した snowl-i，提案アルゴリズム 2 を実装した snowl-s，提案アルゴリズム 3 を実装した jsnowl を作成した．そして，これらのプログラムを他のプログラムと対戦させることで，もとの snowl との比較を行った．各実験は，評価プログラムを snowl，snowl-i，snowl-s，jsnowl とし，評価プログラムとある 1 つの対戦プログラム 4 機の組み合わせで対戦を行った．ここで対戦プログラムの組み合わせによって結果が異なる可能性があるため，対戦プログラムとして default，Kou2，snowl，Blauweregen を使用した．表 1 に対戦カードを示す．default はもっとも基本的な動作をするため，Kou2，Blauweregen(以後，Blau と略す) はそれぞれライト級と無差別級で優勝し，現状もっとも強いプログラムの一つであるため，対戦相手として選択した．対戦プログラムの強さは，Blau，snowl，Kou2，default の順に強いと知られている．またカード交換はあり，席替えは 3 ゲームごと，階級リセットは 100 ゲームごととした．

### 5.2 snowl の評価

比較対象となる snowl を用いたときの各プログラムの得点を，計算機実験により求めた．以後ベース実験と呼ぶ．ゲーム数は対戦 1 から対戦 3 までは 3000 ゲームを 1 セットとし 100 セット，対戦 4 では 3000 ゲームを 1 セットとし 50 セット行った．

表 2 に各対戦における平均得点を示す．ここで，P1 から P4 は各プログラムを示す．snowl の得点は default や Kou2 よりも高く，snowl 同士の対戦ではほぼ同じであり，

表 1　計算機実験の対戦カード

| | P1 | P2 | P3 | P4 | P5 |
|---|---|---|---|---|---|
| 対戦 1 | default | default | default | default | 評価プログラム |
| 対戦 2 | Kou2 | Kou2 | Kou2 | Kou2 | 評価プログラム |
| 対戦 3 | snowl | snowl | snowl | snowl | 評価プログラム |
| 対戦 4 | Blau | Blau | Blau | Blau | 評価プログラム |

表 2　ベース実験の各プログラムの平均得点

| | P1 | P2 | P3 | P4 | snowl |
|---|---|---|---|---|---|
| 対戦 1(default) | 7954 | 7942 | 7950 | 7943 | 13208 |
| 対戦 2(Kou2) | 8930 | 8927 | 8890 | 8959 | 9292 |
| 対戦 3(snowl) | 8994 | 8984 | 9023 | 8984 | 9012 |
| 対戦 4(Blau) | 9406 | 9447 | 9433 | 9373 | 7338 |

表 3　ベース実験の平均得点の P1 との差

| | P2-P1 | P3-P1 | P4-P1 | snowl-P1 |
|---|---|---|---|---|
| 対戦 1(default) | -12 | -4 | -11 | 5254 |
| 対戦 2(Kou2) | -3 | -40 | 29 | 362 |
| 対戦 3(snowl) | -10 | 29 | -10 | 18 |
| 対戦 4(Blau) | 41 | 27 | -33 | -2068 |

表 4　ベース実験の平均得点の P1 との比率

| | P2/P1 | P3/P1 | P4/P1 | snowl/P1 |
|---|---|---|---|---|
| 対戦 1(default) | 0.998 | 0.999 | 0.999 | 1.661 |
| 対戦 2(Kou2) | 1.000 | 0.996 | 1.003 | 1.041 |
| 対戦 3(snowl) | 0.999 | 1.003 | 0.999 | 1.002 |
| 対戦 4(Blau) | 1.004 | 1.003 | 0.996 | 0.780 |

Blau よりは低いことが分かった．

表 3 に各プログラムの平均点と P1 の平均点の差，表 4 に各プログラムの平均点と P1 の平均点との比率を示す．P2 から P4 の各プレイヤの得点と P1 の得点の差は ±50 点程度に収まっており，各比率もおおよそ 1 であるため，P1〜P4 の得点に大きな違いは無いことがわかる．このことから，snowl は特定のプレイヤにのみに大きな影響を与えることはないことがわかる．

### 5.3　snowl-i の評価

提案アルゴリズム 1 を実装した snowl-i を用いたときの各プログラムの得点を，計算機実験により求めた．以後，計算機実験 1 と呼ぶ．ここで snowl-i は P1 を対象のプレイヤとしてプレイする．ゲーム数は対戦 1 から対戦 3 までは 3000 ゲームを 1 セットとし 100 セット，対戦 4 では 3000 ゲームを 1 セットとし 50 セット行った．

本実験では，対象のプレイヤとその他のプレイヤに得点の差があるか否か比較することにより，提案アルゴリズム 1 が狙った影響を及ぼせているか否かを検証する．具体的には，対象のプレイヤ P1 が他のプレイヤ P2〜P4 の得点より低いか否かを検証する．もとの snowl は特定のプレイヤのみに影響を与えるプレイを行わないため，P1 と P2〜P4 の差が，提案アルゴリズムによって与えられた影響となる．

表 5　計算機実験 1 の各プログラムの平均得点

| | P1 | P2 | P3 | P4 | snowl-i |
|---|---|---|---|---|---|
| 対戦 1(default) | 7392 | 9013 | 8986 | 9023 | 10584 |
| 対戦 2(Kou2) | 8537 | 10026 | 10022 | 10032 | 6380 |
| 対戦 3(snowl) | 8798 | 10096 | 10076 | 10078 | 5949 |
| 対戦 4(Blau) | 9021 | 10242 | 10256 | 10208 | 5270 |

表 6　計算機実験 1 の平均得点の P1 との差

| | P2-P1 | P3-P1 | P4-P1 | (snowl-i)-P1 |
|---|---|---|---|---|
| 対戦 1(default) | 1621 | 1594 | 1631 | 3192 |
| 対戦 2(Kou2) | 1489 | 1485 | 1495 | -2157 |
| 対戦 3(snowl) | 1298 | 1278 | 1280 | -2849 |
| 対戦 4(Blau) | 1221 | 1235 | 1187 | -3751 |

表 7　計算機実験 1 の平均得点の P1 との比率

| | P2/P1 | P3/P1 | P4/P1 | snowl-i/P1 |
|---|---|---|---|---|
| 対戦 1(default) | 1.219 | 1.216 | 1.221 | 1.432 |
| 対戦 2(Kou2) | 1.174 | 1.174 | 1.175 | 0.747 |
| 対戦 3(snowl) | 1.148 | 1.145 | 1.145 | 0.676 |
| 対戦 4(Blau) | 1.135 | 1.137 | 1.132 | 0.584 |

表 5 に各対戦における平均得点を示す．snowl-i の得点は P1 から P4 すべての default よりも高く，Kou2，snowl，Blau よりは低いことが分かった．

表 6 に各プログラムの平均点と P1 の平均点の差，表 7 に各プログラムの平均点と P1 の平均点との比率を示す．P2〜P4 の各プレイヤの得点と P1 の得点の差は 1200 点から 1700 点あり，各比率は 1.13 から 1.22 という結果である．このことから，snowl-i は対象のプレイヤの得点を下げており，ある特定のプレイヤに対して狙った影響を与えていることがわかる．また P2〜P4 の各プレイヤの得点と P1 の得点の比率では，対戦 1(default), 対戦 2(Kou2), 対戦 3(snowl), 対戦 4(Blau) の順に比率が低くなるという結果になった．

### 5.4　snowl-s の評価

提案アルゴリズム 2 を実装した snowl-s を用いたときの各プログラムの得点を，計算機実験により求めた．以後，計算機実験 2 と呼ぶ．ここで snowl-s は P1 を対象のプレイヤとしてプレイする．ゲーム数は対戦 1 から対戦 3 までは 3000 ゲームを 1 セットとし 100 セット，対戦 4 では 3000 ゲームを 1 セットとし 30 セット行った．

本実験では，対象のプレイヤとその他のプレイヤに得点の差があるか否か比較することにより，提案アルゴリズム 2 が狙った影響を及ぼせているか否かを検証する．具体的には，対象のプレイヤ P1 が他のプレイヤ P2〜P4 の得点より高いか否かを検証する．もとの snowl は特定のプレイヤのみに影響を与えるプレイを行わないため，P1 と P2〜

表 8　計算機実験 2 の各プログラムの平均得点

| | P1 | P2 | P3 | P4 | snowl-s |
|---|---|---|---|---|---|
| 対戦 1(default) | 11284 | 9628 | 9639 | 9608 | 4840 |
| 対戦 2(Kou2) | 11043 | 10044 | 10044 | 10019 | 3850 |
| 対戦 3(snowl) | 11065 | 10118 | 10139 | 10130 | 3548 |
| 対戦 4(Blau) | 10964 | 10144 | 10108 | 10161 | 3623 |

表 9　計算機実験 2 の平均得点の P1 との差

| | P2-P1 | P3-P1 | P4-P1 | (snowl-s)-P1 |
|---|---|---|---|---|
| 対戦 1(default) | -1656 | -1645 | -1676 | -6444 |
| 対戦 2(Kou2) | -999 | -999 | -1024 | -7193 |
| 対戦 3(snowl) | -947 | -926 | -935 | -7517 |
| 対戦 4(Blau) | -820 | -856 | -803 | -7341 |

表 10　計算機実験 2 の平均得点の P1 との比率

| | P2/P1 | P3/P1 | P4/P1 | snowl-s/P1 |
|---|---|---|---|---|
| 対戦 1(default) | 0.853 | 0.854 | 0.851 | 0.429 |
| 対戦 2(Kou2) | 0.910 | 0.910 | 0.907 | 0.349 |
| 対戦 3(snowl) | 0.914 | 0.916 | 0.915 | 0.321 |
| 対戦 4(Blau) | 0.925 | 0.922 | 0.927 | 0.330 |

P4 の差が，提案アルゴリズムによって与えられた影響となる．

表 8 に各対戦における平均得点を示す．snowl-s の得点は P1 から P4 すべての default，Kou2，snowl，Blau よりは低いことが分かった．

表 9 に各プログラムの平均点と P1 の平均点の差，表 10 に各プログラムの平均点と P1 の平均点との比率を示す．P2〜P4 の各プレイヤの得点と P1 の得点の差は 800 点から 1700 点あり，各比率は 0.85 から 0.93 という結果である．このことから，snowl-i は対象のプレイヤの得点を上げており，ある特定のプレイヤに対して狙った影響を与えていることがわかる．また P2〜P4 の各プレイヤの得点と P1 の得点の比率では，対戦 1(default), 対戦 2(Kou2), 対戦 3(snowl), 対戦 4(Blau) の順に比率が高くなるという結果になった．

### 5.5　jsnowl の評価

提案アルゴリズム 3 を実装した jsnowl を用いたときの各プログラムの得点を，計算機実験により求めた．以後，計算機実験 3 と呼ぶ．ここで jsnowl は P1 を対象のプレイヤとしてプレイする．ゲーム数は対戦 1 から対戦 3 までは 3000 ゲームを 1 セットとし 100 セット，対戦 4 では 3000 ゲームを 1 セットとし 50 セット行った．

本実験では，対象のプレイヤとその他のプレイヤに得点の差があるか否か比較することにより，提案アルゴリズム 3 が狙った影響を及ぼせているか否かを検証する．具体的には，対象のプレイヤ P1 が他のプレイヤ P2〜P4 の得点より低いか否かを検証する．もとの snowl は特定のプレイヤのみに影響を与えるプレイを行わないため，P1 と P2〜P4 の差が，提案アルゴリズムによって与えられた影響となる．

表 11　計算機実験 3 の各プログラムの平均得点

| | P1 | P2 | P3 | P4 | jsnowl |
|---|---|---|---|---|---|
| 対戦 1(default) | 7639 | 8379 | 8373 | 8393 | 12214 |
| 対戦 2(Kou2) | 8684 | 9452 | 9422 | 9427 | 8012 |
| 対戦 3(snowl) | 8855 | 9535 | 9523 | 9538 | 7548 |
| 対戦 4(Blau) | 9195 | 9782 | 9805 | 9804 | 6411 |

表 12　計算機実験 3 の平均得点の P1 との差

| | P2-P1 | P3-P1 | P4-P1 | jsnowl-P1 |
|---|---|---|---|---|
| 対戦 1(default) | 740 | 734 | 754 | 4575 |
| 対戦 2(Kou2) | 768 | 738 | 743 | -672 |
| 対戦 3(snowl) | 680 | 668 | 683 | -1307 |
| 対戦 4(Blau) | 587 | 610 | 609 | -2784 |

表 13　計算機実験 3 の平均得点の P1 との比率

| | P2/P1 | P3/P1 | P4/P1 | jsnowl/P1 |
|---|---|---|---|---|
| 対戦 1(default) | 1.097 | 1.096 | 1.099 | 1.599 |
| 対戦 2(Kou2) | 1.088 | 1.085 | 1.086 | 0.923 |
| 対戦 3(snowl) | 1.077 | 1.075 | 1.077 | 0.852 |
| 対戦 4(Blau) | 1.064 | 1.066 | 1.066 | 0.697 |

表 11 に各対戦における平均得点を示す．jsnowl の得点は P1 から P4 すべての default よりも高く，Kou2，snowl，Blau よりは低いことが分かった．

表 12 に各プログラムの平均点と P1 の平均点の差，表 13 に各プログラムの平均点と P1 の平均点との比率を示す．P2〜P4 の各プレイヤの得点と P1 の得点の差は 580 点から 770 点あり，各比率は 1.06 から 1.1 という結果である．このことから，jsnowl は対象のプレイヤの得点を下げており，ある特定のプレイヤに対して狙った影響を与えていることがわかる．また P2〜P4 の各プレイヤの得点と P1 の得点の比率では，対戦 1(default), 対戦 2(Kou2), 対戦 3(snowl), 対戦 4(Blau) の順に比率が低くなるという結果になった．

## 6.　考察

計算機実験 1〜計算機実験 3 を通して，提案アルゴリズムの期待通りの狙った影響が与えられ，提案アルゴリズムの有効性が示された．同時に提案アルゴリズムはモンテカルロ法を用いているため，モンテカルロ法がコンピュータ大貧民において，狙った影響を及ぼす手法として有用なことがわかった．これは，他の多人数ゲームにも応用できる可能性がある．

またすべての実験で対戦相手が強いほど，影響が及ぼしづらいという結果になった．自身が貧民や大貧民など手札が弱い場合には影響が及ぼしづらいと考えられる．そのため，対戦相手が強い場合は貧民や大貧民になりやすく，影

表 14　ベース実験，計算機実験 1，計算機実験 3 の snowl，snowl-i，jsnowl の平均得点比較

| | snowl | snowl-i | jsnowl |
|---|---|---|---|
| 対戦 1(default) | 13208 | 10584 | 12214 |
| 対戦 2(Kou2) | 9292 | 6380 | 8012 |
| 対戦 3(snowl) | 9012 | 5949 | 7548 |
| 対戦 4(Blau) | 7338 | 5270 | 6411 |

表 15　ベース実験，計算機実験 1，計算機実験 3 の P1 の平均得点比較

| | ベース実験 | 計算機実験 1 | 計算機実験 3 |
|---|---|---|---|
| 対戦 1(default) | 7954 | 7392 | 7639 |
| 対戦 2(Kou2) | 8930 | 8537 | 8684 |
| 対戦 3(snowl) | 8994 | 8798 | 8855 |
| 対戦 4(Blau) | 9406 | 9021 | 9195 |

表 16　ベース実験，計算機実験 1，計算機実験 3 の P1 と P2 の平均得点の差と比率の比較

| | 差 | | | 比率 | | |
|---|---|---|---|---|---|---|
| 計算機実験 | ベース | 1 | 3 | ベース | 1 | 3 |
| 1(default) | -12 | 1621 | 740 | 0.998 | 1.219 | 1.097 |
| 2(Kou2) | -3 | 1489 | 768 | 1.000 | 1.174 | 1.088 |
| 3(snowl) | -10 | 1298 | 680 | 0.999 | 1.148 | 1.077 |
| 4(Blau) | 41 | 1221 | 587 | 1.004 | 1.135 | 1.064 |

表 17　ベース実験，計算機実験 1，計算機実験 3 の P1 と snowl，snowl-i，jsnowl の平均得点の差と比率の比較

| | 差 | | | 比率 | | |
|---|---|---|---|---|---|---|
| 計算機実験 | ベース | 1 | 3 | ベース | 1 | 3 |
| 1(default) | 5254 | 3192 | 4575 | 1.661 | 1.432 | 1.599 |
| 2(Kou2) | 362 | -2157 | -672 | 1.041 | 0.747 | 0.923 |
| 3(snowl) | 18 | -2849 | -1307 | 1.002 | 0.676 | 0.852 |
| 4(Blau) | -2068 | -3751 | -2784 | 0.780 | 0.584 | 0.697 |

響が及ぼしづらいという結果になったと推測される．

次に評価プログラムの得点と対象のプレイヤ，また他のプレイヤの得点の関連性を模索する．対象のプレイヤの得点を下げるプログラムの比較として，計算機実験 1 と計算機実験 3 とベース実験の比較を行う．表 14 にベース実験の snowl，計算機実験 1 の snowl-i，計算機実験 3 の jsnowl の平均得点を示す．jsnowl は snowl よりも低く，snowl-i よりも高い得点を取っていることがわかる．表 15 にベース実験，計算機実験 1，計算機実験 3 の P1 の平均得点を示す．計算機実験 3 の P1 はベース実験の P1 よりも低く，計算機実験 1 の P1 よりも高い得点を取っていることがわかる．表 16 にベース実験，計算機実験 1，計算機実験 3 の P1 と P2 の差と比率を示す．差は “各実験の P2 の平均得点” – “各実験の P1 の平均得点”を表し，比率は “各実験の P2 の平均得点” / “各実験の P1 の平均得点”を表す．計算機実験 3 はベース実験に比べ差及び比率が大きく，計算機実験 1 に比べ，差及び比率が小さいことがわかる．以上のことから jsnowl は snowl-i のように自身を犠牲にしてできるだけ対象の得点を下げるのではなく，ある程度自身の得点を高い水準で保ったまま対象の得点を下げていることがわかる．表 17 に各実験の snowl，snowl-i，jsnowl と P1 の差と比率を示す．差は “各実験の snowl，snowl-i，jsnowl の平均得点” – “各実験の P1 の平均得点”を表し，比率は “各実験の snowl，snowl-i，jsnowl の平均得点” / “各実験の P1 の平均得点”を表す．表の比率を比較すると，jsnowl は snowl-i に比べ P1 に対して得点を多く獲得しているが，snowl と比較すると，P1 に対して得点を多く獲得できていないことがわかる．

これらのことから，snowl は自らの点数は高いが影響は及ぼさないプログラム，snowl-i は自らの点数は低いが大きな影響を及ぼすプログラム，また jsnowl はある程度自身の得点を高い水準で保ったまま影響を及ぼすプログラムであることがわかる．このことから，モンテカルロアルゴリズムを改変することで，強さと及ぼす影響の度合いを調整できることがわかる．また調整次第では，snowl より試合を有利に進められる可能性がある．

## 7. おわりに

本研究では，コンピュータ大貧民において，特定のプレイヤに狙った影響が及ぼせるか否かについて検証を行った．モンテカルロ法を用いたアルゴリズムを 3 つ提案するとともに，それらをプログラムに実装し，計算機実験により評価した．その結果，提案手法の有効性が示された．

今後の課題としては，狙った影響を与えることで試合を有利に進めることができるか否か，またそれらは戦略として実現可能なのか，検証することがあげられる．

### 参考文献

[1] 人間対人工知能 (AI) 将棋，囲碁，チェス，オセロ．https://eco-notes.com/1418/daihinmin/2018/
[2] Mitsuo Wakatsuki, Yasuki Dobashi, Tasuku Mitsuishi, Seiya Okubo and Tetsuro Nishino: Strengthening methods of computer Daihinmin programs, Proceedings of the CAINE 2017, ISCA, pp.229-236, 2017.10.
[3] UECda-2018 コンピュータ大貧民大会. http://www.tnlab.inf.uec.ac.jp/daihinmin/2018/
[4] Volodymyr Kuleshov and Doina Precup. Algorithms for multi-armed bandit problems, arXiv preprint arXiv:1402.6028, 2014.
[5] 今川孝久，金子知適，多腕バンディットアルゴリズムの mcts への応用と性能の分析，ゲームプログラミングワークショップ 2014 論文集，Vol. 2014，pp. 145-150，2014.
[6] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. Machine learning, Vol. 47, No. 2-3, pp. 235-256, 2002.
[7] 須藤 郁弥，成澤 和志，篠原 歩，UEC コンピュータ大貧民大会向けクライアント 「snowl」 の開発，第 2 回 UEC コンピュータ大貧民シンポジウム講演予稿集，電気通信大学 (2010)