二人一般化七並べの必勝判定 *

木谷 裕紀 †1 末續 鴻輝 ‡2

1 名古屋大学大学院情報学研究科数理情報学専攻
2 国立情報学研究所

概要

七並べは日本で遊ばれるトランプカードゲームの中でも認知度, 人気が高い遊びである. 本研究では二人で行う札の枚数を一般化した七並べについて, その解析を行う. まず, オールマイティ札と呼ばれる特殊札を含めてもパス七並べが線形時間で必勝戦略保持者を計算することができることを示す. また, 七並べに関しても線形時間で必勝戦略保持者を計算できることを示す.

1 はじめに

七並べはカードを用いて行う多人数不完全情報ゲームであり, 日本全国で人気が高い遊びである他, Fan-Tan や Sevens など海外にも多くの類似の遊びが存在する. このゲームは多人数不完全情報ゲームであり, 一般に必勝戦略は存在しない. 近年, 将棋やオセロなどの完全情報ゲームに関する研究が AI 分野, 解析分野共に進んでいるとともに, このような不完全情報ゲームについても解析の試みが進められている [1, 2].

本研究では二人で行う札の枚数を一般化した七並べについて組合せゲーム理論をベースにした解析を行う. 組合せゲーム理論はニムと呼ばれる石取りゲームをはじめとする部分局面に分解できるゲームにおける必勝戦略保持者や有用な戦略を導くために培われた学問体系の一つで近年は囲碁やチェスの終盤局面の解析に用いられている [3, 5, 6]. 本研究では, 最初に先にパスをしたプレイヤが負けという亜種のゲームである「パス七並べ」を定義し, その必勝戦略保持者が札の総枚数の線形時間で得られることを示す. また, オールマイティ札と呼ばれる特殊札を含めても線形時間で必勝戦略保持者を計算することができることを示す. また一般の七並べゲームについても, 必勝戦略保持者やその必勝戦略がその札の総枚数の線形時間で得られることを示す.

2 定義

まず本研究の基礎となる k スート m 枚で行う二人七並べを以下のようにモデル化する: 組合せゲーム理論の慣習に従い, ゲームを行う二人のプレイヤを片方を左あるいは**左プレイヤ**, もう片方のプレイヤを右あるいは**右プレイヤ**と呼ぶ. 各札にはスートがついており, それぞれの札は値とスートを持つ (一般に行われる七並べはスートの長さが 6, スート数が 8 である. スート数が 4 ではなく 8 であるのは, 各スートの 7 より大きい部分と 7 より小さい部分を独立したものとしてみなすためである. また, スートの長さが 6 であるのはそれぞれの 1 から 6 の札と 8 から K の札に対応する).

プレイヤに配布された札 (**手札**と呼ぶ) の札数は必ずしも等しくなくてよい. 札の番号は札を出すことができる場所に対応している. この設定の下, 以下の形でゲームを進める.

二人七並べのルール

- 先手後手を決め, **先手プレイヤ**, **後手プレイヤ**の順に交代で, 手札から**場**に 1 枚ずつ札を出していく. 各プレイヤが札を出すことのできる機会のことを**手番**と呼ぶ.
- 場は最初, 各スートに 0 の札のみ置かれている.
- 手番のプレイヤは, 自身の手札の中から場に出ている札のと同じスートの札である札の中から場に出ている札の値よりも真に 1 のみ大きい値の札を 1 枚出すことができる. このとき, 出した札はそれまで出ていた札の上に置かれる (場に出ている札は今出した札に代わる). この場合, 手番はもう一人のプレイヤに移る.
- いずれかのプレイヤの場に出すことができる札がなくなった時点で終了であり, 手札が 0 枚となり場に出す札がなくなった場合は手札が 0 枚にで場に出す札がなくなったプレイヤを, 手札が 1 枚でも存在する場合は, 直前のプレイヤつまり最後に手札を出したプレイヤが勝ちである.

このゲームは一般に遊ばれている七並べを二人ゲームにし, 「ゲーム中の脱落」として多く採用されている各プレイヤの n 回目のパスの宣言時にゲームを脱落するというルールを 0 回に限定したものである. また, 本研究では七

* 本研究は JSPS 科研費 20J14990 の助成を受けたものである.
† kiya.hironori@f.mbox.nagoya-u.ac.jp
‡ suetsugu.koki@gmail.com

並べの簡易化したパス七並べの勝敗についても勝敗を扱うが, このパス七並べは上記の七並べルールと異なる点として勝敗の決め方がある; 通常の七並べルールでは, ゲーム中に脱落することなく, 手札の最後の札を出し切ったプレイヤが勝ちというルールが一般的である. 本研究で扱うパス七並べのルールは通常の七並べから「最後の札を出し切ったプレイヤの勝ち」というルールを取り除いたものである.

また, 上述の通り, 一般の七並べにおいては一人当たりのパスの回数として 3 回や 5 回に限定しているものが多いが本研究で扱うゲームは 0 回に限定する. これは以下の定理より自然な拡張であるといえる.

定理 1. 二人パス七並べあるいは二人七並べの局面 G において以下の二つは必要十分条件である.

- パス 0 回制約ゲームにおいて G は左プレイヤ必勝である.
- パス p 回制約ゲームにおいて G は左プレイヤ必勝である.

この定理の証明に以下のチェルメロの定理を利用する.

命題 1. ゲームの長さが有限の引き分けのない二人完全情報ゲームのうち, 逐次手番で各プレイヤの選択に偶然の要素が影響しないゲームはいずれかのプレイヤに必勝戦略が存在する [8].

本研究で扱うパス七並べはチェルメロの定理の適用条件である「引き分けのないゲームである」, 「二人プレイヤゲームである」, 「完全情報ゲームである」, 「逐次手番ゲームである」「各プレイヤの行動に偶然の要素が影響しない」のすべての条件を満たしていることに留意されたい. このチェルメロの定理を認めた上で以下では証明を行う.

証明. まず, 十分性が成立することを示す.

[十分性の証明] パス 0 回制約ゲームにおいて左プレイヤ必勝である局面 G を考える. パス p 回制約ゲームにおいて, この局面をプレイするとき, 左プレイヤはパス 0 回制約ゲームにおける必勝戦略を用いる. また, 右プレイヤがパスをしてきたときは必ず左プレイヤもパスをする. このようにすることで, 右プレイヤはパス 0 回制約ゲームと同じ手順で勝つことができる.

次に必要性が成立することを示す.

[必要性の証明] 必要性を示すには, 「パス p 回制約ゲームで G が左プレイヤ必勝ならばパス 0 回制約ゲームにおいて G は左プレイヤ必勝である」を示せばよい. これを命題 P とする. この対偶 P' を考えると「パス 0 回制約ゲームで G が左プレイヤ必勝でないならばパス p 回制約ゲームにおいて G は左プレイヤ必勝でない」となる. またチェルメロの定理より, 左プレイヤ必勝でないすべてのゲームは右プレイヤ必勝であるので, 対偶 P' は「パス 0 回制約ゲームにおいて G' が右プレイヤ必勝であるならばパス p 回制約ゲームにおいても右プレイヤ必勝である」と同値である. この命題 P' は十分性とほとんど同一の議論により真であることが分かる. よって命題 P も真となる.

以上の議論より本定理は示された. □

本研究ではパス七並べ, 七並べにおける必勝戦略保持者を示す他に, パス七並べの亜種として, オールマイティ札が入ったパス七並べ関しても議論を行う. オールマイティ札は七並べにおける JOKER 札の役割の一つであり, 地域やコミュニティによっていくつかの異なったルールで遊ばれているが本研究では「任意のタイミングで場に出ている札の値よりも真に 1 のみ大きい値の札の代わりに出すことができる札 (ただし手番に含まれない)」とする. つまり, 任意のタイミングで自分の保有していない札の代わりに一枚出すことができる札となる. この札は他の札同様, 使用後は手札に残らないことに留意されたい (いくつかのルールでは使用後に相手プレイヤに渡るルールが存在するが今回は採用しない).

3 局面の値

七並べにおいて, プレイヤはいずれかのスートを選び一手着手を行う. 複数のスートに同時に着手することはできない. したがって, それぞれのスートは独立していると考えることができる. 本研究ではゲームを集合による定義をすることにより, このようなゲームにおいて有効である組合せゲーム理論を用いた解析を適用する.

定義 1.
- $G = \{|\}$ はゲームである.
- $G = \{G_1^L, G_2^L, \ldots, G_k^L | G_1^R, G_2^R, \ldots, G_j^R\}$ は (片方が空であっても) ゲームである.

以上の定義の G_k^L, G_j^R はそれぞれ左選択肢, 右選択肢とよばれ, 左選択肢は左が一手その局面に着手した直後の局面, 右選択肢は右が一手その局面に着手した直後の局面を表す. その他本研究に用いない詳細な定義は省略する. 詳しい定義は参考文献 [4, 7] を参照されたい.

本節ではこの定義の下, パス七並べの局面に値を与える. 組合せゲーム理論で使用する局面の値の定義のうち, パス七並べの局面で現れるもののみ以下では記載する.

定義 2.
- $0 = \{|\}$,
- $k = \{k-1|\}$ *(ただし k は正の整数),*
- $-k = \{|1-k\}$ *(ただし k は正の整数),*
- $0 = \{|k\}$ *(ただし k は正の整数),*
- $0 = \{-k|\}$ *(ただし k は正の整数).*

上記 $0, k, -k$ は局面の値が $0, k, -k$ である局面 (集合) を表す. この局面の値はパス七並べにおいて, この局面に与える値はどちらがどれだけパスをせずにその手を打つことができるかに対応しており, お互いが自身にとって最適な着手をした場合において, (ゲームの終了時までに) 局

面の値が正ならば左プレイヤが, 負ならば右プレイヤがより多く (パスをすることなく) 盤面に手を打つことができる.

3.1 パス七並べの局面の値に関する定義

ゲーム全体の局面の値は, それぞれのゲーム和で表せることが知られているため, 以下では単独のスートの値のみを考える. ゲームの局面を数と L, R の文字列を用いて表す. すなわち, 既に場に出されている札は数, 場に提出されていない札, つまりプレイヤの手札内にある札は, それぞれを持っているプレイヤに応じて左が持っているなら L, 右がもっているなら R で表す. つまり, 局面が $\underline{01LLRL}$ と表された場合, 既に場に出ている札が 0 と 1 であり, 左が持っている札が $2, 3, 5$, 右が持っている札が 4 となる. また, 任意の数を k, 任意の文字列を X, Y と表し, 各スートの局面の値をゲーム全体のすべてのスートに対して計算したその和をゲーム和と呼ぶ.

3.2 パス七並べにおける局面の値

以下では, パス七並べにおけるゲームの値に関する補題を述べる. 等号は組合せゲーム理論の意味での等号, すなわちゲームの値が等しいことを意味している.

補題 1. 以下の等式が成立する : $\underline{0XkY} = \underline{0XY}$,
特に, $\underline{01\cdots(k-1)kX} = \underline{0X}$.

補題 1 は場に置かれている札を無視して局面の値を計算してよいことを表している. この成立は明らかであるため, 証明は省略する. また補題 1 で起こるようなスートの途中に場の札が置かれている状況は本問題設定では起こりえないが, 3 人以上で行う七並べの終盤においてあるプレイヤが脱落することによ同一の局面が現れる可能性がある. また, 以下の補題 2 が成立する.

補題 2. 以下の二つの等式が成立する :

$$\underline{0LX} = \begin{cases} \underline{0X} + 1 & (\underline{0X} \geq 0), \\ 0 & (\underline{0X} < 0). \end{cases}$$

$$\underline{0RX} = \begin{cases} \underline{0X} - 1 & (\underline{0X} \leq 0), \\ 0 & (\underline{0X} > 0). \end{cases}$$

証明. $\underline{0LX} = \{\underline{0X} \mid\}, \underline{0RX} = \{\mid \underline{0X}\}$ より成立する. □

この証明を組合せゲーム理論の言葉を用いずに解釈すると次のようになる: 一文目の $\underline{0LX} = \{\underline{0X} \mid\}$ は盤面 $\underline{0LX}$ に対して, 左プレイヤが手を打つことによって盤面 $\underline{0X}$ となり, 盤面 $\underline{0LX}$ に対して, 右プレイヤは手を打つことができないことを表している. 二文目の $\underline{0RX} = \{\mid \underline{0X}\}$ は盤面 $\underline{0RX}$ に対して, 右プレイヤが手を打つことによって盤面 $\underline{0X}$ となり, 盤面 $\underline{0RX}$ に対して, 左プレイヤは手を打つことができないことを表している. また, この定理より次の系が直ちに導かれる.

系 1. $\underline{0LRRX} = \underline{0RLLX} = 0$

系 1 の重要な点は, パス七並べの局面が与えられたとき, 連続する二つのカードの手札を同じプレイヤが持っており, その前のカードが別のプレイヤの手札であれば, その三枚より大きい範囲においてはいずれのプレイヤが手札を持っているかに関わらず, ゲームの値を求めることができるという点である.

系 2.

$$\underline{0LL\cdots L} = k,$$
$$\underline{0RR\cdots R} = -k.$$

ただしそれぞれ, L, R は k 個連続しているとする.

系 3.

$$\underline{0LRLR\cdots LR} = \underline{0LRLR\cdots LRRX} = 0,$$
$$\underline{0LRLR\cdots LRL} = \underline{0LRLR\cdots LRLLX} = 1,$$
$$\underline{0RLRL\cdots RL} = \underline{0RLRL\cdots RLLX} = 0,$$
$$\underline{0RLR\cdots RLR} = \underline{0RLRL\cdots RLRRX} = -1.$$

以上の補題により, パス七並べの局面は簡単にゲームの値を求めることができるとわかる. 具体的には, 以下の Algorithm 1 を用いればよい. 文字列の順序は 0 に近いほうを前, 数字がより大きいほうを後とする. また, 補題 1 を用いることで以下では 0 以外の場に置かれている札を無視して考える. また, これらの補題より次の系を導くことができる.

系 4. 各文字列の最も左の L を数に変更したとき, そのスートの値は 1 以上減少する.

系 5. 各文字列の最も左の R を数に変更したとき, そのスートの値は 1 以上増加する.

これらの補題, 系, アルゴリズムを用いて 4 章では証明を行う.

4 パス七並べの解析

本節ではパス七並べとオールマイティ札が入ったその亜種についての必勝戦略保持者を決定する計算時間についての結果を述べる.

4.1 パス七並べの必勝戦略保持者判定

本節ではパス七並べの必勝戦略保持者判定の計算時間を議論する.

定理 2. パス七並べの必勝戦略保持者がどちらであるかという判定は札の総枚数に関して線形時間で行うことができる.

証明. 以下の Algorithm 2 を使うことによって線形時間で解くことができることを示す. このとき, Algorithm 2 は各札の参照を高々定数回しか行わないことを留意したい. したがって Algorithm 2 は線形時間で終了する. また, 続いて正当性を示す. 一般性を失うことなくゲーム

Algorithm 1 ゲームの値判定アルゴリズム

```
if 最も前の非数が L. then
  if k 個の L だけでスートが終わる. then
    return k
  else {k 個の L のあとに, 2 個以上 R が続く.}
    return k − 1
  else {k 個の L の後に R と L が交互に続く.}
    if RLRL···RLL となり, ここで終わるかさらに何か続く. then
      return k
    else {RLRL···RLRR となり, ここで終わるかさらに何か続く.}
      return k − 1
    else {RLRL···RL となり, 終わる.}
      return k
    else {RLRL···RLR となり, 終わる.}
      return k − 1
    end if
  end if
else { 最も前の非数が R.}
  if k 個の R だけで局面が終わる. then
    return −k
  else {k 個の R の後に 2 個以上 L が続く.}
    return −k + 1
  else {k 個の R が後に L と R が交互に続く.}
    if LRLR···LRR となり, ここで終わるかさらに何か続く. then
      return −k
    else {LRLR···LRLL となり, ここで終わるかさらに何か続く.}
      return −k + 1
    else {LRLR···LR となり, 終わる.}
      return −k
    else {LRLR···LRL となり, 終わる.}
      return −k + 1
    end if
  end if
else
  return 0
end if
```

Algorithm 2 パス七並べ判定アルゴリズム

1: Algorithm 1 を用いて各スートの値を求める.
2: 各ゲーム和を計算する.
3: ゲーム和が 0 より大きいなら左プレイヤ, 0 より小さいなら右プレイヤ, 0 なら後手プレイヤを勝利プレイヤとして出力する.

和が 0 より大きいとする. 左プレイヤは各手番において Algorithm 3 にしたがって手を打つことによって必ず勝てることを示す. このとき, 正のスートに対して手を打つとスートは必ず 1 減少し, 相手プレイヤの手によって相手プレイヤがパスをしないのであればゲーム内のいずれかのスートは 1 以上必ず増加する. このとき, 各ゲーム和は 0 より大きい状態が維持される. 全てのスートの値が 0 より大きくなったとき, 定義より右プレイヤはどのスートに

Algorithm 3 パス七並べにおける左プレイヤの戦略

1: Algorithm 1 を用いて各スートの値を求める.
2: 値が正である任意のスートに対し, 手を打つ. 操作 1 に戻る.

も着手できない. したがってこのとき, 左プレイヤ必勝である. 同様の議論でゲーム和が 0 以下の場合も示すことができる. 以上より本定理は示された. □

4.2 オールマイティ札 1 枚のパス七並べの必勝戦略保持者判定

本節では以下の定理を記述する.

定理 3. パス七並べにおいて, 札の総枚数を n とし, どちらかのプレイヤが 1 枚のみオールマイティ札を手札に保持しているとする. このとき, 必勝戦略保持者がどちらであるかという判定は n に関して線形時間で行うことができる.

証明. 一般性を失うことなく左プレイヤがオールマイティ札を持つとする. このとき, 以下の Algorithm 4 を使うことによって線形時間で解くことができることを示す.

Algorithm 4 オールマイティ札有りパス七並べ判定アルゴリズム

1: Algorithm 1 を用いて各スートの値を求める.
2: 各スートに対し, 右プレイヤの持つ一番数字の小さい札 (文字列のうち一番最初に出てきた R)1 枚を削除した場合の値を Algorithm 1 を用いて求める.
3: 各スートにおいて, 操作 1 で求めた値と操作 2 で求めた値の差を求め, そのうち最も大きい差の値の絶対値を求める.
4: 操作 1 で求めた値の和を計算する.
5: 操作 3 で求めた値と操作 4 で求めた値の和を計算する.
6: 操作 5 で求めた値が 0 より大きいなら左プレイヤ, 0 より小さいなら右プレイヤ, 0 なら後手プレイヤを勝利プレイヤとして出力する.

このとき, Algorithm 4 は各札の参照を高々定数回しか行わないことを留意したい. したがって Algorithm 4 は線形時間で終了する. 次に Algorithm 4 が正しい勝者を出力していることを確認する. まず. 操作 4 で求めた値が正の数のとき, オールマイティ札を用いずとも左プレイヤは勝利できるのでこのとき Algorthm 4 は正しく出力していることは明らかである. 次に, Algorithm 4 において操作 4 で求めた値が 0 以下であり, 操作 5 で求めた値が正のとき, Algorithm 6 を用いることによって勝利できることを示す. まず, Algorithm 6 の補助アルゴリズムとして Algorithm 5 を設計する.

次に, Algorithm 6 を以下のように設計する.

Algorithm 4 において操作 4 で求めた値が 0 以下であり, 操作 5 で求めた値が正ならば, Algorithm 6 を使用することで, オールマイティ札を必ず使用でき, また, オー

Algorithm 5 補助アルゴリズム

1: Algorithm 1 を用いて各スートの値を求める.
2: 各スートに対し, 右プレイヤの持つ一番数字の小さい札 (文字列のうち一番最初に出てきた R)1 枚を削除した場合の値を Algorithm 1 を用いて求める.
3: 各スートにおいて, 操作 1 で求めた値と操作 2 で求めた値の差を求め, その差が最も大きいスートをスート A とする. 操作 1 で求めた各スートの値をスート A のみ操作 2 の値に更新する.
4: スート A に対して, オールマイティ札が出せるならばオールマイティ札を出す.

Algorithm 6 オールマイティ札有りパス七並べにおける左プレイヤの戦略

1: オールマイティ札を保持しているならば, Algorithm 5 を行う. その後各手番において操作 2 から操作 8 を行う.
2: オールマイティ札を保持していないとき, Algorithm 3 を行う.
3: オールマイティ札を保持しているならば, 以下の操作 4 から操作 8 を行う.
4: Algorithm 1 を用いて各スートの値を求める.
5: 各スートに対し, 右プレイヤの持つ一番数字の小さい札 (文字列のうち一番最初に出てきた R)1 枚を削除した場合の値を Algorithm 1 を用いて求める.
6: 各スートにおいて, 操作 4 で求めた値と操作 5 で求めた値の差を求め, その差が最も大きいスートを新たにスート A とする. 操作 4 で求めた各スートの値をスート A のみ操作 5 の値に更新する.
7: 操作 6 で更新した値が正である任意のスートに対し, 手を打つ.
8: スート A に対して, オールマイティ札が出せるならばオールマイティ札を出す.

ルマイティ札使用後の盤面において必ずゲームにおける各ゲーム和は正になっている. したがって定理 2 と同様の議論により, 左プレイヤ必勝である. また, Algorithm 4 において操作 4 で求めた値が 0 以下であり, 操作 5 で求めた値も 0 のとき, オールマイティ札をどこに使用しても各ゲーム和は最大 0 である. また, Algorithm 6 を使用することでゲーム和 0 を必ず実現可能である. したがって, 定理 2 と同様の議論により, 後手必勝である. 最後に, Algorithm 4 において操作 4 で求めた値が 0 以下であり, 操作 5 で求めた値も負のとき, オールマイティ札をどこに使用しても各ゲーム和は負になるので定理 2 と同様の議論により, 右プレイヤ必勝である. したがって定理 3 は示された. □

5 七並べの解析

前節にて議論したのパス七並べは通常の七並べのルールのうち,「先に全ての手札をプレイしたプレイヤが勝ち」というルールないし勝敗条件を取り払ったものである. 本節以下では通常の七並べ, つまりパス七並べの勝敗条件に「手札を全ての手札を消費したプレイヤが勝ち」という条件を加えた七並べの必勝判定においても線形時間で判定可能であることを示す.

5.1 準備

一般的な組合せゲーム理論の文脈では,「手札を全ての手札を消費したプレイヤが勝ち」というルールは存在しないため, 組合せゲーム理論の局面の値をそのまま七並べに適用すると勝敗を正しく判定することはできない. 本研究では, 七並べの解析のため, 七並べで用いる新たなスートの定義のほかに 7 種類のスートの特徴づけを与える. その特徴づけを**スートの種類**と呼ぶ. 七並べにおけるスートの種類は以下のように定義する.

定義 3. $0_0 = \{|\}$,
$(-k)_0 = \{|(-k+1)_0\}$, $k_0 = \{(k-1)_0|\}$,
$0_{LL} = \{(-k)_0|\}$, $0_{RR} = \{|k_0\}$,
$k_{LL} = \{(k-1)_{LL}|\}$, $k_{RR} = \{|-(k-1)_{RR}\}$,
$-1_L = \{|0_{LL}\}$, $1_R = \{0_{RR}|\}$,
$0_L = \{(-1)_L|\}$, $0_R = \{|1_R\}$,
$k_L = \{(k-1)_L|\}$, $k_R = \{(k-1)_R|\}$,
$(-k)_L = \{|(-k+1)_L\}$, $-k_R = \{|(-k+1)_R\}$
$0_{NR} = \{|k_{LL}\}$, $0_{NL} = \{-k_{RR}|\}$(ともに k は 1 以上),
$0_{NR} = \{|k_L\}$, $0_{NL} = \{-k_R|\}$(ともに k は 1 以上),
$0_{NL} = \{(-k)_L|\}$, $0_{NR} = \{|k_R\}$(ともに k は 2 以上),
$(-k)_{NR} = \{|(-k+1)_{NR}\}$, $(-k)_{NL} = \{|(-k+1)_{NL}\}$,
$k_{NL} = \{(k-1)_{NL}|\}$, $k_{NR} = \{(k-1)_{NR}|\}$
$0_{NR} = \{|k_{NR}\}$, $0_{NR} = \{|k_{NL}\}$,
$0_{NL} = \{(-k)_{NR}|\}$, $0_{NL} = \{(-k)_{NL}|\}$.

以下の証明ではスートの値が k, スートの種類が $i(i = 0, LL, L, R, RR, NR, NL)$ のスートを k_i と表記する. この局面の値は定理 2 で得られる局面の値と等しくなる. これを以下では示す.

定理 4. 七並べにおけるどのようなスート札列に対しても定義 3 により, 局面の値は定義できる.

定理 5. *あるスート列に対するパス七並べの値を k とする. 七並べにおけるそのスート列の値は k_i(ただし, i は $0, L, LL, R, RR, NR, NL$ のいずれか) である.*

定理 4, 定理 5 の証明. k が正のとき, 対象のスート列の文字列は以下のいずれかである.
(1)$\underline{0L^k}$,
(2)$\underline{0L^k(RL)^m}$,
(3)$\underline{0L^{k+1}\ (RL)^m R}(m \geq 0)$,
(4)$\underline{0L^{k+1}(RL)^m RRX}(m \geq 0)$,
(5)$\underline{0L^k(RL)^m LX}(m \geq 0)$.
このそれぞれに対し, 値が等しいことを確認すればよい. 以下の補題 3-7 を用いて示す.

補題 3. 任意の正の整数 k に対し, $\underline{0L^k} = k_0$, $\underline{0R^k} = -k_0$ が成立する.

証明. 定義より, $0_0 = \{|\} = \underline{0}, 1_0 = \{0|\} = \underline{0L}, 2_0 = \{1_0|\} = \underline{0LL}, \dots, k_0 = \underline{0L^k}$. したがって補題 3 は示された. □

補題 4. 任意の正の整数 k, 0 以上の整数 m に対し, $\underline{0L^k(RL)^m} = k_R$ が成立する.

証明. 定義より, $\underline{0RL} = \{|1_0\} = 0_R, \underline{0LRL} = \{0_R|\} = 1_R, \underline{0RLRL} = \{|1_R\} = 0_R, \dots, \underline{0L(RL)^m} = 1_R, \underline{0LL(RL)^m} = 2_R, \dots, \underline{0L^k(RL)^m} = k_R$. したがって補題 4 は示された. □

補題 5. 任意の正の整数 k, 0 以上の整数 m に対し, $\underline{L^{k+1}(RL)^mR} = k_L$ が成立する.

証明. 定義より, $\underline{0LR} = \{1_0|\} = 0_L, \underline{0RLR} = \{|0_L\} = -1_L, \underline{0LRLR} = \{-1_L|\} = 0_L, \dots, \underline{0L(RL)^mR} = 0_L, \underline{0LL(RL)^m} = 1_L, \dots, \underline{0L^k(RL)^m} = k_R$. したがって補題 5 は示された. □

補題 6. 任意の正の整数 k, 0 以上の整数 m に対し, $\underline{L^{k+1}(RL)^mRRX}(m \geq 0) = k_{NL}$ または k_L または k_{LL} が成立する.

証明. 2 つのケースに分けて場合分けを行う.

Case 1, 文字列 X が R のみからなる場合,

定義より, $\underline{0LR^z} = \{z_0|\} = 0_LL$ が成立するので, 以下補題 4 と同様に $\underline{0L^{k+1}(RL)^mR^z}(m \geq 0) = k_L(m > 0)$, $\underline{0L^{k+1}(RL)^mR^z}(m \geq 0) = k_{LL}(m = 0)$.

Case 2, X に L が一つでもある場合,

$\underline{0RX} = \{|k_L\}$($k$ は 0 以下) $= -k - 1_L$, もしくは

$\underline{0RX} = \{|k_0\} = 0_R$, もしくは

$\underline{0RX} = \{|k_L\}$($k$ は 1 以上)$= 0_N$, もしくは

$\underline{0RX} = \{|k_R\} = 0_{NR}$, もしくは

$\underline{0RX} = \{|k_{NR}\} = 0_{NR}$, もしくは

$\underline{0RX} = \{|k_{NL}\} = 0_{NR}$, もしくは

$\underline{0RX} = \{|(-k_+1)_{NR}\} = -k_{NR}$, もしくは

$\underline{0RX} = \{|(-k_+1)_{NL}\} = -k_{NL}$ のいずれかが成立する.

したがって $\underline{0RRX} = \{|-k - 1_L\}$, もしくは

$\underline{0RRX} = \{|-k - 1_L\}) = (-k - 2)_L$, もしくは

$\underline{0RRX} = \{|0_R\}) = -1_R$, もしくは

$\underline{0RRX} = \{|0_{NR}\}) = -1_{NR}$, もしくは

$\underline{0RRX} = \{|0_{NL}\}) = -1_{NL}$, もしくは

$\underline{0RRX} = \{|(k_N\} = (-k - 1)_{NR}$, もしくは

$\underline{0RRX} = \{|(k_N\} = (-k - 1)_{NL}$ のいずれかが成立する.

以上より, $\underline{0LRRX} = 0_{NL}$,

したがって $\underline{0(RL)^mRRX}(m \geq 0) = -1_{NL}$.

以上より, $\underline{0L^{k+1}(RL)^mRRX}(m \geq 0) = k_{NL}$.

したがって補題 6 は示された. □

補題 7. 任意の正の整数 k, 0 以上の整数 m に対し, $\underline{0L^k(RL)^mLX}(m \geq 0) = k_{NL}$ または k_{NR} または k_R または k_{RR} が成立する.

補題 6 と全く同様の手法で示すことができるので本稿では証明を省略する.

以上より定理 4, 定理 5 は示された. □

本小節の最後に, 証明に用いるこの七並べの値の性質を述べる. 以下の性質は定義より自明に導かれる. また以下本稿では, スートの種類が i(ただし, i は $0, L, LL, R, RR, NR, NL$ のスートを単に i スートと呼ぶ.

性質 1. 左が着手可能なスートはスートの値が正のスートあるいはスートの値が 0 の L スート, LL スート, NL スートのみである.

性質 2. 左はスートの値が正のスートに着手した後, そのスートの値はちょうど 1 だけ減少する. また値が 0 のスートに左が着手した後, そのスートの値は 1 以上の減少をする.

性質 3. 着手による状態遷移は図 1 のように記述できる. 特に 0_{NL} スートから L スートに状態が遷移するような着手を左がした場合, その局面の値は -2 以下である.

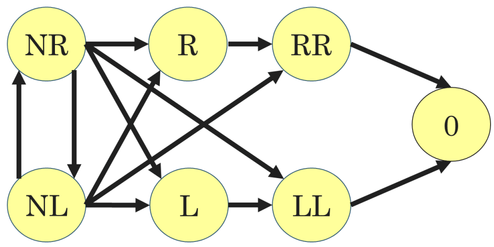

図 1 スートの種類の状態遷移図, 各文字はスートの種類を表す.

以上の性質は符号を反転させることで右についても同様のことが言えることに留意されたい. 以上の定義を用いて次小節にて証明を行う.

5.2 七並べの必勝判定

本小節では以下の定理を示す.

定理 6. 七並べの必勝戦略保持者がどちらであるかという判定は札の総枚数に関して線形時間で行うことができる.

以下の Algorithm 7 を使うことによって線形時間で七並べの必勝戦略保持者が正しく判定できることを示す.

また, 左が勝利プレイヤであるとき, 以下の札提出 Algorithm 8 が必勝戦略となる.

Algorithm 7 七並べ判定アルゴリズム

1: 各スートの値, 種類, ゲーム和を求める.
2: 全てのスートが 0 スートの場合, ゲーム和が 0 より大きいなら右プレイヤ, 0 より小さいなら左プレイヤ, 0 なら先手プレイヤを勝利プレイヤとして出力する. 全てのスートが 0 スートでない場合操作 3 に進む.
3: 全てのスートが 0 スートと $LL(RR)$ スートからなる場合, 左 (右) プレイヤを勝利プレイヤとして出力する. 0 スートと $LL(RR)$ スート以外のスートを含む場合操作 4 に進む.
4: 全てのスートが 0 スートと $LL(RR)$ スート,$L(R)$ スートからなる場合, ゲーム和が -1 より大きい (1 より小さい) なら左 (右) プレイヤ, -1 より小さい (1 より大きい) なら右 (左) プレイヤ, $-1(1)$ なら後手プレイヤを勝利プレイヤとして出力する. 他のスートを含む場合, 操作 5 に進む.
5: ゲーム和が 0 より大きいなら左プレイヤ, 0 より小さいなら右プレイヤ, 0 なら後手プレイヤを勝利プレイヤとして出力する.

Algorithm 8 七並べの左プレイヤの必勝戦略

1: Algorithm 1 を用いて各スートの値を求める.
2: 各スートの種類を求める.
3: スートの値が 1_R でない正のスートがあればそのスートに着手する. 無いとき, スートの値が 1_R のスートに着手する. スートの値が正のスートがなければ操作 4 を行う.
4: スートの種類が 0_L のスートがあればそのスートに着手する. 無いとき着手可能なスートに着手する.

このアルゴリズムが正しく動作することを確認するためには以下の 4 つの補題を証明すればよい.

補題 8. *七並べにおいて, 全てのスートが 0 スートの場合, ゲーム和が 0 より大きいなら右プレイヤが, 0 より小さいなら左プレイヤが, 0 なら先手プレイヤが必勝戦略を持つ.*

補題 9. *全てのスートが 0 スートと LL スートのみであり, LL スートを一つ以上含む七並べにおいて, 左プレイヤが必勝戦略を持つ.*

補題 10. *七並べにおいて, 全てのスートが 0 スートまたは LL スートまたは L スートであり, かつ L スートが一つ以上あるとする. このとき, ゲーム和が 0 または正ならば, 左必勝, -1 ならば後手必勝, -2 以下ならば右必勝である.*

補題 11. *NR または NL スートを一つ以上含むまたは, LL スートと L スートのどちらかと RR スートと R スートのどちらかが混在する場合, ゲーム和が正ならば, 左必勝, 0 ならば後手必勝, 負ならば右必勝である.*

以下ではこの 4 つの補題を証明する.

補題 8 の証明. スートの値が正のゲーム和を j, スートの値が負のゲーム和を $-k$ とする. 全てのスートが 0 スートの場合, それぞれの手番において, 相手がどの順序で手を打とうとも左は j 手, 右は k 手パスすることなく打つことが可能である. $j-k>0$ つまりゲーム和が 0 より大きいなら右プレイヤが, $k-j>0$ つまりゲーム和が 0 より小さいなら左プレイヤがより少ない手数で全ての札を出すことができるので, 勝利プレイヤである. また, ゲーム和が 0 のとき, 左も右もすべての手札を出すのに同手数かかるため, この場合, 先に出し終えることができる先手が必勝戦略を持つ.

したがって補題 8 は示された. □

補題 9 の証明. スートの値が正のゲーム和を j, スートの値が負のゲーム和を $-k$ とする. $j-k>0$ のとき, 左プレイヤはスートの値が正のスートにのみ着手することにより, k 手後の局面はすべてのスートが LL スートまたは L 札のみのスートになる. この局面において, 右の手番の場合この局面が, 左の手番の場合, 一手正のスートに打った後の局面が「右が手番かつ $j-k \geq 0$ が成立している」局面となる. したがってすべてのスートが LL スートまたは L 札のみのスートである. このとき, 右が着手可能なスートが存在しないため, 左が必勝戦略を持つ. $j-k \geq 0$ のとき, 仮定より一つ以上 LL スートがあるのでそのうち一つのスートを最後に着手することで左がパスすることなくすべての手札を出し切ることができる. 右は LL スートの L 札の後に出す必要がある札を抱えているためやはり $j-k \geq 0$ のときも左が必勝戦略を持つ. □

補題 10 の証明. ゲーム和の値によって 3 つの場合分けして示す.

(1) ゲーム和が正, または 0 で右先手のとき, ゲーム和が正, または 0 で右先手のとき, 左の手番時に常に値が正のスートが存在するため, 左はそのスートに着手できる. 性質 2 より, これを繰り返すとゲーム和が 0 以上でスートの値が負のスートが一つもない状態になる. このとき, 全てのスートの値は $0_L, 0_{LL}(, 0_0)$ であるので, 性質 1 より, 右がその局面において着手できる手がないので左が勝ちである.

(2) ゲーム和が 0 で左先手, またはゲーム和が -1 で右先手のとき, 左の手番時に値が正のスートが存在すればそのスートに着手するという操作を左が繰り返すことによって, 全てのスートの値は $0_L, 0_{LL}$ のみとなる. このとき右は $0_L, 0_{LL}$ に着手できないので値が正のスートが存在しないことに留意されたい. 左が手番の場合, スートの値が 0_L のスートがあればそのスートに着手すると, そのスートの値は必ず -1_L となる. この局面において右はこのスートにしか着手できないため, 左と右が一手ずつ売った後の局面も全てのスートの値は $0_L, 0_{LL}$ のいずれかとなる. これを繰り返すとスートの値が 0_{LL} のスートのみとなるが, この局面は補題 9 より, 左必勝である.

(3) ゲーム和が -1 で左手番, またはゲーム和が -2 以下のとき, 右の手番時に「値がスートの値が -1_L でない負のスートが存在すればそのスートに着手し, スートの値

が -1_L のスート以外着手できなければそのスートに着手する」という操作を右が繰り返すことによって, ある右着手後において全てのスートの値は $0_L, 0_{LL}, -k_0, -j_L$ のいずれか, ゲーム和は -1 以下となる. このとき, 左は 0_L または, 0_{LL} にしか着手できない. 0_{LL} に対して着手を行った場合, そのスートの新しい値は $-k_0$ となるので必ずこのスートに着手可能である. 左が 0_L に着手した場合, そのスートの新しい値は必ず -1_L となる. このとき, 上記の「値がスートの値が -1_L でない負のスートが存在すればそのスートに着手し, スートの値が -1_L のスート以外着手できなければそのスートに着手する」を続けることによって, (初期局面に L スートが一つ以上存在するため,) 必ず左手番開始時に -1_L が一つ以上と値が負のスートのみになる. このどちらにも左が着手不可能であり, -1_L スートに属する L 札を左は手札に必ず抱えているので, ゲーム和が -1 で左手番, またはゲーム和が -2 以下の時右が必勝戦略を持つ.

したがって題意は示された. □

補題 11 の証明. 対称性を利用することで補題 11 の証明は NR スート, NL スート, R スート, RR スートのいずれかが一つでもある場合, ゲーム和が負のときあるいはゲーム和が 0 で左が先手のときに右勝ちであることを示すのみでよい. ゲーム和が 0 で左が先手のとき, あるいはゲーム和が負のとき, 右は毎手番, スートの値が負のスートにのみ着手し続けることによって, ゲーム和が 0 以下の状態でスートの値が 0_R あるいは 0_{RR} であるスートが一つ以上ある局面にできる. このとき, スートの値が負で右の手番であればそのスートに着手することで左手番且つ, ゲーム和が 0 以下の状態でスートの値が 0_R あるいは 0_{RR} であるスートが一つ以上ある状態にできる. これが成り立つ局面を基本局面とする. 左は値が正のスートに着手する場合, 値の和は 0 以下なので必ず着手した後にスートの値が負のスートが存在するので右はそこに着手することで基本局面となる. 左が 0_L あるいは 0_{LL} のスートに着手した場合, 右は必ずそのスートに着手でき, そのスートに着手することでやはり基本局面にできる. 左が 0_N スートに着手した場合, その局面の値は性質 3 より, ゲーム和が -1 以下でかつスートの種類は NR または NL または RR または R となる, またはゲーム和が -2 以下で L スートが一つは存在する局面になる. したがって前者は右はそのスートに着手することにより, 基本局面にでき, 後者は補題 10 より右必勝局面となる. これを繰り返すと 0_R スート, 0_{RR} スート, 0_{NR} スートのみとなり, 左が着手できるスートがなくなる. 従って補題 11 は示された. □

補題 8, 9, 10, 11 より, Algorithm 7, 8 は正しく動作することが確認できる.

また以下の系を定義から直ちに示すことができる.

系 6. 0 以外の場に置かれている札がないとする. このとき正の整数 x, y と 0 または正の整数の z に対して以下が成立する;

- 0 スートならばその札列は $\underline{0L^x}$, または$\underline{0R^x}$ が成立する.
- LL スートならばその札列 $\underline{0L^xR^y}$ が成立する.
- L スートならばその札列は $\underline{0L^x(LR)^yR^z}$ または $\underline{0R^x(LR)^yR^z}$ が成立する.

したがって, 札列を一度チェックするだけでスートの種類が判定できる. したがって Algorithm 7 は札の総枚数 n に対して線形時間で動作する. 以上より定理 6 は示された. □

6 まとめと今後の展望

本研究では二人で行う七並べの亜種を定義し, その最適戦略を調べた. まず, 定理 2, 3 に示すようにパスを最初にしたプレイヤの負けとする七並べとその亜種に対して, 枚数に関して線形時間でその必勝戦略保持者がどちらであるかという判定を行うことができることを示した. また, 本研究では七並べにおいてもその必勝戦略保持者を線形時間で計算可能であることを示した. この七並べのような「手札を全て消費したプレイヤが勝ち」というルールのゲームは従来の組合せゲーム理論の解析では対象として扱えないものである. 本研究における最大の貢献はこのような従来の組合せゲーム理論解析で扱うことができなかった問題群に対し, 局面の値を再定義することで従来の枠組みに近い形で扱えるようにした点である. この解析手法は他の「手札を全て出し切ったら勝ちである」ゲーム群のみに限定されない幅広いゲームの解析に有効に働くことが期待される.

また, 「ある札に関して両方のプレイヤが所持あるいは両方のプレイヤが未所持であってもよい. 」という拡張や, 実際の七並べの拡張でもある「ある札を出したとき, 別のスートの最大札からも札を出すことが可能である (七並べにおいて K を出すとそのスートの A からも出すことができるルールがある). 」という拡張が考えられるが, 本研究の結果はそのような問題においても比較的高速に必勝戦略保持者がどちらであるかという判定できる可能性を示唆している. 一方で一般で広く遊ばれているような 3 人以上で行う場合などは本研究の提案アルゴリズムはそのままでは対応できない. しかしながらそういったより一般化した設定において, よりよい戦略を考える上での終盤解法の一つとしても, 本研究が有効に働くことを期待している.

参考文献

[1] 木村 勇太, 伊藤 毅志, :深層強化学習を用いたガイスター AI の構築, ゲームプログラミングワークショッ

プ 2019 論文集, pp.130–pp.135, (2019).

[2] 畢暁恒, 田中哲朗, :対話のない人狼ゲームの戦略. ゲームプログラミングワークショップ 2015 論文集, pp.25–pp.30(2015).

[3] Elkies, N D., :On numbers and endgames: combinatorial game theory in chess endgames. Games of No Chance 29 : pp.135–pp.150, (1996).

[4] Albert M. H., Nowakowski R. J., Wolfe D., 川辺治之訳, ：組合せゲーム理論　勝利の方程式, 共立出版 (2011).

[5] Bouton C. L., :NIM, A. Game With A Complete Mathematical Theory. Annals of Mathematics, 3.2: pp.35–pp.39(1902).

[6] Muller, M., :Computer Go as a sum of local games: an application of combinatorial game theory. PhD Thesis. Verlag nicht ermittelbar, (1995).

[7] Siegel A. N., :Combinatorial Game Theory. American Mathematical Society, (2013).

[8] Ulrich S., Paul W., :Zermelo and the Early History of Game Theory, Games and Economic Behavior, vol.34, number 1, pp.123 − 137, (2001).