修 士 論 文 の 和 文 要 旨

研究科・専攻	大学院　情報理工学系学　研究科　　　情報･通信工学　専攻　博士前期課程		
氏　　　名	金広　尚平	学籍番号	1431034
論 文 題 目	得点札を持つ二人トリックテイキングゲームの解析		

要　　旨

本研究ではトリックテイキングと呼ばれるカードゲームのジャンルの中でも、得点となる特定のカードを多く獲得することを目指すゲームについてその性質の解析と必勝性判定の方法の解明を行った。トリックテイキングゲームは複数のプレイヤーで遊ぶゲームである。先手のプレイヤーから順番にカードを出していく。その中で一番強いカードを出したプレイヤーがトリックと呼ばれるラウンドを取り、次のトリックの先手プレイヤーになる。トリックを繰り返し、勝者が決められる。

二人のプレイヤーで行うシングルスートのトリックテイキングゲームの必勝性判定について、中井ら、福澤、Wästlud および Kahn らの先行研究の拡張を目指した。本論文では、得点札を多く獲得する事を目的としたゲームで、カード全体の中で強い方から複数枚が得点札となる場合を考える。先行研究で扱われている Whist の勝利条件はより多くのトリックを獲得したプレイヤーの勝利であるが、これは多くの得点札を得たプレイヤーが勝利するルールにおいて全ての使用するカードが得点札の状況とみなすことができる。したがって本論文のルールは Whist の一般化とみなせる。

本研究ではまず、得点札が 2 枚のとき、両者が最善を尽くした場合にプレイヤーが獲得できる得点札の枚数を求める方法を示した。この場合は両プレイヤーの手札にある得点札でないカード同士の大小関係からプレイヤーが獲得可能な得点札の枚数を求める事ができる。つぎに Kahn らの手法を、得点札であるカードと得点札でないカードが存在する状況に拡張した。一方のプレイヤーが得点札のうち最も弱いカードを持ち、さらにそのプレイヤーの持つ非得点札は全て、もう一人のプレイヤーの持つどの非得点札よりも強いという条件のもとで、先手後手の優位性や手札に持っているカードによる有利不利がどの程度かを示した。得点札が複数枚ある場合においても先手番であるより後手番である事が有利であることを示した。また、手札に存在する得点札の数字は大きい方が有利であること、得点札でないカードの数字は小さい方が有利であることを示した。

平成 27 年度　修士論文

得点札を持つ
二人トリックテイキングゲームの
解析

学籍番号 1431034

金広　尚平

情報・通信工学専攻　情報数理工学コース

指導教員:武永康彦准教授
副指導教員:垂井淳准教授

目次

1　はじめに

本研究ではトリックテイキングと呼ばれるカードゲームの中でも、得点となる特定のカードを多く獲得することを目指すゲームについてその性質の解析と必勝性判定の方法の解明を行う。

トリックテイキングゲームとは、トランプ等のカードで遊ばれるゲームのルールの一種である。古い歴史を持つゲームジャンルであり、世界中で様々なトリックテイキングゲームが遊ばれている。有名なゲームとしてはブリッジ、ハーツ、ナポレオンなどが挙げられる。これらには以下の一般的なルールが共通している。

1. 各プレイヤーには同じ枚数の手札が配られる。
2. 1人の先手プレイヤーと呼ばれるプレイヤーが手札からカードを1枚公開して場に出す。
3. その後、他のプレイヤーも順々にカードを出していく。
4. 全員の出したカードの数字を比較する。一番大きいカードを出したプレイヤーがそのラウンド（トリック）に勝利する（トリックを取る)。
5. トリックを取ったプレイヤーはそのトリックに出たカード全てを得て、次のトリックの先手プレイヤーになる。この時得たカードは手札とは別に手元に置く。
6. これを手札が無くなるまで繰り返して勝利条件を満たしたプレイヤーが勝利する。

勝利条件はゲームによって異なっている。シンプルなトリックテイキングゲームの1つである whist はゲーム中に獲得したトリックの回数で勝利が決定する。このようにトリックの回数を競うものの他に、様々な勝利条件のゲームが存在する。FiveCard はトリックの回数ではなく、最後のトリックを獲得したプレイヤーが勝利する。また使用するカードの中で得点になるカードを決めてそれらを獲得する事を目的とするゲームも存在する。Skat と呼ばれるトリックテイキングゲームはトランプを用いて行われるが、そのうちの絵柄のあるカードおよび A のカードを獲得するとそれぞれに応じた得点が得られるというルールである。66 という名前のトリックテイキングゲームでも Skat と同様に絵柄のカードに得点が設定されているが、獲得した得点札の合計が 66 以上になると上がりを宣言できるというルールである。他には複数のプレイヤーの中でペアを組んで合計の得点を競うものや、ゲーム開始前に目標となる得点を設定しそれを目指すなどのルールも存在している。また多くのゲームではトランプが用いられるが、先手プレイヤーの出したカードのスート（マーク）と同じスートのカードでなければ出すことができないマストフォローなどのルールを持つゲームも多く存在している。

本論文では二人のプレイヤーで行うシングルスートのトリックテイキングゲームの必勝性

判定について、中井ら [1]、福澤 [2]、Wästlund[3] および Kahn ら [4][5]、の先行研究の拡張を目指した。中井、福澤の論文では一般化 66 という一般化したトリックテイキングゲームについての必勝性判定をゲームにおいて取るべきカードである得点札が 1 枚の時について研究が行われた。[1] では一般化 66 の必勝性判定問題の計算量と終盤の山札が無くなった場合の必勝性の判定手法が示された。[2] では山札がある場合での必勝性判定および統計解析が示された。Kahn の論文では whist をはじめとするゲームについて、後手番であることの優位性や手札の数字の大きさによる優位性などの性質が示された。Wästlund[3] では、Whist,GreedyWhist, および FiveCard について、最適な戦略とプレイヤーが何回トリックを獲得できるかを基準とした必勝判定法が示された。

本論文では、得点札を多く獲得する事を目的としたゲームで、得点札が複数枚存在する場合を考える。ただし、得点札は使用するカード全体で大きい数字から複数枚である。Whist の勝利条件はより多くのトリックを獲得したプレイヤーの勝利であるが、これは多くの得点札を得たプレイヤーが勝利するルールにおいて全ての使用するカードが得点札の状況とみなすことができる。したがって本論文のルールは Whist の一般化したルールになっている。

本研究ではまず、トリックテイキングゲームについて、まず得点札が 2 枚のとき、プレイヤーが獲得できる得点札の枚数を求める方法を示した。つぎに Kahn らの手法を、得点札であるカードと得点札でないカードが存在する状況に拡張することにより、プレイヤーの手札のある条件のもとで、先手後手の優位性や手札に持っているカードによる有利不利がどの程度かを考案する。

2　ルール

本研究ではカードの中に得点となるカード（得点札）があり、それを多く集める事を目的としたトリックテイキングゲームをモデル化して取り扱う。プレイヤーの人数は二人で、一人当りの手札は n 枚である。したがってゲームで使用するカードの総数は $2n$ 枚である。本論文ではこの $2n$ 枚中の大きい方から k 枚のカードを得点札とする場合について扱う。なお、得点札でない残りの $2n-k$ 枚のカードを非得点札と呼称する。

本研究のゲームの流れは、n 枚ずつの手札が配られた後、次の手順を繰り返すことで進行する。

1. 先手が 1 枚カードを出し、後手が 1 枚出す。
2. 強いカードを出したプレイヤーがそのトリックを取る。
3. トリックを取ったプレイヤーがそのトリックで出たカードを獲得する。ただしこのカードは手札とは別に保持する。
4. トリックを取ったプレイヤーが次の先手となる。

全ての手札を使いきった時にゲームは終了する。この時、取得した得点札の枚数が多いプレイヤーが勝利する。

本論文では二人のプレイヤーを Left と Right と呼称する。A を Left の手札、B を Right の手札とする。ゲームに使われるカードの数字は 1 から $2n$ であり、ゲーム開始時の二人の手札 A,B を $A = \{a_1, a_2, ..., a_n\}, B = \{b_1, b_2, ..., b_n\}$ と表わす。ただし $a_1 > a_2 > ... > a_n, b_1 > b_2 > ... > b_n$ をみたすものとする。

ゲーム中で使用されるカード全体が A と B のどちらに所属しているかを表す降順のカードの並びを標準表現とする。標準表現は残りのカード全体 $A \cup B$ の中で強いカードから順にどちらに属しているかを A または B で表現される。例えば $A = \{8, 7, 3, 1\}, B = \{6, 5, 4, 2\}$ の時の標準表現は AABBBABA である。この標準表現はカードが使用されるカード全体の中でどれほどの強さなのかを示すものである。また、標準表現が同じカード A,B とカード A',B' があるとき、それぞれに含まれる得点札の枚数が同じであれば、それぞれのゲームの結果は全く同じになる。

$2n$ 枚のカードのうち、得点札の枚数を k 枚とし、A に含まれる得点札が p 枚、B に含まれる得点札が $k - p$ 枚とする。ϵ を手番を表す変数とする。$\epsilon = 0$ の時は次のトリックにおいて Right が先手で、1 の時は Left が先手である。ゲームの途中での状態を (ϵ,A,B) と表す。そして $V_n^\epsilon(A, B)$ を Left の状態の値とする。この値は Left と Right が最適なプレイを行うと仮定した時のゲーム終了時に Left が取る得点カードの枚数である。

またカード a, b の大小関係を示す記号を以下のように定義する。

$$[a, b] = \begin{cases} 1 & \text{if b<a} \\ 0 & \text{if a<b} \end{cases}$$

$\epsilon = 0$ のとき、Right が先にカード $b_j \in B$ を選び、その後に Left がカード $a_i \in A$ を Right が出したカード b_j が何か知っている状態で選ぶ。$s_{i,j}$ を a_iと b_j に含まれる得点札の枚数とすると、このときのゲームの状態は $([a_i, b_j], A - \{a_i\}, B - \{b_j\})$ であり Left は $V_{n-1}^{[a_i,b_j]}(A - \{a_i\}, B - \{b_j\})$ 枚の得点札を得る。このときの式を示すと以下のようになる。

$$V_n^0(A, B) = \min_{b \in B} \max_{a \in A}([s_{i,j} \cdot [a_i, b_j] + V_{n-1}^{[a_i,b_j]}(A - \{a_i\}, B - \{b_j\}))$$

$\epsilon = 1$ のとき、同様に以下の式で示される。

$$V_n^1(A, B) = \max_{a \in A} \min_{b \in B}(s_{i,j} \cdot [a_i, b_j] + V_{n-1}^{[a_i,b_j]}(A - \{a_i\}, B - \{b_j\}))$$

利得行列 G とはプレイヤー 1 とプレイヤー 2 がそれぞれ n 個、m 個 の戦略を持つときに得られる $n \times m$ 行列 である。i 行 j 列の要素 $g_{i,j}$ はプレイヤーがそれぞれ i 番目と j 番目の戦略をとった時のプレイヤー 1 の利得である。本研究の利得行列は、手札のカード 1 枚ずつがそれぞれ 1 つの戦略としてみなされ、ゲーム開始時に $n \times n$ 行列として得られる。

その要素は両者が最適なプレイをした時に Left の取得する得点札の枚数である。利得行列 $G(A,B)=(g_{ij})$ は以下のように表せる。

$$g_{ij}=s_{i,j}\cdot[a_ib_j]+V_{n-1}^{[a_i,b_j]}(A-\{a_i\},B-\{b_j\}) \tag{1}$$

(1) 式は a_i,b_j を出したトリックでの Left の獲得枚数と a_i,b_j を出した後に Left が獲得する得点札の枚数の合計になっている。それぞれを別の行列 T と行列 W として以下のように定義する。行列 G および行列 T, 行列 W の例を図 1 に示す。

$$\begin{aligned} t_{ij}&=s_{i,j}\cdot[a_ib_j] \\ w_{ij}&=V_{n-1}^{t_{ij}}(A-\{a_i\},B-\{b_j\}) \end{aligned}$$

$$\begin{array}{c|c|c|c|} & 6 & 4 & 2 \\ \hline 5 & 0 & 1 & 1 \\ \hline 3 & 1 & 1 & 0 \\ \hline 1 & 1 & 1 & 1 \\ \hline \end{array} \qquad \begin{array}{c|c|c|c|} & 6 & 4 & 2 \\ \hline 5 & 0 & 1 & 1 \\ \hline 3 & 0 & 0 & 0 \\ \hline 1 & 0 & 0 & 0 \\ \hline \end{array} \qquad \begin{array}{c|c|c|c|} & 6 & 4 & 2 \\ \hline 5 & 0 & 0 & 0 \\ \hline 3 & 1 & 1 & 0 \\ \hline 1 & 1 & 1 & 1 \\ \hline \end{array}$$

G　　T　　W

図 1　$k=6$ のときの利得行列 G({6,3,1},{5,4,2}) と行列 T({6,3,1},{5,4,2}) と行列 W({6,3,1},{5,4,2}).

3　Kahn らの研究について

[4] で扱われるゲーム Whist の勝利条件は多くのトリックを獲得したプレイヤーの勝利である。この勝利条件は本研究における得点札の枚数 $k=2n$ すなわち、使用する全てのカードが得点札である場合と等価である。その場合について [4] では以下の定理が示されている。

以下の定理 1 は後手のプレイヤーの優位性を示すものである。プレイヤーが後手である事は重要であるがこの定理 1 はゲーム中においてそれがどれほど重要であるかを、取れるトリック数を用いて示している。

定理 1　先手である事は先手でない事に対して高々 1 トリック分の差しか生じない。つまり以下の関係をみたす。

$$V_n^1(A,B)\le V_n^0(A,B)\le 1+V_n^1(A,B)$$

定理 2 は手札に大きな数字のカードを持つ事はプレイヤーにとって得であるということを示している。

定理 2　$A \cup B$ の中に a と b の間の値をとる要素がなく、$a \in A, b \in B, b > a$ と仮定する。$A' = (A - \{a\}) \cup \{b\}, B' = (B - \{b\}) \cup \{a\}$ とする。その時以下の関係を満たす。

$$V_n^\epsilon(A, B) \leq V_n^\epsilon(A', B') \leq V_n^\epsilon(A, B) + 1$$

4　2 枚の得点札の場合の解析

本章では得点札が使用するカードの中で大きい方から 2 枚の場合について、Left と Right の両プレイヤーが最善を尽くした場合に Left が獲得する得点札の枚数が 2 枚の時は勝利、1 枚の時は引き分け、0 枚の時は敗北として必勝性判定を行う。手札行列 H の要素 $h_{i,j}$ を以下のように定義する [3]。

$$h_{i,j} = \begin{cases} 1, & \text{if } a_i > b_j \\ -1, & \text{if } a_i < b_j \end{cases}$$

この手札行列は 1 トリックを終えるごとに更新されていく。a_i, b_j が出された直後のトリックの手札行列は元の手札行列から i 行目と j 列目 を削除する事で得られる。

2 枚の得点札が A にのみ存在する場合、Left は明らかに得点札を 2 枚獲得する。これは得点札が他のどのようなカードより大きいため、Left が得点札を出したトリックでは Right がいかなるカードを出しても Left がトリックを取り、得点札を獲得するためである。得点札が B にのみ存在する場合も同様に Left は得点札を得ることはできない。

次に得点札 2 枚が異なるプレイヤーの手札に含まれる場合を考える。一般生を失うことなく得点札 $2n$ が A に、$2n-1$ が B に含まれると仮定する。このとき $a_1 = 2n, b_1 = 2n-1$ である。この場合、次の定理が成り立つ。

定理 3　Left が最初のトリックで先手プレイヤーであるとする。Left と Right は最適な戦略を取るものとする。Left は得点札を 1 枚のみ獲得する。

証明　最初のトリックで Left が a_1 を出したとき、Right が b_1 以外を出す。この後のトリックでは、Left は Right の得点札 b_1 より大きいカードを手札に持たない。したがって Left は得点札を 1 枚しか獲得できない。最初のトリックで Left が a_1 以外を出したとき、Right が b_1 を出す。したがって以降のトリックで Left は得点札 $a_1 = 2n$ で $b_1 = 2n-1$ に勝ち、得点札を 2 枚獲得する機会を失う。よって Left は得点札を 1 枚しか獲得できない。□

定理 4　Right が最初のトリックの先手プレイヤーである。Left と Right は最適な戦略を取るものとする。$h_{i,i} = -1 (3 \leq i \leq n)$ である場合、Left は得点札を 2 枚獲得する。それ以外の場合、Left は得点札を 1 枚だけ獲得する。

証明　まず $h_{i,i} = -1 (3 \leq i \leq n)$ のときに Left が得点札を 2 枚獲得する戦略を示す。最初のトリックで Right が b_1 を出した場合、Left は a_1 を出して得点札 2 枚を獲得する。

Right が $b_j(3 \leq j \leq n)$ を出した場合、$h_{i,i} = -1$ であるので Left は a_j を出すことで次のトリックの先手プレイヤーを Right にすることができる。また、トリック終了後の手札行列 H' はゲーム開始時の手札行列から j 行 j 列 を削除して得られるため、H' の要素は $h'_{i,i} = -1(3 \leq i \leq n-1)$ をみたす。したがって各プレイヤーの手札が 1 枚ずつ減ったが、Right が先手プレイヤーであり、手札行列 H の要素が $h_{i,i} = (3 \leq i \leq n-1)$ であるという定理 4 の仮定が維持されている。Right が b_2 を出した場合について考える。$h_{2,2} = -1$ であれば L は a_2 を出すことで $b_j(3 \leq j \leq n)$ を出された場合と同様に定理 4 の先手プレイヤーが Right である事と手札行列の要素 $h_{i,i} = -1(3 \leq i \leq n-1)$ の仮定が維持されている。もし Right が b_2 を出して $h_{2,2} = 1$ の場合、L は a_3 を出す。$h_{3,3} = -1$ であるので $a_3 < b_3$ より $a_2 < b_3$ である。このとき、手札行列 H' は H の 3 行目と 2 列 目を削除して得られるが、$h_{2,3} = 1$ であるので手札行列 H' の要素 h' は $h'_{2,2} = 1$ となる。$h'_{i,i}(3 \leq i \leq n-1)$ は 2 行目と 3 列目の削除により、$h'_{i,i} = h_{i+1,i+1} = -1$ となる。また次のトリックの先手プレイヤーは Right のままであるので、定理 4 の仮定が維持されている。

Right が先手プレイヤーであり、$h_{i,i} = -1(3 \leq i \leq n)$ の状態を保ちながら以上のケースを繰り返すと、常に Right が先手プレイヤーのままで $A = \{a_1, a_2\}, B = \{b_1, b_c\}(2 \leq c \leq n)$ の状態に到達する。このとき、Right が得点札 b_1 を出した時は Left も得点札 a_1 を出すことで得点札を 2 枚獲得する。Right が非得点札を出した時は、Left も非得点札を出し、次のトリックで Left が得点札を 2 枚獲得する。この際、最後のトリックでは先手と後手が関係ないため、非得点札同士のカードの大小は獲得枚数に影響しない。以上より、Left は得点札を 2 枚獲得する。

次に $3 \leq i \leq n$ の範囲で $h_{d,d} = 1$ となるような $d(3 \leq d \leq n)$ が存在するときに Right が得点札を 1 枚取る戦略を示す。最初のトリックで Right は b_d を出す。Left が a_d 以上のカードを出すと $b_d < a_d$ であるため Left がトリックを獲得し次のトリックの先手プレイヤーになる。このとき、定理 3 より Right は得点札を 1 枚獲得できる。最初のトリックで Right が b_d を出し、Left が b_dに負けるカード $a_t(d < t \leq n)$ を出した場合を考える。次のトリックの手札行列 H' は H の t 行目と d 列目 を削除して得られる。このときの $h'_{d,d}$ について考える。最初の手札行列 H において $b_d < a_d$および $b_{d+1} < b_d$ より $h_{d,d+1} = 1$ である。ここから t 行目と d 列目 を削除するので、$h'_{d,d} = h_{d,d+1}$ であるので $h'_{d,d} = 1$ である。そして次のトリックの先手プレイヤーは Right であるので、定理 4 の先手プレイヤーの仮定と、手札行列の要素に $h_{d,d} = 1$ となる $d(3 \leq d \leq n-1)$ が存在するという仮定が維持されている。これを繰り返していくと少なくとも手札行列が $d \times d$ 行列になるときには任意の i に対し $h_{i,d} = 1$ となるため、L がいかなるカードを出しても L が先手プレイヤーになる。このとき、定理 3 より Right は得点札を 1 枚のみ獲得する。 □

以上から得点札が 2 枚の場合の必勝判定を行うことができる。

5　複数枚の得点札を持つゲームの解析

5.1　ゲームの解析

本章では本論文で扱うゲームについて、Kahn らの結果と同様の解析を行う。定理 1 の後手番の優位性は得点札が存在するルールでも手番のルールは共通しているので必勝性に深く関わってくる。また、定理 2 の大きい数字のカードを所持する事の優位性は、得点札と得点札でないカードの 2 種類にカードが別けられるので、得点札同士の交換、得点札でないカード同士の交換、得点札と得点札でないカードの交換の 3 つの場合に分けて証明を行う。

以下本章で扱う手札 A と B を次のように仮定する。A の持つ最小の得点札 a_p と B の持つ最小の得点札 b_{k-p} の大小関係が $a_p > b_{k-p}$ であり、また A に含まれる全ての非得点札 $a_{p+1}...a_n$ が B に含まれる全ての非得点札 $b_{k-p+1}...b_n$ より小さい。

図 3 は a_i,b_j の組み合わせでどちらがトリックを取るかを示した図である。(a) は一般的な場合について示している。a_i が得点札であり b_j が非得点札であるような領域は $a_i > b_j$ であるので明らかに Left がトリックを取る。a_i が非得点札で b_j が得点札であるような領域についても同様に $a_i < b_j$ であるので明らかに Right がトリックを取る。a_i, b_j がともに得点札、または非得点札であるような領域は、$a_i > b_j$ である部分は Left がトリックを取り、$a_i < b_j$ である部分は Right がトリックを取るが、手札のカードによってその範囲は異なる。ある行について見れば $b_j > b_{j+1}$ であるので右側の要素ほど Left がトリックを取りやすく、また列について見れば $a_i > a_{i+1}$ であるので上側の要素ほど Left がトリックを取りやすい。

(b) は手札の仮定を満たす場合についての図となっている。A の持つ最小の得点札 a_p が B のもつ最小の得点札 b_{k-p} よりも大きいため、図 3(b) の左上の両者が得点札同士を出す領域の内、一番右の列は全て Left がトリックを取る。また A に含まれる非得点札全てが B の全ての非得点札よりも小さいため、図 3(b) の右下の非得点札同士の領域は全て Right がトリックを取る。

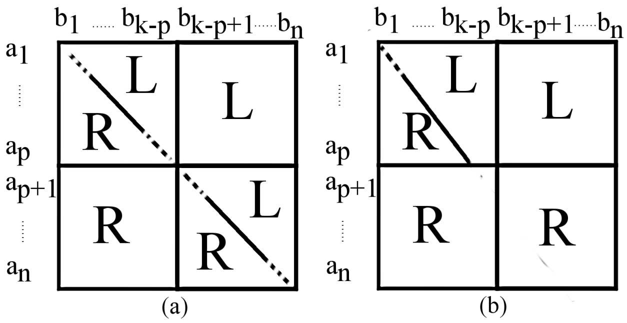

図 2　手札 a_i,b_j の組み合わせでどちらがトリックを取るかを示した図

次に手札が同じ時に得点札の枚数 k を変化させた場合について考える。得点札を獲得するためには自分の所持する得点札を出してトリックを取る、または相手の所持する得点札に対して得点札を出してトリックを取る事が必要である。得点札は使用するカードの大きい方から k 枚なので、標準表現が同じならば得点札の枚数が多い方が得点札を獲得する機会が多くなる。以下の性質は A、B の各カードの大小関係が同じ時に、得点札の枚数が 1 枚変わる事による $V_n^\epsilon(A,B)$ の差を示している。

性質 1　$V_{(m)}^\epsilon(A,B)$ を $k=m$ の時の $V_n^\epsilon(A,B)$ とする。この時、以下の関係を満たす。

$$V_{(m)}^\epsilon(A,B) \le V_{(m+1)}^\epsilon(A,B) \le V_{(m)}^\epsilon(A,B)+1$$

証明　まず、左の不等式を示す。得点札の枚数を変化させても、カードの数字は変わらないので A と B の各カードの大小関係は同じである。したがって、Left は $k=m$ の時の戦略を $k=m+1$ の時に用いることで少なくとも $k=m$ で Left が取得した得点札は獲得できる。よって左側の不等式が成り立つ。次に右の不等式を示す。左の不等式の場合と同様に得点札の枚数を変化させてもカードの大小関係が同じであるので、Right は $k=m$ の時の戦略を $k=m+1$ の時に用いることで、少なくとも $k=m$ で Right が取得した得点札は獲得できる。また、このとき $k=m$ で非得点札であったカードが $k=m+1$ で得点札になったとき、Right がそのカードを獲得していた場合は $V_{(m+1)}^\epsilon(A,B)$ は $V_{(m)}^\epsilon(A,B)$ のままである。Left がそのようなカードを獲得していれば明らかに $V_{(m+1)}^\epsilon(A,B)=V_{(m)}^\epsilon+1$ である。よって右側の不等号が成り立つ。

以下の補題は以下の定理の証明中において $\min_j \max_i w_{i,j}$ を求める際に用いる。行列 W の要素の大小関係からその候補となる要素を求める補題である。

補題 1　$r_1 \ge ... \ge r_l \le r_{l+1} \le ... \le r_n, s_1 \ge ... \ge s_l \le s_{l+1} \le ... \le s_n$ を満たす時、$\min(\max(r_1,s_1),\max(r_2,s_2),...,\max(r_n,s_n)) = \max(r_l,s_l)$ である。

証明　$1 \le i \le l-1$ の時、$r_i \ge r_{i+1}$ および $s_i \ge s_{i+1}$ より $\max(r_{i+1},s_{i+1}) = r_{i+1} \le r_i \le \max(r_i,s_i)$ であり $\max(r_i,s_i) = s_i$ のとき、$s_i > r_i \ge r_{i+1}$ より $\max(r_i,s_i) > r_{i+1}$ である。また $\max(r_{i+1},s_{i+1}) = s_{i+1}$ のときも同様に $\max(r_i,s_i) > s_{i+1}$ である。したがって $\max(r_i,s_i) \ge \max(r_{i+1},s_{i+1})$ である。同様に $l \le i \le n-1$ の時、$r_i \le r_{i+1}$および $s_i \le s_{i+1}$ より $\max(r_i,s_i) \le \max(r_{i+1},s_{i+1})$ である。したがって $\min(\max(r_1,s_1),\max(r_2,s_2),...,\max(r_n,s_n)) = \max(r_l,s_l)$ である。

以下の定理は本論文のゲームのルールにおける性質を示している。これらの証明は後に行う。同じ手札で先手である場合と後手である場合の得点の差は以下の式の関係をみたす。

定理 5　手札 A,B について最小の得点札が B に含まれていて、A に含まれている非得点札

が全て B に含まれている非得点札より小さいとする。後手番であることは先手番であることより高々得点札 2 枚分有利である。それは以下の式で示される。

$$V_n^1(A,B) \leq V_n^0(A,B) \leq 2 + V_n^1(A,B)$$

次に A と B の連続したカードを交換する事を考える。以下の定理は得点札同士の交換、得点札でないカード同士の交換、そして得点札と得点札でないカードの交換において Left の得点が交換前後でどの程度増減する可能性があるかを示している。

定理 6 手札 A,B について、最小の得点札が B に含まれていて、A に含まれている非得点札が全て B に含まれている非得点札より小さいとする。$a^* \in A, b^* \in B, a^* < b^*, a^*$と b^* は連続であり、また a^*、b^*が共に得点カード、$A' = (A - \{a^*\}) \cup \{b^*\}, B' = (B - \{b^*\}) \cup \{a^*\}$ とする。カードの交換前後の $V_n^\epsilon(A,B)$ は $\epsilon = 0$ または 1 どちらでも以下の関係を満たす。

$$V_n^\varepsilon(A,B) \leq V_n^\varepsilon(A',B') \leq V_n^\varepsilon(A,B) + 2$$

定理 7 手札 A,B について最小の得点札が B に含まれていて、A に含まれている非得点札が全て B に含まれている非得点札より小さいとする。$a^* \in A, b^* \in B, b^* < a^*, a^*$と b^* は連続であり、また a^*、b^*は共に非得点カード、$A' = (A - \{a^*\}) \cup \{b^*\}, B' = (B - \{b^*\}) \cup \{a^*\}$ とする。カードの交換前後の $V_n^\epsilon(A,B)$ は $\epsilon = 0$ または 1 どちらでも以下の関係を満たす。

$$V_n^\varepsilon(A,B) \leq V_n^\varepsilon(A',B') \leq V_n^\varepsilon(A,B) + 2$$

定理 8 手札 A,B について最小の得点札が B に含まれていて、A に含まれている非得点札が全て B に含まれている非得点札より小さいとする。$a^* \in A, b^* \in B, a^* < b^* a^*$と b^* は連続であり、また a^*は非得点カード、b^*は得点カード である時、$A' = (A - \{a^*\}) \cup \{b^*\}, B' = (B - \{b^*\}) \cup \{a^*\}$ とする。カードの交換前後の $V_n^\epsilon(A,B)$ は $\epsilon = 0$ または 1 どちらでも以下の関係を満たす。

$$V_n^\varepsilon(A,B) - 1 \leq V_n^\varepsilon(A',B') \leq V_n^\varepsilon(A,B) + 1$$

上記の 4 つの定理の証明は帰納法で行われ、それぞれの証明は個別ではなく 1 つの証明の中で行う。以上の定理が全て $n-1$ のときに成り立つと仮定し、それぞれの定理で n のときについて証明を行う。その証明の中で、定理 5 の証明のために定理 6 から定理 8 の $n-1$ のときの仮定を用い、また定理 6 から定理 8 の証明のために定理 5 の $n-1$ のときの仮定を用いる。

証明 まず各定理について $n = 1$ の場合を考える。定理 5 に関して、$n = 1$ の時に先手と後手が入れ替わる事は $V_n^\epsilon(A,B)$ に影響を及ぼさないので成立している。

定理 6 に関して、$n = 1$ の時は $a = 1, b = 2$ かつ共に得点札である状況である。この場合に交換を行えば $\epsilon = 0,1$ 両方で 明らかに $V_n^\epsilon(A',B') = V_n^\epsilon(A,B) + 2$ を満たす。得点札

でないカード同士の交換を行う定理 7 に関して $n=1$ の時は $a=2, b=1$ であるが交換を行っても得点札が存在せず交換前後で $V_n^{\epsilon}(A,B)$ の変化がないため、成立している。定理 8 の場合の $n=1$ の時は $a=1, b=2$ かつ 2 のカードのみ得点札の場合のみ、この定理に示す交換が可能である。この時 $V_n^{\epsilon}(A',B') = V_n^{\epsilon}(A,B)+1$ を満たす。

ここから帰納ステップの証明に入る。まず、n の時の定理 6 について考える。$a^* \in A, b^* \in B, a^* < b^*$で共に得点札であり a^*と b^*は $A \cup B$ において連続した要素である。$\epsilon = 0$ および$\epsilon = 1$ の場合の $V_n^{\epsilon}(A,B)$ はそれぞれ以下の式で表される。

$$V_n^0(A,B) = \min_{b\in B}\max_{a\in A}(2\cdot[ab] + V_{n-1}^{[ab]}(A-\{a\}, B-\{b\}))$$
$$V_n^1(A,B) = \max_{a\in A}\min_{b\in B}(2\cdot[ab] + V_{n-1}^{[ab]}(A-\{a\}, B-\{b\}))$$

これらの値が a^*と b^* の交換で変わらないか、高々 2 だけ増える事を、各カード a_i, b_j の組み合わせにおいて確かめる。

(i)$a \neq a^*, b \neq b^*$ の時 $g_{a,b}$ は以下の式になる。

交換前

$$s_{a,b}\cdot[a,b] + V_{n-1}^{[a,b]}(A-\{a\}, B-\{b\})$$

交換後

$$s_{a,b}\cdot[a,b] + V_{n-1}^{[a,b]}(A'-\{a\}, B'-\{b\})$$

定理 6 の $n-1$ の場合より、

$$\begin{aligned} V_{n-1}^{[a,b]}(A-\{a\}, B-\{b\}) &\leq V_{n-1}^{[a,b]}(A'-\{a\}, B'-\{b\}) \\ &\leq V_{n-1}^{[a,b]}(A-\{a\}, B-\{b\}) + 2 \end{aligned} \tag{2}$$

残された手札は交換した連続しているカード a^*, b^* のみが交換前後で異なるだけなので、上の不等式に $s_{a,b}\cdot[ab]$ を追加するだけである。

(ii)$a = a^*, b \neq b^*$ かつ b が得点札のとき

a^*と b 及び b^*と b の大小関係は不明なため $[a^*,b], [b^*,b]$ の値が決まらないが、$g_{a^*,b}$ は以下の式になる。

交換前

$$2\cdot[a^*,b] + V_{n-1}^{[a^*,b]}(A-\{a^*\}, B-\{b\})$$

交換後

$$2\cdot[b^*,b] + V_{n-1}^{[b^*,b]}(A'-\{b^*\}, B'-\{b\})$$

a^*と b^* が連続したカードであるので $[a^*,b] = [b^*,b]$ である。したがってトリック後の手札の組 $(A-\{a^*\}, B-\{b\})$ は $(A'-\{b^*\}, B'-\{b\})$ と同じ標準表現を持ち、得点札の枚

数も同じである。標準表現は使用されるカード全体の並びであるので、これが全く同じ時、Left は $(A-\{a^*\}, B-\{b\})$ の戦略を用いることで $(A'-\{b^*\}, B'-\{b\})$ のときに全く同じだけ得点札を獲得できる。このため交換前と交換後で同じ値となる。

(iii)$a=a^*, b\neq b^*$ かつ b が得点札でないとき
$a^*>b, b^*>b$ であるので $g_{a^*,b}$ は以下の式になる。
交換前

$$1\cdot[a^*,b]+V_{n-1}^{[a^*,b]}(A-\{a^*\}, B-\{b\})=1+V_{n-1}^{1}(A-\{a^*\}, B-\{b\})$$

交換後

$$1\cdot[b^*,b]+V_{n-1}^{[b^*,b]}(A'-\{b^*\}, B'-\{b\})=1+V_{n-1}^{1}(A'-\{b^*\}, B'-\{b\})$$

(ii) と同様の理由により交換前と交換後で値は変わらない。

(iv)$a\neq a^*, b=b^*$かつ a が得点札のとき
a と a^*および a と b^* の大小関係が不明であるが a^*と b^* は連続したカードであるので、$[a,b^*]=[a,a^*]$ である。(ii) と同様に g_{a,b^*} は以下の式になる。
交換前

$$2\cdot[a,b^*]+V_{n-1}^{[a,b^*]}(A-\{a\}, B-\{b^*\})$$

交換後

$$2\cdot[a,a^*]+V_{n-1}^{[a,a^*]}(A'-\{a\}, B'-\{a^*\})$$

(ii) と同様の理由により交換前と交換後で値は変わらない。

(v)$a\neq a^*, b=b^*$ かつ a が非得点札のとき
$a<a^*, a<b^*$ であるので g_{a,b^*} は以下の式になる。
交換前

$$1\cdot[a,b^*]+V_{n-1}^{[a,b^*]}(A-\{a\}, B-\{b^*\})=V_{n-1}^{0}(A-\{a\}, B-\{b^*\})$$

交換後

$$1\cdot[a,a^*]+V_{n-1}^{[a,a^*]}(A'-\{a\}, B'-\{a^*\})=V_{n-1}^{0}(A'-\{a\}, B'-\{a^*\})$$

(ii) と同様の理由により交換前と交換後で値は変わらない。

(vi)$a=a^*, b=b^*$ のとき
g_{a^*,b^*} は以下の式になる。
交換前

$$2\cdot[a^*,b^*]+V_{n-1}^{[a^*,b^*]}(A-\{a^*\}, B-\{b^*\})=V_{n-1}^{0}(A-\{a^*\}, B-\{b^*\}) \tag{3}$$

交換後

$$\begin{aligned}&2\cdot[b^*,a^*]+V_{n-1}^{[b^*,a^*]}(A'-\{b^*\},B'-\{a^*\})\\&=2+V_{n-1}^{1}(A'-\{b^*\},B'-\{a^*\})\end{aligned}$$

この時、$(A'-\{b^*\},B'-\{a^*\})=(A-\{a^*\},B-\{b^*\})$ であるので以下の関係が成り立つ。

$$\begin{aligned}&2+V_{n-1}^{1}(A'-\{b^*\},B'-\{a^*\})\\&=2+V_{n-1}^{1}(A-\{a^*\},B-\{b^*\})\end{aligned}\tag{4}$$

また $n-1$ の時の定理 5 より、以下の関係を満たしている。

$$\begin{aligned}&2+V_{n-1}^{1}(A-\{a^*\},B-\{b^*\})\\&\geq V_{n-1}^{0}(A-\{a^*\},B-\{b^*\})\end{aligned}\tag{5}$$

また (4) において $n-1$ の時の定理 5 より

$$2+V_{n-1}^{1}(A-\{a^*\},B-\{b^*\})\leq 2+V_{n-1}^{0}(A-\{a^*\},B-\{b^*\})\tag{6}$$

である。以上の (i) から (vi) の場合において、カードの交換によって $V_n^{\epsilon}(A,B)$ の交換後の値が交換前の値と変わらないか、高々 2 増えるのみであることが示された。

n の時の定理 7 について考える。a^*と b^* は $a^*\in A, b^*\in B, b^*<a^*$を満たす共に得点札ではないカードで、$a^*$と b^*は $A\cup B$ において連続した要素である。定理 6 と同様に証明を行う。

(i)$a\neq a^*, b\neq b^*$ のとき $g_{a,b}$ は以下の式になる。

交換前

$$s_{a,b}\cdot[a,b]+V_{n-1}^{[a,b]}(A-\{a\},B-\{b\})$$

交換後

$$s_{a,b}\cdot[a,b]+V_{n-1}^{[a,b]}(A'-\{a\},B'-\{b\})$$

定理 7 の $n-1$ の場合より以下の関係が成り立つ。

$$\begin{aligned}V_{n-1}^{[ab]}(A-\{a\},B-\{b\})&\leq V_{n-1}^{[ab]}(A'-\{a\},B'-\{b\})\\&\leq V_{n-1}^{[ab]}(A-\{a\},B-\{b\})+1\end{aligned}$$

定理 6 の (i) の場合と同様の理由により、残された手札は交換した連続しているカード a^*, b^* だけが交換前後で異なるだけなので、上の不等式に [ab] を追加するだけである。

(ii)$a=a^*, b\neq b^*$ かつ b は得点札でないとき

a^*と b 及び b^*と b の大小関係は不明なため $[a^*b], [b^*b]$ の値が決まらないが、大小関係がわかっていても $V_n^\epsilon(A,B)$ は以下の式になる。
交換前

$$0 \cdot [a^*b] + V_{n-1}^{[a^*b]}(A - \{a^*\}, B - \{b\}) = V_{n-1}^{[a^*b]}(A - \{a^*\}, B - \{b\})$$

交換後

$$0 \cdot [b^*b] + V_{n-1}^{[b^*b]}(A' - \{b^*\}, B' - \{b\}) = V_{n-1}^{[b^*b]}(A' - \{b^*\}, B' - \{b\})$$

定理 6 の証明における (ii) と同様の理由で交換前と交換後の値は変わらない。

(iii)$a = a^*, b \neq b^*$ かつ b が得点札であるとき
$a^* < b, b^* < b$ であるので $g_{a^*,b}$ は以下の式になる。
交換前

$$1 \cdot [a^*b] + V_{n-1}^{[a^*b]}(A - \{a^*\}, B - \{b\}) = V_{n-1}^{0}(A - \{a^*\}, B - \{b\})$$

交換後

$$1 \cdot [b^*b] + V_{n-1}^{[b^*b]}(A' - \{b^*\}, B' - \{b\}) = V_{n-1}^{0}(A' - \{b^*\}, B' - \{b\})$$

上記 (ii) と同様の理由により交換前と交換後で値は変わらない。

(iv)$a \neq a^*, b = b^*$かつ a が非得点札であるとき
a^*と a 及び b^*と a の大小関係は不明なため $[a, b^*]$ と $[a, a^*]$ の値が決まらないが $[a, b^*] = [a, a^*]$ である。g_{a,b^*} は以下の式になる。
交換前

$$0 \cdot [ab^*] + V_{n-1}^{[ab^*]}(A - \{a\}, B - \{b^*\}) = V_{n-1}^{[ab^*]}(A - \{a\}, B - \{b^*\})$$

交換後

$$0 \cdot [aa^*] + V_{n-1}^{[aa^*]}(A' - \{a\}, B' - \{a^*\}) = V_{n-1}^{[aa^*]}(A' - \{a\}, B' - \{a^*\})$$

(ii) と同様の理由により交換前と交換後で値は変わらない。

(v)$a \neq a^*, b = b^*$ かつ a は得点札であるとき
$a > a^*, a > b^*b$ であるので g_{a,b^*} は以下の式になる。
交換前

$$1 \cdot [ab^*] + V_{n-1}^{[ab^*]}(A - \{a\}, B - \{b^*\}) = 1 + V_{n-1}^{1}(A - \{a\}, B - \{b^*\})$$

交換後

$$1 \cdot [aa^*] + V_{n-1}^{[aa^*]}(A' - \{a\}, B' - \{a^*\}) = 1 + V_{n-1}^{1}(A' - \{a\}, B' - \{a^*\})$$

(ii) と同様の理由により交換前と交換後で値は変わらない。

(vi)$a = a^*, b = b^*$ のとき
g_{a^*,b^*} は以下の式になる。
交換前

$$0 \cdot [a^* b^*] + V_{n-1}^{[a^* b^*]}(A - \{a^*\}, B - \{b^*\}) = V_{n-1}^{1}(A - \{a^*\}, B - \{b^*\}) \tag{7}$$

交換後

$$\begin{aligned} &0 \cdot [b^* a^*] + V_{n-1}^{[b^* a^*]}(A' - \{b^*\}, B' - \{a^*\}) \\ &= 0 + V_{n-1}^{0}(A' - \{b^*\}, B' - \{a^*\}) \end{aligned}$$

この時、$(A' - \{b^*\}, B' - \{a^*\}) = (A - \{a^*\}, B - \{b^*\})$ であるので式は次のように変形できる。

$$\begin{aligned} &V_{n-1}^{0}(A' - \{b^*\}, B' - \{a^*\}) \\ &= V_{n-1}^{0}(A - \{a^*\}, B - \{b^*\}) \end{aligned} \tag{8}$$

また $n-1$ の時の定理 5 より以下の関係を満たしている。

$$\begin{aligned} &V_{n-1}^{0}(A - \{a^*\}, B - \{b^*\}) \\ &\geq V_{n-1}^{1}(A - \{a^*\}, B - \{b^*\}) \end{aligned} \tag{9}$$

また $n-1$ の時の定理 5 より

$$V_{n-1}^{0}(A - \{a^*\}, B - \{b^*\}) \leq 2 + V_{n-1}^{1}(A - \{a^*\}, B - \{b^*\}) \tag{10}$$

以上の (i) から (vi) の場合において、カードの交換によって $V_n^{\epsilon}(A, B)$ の交換後の値が交換前の値と変わらないか、高々 2 増えるのみであることが示された。

同様に n の時の定理 8 について考える。a^*と b^*は $a^* \in A, b^* \in B, a^* < b^*$を満たしており、$a^*$は得点札ではないカード、$b^*$は得点札である。$a^*$と b^*は $A \cup B$ において連続した要素である。定理 6 と同様に証明を行う。

a^*と b^* は連続かつ a^* は非得点札、b^* は得点札なので、a^* は非得点札の中で最大のカードであり、b^* は得点札の中で最小のカードである。
(i)$a \neq a^*, b \neq b^*$ のとき
交換前

$$s_{a,b} \cdot [a, b] + V_{n-1}^{[a,b]}(A - \{a\}, B - \{b\})$$

交換後

$$s_{a,b} \cdot [a, b] + V_{n-1}^{[a,b]}(A' - \{a\}, B' - \{b\})$$

定理 8 の $n-1$ の場合より以下の関係を満たしている。

$$\begin{aligned} V_{n-1}^{[a,b]}(A-\{a\},B-\{b\})-1 &\leq V_{n-1}^{[a,b]}(A'-\{a\},B'-\{b\}) \\ &\leq V_{n-1}^{[ab]}(A-\{a\},B-\{b\})+1 \end{aligned} \tag{11}$$

定理 6 のときの (i) の場合と同様の理由により、残された手札は交換した連続しているカード a^*, b^* だけが交換前後で異なるだけなので、上の不等式に [a,b] を追加するだけである。

(ii)$a = a^*, b \neq b^*$ かつ b が得点札であるとき
$a^* < b, b^* < b$ であるので $g_{a^*,b}$は以下の式になる。
交換前

$$1 \cdot [a^*, b] + V_{n-1}^{[a^*,b]}(A-\{a^*\},B-\{b\}) = V_{n-1}^{0}(A-\{a^*\},B-\{b\}) \tag{12}$$

交換後

$$2 \cdot [b^*, b] + V_{n-1}^{[b^*,b]}(A'-\{b^*\},B'-\{b\}) = V_{n-1}^{0}(A'-\{b^*\},B'-\{b\}) \tag{13}$$

交換前の手札 $A-\{a^*\}, B-\{b\}$ と交換後の手札 $A'-\{b^*\}, B'-\{b\}$ に含まれているカード全体の得点札の枚数を比較する。Left が交換前に a^*、交換後に b^* を出しているため交換前の方が得点札が 1 枚多くなっている。しかし交換前後の手札で標準表現に変化はないため、性質 1 から $V_{n-1}^{0}(A'-\{b^*\},B'-\{b\}) \leq V_{n-1}^{0}(A-\{a^*\},B-\{b\}) \leq V_{n-1}^{0}(A'-\{b^*\},B'-\{b\})+1$ の関係が成り立つ。よって交換後の $V_n^{\epsilon}(A,B)$ は 交換前より等しいか 1 だけ小さい値となる。
(iii)$a = a^*, b \neq b^*$ かつ b が得点札でない時
$a^* > b, b^* > b$ であるので $V_n^{\epsilon}(A,B)$ は以下の式になる。
交換前

$$0 \cdot [a^*, b] + V_{n-1}^{[a^*,b]}(A-\{a^*\},B-\{b\}) = V_{n-1}^{1}(A-\{a^*\},B-\{b\}) \tag{14}$$

交換後

$$1 \cdot [b^*, b] + V_{n-1}^{[b^*,b]}(A'-\{b^*\},B'-\{b\}) = 1 + V_{n-1}^{1}(A'-\{b^*\},B'-\{b\}) \tag{15}$$

(ii) と同様に交換前の手札 $A-\{a^*\}, B-\{b\}$ と交換後の手札 $A'-\{b^*\}, B'-\{b\}$ に含まれているカード全体の得点札の枚数を比較すると、Left が交換前に a^*、交換後に b^* を出しているため、交換前の方が得点札の枚数が 1 枚多い。また交換前後の手札で標準表現に変化はないため、性質 1 から $V_{n-1}^{1}(A'-\{b^*\},B'-\{b\}) \leq V_{n-1}^{1}(A-\{a^*\},B-\{b\}) \leq 1+V_{n-1}^{1}(A'-\{b^*\},B'-\{b\})$ の関係を満たしている。よって交換後の値は交換前と変わらないか 1 だけ増えた値になる。

(iv)$a \neq a^*, b = b^*$かつ a は得点札でないとき

$a < a^*, a < b^*$ であるので $V_n^{\epsilon}(A, B)$ の式は以下の式になる。

交換前

$$1 \cdot [a, b^*] + V_{n-1}^{[a,b^*]}(A - \{a\}, B - \{b^*\}) = V_{n-1}^0(A - \{a\}, B - \{b^*\}) \tag{16}$$

交換後

$$0 \cdot [a, a^*] + V_{n-1}^{[a,a^*]}(A' - \{a\}, B' - \{a^*\}) = V_{n-1}^0(A' - \{a\}, B' - \{a^*\}) \tag{17}$$

交換前後の手札 $(A - \{a\}, B - \{b^*\})$ と $(A' - \{a\}, B' - \{a^*\})$ に残された得点札の枚数を比較すると、交換後の手札の方が得点札の枚数が多いので性質 1 から $V_{n-1}^0(A - \{a\}, B - \{b^*\}) \leq V_{n-1}^0(A' - \{a\}, B' - \{a^*\}) \leq V_{n-1}^0(A - \{a\}, B - \{b^*\}) + 1$ である。したがって交換後の値は変わらないか 1 増えるのみである。

(v)$a \neq a^*, b = b^*$ かつ a が得点札であるとき

$a > a^*, a > b^*$ であるので $V_n^{\epsilon}(A, B)$ は以下の式になる。

交換前

$$2 \cdot [a, b^*] + V_{n-1}^{[a,b^*]}(A - \{a\}, B - \{b^*\}) = 2 + V_{n-1}^1(A - \{a\}, B - \{b^*\}) \tag{18}$$

交換後

$$1 \cdot [a, a^*] + V_{n-1}^{[a,a^*]}(A' - \{a\}, B' - \{a^*\}) = 1 + V_{n-1}^1(A' - \{a\}, B' - \{a^*\}) \tag{19}$$

(ii) と同様に交換前の $A - \{a\}, B - \{b^*\}$ と交換後の手札 $A' - \{a\}, B' - \{a^*\}$ のカード全体に含まれている得点札の枚数を比較する。Right が交換前に b^*、交換後に a^* を出しているので交換後の手札は交換前の手札より得点札が 1 枚多い。ここで性質 1 から $V_{n-1}^1(A - \{a\}, B - \{b^*\}) \leq V_{n-1}^1(A' - \{a\}, B' - \{a^*\}) \leq V_{n-1}^1(A - \{a\}, B - \{b^*\}) + 1$ である。このとき、$V_{n-1}^1(A - \{a\}, B - \{b^*\}) \leq V_{n-1}^1(A' - \{a\}, B' - \{a^*\})$ であれば交換前と交換後の値の差は $2 \cdot [ab^*]$ と $1 \cdot [aa^*]1$ だけ減るのみであり、$V_{n-1}^1(A' - \{a\}, B' - \{a^*\}) \leq V_{n-1}^1(A - \{a\}, B - \{b^*\}) + 1$ であれば交換前と交換後の値の差はなくなる。よって値は変わらないか 1 減るのみである。

(vi)$a = a^*, b = b^*$ のとき

g_{a^*,b^*} は以下の式になる。

交換前

$$1 \cdot [a^*, b^*] + V_{n-1}^{[a^*,b^*]}(A - \{a^*\}, B - \{b^*\}) = V_{n-1}^0(A - \{a^*\}, B - \{b^*\}) \tag{20}$$

交換後

$$\begin{aligned}
&1 \cdot [b^*, a^*] + V_{n-1}^{[b^*,a^*]}(A' - \{b^*\}, B' - \{a^*\}) \\
&= 1 + V_{n-1}^1(A' - \{b^*\}, B' - \{a^*\})
\end{aligned}$$

この時、交換前の手札 $A' - \{b^*\}, B' - \{a^*\}$ と交換後の手札 $A - \{a^*\}, B - \{b^*\}$ の標準表現およびカードも全く同じであるので式は次のように変形できる。

$$\begin{aligned} &1 + V_{n-1}^1(A' - \{b^*\}, B' - \{a^*\}) \\ &= 1 + V_{n-1}^1(A - \{a^*\}, B - \{b^*\}) \end{aligned} \tag{21}$$

$n-1$ の時の定理 5 より以下の関係を満たしている。

$$\begin{aligned} &1 + V_{n-1}^1(A - \{a^*\}, B - \{b^*\}) \\ &\geq V_{n-1}^0(A - \{a^*\}, B - \{b^*\}) - 1 \end{aligned} \tag{22}$$

また $n-1$ の時の定理 5 より

$$(21) \leq 1 + V_{n-1}^0(A - \{a^*\}, B - \{b^*\}) \tag{23}$$

以上の (i) から (vi) の場合において、カードの交換によって $V_n^\epsilon(A, B)$ の交換後の値が交換前の値と変わらないか、高々 1 だけ増減する事が示された。

つぎに定理 5 の証明を行う。n の時の定理 5 を証明するためにカードの大小関係と二つの行列 T と W を用いる。まず、行列 T について考える。T の要素 $t_{i,j}$ は以下のようになっている。

$$t_{i,j} = \begin{cases} 0 \ (i \geq p+1 \text{ のとき}) \\ 1 \ (i \leq p, j \geq k-p+1 \text{ のとき}) \\ 2 \ (i \leq p, j \leq k-p \text{ かつ } a_i > b_j \text{のとき}) \\ 0 \ (i \leq p, j \leq k-p \text{ かつ } a_i < b_j \text{のとき}) \end{cases}$$

それぞれの T の要素について示す。$i \geq p+1$ の時は、a_i では $j \leq k-p$ のとき得点札 b_j に勝てず $[a_i b_j] = 0$ となり、$j \geq p$ のときには得点札が存在しないので $s_{i,j} = 0$ になるため、$t_{ij} = 0$ である。$i \leq p$ の時の T の要素は、手札 A,B の条件から $j \leq k-p$ では $a_i > b_j$ のき $t_{ij} = 2$、$a_i < b_j$ のとき $t_{i,j} = 0$ である。また $j \geq k-p$ では $t_{ij} = 1$ である。

次に行列 W について考える。W の要素は手札 A,B の仮定より以下の関係を満たす。

$$\forall i \ \ w_{i,k-p} \leq w_{i,k-p+1} \tag{24}$$

これは手札 A,B の仮定から成立する。$w_{i,k-p}$と $w_{i,k-p+1}$ を比較すると、$a_i > b_{k-p}$ よりどちらの場合でも次のトリックの先手プレイヤーは Left である。また、$w_{i,k-p}$ と $w_{i,k-p+1}$ のときの手札 A,B は同じ標準表現を持ち、それぞれの手札全体に含まれる得点札の枚数を比較すると、b_{k-p} が得点札であり、b_{k-p+1} は非得点札であるので $w_{i,k-p}$ のときのカード全体の得点札は 1 枚少ない。よって性質 1 より $w_{i,k-p} \leq w_{i,k-p+1}$ である。

また、行列 W の上下左右に隣接した要素同士は次のような大小関係である (図 3 参照)。

$$w_{i+1,j} \geq w_{i,j}(1 \leq i < p, 1 \leq j \leq k-p), w_{i,j} \geq w_{i,j+1}(1 \leq i \leq p, 1 \leq j < k-p) \tag{25}$$

$$w_{i+1,j} \geq w_{i,j}(p+1 \leq i < n, 1 \leq j \leq k-p), w_{i,j+1} \geq w_{i,j}(p+1 \leq i \leq n, 1 \leq j < n) \tag{26}$$

$$w_{i+1,j} \leq w_{i,j}(1 \leq i < p, k-p+1 \leq j \leq n), w_{i,j+1} \leq w_{i,j}(1 \leq i \leq p, k-p+1 \leq j < n) \tag{27}$$

$$w_{i+1,j} = w_{i,j}(p+1 \leq i < n, k-p+1 \leq j \leq n), w_{i,j} = w_{i,j+1}(p+1 \leq i \leq n, k-p+1 \leq j < n) \tag{28}$$

(25) 式から (28) 式のそれぞれについてこの関係が成り立つことを示す。

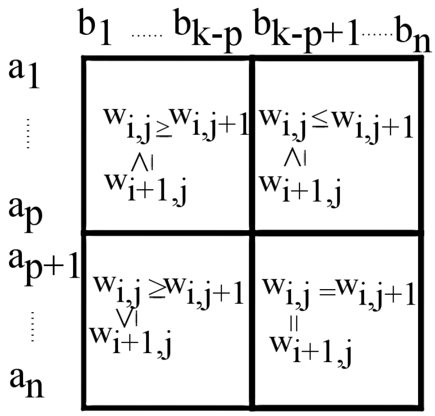

図 3　行列 W 内の各範囲における上下左右に隣接した要素の大小関係

(i) $1 \leq i \leq p, 1 \leq j \leq k-p$ のとき

$[a_{i+1}, b_j] = [a_i, b_j]$ である場合について考える。a_{i+1} を出した後の手札 $A - \{a_{i+1}\}$ と a_i を出した後の手札 $A - \{a_i\}$ に残っている a_i と a_{i+1} を比較すると、$a_{i+1} < a_i$ であり、$w_{i+1,j}$と $w_{i,j}$ のときの手札 A,B の標準表現は同一である。よって a_i と a_{i+1} を交換したものと見なすことで定理 6 の $n-1$ の仮定より $w_{i,j} \leq w_{i+1,j}$ である。

$[a_{i+1}, b_j] \neq [a_i, b_j]$ である場合には、$a_{i+1} < b_j < a_i$ なので、$[a_{i+1}, b_j] = 0$,$[a_i, b_j] = 1$ である。このとき、定理 5 の $n-1$ の仮定より

$$\begin{aligned} w_{i+1,j} &= V^0_{n-1}(A - \{a_{i+1}\}, B - \{b_j\}) \\ &\geq V^1_{n-1}(A - \{a_{i+1}\}, B - \{b_j\}) \\ &\geq V^1_{n-1}(A - \{a_i\}, B - \{b_j\}) \\ &= w_{i,j} \end{aligned} \tag{29}$$

となる。最後の不等式は定理 6 の $n-1$ の仮定に従っている。

$w_{i,j}$と $w_{i,j+1}$ についても同様に関係を示す事ができる。

(ii) $1 \leq i \leq p, k-q+1 \leq j < n$ のとき

まず $w_{i,j}$ と $w_{i+1,j}$ の関係を考える。この範囲の任意の i,j について $a_i > b_j$、つまり $[a_i, b_j] = 1$ であるのでどのカードの組み合わせでも Left が勝利する。$w_{i+1,j}$ と $w_{i,j}$ のそれぞれの手札 $A - \{a_{i+1,j}\}$ と $A - \{a_i\}$ に残された得点札について考えると、$a_{i+1} < a_i$ であるので、定理 6 の $n-1$ の場合より $w_{i+1,j} \geq w_{i,j}$ である。次に $w_{i,j}$と $w_{i,j+1}$ の関係を考える。$w_{i,j+1}$ の時の R の手札 $B - \{b_{j+1}\}$ と $w_{i,j}$ の時の手札 $B - \{b_j\}$ を比較すると、それぞれで異なるカードの関係は $b_j > b_{j+1}$ である。定理 7 の $n-1$ の仮定より、b_jと b_{j+1} の非得点カードのうち小さい値の b_{j+1} が手札に残っている方が Right は良い結果が得られる可能性がある。したがって $w_{i,j} \geq w_{i,j+1}$ となる。

(iii) $p+1 \leq i \leq n, 1 \leq j \leq k-p$ のとき

(ii) と同様にこの範囲の全ての i,j について $[a_i, b_j] = 0$ である。$a_{i+1} < a_i$ から (ii) と同様に定理 7 より $w_{i+1,j} \leq w_{i,j}$ である。

$w_{i,j}$と $w_{i,j+1}$ の関係を考えると、$b_j > b_{j+1}$ なので、定理 6 の $n-1$ の場合より、b_jと b_{j+1} の得点札であるカードのうち、大きい値の b_j が手札に残っている方が Right は良い結果が得られる可能性がある。したがって、$w_{i,j+1} \leq w_{i,j}$ である。

(iv) $p+1 \leq i \leq n, k-p+1 \leq j \leq n$ のとき

手札 A と B の仮定より、この範囲の $w_{i,j}$ は連続したカードのどれを出しても全て同じ値である。したがって明らかに $w_{i+1,j} = w_{i,j} = w_{i,j+1}$ の関係を満たしている。

次に行列 W の $\max_i \min_j w_{i,j}$ について考える。$1 \leq i \leq p$ のとき、(25)、(27) より $\min_j w_{i,j} = \min(w_{i,k-p}, w_{i,k-p+1})$ である。また、$p+1 \leq i \leq n$ のときも、(26)、(28) から $\min_j w_{i,j} = \min(w_{i,k-p}, w_{i,k-p+1})$ である。

したがって $\max_i \min_j w_{i,j}$ は次の式で表せる。

$$\max_i \min_j w_{i,j} = \max(\min(w_{1,k-p}, w_{1,k-p+1}), ..., \min(w_{p,k-p}, w_{p,k-p+1}),$$
$$\min(w_{p+1,k-p}, w_{p+1,k-p+1}), ..., \min(w_{n,k-p}, w_{n,k-p+1})) \quad (30)$$

また、(24),(25),(26) より任意の i に対して $w_{i,k-p} \leq w_{i,k-p+1}$ であるので、以下の式が成り立つ。

$$\begin{aligned}\max_i \min_j w_{i,j} &= \max(w_{1,k-p}, ..., w_{n,k-p}) \\ &= \max(w_{p,k-p}, w_{p+1,k-p}) \end{aligned} \quad (31)$$

$\max_i \min_j w_{i,j}$ と同様に行列 W の $\min_j \max_i w_{i,j}$ について考える。(25),(26) および (27),(28) より $\max_i w_{i,j} = \max(w_{p,j}, w_{p+1,j})$ である。したがって $\min_j \max_i w_{i,j}$ は次の式で表せる。

$$\min_j \max_i w_{i,j} = \min(\max(w_{p,1}, w_{p+1,1}), ..., \max(w_{p,n}, w_{p+1,n}))$$

また (24),(26) および (28) と補題 1 より $\min(w_{p,j}, w_{p+1,j}) = \min(w_{p,k-p}, w_{p+1,k-p})$ であ

る。したがって $\min_j \max_i w_{i,j}$ は次のようになる。

$$\min_j \max_i w_{i,j} = \max(w_{p,k-p}, w_{p+1,k-p}) \tag{32}$$

$V_n^1(A,B)$ および $V_n^0(A,B)$ はそれぞれ (31)、(32) より以下の式が得られる。

$$\begin{aligned} V_n^1(A,B) &= \max_i \min_j (t_{ij} + w_{ij}) \\ &\geq \max_i \min_j w_{ij} \\ &= \max(w_{p,k-p}, w_{p+1,k-p}) \end{aligned} \tag{33}$$

$$\begin{aligned} V_n^0(A,B) &= \min_j \max_i (t_{ij} + w_{ij}) \\ &\leq 2 + \min_j \max_i w_{ij} \\ &= 2 + \max(w_{p,k-p}, w_{p+1,k-p}) \end{aligned} \tag{34}$$

n の時の定理 5 の右側の不等式について (33) 式と (34) を当てはめることで、$V_n^0(A,B) \leq 2 + w_{p+1,k-p} \leq 2 + V_n^1(A,B)$ という式が得られる。左側の不等式についても $V_n^0(A,B) = 2 + V_n^1(A,B)$ のとき $V_n^1(A,B) < V_n^0(A,B)$ であるので $V_n^1(A,B) \leq V_n^0(A,B)$ である。よって定理 5 の行列 W の関係式が示された。 □

5.2　例

前節で示した定理 5 から定理 8 について具体的なゲームの例をあげる。

まず、定理 6 において獲得枚数の変化が最大になる例を示す。$n = 4$, $k = 4$ のとき、交換前の手札を $A = \{7,6,4,3\}, B = \{8,5,2,1\}$ とする。この手札では、Right が先手の場合には Left は得点札を 1 枚しか獲得できない。ここで連続した数字の得点札の 7 と 8 を交換すると手札はそれぞれ $A' = \{8,6,4,3\}, B' = \{7,5,2,1\}$ となる。交換後の手札では Right が先手であれば Left は得点札を 3 枚獲得できるため、交換前の手札より得点札 2 枚分有利である。

定理 6 は数字が連続した得点札同士の交換を行っているが、連続していない得点札同士の交換についてどれほどの変化があるかを確認する。$n = 2, k = 4$ のとき、$A = \{3,1\}, B = \{4,2\}$ のように手札が配られたとする。この手札では、Left が先手の場合、Left は得点札を獲得することができない。連続でない得点札の 4 と 1 を交換して交換後の手札を $A' = \{4,3\}, B' = \{2,1\}$ にすることを考える。この交換後の手札 A', B' では Left は得点札を明らかに 4 枚獲得できるため、Left が先手の場合には交換前と比べて 4 枚分得である。では、この連続でないカードの交換を連続したカードの交換で、Left がどれほど有利になるかを調べていく。まず 1 と 2 を交換して $A'' = \{3,2\}, B'' = \{4,1\}$ にする。このとき Left は

どちらが先手であっても得点札を 2 枚獲得出来るため、Left が先手の場合には交換前とくらべて 2 枚分有利である。次に 3 と 4 を交換して $A''' = \{4,2\}, B''' = \{3,1\}$ にする。このとき、Left が先手の場合には得点札は 2 枚獲得出来るため変化はない。最後に 2 と 3 を交換すると A', B' になる。よって連続でない得点札の交換における獲得できる得点札の枚数の変化は、段階的に見れば定理 6 の範囲内に収まっていることがわかる。

次に定理 7 において獲得枚数が 1 枚増える例を示す。$n = 4, k = 4$ のとき、交換前の手札を $A = \{8,7,5,2\}, B = \{6,4,3,1\}$ とする。このとき、Right が先手の場合には Left は得点札を 3 枚獲得する。しかし、2 と 1 を交換した時、つまり $A' = \{8,7,5,1\}, B' = \{6,4,3,2\}$ のような手札になった時、Right が先手であれば Left は得点札を 4 枚獲得できるため、交換前よりも得点札 1 枚分有利である。

最後に定理 8 において獲得枚数が 1 枚減る例を示す。$n = 4, k = 4$ で交換前の手札を $A = \{8,6,4,2\}, B = \{7,5,3,1\}$ とする。このとき、Right が先手の場合には Left は得点札を 3 枚獲得する。ここで 4 と 5 を交換すると手札はそれぞれ $\{8,6,5,2\}, \{7,4,3,1\}$ となる。一見すると Left は得点札を 3 枚手札に持っているため有利に見えるが、Right が先手の場合には Left は得点札を 2 枚しか獲得できず交換前より得点札 1 枚分不利になっている。

5.3 一般化した場合の証明について

定理 5 から定理 8 の証明は手札 A,B の条件を用いているため、一般的な手札の場合について成りたたないことがある。まず本性での仮定と異なり最小の得点札が、B に含まれていない場合を考える。このとき、行列 W において $w_{p,k-p}$ と $w_{p,k-p+1}$ の 2 つの要素の大小関係が $w_{p,k-p} \leq w_{p,k-p+1}$ が成り立つとは限らないため、$\min_j \max_i w_{i,j}$ および $\max_i \min_j w_{i,j}$ の候補を $max(w_{p,k-p}, w_{p+1,k-p})$ に絞ることができない。

次に、手札の条件の A に含まれる非得点札が B に含まれる非得点札よりも全て小さいという仮定を満たさない場合について考える。このときは行列 W 内の非得点札同士を比較する部分の要素（図 2 右下部分）の大小関係について、隣接した行列 W の要素のトリックの勝者が同じ場合について以下の補題が成り立つ。

補題 2 $[a_i, b_j] = [a_{i+1}, b_j]$ および $[a_i, b_j] = [a_i, b_{j+1}]$ のときの $w_{i,j}$ は以下の関係にある。

$$w_{i+1,j} \leq w_{i,j} \ (p+1 \leq i < n, k-p+1 \leq j \leq n)$$
$$w_{i,j} \leq w_{i,j+1} \ (p+1 \leq i \leq n, k-p+1 \leq j < n)$$

証明 まず上の不等式について示す。$w_{i,j}$ の後のトリックの手札 A と $w_{i+1,j}$ の後のトリックの手札 A を比較する。二つの手札で相違している部分は a_{i+1} と a_i であるが、この 2 枚のカードは $a_{i+1} < a_i$ という関係にある。したがって定理 7 より $w_{i+1,j} \leq w_{i,j}$ である。

次に下の不等式について示す。上の不等式と同様に $w_{i,j}$ の後のトリックの手札 B と $w_{i,j+1}$ の後のトリックの手札 B を比較すると、定理 7 より $w_{i,j}$ の方が $w_{i,j+1}$ より Right

は小さい値にできる可能性がある。したがって $w_{i,j} \leq w_{i,j+1}$ である。 □

補題 2 は隣接した行列 W の要素のトリックの勝者が同じ場合について成り立つものであるため、隣接した要素でトリックの勝者が異なる場合には行列の要素の大小関係の式は成り立たない。その例として、$n = 4, k = 4$ のときの手札 $A = \{7, 6, 4, 1\}, B = \{8, 5, 3, 2\}$ を考える。$a_3 = 4, a_4 = 1$ と $b_3 = 3$ の大小関係は $a_3 > b_3, a_4 < b_3$ であるので、隣接した行列 W の要素でトリックの勝者が異なっている。図 4 の右下の部分に注目すると、$w_{3,3} = 1, w_{4,3} = 2$ のため $w_{i+1,j} \leq w_{i,j}$ が成り立っていないことがわかる。

$$
\begin{array}{c|cc|cc|}
 & 8 & 5 & 3 & 2 \\
\hline
7 & 2 & 1 & 1 & 1 \\
6 & 2 & 1 & 1 & 1 \\
\hline
4 & 3 & 1 & 1 & 1 \\
1 & 3 & 1 & 2 & 2 \\
\hline
\end{array}
$$

図 4　A=$\{7, 6, 4, 1\}, B = \{8, 5, 3, 2\}$ のときの行列 W

6　おわりに

本論文では 4 章で得点札がカード全体のうち最大の値を持つ 2 枚の場合について、プレイヤーが獲得できる得点札の枚数を求めるアルゴリズムを示した。

5 章では得点札が複数枚ある場合について、プレイヤーの手札のある条件のもとで、得点札の枚数や先手番後手番の変化、またカードの大小関係の変化によって獲得できる得点札の枚数がどれほど変化するかを明らかにした。

全く同じ標準表現である手札であれば、得点札の枚数が多いほどプレイヤーにとって有利である。また、先手番であることは後手番であることよりも高々得点札 2 枚分不利になることがわかった。所有する得点札が数字が大きいカードであるほうが有利であり、所有する非得点札は数字が小さいカードである方が有利であることがわかった。また最も小さい得点札と最も大きい非得点札はどちらを持っていても高々得点札 1 枚分損をするか得をすることがわかった。

本論文では複数枚の得点札が存在する場合において、手札として持つカードに条件をつけ

た場合について定理を証明したが、より手札が一般的な場合についても解析を行い、問題の性質を明らかにすることが大きな課題となる。また、得点札の大きさをカード全体の最大から k 枚としたが、カード全体の最小から k 枚の場合やカード全体の中ほどの大きさであっても、得点札が連続して存在していれば今回と同様の手法を用いることで、後手番の優位性や得点札が大きい方が有利であること、非得点札が小さい方が有利であるなどの性質があると予測している。ハーツに代表されるような得点札を獲得してはならないようなルールについても、得点札をカード全体から連続した k 枚のカードであると設定することで、今回の手法と同様に解析を行い、性質を示すことが可能であると予測している。この場合、プレイヤーの目的は獲得する得点札の枚数を最大化する事から最小化する事に変わるので、定理 6 は成り立たないと思われる。しかし先手プレイヤーのカードを見てから自分の出すカードを決められる後手番の優位性や非得点札が小さいほうが有利である事は得点札を獲得しない事にも有用であると考えている。

参考文献

[1] 中井健一郎, 武永康彦, トリックテイキングゲームの計算量と必勝戦略, 数理解析研究所講究録 1799, pp.183-186, 2012.

[2] 福澤達洋, 二人トリックテイキングゲームにおける必勝性判定問題, 平成 24 年度電気通信大学大学院情報・通信工学専攻修士論文, 2013.

[3] J.Wästlund,A solution of two-person single-suit whist, The Electronic Journal of Combinatorics vol.12, 2005.

[4] J.Kahn, J.C.Lagarias, H.S.Witsenhausen, Single-Suit Two-Person Card Play, International Journal of Game Theory, Vol.16, Issue4, pp.291-320, 1987.

[5] J.Kahn, J.C.Lagarias, H.S.Witsenhausen, Single-Suit Two-Person Card Play II, Order, Vol.5, pp.45-60, 1988.

付録

本論文の性質の解明において C# にて $V_n^\epsilon(A,B)$ を求めるプログラムを作成した。そのソースコードと結果の一部を以下に示す。

ソースコード

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;

namespace ConsoleApplication1
{
  class Program
  {
    static void Main(string[] args)
    {
      //WriteFile();
      //   string filePath = @"c:\Users\0911017\Desktop\4by4_4Scorecard.txt";//出力先ファ
イル名の指定
       //   StreamWriter sw = new StreamWriter(filePath,true,System.Text.Encoding.GetEncoding("Shift_
       JIS"));//出力先、追記するかどうか、文字コード
       //   Console.SetOut(sw);//出力先設定
        int num=4;
        int k = 4;

            int[] A = new int[4] { 8,7,3,2};
            int[] B = new int[4] { 6,5,4,1};
            int[] S = new int[num*2];
            for (int i = 0; i < k; i++)
            {
                S[num*2 - i-1] = 1;

            }

           int[,] G = GenerateG(num, k, A, B,S);
           int[,] W = GenerateW(num, k, A, B, S);
          //   int[,] G = new int[2, 2] { { 1,1 }, { 0,0}};
           Console.WriteLine("---------------------------------------");
           Console.WriteLine("A is");
           showhand(num, A);
           Console.WriteLine("B is");
           showhand(num, B);
           Console.WriteLine("payoff matrix(G) value matrix(W)");
```

```
    showtwomatrix(num, G, W);
/*      Console.WriteLine("payoffmatrix is ");
     showmatrix(num, G);
     Console.WriteLine("valuematrix is");
     showmatrix(num, W);*/
  //    Console.WriteLine("A-B is");
  //    showmatrix(num,subtractionMatrix(num, G, W));
     int mami = maxmin(8, G);
     int mima = minmax(8, G);
     int mamW = maxmin(8, W);
     int mimW = minmax(8, W);
     Console.WriteLine("maxmin in G :{0} maxmin in W:{1}", mami, mamW);
     Console.WriteLine("minmax in G :{0} minmax in W:{1}", mima,mimW);
     Console.WriteLine("-----Swap cards----");
     SearchSwapG(num, k, A, B, S);

    // sw.Dispose();
    // Console.ReadKey();
 }

 public static int[,] GenerateG(int num,int k,int[] A,int[] B,int[] S){
     int[,] G = new int[num, num];
     if(num > 1){
         for (int i = 0; i < num; i++)
         {
             for (int j = 0; j < num; j++)
             {
               //  if ((A[i] != 0 )&&( B[j] != 0)){//削除されてなければ

            //          Console.WriteLine("num={0}", num);
            //          Console.WriteLine("A[{0}]={1} B[{2}]={3}", i, A[i], j, B[j]);

                     int[] AA = new int[num];
                     int[] BB = new int[num];
                     for (int l = 0; l < num; l++)
                     {
                         AA[l] = A[l];
                         BB[l] = B[l];
                     }
                         Array.Clear(AA, i, 1); //手札から使用したものを削除する
                     int[] neoA = new int[num - 1];


                     Array.Clear(BB, j, 1);
                     int[] neoB = new int[num - 1];
                     int countA = 0, countB = 0;
                 //    if (num - 1 > 1){
                         for (int l = 0; l < num; l++)
                         {
                 //            Console.WriteLine("l={0},AA[{1}]={2}", l, l, AA[l]);
                 //            Console.WriteLine("l={0},BB[{1}]={2}", l, l, BB[l]);
```

```
                        if (AA[l] != 0)
                        {
                            neoA[countA] = AA[l];
                            countA++;
                 //            Console.WriteLine("countA={0}", countA);
                        }
                        if (BB[l] != 0)
                        {
                            neoB[countB] = BB[l];
                            countB++;
                    //        Console.WriteLine("countB={0}", countB);
                        }
                    }
               //  }
                G[i, j] = CalcT(num, k, A[i], B[j],S);
                if (A[i] < B[j]) //Rが勝つとき
                {
                    G[i, j] += minmax(num-1,GenerateG(num-1, k, neoA, neoB,S));
                }
                else//Lが勝つとき
                {
                    G[i, j] += maxmin(num - 1, GenerateG(num-1, k, neoA, neoB,S));
                }
            // }
          }
        }
    }else{
      //  Console.WriteLine("num={0},a={1},b={2}",num,A[0],B[0]);
        G[0,0]=CalcT(num,k,A[0],B[0],S);
    }
    return G;
}
public static int  CalcV(int num,int k,int[] A,int[] B,int a,int b){
    int V=0;

    return V;
}
public static int CalcT(int num,int k,int a, int b,int[] S)
{
    int T = 0, scoreA=0, scoreB=0;
    if (S[a-1] == 1)
    {
        scoreA = 1;
    }
    if (S[b-1] == 1)
    {
        scoreB = 1;
    }
    if (a > b)
    {
        T = scoreA + scoreB;
```

```
 //       Console.WriteLine("Left Wins:a={0} b={1} T={2}", a, b, T);
    }
   // Console.WriteLine("T={0}", T);
    return T;
}
public static int[,] GenerateW(int num, int k, int[] A, int[] B, int[] S)
{
    int[,] G = new int[num, num];
    if (num > 1)
    {
        for (int i = 0; i < num; i++)
        {
            for (int j = 0; j < num; j++)
            {
                //  if ((A[i] != 0 )&&( B[j] != 0)){//削除されてなければ

                //         Console.WriteLine("num={0}", num);
                //         Console.WriteLine("A[{0}]={1} B[{2}]={3}", i, A[i], j, B[j]);

                int[] AA = new int[num];
                int[] BB = new int[num];
                for (int l = 0; l < num; l++)
                {
                    AA[l] = A[l];
                    BB[l] = B[l];
                }
                Array.Clear(AA, i, 1); //手札から使用したものを削除する
                int[] neoA = new int[num - 1];


                Array.Clear(BB, j, 1);
                int[] neoB = new int[num - 1];
                int countA = 0, countB = 0;
                //    if (num - 1 > 1){
                for (int l = 0; l < num; l++)
                {
                    //            Console.WriteLine("l={0},AA[{1}]={2}", l, l, AA[l]);
                    //            Console.WriteLine("l={0},BB[{1}]={2}", l, l, BB[l]);
                    if (AA[l] != 0)
                    {
                        neoA[countA] = AA[l];
                        countA++;
                        //           Console.WriteLine("countA={0}", countA);
                    }
                    if (BB[l] != 0)
                    {
                        neoB[countB] = BB[l];
                        countB++;
                        //       Console.WriteLine("countB={0}", countB);
                    }
                }
```

```
                    //  }
                 //   G[i, j] = CalcT(num, k, A[i], B[j], S);
                    if (A[i] < B[j]) //Rが勝つとき
                    {
                        G[i, j] += minmax(num - 1, GenerateG(num - 1, k, neoA, neoB, S));
                    }
                    else//Lが勝つとき
                    {
                        G[i, j] += maxmin(num - 1, GenerateG(num - 1, k, neoA, neoB, S));
                    }
                    // }
                }
            }
        }
        else
        {
            //  Console.WriteLine("num={0},a={1},b={2}",num,A[0],B[0]);
            G[0, 0] = CalcT(num, k, A[0], B[0], S);
        }
        return G;
    }

    public static void SearchSwapG(int num, int k, int[] A, int[] B, int[] S)
    {
        int[,] SS = new int[num, num];
        int[] NA = new int[num];
        int[] NB=new int[num];
        for (int i = 0; i < num; i++)
        {
            NA[i] = A[i];
            NB[i] = B[i];
        }

        for (int i = 0; i < num; i++)
        {
            for (int j = 0; j < num; j++)
            {
                if( ( (NA[i]==NB[j]+1)&&(S[NA[i]-1]==0&&S[NB[j]-1]==0) )||( (NA[i]==NB[j]-1)&&(S[NB
                [j]-1]==1))){
                    Console.WriteLine("swap a={0} to b={1}",NA[i],NB[j]);
                    int[,] SG=swapGenerateG(num, k, A, B, S, i, j);
                    int[,] SW = swapGenerateW(num, k, A, B, S, i, j);
                    Console.WriteLine("swap matrix G      swap matrix W");
                    //showmatrix(num, SG);
                    showtwomatrix(num, SG, SW);
                    Console.WriteLine("swap G maxmin = {0},swap W maxmin = {1}", maxmin(num, SG),ma
                    xmin(num,SW));
                    Console.WriteLine("swap G minmax = {0},swap W minmax = {1}", minmax(num, SG),mi
                    nmax(num,SW));
                }
                else if ((NA[i] == 1 && NB[j] == 5)||(NA[i] == 5 && NB[j] == 1))
```

```
            {
                Console.WriteLine("swap a={0} to b={1}", NA[i], NB[j]);
                int[,] SG = swapGenerateG(num, k, A, B, S, i, j);
                int[,] SW = swapGenerateW(num, k, A, B, S, i, j);
                Console.WriteLine("swap matrix G      swap matrix W");
                //showmatrix(num, SG);
                showtwomatrix(num, SG, SW);
                Console.WriteLine("swap G maxmin = {0},swap W maxmin = {1}", maxmin(num, SG), ma
                Console.WriteLine("swap G minmax = {0},swap W minmax = {1}", minmax(num, SG), mi
            }
        }

    }
}

public static void SearchSwapW(int num, int k, int[] A, int[] B, int[] S)
{
    int[,] SS = new int[num, num];
    int[] NA = new int[num];
    int[] NB = new int[num];
    for (int i = 0; i < num; i++)
    {
        NA[i] = A[i];
        NB[i] = B[i];
    }

    for (int i = 0; i < num; i++)
    {
        for (int j = 0; j < num; j++)
        {
            if ((NA[i] == NB[j] + 1) || (NA[i] == NB[j] - 1))
            {
                swapGenerateW(num, k, A, B, S, i, j);
            }
        }

    }
}

public static int[,] swapGenerateG(int num, int k, int[] A, int[] B, int[] S, int Ai, int Bj)
{
    int[] SA=new int[num];
    int[] SB=new int[num];
    for(int i=0;i<num;i++){
        SA[i]=A[i];
        SB[i]=B[i];
    }

    int swap = SA[Ai];
    SA[Ai] = SB[Bj];
    SB[Bj] = swap;
```

```
        sorthand(num, SA);
        sorthand(num, SB);
        Console.WriteLine("A is");
        showhand(num, SA);
        Console.WriteLine("B is");
        showhand(num, SB);
        return GenerateG(num, k, SA, SB, S);
    }
    public static int[,] swapGenerateW(int num, int k, int[] A, int[] B, int[] S, int Ai, int Bj)
    {
        int[] SA = new int[num];
        int[] SB = new int[num];
        for (int i = 0; i < num; i++)
        {
            SA[i] = A[i];
            SB[i] = B[i];
        }

        int swap = SA[Ai];
        SA[Ai] = SB[Bj];
        SB[Bj] = swap;

        /*
        Console.WriteLine("A is");
        showhand(num, SA);
        Console.WriteLine("B is");
        showhand(num, SB);
         * */
        sorthand(num, SA);
        sorthand(num, SB);
        return GenerateW(num, k, SA, SB, S);
    }

    public static int maxmin(int num,int[,] hand)
    {
   //     Console.WriteLine("------maxmin-------");
      //  showmatrix(num, hand);

        int i=0, j=0;
        int[] imin=new int[num];
        for(i=0;i<num;i++){
            imin[i]=10;
        }
        int[,] w = hand;
        for (i = 0; i < num; i++)
        {
            for (j = 0; j < num; j++)
            {
             //   Console.WriteLine("maxmin w[{0},{1}]={2}", i, j, w[i, j]);
                if (imin[i] >= w[i, j])//各行の最小値を求める
                {
```

```
                    imin[i] = w[i, j];
                }
            }
        }
        for (i = 0; i < num; i++)
        {
            if (imin[0] <= imin[i])//各行の最小の最大を求める
            {
                imin[0] = imin[i];
            }
        }
//      Console.WriteLine("maxmin ={0}", imin[0]);
  //    Console.WriteLine("------------------");
            return imin[0];
    }

    public static int minmax(int num,int[,] hand)
    {
      //  Console.WriteLine("------minmax-------");
     //   showmatrix(num,hand);//行列の表示

        int i = 0, j = 0;
        int[] jmax = new int[num];
        for (i = 0; i < num; i++)
        {
            jmax[i] = 0;
        }
        int[,] w = hand;
        for (j = 0; j < num; j++)
        {
            for (i = 0; i < num; i++)
            {
        //          Console.WriteLine("minmax w[{0},{1}]={2}", i,j, w[i,j]);
          //        Console.WriteLine("minmax jmax[{0}]={1}", i, jmax[i]);
                if (jmax[j] <= w[i, j])//各列の最大値を求める
                {
                    jmax[j] = w[i, j];
     //                 Console.WriteLine("minmax Change jmax[{0}]={1}", j, jmax[j]);
                }
            }
        }
        for (j = 0; j < num; j++)
        {
//              Console.WriteLine("minmax jmax[{0}]={1}", j, jmax[j]);
            if (jmax[0] >= jmax[j]) //各列の最大の中の最小を求める
            {
                jmax[0] = jmax[j];
            }
        }
//          Console.WriteLine("minmax ={0}", jmax[0]);
        //  Console.WriteLine("------------------");
```

```
    return jmax[0];
}

public static void showmatrix(int num,int[,] mat){
     for (int l = 0; l < num; l++)
    {
        for (int k = 0; k < num; k++)
        {
            if (k == num - 1)
            {
                Console.WriteLine("{0}", mat[l, k]);
            }
            else
            {
                Console.Write("{0}", mat[l, k]);
            }
        }
    }
}
public static void showhand(int num, int[] hand)
{
    for (int i = 0; i < num-1; i++)
    {
        Console.Write("{0} ", hand[i]);
    }
    Console.WriteLine("{0}", hand[num-1]);
}
public static void showtwomatrix(int num, int[,] A, int[,] B)
{
    for (int i = 0; i < num; i++)
    {
        for (int j = 0; j < 2*num+1; j++)
        {
            if (j < num)
            {
                Console.Write(A[i, j]);
            }
            else if (j == num)
            {
                Console.Write(" ");
            }
            else if (j > num && j < 2 * num )
            {
                Console.Write(B[i,j-(num+1)]);
            }else if(j== 2*num){
                Console.WriteLine(B[i, j - (num + 1)]);
            }
        }
    }
}
```

```
        public static void sorthand(int num,int[] A){
            int swap=0;
            for(int i=1;i<num-1;i++){
                for (int j = num-1; j >= i; j--)
                {
                    if (A[j-1] < A[j])
                    {
                        swap = A[j];
                        A[j] = A[j - 1];
                        A[j - 1] = swap;
                    }
                }
            }
        }
        public static int[,] subtractionMatrix(int num, int[,] matA ,int[,] matB)
        {
            int[,] diff=new int[num,num];
            for (int l = 0; l < num; l++)
            {
                for (int k = 0; k < num; k++)
                {
                    diff[l, k] = matA[l, k] - matB[l, k];
                }
            }
            return diff;
        }
        public static void WriteFile()
        {
            string filePath = @"c:\Users\0911017\Desktop\text.txt";
            string text = "AAA BBB\r\n CCC\r\n";

            StreamWriter sw = new StreamWriter(filePath, true, Encoding.ASCII);
            sw.Write(text);
            sw.Close();
        }
    }
}
```

実行結果の例

実行結果中の A、B はそれぞれ Left、Right の手札であり、Payoffmatrix は利得行列 G、value matrix は行列 W である。また各手札 A,B の場合について交換可能なカードがある場合にどのように交換した時に利得行列 G と行列 W の $\max_i \min_j w_{i,j}$ および $\min_j \max_i w_{i,j}$ が交換前後でどう変化したかを表している。

```
---------------------------------------
A is
8 7 6 5
```

```
B is
4 3 2 1
payoff matrix(G) value matrix(W)
4444 3333
4444 3333
4444 3333
4444 3333
maxmin in G :4 maxmin in W:3
minmax in G :4 minmax in W:3
-----Swap cards----
swap a=5 to b=1
A is
8 7 6 1
B is
5 4 3 2
swap matrix G      swap matrix W
4333 2222
4333 2222
4333 2222
3444 3444
swap G maxmin = 3,swap W maxmin = 3
swap G minmax = 4,swap W minmax = 3
---------------------------------------
A is
8 7 6 4
B is
5 3 2 1
payoff matrix(G) value matrix(W)
4333 2222
4333 2222
4333 2222
3444 3444
maxmin in G :3 maxmin in W:3
minmax in G :4 minmax in W:3
-----Swap cards----
swap a=4 to b=5
A is
8 7 6 5
B is
4 3 2 1
swap matrix G      swap matrix W
4444 3333
4444 3333
4444 3333
4444 3333
swap G maxmin = 4,swap W maxmin = 3
swap G minmax = 4,swap W minmax = 3
swap a=4 to b=3
A is
8 7 6 3
B is
```

```
5 4 2 1
swap matrix G      swap matrix W
4333 2222
4333 2222
4333 2222
3444 3444
swap G maxmin = 3,swap W maxmin = 3
swap G minmax = 4,swap W minmax = 3
---------------------------------------
A is
8 7 6 3
B is
5 4 2 1
payoff matrix(G) value matrix(W)
4333 2222
4333 2222
4333 2222
3444 3444
maxmin in G :3 maxmin in W:3
minmax in G :4 minmax in W:3
-----Swap cards----
swap a=3 to b=2
A is
8 7 6 2
B is
5 4 3 1
swap matrix G      swap matrix W
4333 2222
4333 2222
4333 2222
3444 3444
swap G maxmin = 3,swap W maxmin = 3
swap G minmax = 4,swap W minmax = 3
---------------------------------------
A is
8 7 6 2
B is
5 4 3 1
payoff matrix(G) value matrix(W)
4333 2222
4333 2222
4333 2222
3444 3444
maxmin in G :3 maxmin in W:3
minmax in G :4 minmax in W:3
-----Swap cards----
swap a=2 to b=1
A is
8 7 6 1
B is
5 4 3 2
```

```
swap matrix G      swap matrix W
4333 2222
4333 2222
4333 2222
3444 3444
swap G maxmin = 3,swap W maxmin = 3
swap G minmax = 4,swap W minmax = 3
---------------------------------------
A is
8 7 6 1
B is
5 4 3 2
payoff matrix(G) value matrix(W)
4333 2222
4333 2222
4333 2222
3444 3444
maxmin in G :3 maxmin in W:3
minmax in G :4 minmax in W:3
-----Swap cards----
swap a=1 to b=5
A is
8 7 6 5
B is
4 3 1 2
swap matrix G      swap matrix W
4444 3333
4444 3333
4444 3333
4444 3333
swap G maxmin = 4,swap W maxmin = 3
swap G minmax = 4,swap W minmax = 3
---------------------------------------
```