経験則を用いないカルキュレーションのプレイ

田　中　哲　朗[†]

カルキュレーション (calculation) は古くから知られたトランプの一人遊びゲームである．人間のエキスパートによるこのゲームの成功率は 3 スタックで 60% 程度，4 スタックで約 95% と言われている．

日本では以前からカルキュレーションをプレイするプログラムの作成が盛んで，4 スタックのゲームに関しては人間のエキスパートに近い成功率が達成されている．しかし，従来の手法はいずれも経験則を用いたもので，成功率を上げるには，ルールベースの場合はルールの追加や削除，評価関数を用いる場合は人手によるパラメータの調節をしながら試行錯誤を繰り返す必要があった．また，3 スタックのゲームに関しては未だにエキスパートレベルに達した例はない．

そこで，本研究では経験則を用いずにカルキュレーションをプレイする手法を提案する．これは，カルキュレーションを単純化したゲームを考え，このゲームの全局面の成功率を完全探索で求めてテーブルを作成し，そのテーブルを用いてカルキュレーションの局面における成功率を見積もる手法である．

提案する手法に従って作成したプログラムは，以前の経験則に基づくプログラムでの成功率の記録 (3 スタックで 44.6%, 4 スタックで 93%) を大幅に上回る，3 スタックで 約 67 %, 4 スタックで，約 98 %, という成功率を記録した．

A caluculation player program without using heuristic rules

Tetsuro Tanaka[†]

"Calculation" is a well known solitaire game. The success rate of the game by human experts is about 60 % with 3 stacks and about 95 % with 4 stacks.

Many computer program of the game have been written in Japan, and the success rates of these programs are very close to that of human expert players. But it is difficult to improve these programs, because they relies on heuristic rules.

In this paper, we analyse a simplified variant of the game, and constructed a huge table which include the success rates of all states in the game. Using the table, we estimate success rates of states in the original game.

With the method, the success rate of our program reached about 67 % with 3 stacks and about 98 % with 4 stacks.

1. はじめに

1.1 ゲームの説明

カルキュレーションのルールを，文献 1) から引用する．なお，用語の一部を変更してある．

目標　4 つの台に次のような列を昇順に作ること (10 は T と記述する).

台 0: `A,2,3,4,5,6,7,8,9,T,J,Q,K`
台 1: `2,4,6,8,T,Q,1,3,5,7,9,J,K`
台 2: `3,6,9,Q,2,5,8,J,1,4,7,T,K`
台 3: `4,8,Q,3,7,J,2,6,T,1,5,9,K`

† 東京大学情報基盤センター
Information Technology Center, University of Tokyo
ktanaka@ecc.u-tokyo.ac.jp

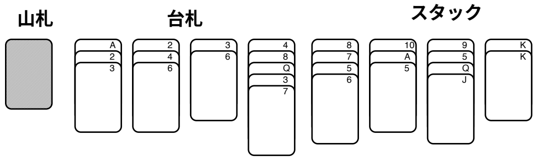

図 1　ゲーム中の様子

開始　52 枚のカードをよく切って，裏返しにしたままで山に置く．

山から引く　山から 1 枚引いて手にもつ．よく状況を考えて次の動作をする．

- **台に出す**　4 つの台のうちのどれかに出すことができるときは，出してもよい．
- **スタックに置く**　台に出さなかったときは，4 つのスタック (場あるいは屑とも呼ばれる) のどれかに表向きに置く．

これを済ませたら，次にはカードを移すことができる．

カードを移す　スタックの先頭のカードのうちに，台に出すことのできるものがあれば，それを移してもよい．これは，何回でも行なえるし，まったく行なわなくてもよい．そしてまた山からカードを引く．

終了の判定　台の 4 つの列が完成し，表面にキングが 4 枚並べば成功．山が空なら，スタックから台にカードを移せるだけ移す．どうしても移せず，スタックが残り，台が完成しなければ失敗．

スタック 5 つ以上では成功率が高すぎ，スタック 2 つでは成功率が低すぎ，いずれもゲームとしての面白味が欠けるので，スタック数 3，4 でプレイするのが一般的である．

1.2　人間によるプレイ

カルキュレーションはルールだけを習ってプレイしてみても成功率は低いが，ゲームに習熟するにしたがって，成功率はかなりのレベルまで達する．そういう意味で，カルキュレーションは「面白い」ゲームである．人間のエキスパートの値としては，3 スタックに関しては文献 1) に 6 割程度，4 スタックに関しては文献 2) に約 95 ％ と書かれている．

人間がプレイする上で役に立つ経験的規則をいくつかあげる．

(1)　スタックにカードを置く時は，後でどの台に出すつもりかどうか決めておく．

(2)　台の後の方で出すカードを積むスタックと台の前の方で出すカードと積むスタックとを区別する．

(3)　ゲーム序盤で引いたカードは台の後の方で出すカードと思うことにする．

(1) はプレイの自由度を制限しているようだが，このゲームでは後で困らないようにスタックに積むのが成功率を高めるポイントで，スタックに積む際に役割をどの台に出すつもりか決めておくとそのチェックが容易になる．

「もつれ」(ある台の早めに出るべきカードの上に，別の台の後になるまで出ないカードが乗ることによる失敗) を生じさせないためには (2) が役に立つ．特に，K を 4 枚引くまでは，「K を置ける台」を作っておかないと，K が出た途端に失敗が確定してしまうことがある．

スタックに置いたカードは簡単には台に移せない．たとえば，序盤で 9 を引いた時，台 2 の 3 番目の 9 だと思ってスタックに置いても，先行する 3, 6 と 2 枚のカードが出る前に他のカードを上に置く必要が生ずるかもしれない (他のカードを置かないようにすると，プレイにかなりの制約が加わる)．したがって，(3) のように，台 3 の 9(後ろから 2 番目) と考えて置くのが定石である．

1.3　スタックの数と成功率

4 スタックでは，人間が 95 ％ 程度成功することから，最善のプレイをすれば，100 ％ 成功するのではないかという予想が立つが，そうではないことが容易に証明できる．

スタックの数が 11 個あったとしても必ず失敗する山札のパターンがある．以下のような順で山札を引く場合を例を示す．

```
2,2,2,2,3,3,3,3,4,4,4,4,5,5,5,5,6,6,6,6,
7,7,7,7,8,8,8,8,9,9,9,9,T,T,T,T,J,J,J,J,
Q,Q,Q,Q,K,K,K,K,A,A,A,A
```

最後の 4 枚の A を残して，48 枚まで引いた局面を考える．ここまで A を 1 枚も引いていないため，台 0 には 1 枚も出せず，台 0 に出す予定のカードはすべてスタックに置かれているはずである．

台 0 に出す予定以外のカードを除いて考えると，スタックは 11 個でカードは 12 枚なので，下のように，カードが 2 枚置かれているスタックが存在する．なお以下ではスタックに置かれたカード `a-b` は b の台に出す予定のカード `a` を表す．

スタック 0: 2-0
スタック 1: 3-0
スタック 2: 4-0

```
スタック 3: 5-0
スタック 4: 6-0 7-0
スタック 5: 8-0
スタック 6: 9-0
スタック 7: T-0
スタック 8: J-0
スタック 9: Q-0
スタック 10: K-0
```

その 2 枚は山札の出現順から分かるように，台 0 の後に出す方が上に置かれているが，このようなスタックが一つでもあると必ず失敗する．

逆に，12 個のスタックがあれば必ず成功させることができることの証明も容易である．12 個のスタックをそれぞれ，対応する順位 (それぞれの台の前から何番目のカードかを表す) のカードを置くスタックとする．順位が 1 番のカードはスタックに積まずに台に出せるので，順位が 2 番から 13 番のカードが来たら，対応するスタックに何も考えずに積む．

```
スタック 0: 2-0 8-3 6-2 4-1
スタック 1: Q-3 9-2 6-1 3-0
スタック 2: Q-2 3-3 8-1 4-0
スタック 3: 5-0 T-1 2-2 7-3
スタック 4: 5-2 6-0 J-3 Q-1
スタック 5: 7-0 1-1 2-3 8-2
スタック 6: 8-0 J-2 3-1 6-3
スタック 7: 5-1 1-2 T-3 9-0
スタック 8: T-0 1-3 7-1 4-2
スタック 9: 7-2 5-3 J-0 9-1
スタック 10: J-1 Q-0 9-3 T-2
スタック 11: K-3 K-2 K-1 K-0
```

最後の山札を引いた後で，順位が 2 番目のカードに対応するスタックから順に台に移していくと，かならず成功する．

2. 従来の研究

カルキュレーションは 1978 年以来何度も GPCC の問題として取り上げられたこともあって，日本では盛んにプログラムが作成された．以下は，4 スタックのカルキュレーションに関して，文献 3) に報告されている 1990 年 1 月時点での記録に，その後の記録を加えたものである．

年月	作者	成功率
1978.1	小川貴英	30-40%
	竹内郁雄	約 75 %
1985.7	島内剛一	50 % 以上[2]
1986.5	猪飼	72 %
	孫	約 70 %
	桂川	約 65 %
1987.1	猪飼	79.4 %
	孫	70.9 %
1987.11	花澤正純	85.4[4] %
1988.9	小西憲俊	92.1 %
1990.1	中村健次郎	92.0 %
1997.8	花澤正純	93 % 強[5]

3 スタックのカルキュレーションに関しては，以下の記録がある．

年月	作者	成功率
1989.4	花澤正純	40.0 %
1997.8	花澤正純	約 46.6 %[5]

過去に作られたプログラムについて，代表的なものを 2 つ取り上げる．

まず，文献 2) では，ルールベースのアルゴリズムが紹介されている．4 本のスタックを重さ 1 から 4 までとあらかじめ決めておき，14 のルールに従ってプレイするというアルゴリズムである．簡単なルールで 50 % を超える成功率を達成できたため，カルキュレーションの研究が盛んになるきっかけともなった．

文献 5) のプログラムは文献 4) のものの改良版で，3 スタック，4 スタック共にこれまでの最高の成功率を達成している．これは人間の作成した評価関数を用いて局面評価をおこなうもので，概略は以下のようになっている．

- 4 本のスタックそれぞれに「場の阻害度」という値があらかじめ与えられる．
- 置こうとするカードについて「札の阻害度」というものを計算する．「場の阻害度」と「札の阻害度」から計算できる「有害度」の表を経験的規則により作成する．
- スタックにあるカードをどの台に移すかという割当て案を複数 (70 程度) 保持する．

これら，人間の作成したルール，評価関数を用いた手法は，成功率を上げる際，ルールや評価関数を調節しつつ大量の試行錯誤を繰り返す必要がある．

そのため，3 スタックのゲームにおいては未だ

にエキスパートレベルには達していないし，４スタックのゲームにおいてはエキスパートに近いレベルには達しながらも，それ以降の進歩は止まった状態になっている．

3. 提案する手法

本研究で提案するのは，カルキュレーションにおいて，あるプレイをした結果の局面と別のプレイをした結果の局面のどちらが好ましいかの判断を，以下の単純化したゲームの局面の集合と見なした時の成功率の比較によっておこなう手法である．

(1) スタック上のカードはどの台に出す予定か決まっている．
(2) 山札もどの台に出す予定のカードか決まっている．
(3) 台の先頭のカード (A-0, 2-1, 3-2, 4-3) は最後まで引けない．ゲーム終了時に台に先頭のカードを出し，スタックから台にすべて移せたら成功．

以下ではこのゲームのことをスタックゲームと呼ぶ．

1 の「スタック上のカードはどの台に移す予定か」は文献 5) で割当て案と呼んでいるものだが，この割当て案を複数保持するので，「スタックゲームの局面の集合」となる．

3.1 スタックゲームの完全解析

スタックゲームにおいて，ゲームの 1 局面は以下のデータのみで表すことができる．

(1) 山札の数
(2) 山札の間の順序関係 (同じスタックに置くときに，どちらが上にならなくてはいけないか)
(3) それぞれがどのスタックに置けないか

局面の表現法はいろいろあるが，本論文では連続したカードを 1 つのノードにまとめて，以下のような有向グラフで表す．

```
 (a)         (b)
[7{}] ---> [1{1}]
       (c) /      (d)
       [3{}] --> [1{1}]
```

上の (a) のノードは連続した 7 つのカードを表し，次にどこに置いても構わないことを表す．(b) のノードは 1 つのカードを表し，次に 1 のスタックに置くと成功しないこと，(a), (c) のノード中のカードの上に置くと成功しないことを表す．

この局面の成功率を計算するには，次に出る可能性のあるカード $(7+1+3+1=12)$ すべてについて，成功率最大の局面となるためにはどのスタックに積むのが良いかを求め，それらの成功率の平均を求めれば良い．たとえば，(c) のノードの 2 番目のカード 1 枚が次に出たときは，スタック 0 に置いた場合の

```
[7{}] ---> [1{0,1}]
           /
 [1{}]-> [1{0}] --> [1{0,1}]
```

スタック 1 に置いた場合の，

```
[7{}] ---> [1{1}]
           /
 [1{}]-> [1{1}] --> [1{1}]
```

スタック 2 に置いた場合の

```
[7{}] ---> [1{1,2}]
           /
 [1{}]-> [1{2}] --> [1{1,2}]
```

の 3 種類の局面への移行が考えられる．途中で，どこにも置けないカードが出現したら成功率 0 と分かるし，カードの総数が 0 になれば成功率 1 となる．ある局面から移行する局面はカードの総数が必ず 1 減っているため，初期状態から到達可能な局面数は有限となり，すべての局面を記憶する容量があれば，ゲーム中現われるすべての局面の成功率を求めることができる．

スタックゲームの完全解析をおこなうために以下の工夫をおこなった．

- 局面情報の圧縮
  局面の表現は上の有向グラフでおこなったが，ノードに属するカードの数，どのスタックに置けるかという情報を実験によって得られた頻度情報を元に Huffman 符号化することにより短いビット列で表現した．これにより 1 局面につき必要なメモリ量を 20 バイト以下におさめることができた．
- 同一局面の排除
  ノードの順序やスタックを入れ替えることにより，同じ表現になる局面は正規化により一度しかテーブルに登録しない．
- 複数プロセス
  1 プロセスあたり 2GB のメモリ空間しか使えない OS 上で作業したため，2GB を超えるテーブルを作成する際にはハッシュの管理を複数プロセスに分ける．

表 1および表 2は

[n{}]

というノードが m 個 (すなわちカードの種類が n で台が m) ある状態から始めた局面数である．

局面数は，カード数に対してほぼ指数関数的に増えていくので，手元にあった 12GB のメモリを積んだ計算機を使っても，３つ以上の台を対象にしたスタックゲームの全局面を求めることはできなかった．そこで，以下の近似に基づいて 4 つの台の局面の成功率を見積もることにする．

- 2 つの台を解析する際に途中に現われる局面の成功率は，それぞれの台に属するノードについて独立に成功率を計算してその積を取ったものに，加わった辺 (edge) による低下率を掛けたものと考える．
- 辺が加わることによる低下率は他のノードが加わっても変わらないとする．

この近似により，４つの台の局面の成功率は，台ごとの成功率の積に，４つから 2 つの台を選ぶ 6 通りの低下率を掛けたものとなる．

3.2 局面評価の例

前節のアルゴリズムを具体例で説明する．カルキュレーションの以下の局面の特定の割当て案のもとでの成功率を見積もる．割当て案はスタックのカードに移す予定の台を付記して表している．

台の状態

```
0:     3 4 5 6 7 8 9 T J Q K
1:         T Q A 3 5 7 9 J K
2: 3 6 9 Q 2 5 8 J A 4 7 T K
3:       3 7 J 2 6 T A 5 9 K
```

スタックの状態

```
0: 8-2 5-2 Q-2 A-1
1: 9-1 A-3 Q-0 J-0 6-3
2: K-3 K-1 9-3 5-3 J-1
```

以下では，山札は[] で囲んで表し，スタック上のカードは囲まずに表す．また連続したカードは[4,5,6,7,8,9,T-0] のように一つのノードで表す．台に関する順序関係は以下のようになる．

```
[4,5,6,7,8,9,T-0]->J-0->Q-0->[K-0]
[T,Q-1]->A-1->[3,5,7-1]->9-1->J-1->K-1
[6,9-2]->Q-2->[2-2]->5-2->8-2->[J,A,4,7,T,K-2]
[7,J,2-3]->6-3->[T-3]->A-3->5-3->9-3->K-3
```

スタックに置く順序によって決定される順序関係は以下のようになる．

```
A-1->Q-2->5-2->8-2
6-3->J-0->Q-0->A-3->9-1
J-1->5-3->9-3->K-1->K-3
```

これを合わせると下のようになる．

```
[4,5,6,7,8,9,T-0]->J-0->Q-0->[K-0]
                    /        \
     [7,J,2-3]->6-3->[T-3]->A-3----->5-3->9-3->K-3
                               \      /         /
   [T,Q-1]->A-1->[3,5,7-1]->9-1->J-1------->K-1
                \
     [6,9-2]->Q-2->[2-2]->5-2->8-2->[J,A,4,7,T,K-2]
```

スタックに置かれたカードよりも順序の低い山札は，そのスタックには置けないので，その情報を伝播する．

```
[4,5,6,7,8,9,T-0{}]->J-0---->Q-0->[K-0{1}]
                      /          \
   [7,J,2-3{}]->6-3->[T-3{1}]->A-3->5-3->9-3->K-3
                                   \       /     /
  [T,Q-1{}]->A-1->[3,5,7-1{0}]->9-1->J-1-->K-1
               \
   [6,9-2{}]->Q-2->[2-2{0}]->5-2->8-2->[J,A,4,7,T,K-2{0}]
```

スタック上のカードと，カードの種類はスタックゲームの成功率と関係ないので山札の数と順序関係だけを残したグラフにする．

```
[7{}] ---> [1{1}]
          /
      [3{}] --> [1{1}]
     [2{}] -> [3{0}]
        \
 [2{}] -> [1{0}] ->[6{0}]
```

まだ成功する可能性があることは，すべてのカードを置くことのできるスタックがあること，ループのないことで確かめられるので，成功率を見積もる．

台 0 に出すカードだけに注目すると，以下のグラフになる．

```
[7{}] ->[1{1}]
```

他のカードを無視した成功率を前セクションで作成したテーブルを使って求めると， $p_0 = 0.3910466$ となる，同様に $p_1 = 0.8333333$, $p_2 = 0.06856813$, $p_3 = 0.9583333$ である．

2 つの台に注目して計算してみる．台 0，３に注目したグラフは，

```
[7{}] ---> [1{1}]
          /
      [3{}] --> [1{1}]
```

となる．この成功率は $p_{03} = 0.3513412$ となる．同様に $p_{01} = 0.2739771, p_{02} = 0.01547976, p_{12} = 0.05273561, p_{13} = 0.7986111, p_{23} = 0.05320396.$

となる．全体の成功率は，台ごとの成功率の積と，４つから 2 つの台を選ぶ 6 通りの低下率の積である

$p_0 \cdot p_1 \cdot p_2 \cdot p_3 \cdot \frac{p_{01}}{p_0 p_1} \cdot \frac{p_{02}}{p_0 p_2} \cdot \frac{p_{03}}{p_0 p_3} \cdot \frac{p_{12}}{p_1 p_2} \cdot \frac{p_{13}}{p_1 p_3} \cdot \frac{p_{23}}{p_1 p_3} = 0.007281463$ と見積もれる．

局面をスタックゲームと見なした時の成功率は

表 1 スタックゲームの局面数 (3 スタック)

m \ n	1	2	3	4	5	6	7	8	9	10	11	12
1	1	3	7	14	25	41	63	92	129	175	231	298
2	2	12	65	339	1665	7757	34996	155155	681545	2977599	12964435	56313735
3	3	38	622	11664	214034	-	-	-	-	-	-	-
4	4	114	6730	543588	-	-	-	-	-	-	-	-

表 2 スタックゲームの局面数 (4 スタック)

m \ n	1	2	3	4	5	6	7	8	9	10	11	12
1	1	3	7	15	30	56	98	162	255	385	561	793
2	2	12	71	459	2937	17851	102750	566036	3028205	15680903	79991496	402134982
3	3	39	807	22628	636368	-	-	-	-	-	-	-
4	4	125	11146	1637973	-	-	-	-	-	-	-	-

実際のカルキュレーションのプレイにおける成功率と比べて低い値になっている．しかし，別々のプレイによって到達する局面において，スタックゲームと見なした時の成功率の大小がカルキュレーションの成功率の大小と高い相関を持てば，目的は達せられる．

3.3 割当て案の扱い

本手法では文献 5) と同様に，複数の割当て案を制限値まで保持し，スタックにカードを積む際にそれまで保持した割当て案を元に新たな割当て案を作成する手法を取る．文献 5) ではある局面に複数の割当て案がある時，「困難度」という評価関数を最小にする割当て案をその局面の評価値としている．

これを本手法に当てはめると各割当て案の成功率を $p_i (0 \leq i \leq n)$ とした時，$max(p_i)$ となる．

これに対して，複数の割当ての成功率が独立であると考えて $1 - \prod_{i=0}^{n}(1 - p_i)$ とする考え方もある．

それぞれの割当て案は相関が高いと考えられ，前者が自然であるが，予備実験においても前者の方が高い成功率を上げることが確かめられたので，以下では前者を用いることにする．

保持する割り当て案の数は成功率に関係すると考えられるので，以下では変更可能にして実験することにした．

3.4 先　読　み

ある局面から別々のプレイによって到達する局面において，スタックゲームと見なした時の成功率が，カルキュレーションの成功率に対して線形 (linear) なら，先読みによってより良い評価をすることが可能になる．具体的には，最善のプレイをした時の数手先のスタックゲームの成功率と数手先までの山札の出現確率の積で求められる．

表 3 カルキュレーションの成功回数 (スタック数 3)

	先読みレベル	保持割当て案数	1000 回試行	10000 回試行
(1)	0	50	656	6474
(2)	0	200	687	6749
(3)	0	1000	686	-
(4)	1	50	678	-
(5)	1	200	709	-

スタック 4 のゲームで 13 種類のカードが残っている状態では，1 手先読みするごとに $13 * 4 = 52$ 倍程度は評価の必要な局面が増えてしまい，コストがかかるため，実験では一部のものに関して 1 手だけ先読みをおこなった．

4. 評　　価

前節のアルゴリズムにしたがって，カルキュレーションをプレイするプログラムを作成した．乱数としては，以前の研究では島内乱数を使うものが多かったが，入手できなかったので，BSD 系 Unix の random を使って 10 回完全シャフルをおこなっている．いろいろな条件で比較をするため，再現性のある乱数を使っている．

保持する割当て案数を 50，200，1000 として，1000 回，10000 回試行した際の成功回数を表 3と表 4に示す．スタック 3 については先読みレベルを 1 としたものも含めているが，スタック 4 では時間の関係で試行しなかった．また，時間の関係で多くのものは 10000 回の試行ができなかった．

保持割当て案数に関しては，文献 5) では保持割当て案数 70 で実験して，それ以上は効果がなかったとしているが，先読みレベル 0, 1 の場合共に 50 の場合と比較して 200 にすると有意な改善

表4 カルキュレーションの成功回数 (スタック数 4)

	先読み レベル	保持 割当て案数	1000 回 試行	10000 回 試行
(1)	0	50	979	9822
(2)	0	200	987	-
(2)	0	1000	984	-

がみられた．保持割当て案を 1000 に変更しても成功率は，ほとんど変わらないか少し低下しているが，この試行回数でははっきりとしたことは言えない．

全体として，従来の計算機によるプレイによる成功率を大きく超え，人間のエキスパートも超える成功率を達成していることがわかる．

従来の計算機によるプレイとの違いは，ゲームの進行例を見ても分かる．付録 A.1は保持割当て案数 200，先読みレベル 1 での典型的なプレイの一部を示しているが，すぐ台に移せるカードをスタックに積むなど，従来のプログラムのプレイとは一味違った「人間的な」プレイになっている．

5. ま と め

本論文では経験則を使わずに，単純化したゲームの完全解析によるデータを元にプレイする手法を提案し，人間のエキスパートを超える成功率を達成した．しかし，最適なプレイの成功率はもっと高いと考えられる．

この手法を用いて，更に成功率を向上するためには以下のような改善が有効と考えられる．

- スタックゲームの解析の改良
  2 つの台のスタックゲームの解析を元に，4 つの台のスタックゲームの成功率を近似するという手法は情報を落している可能性がある．
  現在の手法をそのまま 3 つの台，4 つの台に当てはめるのはハードウェア の進歩を 10 年待っても無理と考えられるので，完全探索で求めた成功率を元に，よい近似式を発見するなどの改良が考えられる．
- 高速化により先読みレベルの向上
  先読みをしないのとするのとでは有意な違いが観察されている．高速化により，より先まで読めると成功率が向上することが考えられる．
- 割り当て案の増加
  割り当て案を保持する際に圧縮した表現を用いたり，同じ成功率となる割り当て案を正規化するなどして，保持可能な割り当て案の数を増やすことも有効と考えられる．

なお，今回用いたプログラムは，WWW 上 (http://www.tanaka.ecc.u-tokyo.ac.jp/~ktanaka/calc/) で公開しているので，更なる記録を目指す人は参考にして欲しい．

参 考 文 献

1) 竹内郁雄: GPCC ウルトラナノピコ問題, bit, Vol. 18, No. 3, pp. 341(1986).
2) System5,「計算」について (続)，数学セミナー, Vol. 24, No. 7, pp. 53-57(1985).
3) 小谷善行，南雲夏彦: GPCC 報告, 第 31 回プログラミング・シンポジウム報告集, pp. 189-192(1990).
4) 花澤正純: カードゲーム「計算」の計算機用アルゴリズム，情報処理学会記号処理研究会 43-4(1987).
5) 花澤正純: カリキュレーション (計算), bit 別冊「ゲームプログラミング」 松原仁，竹内郁雄編, 共立出版社, pp. 109-117(1997).
6) 小谷善行，南雲夏彦，飯田弘之，竹内郁雄，一松信: プログラミングシンポジウム GPCC のゲームとパズル, 情報処理学会ゲーム情報学研究会研究報告 1-8, pp. 55-61(1999).

付 録

A.1 ゲームの進行例

スタック数 3 で，先読みレベル 1，保持する割り当て案は 200 のゲームの進行例の一部を示す．

```
....
====================================================
7 : Q

0: A 2 3 4 5 6 7 8 9 T J Q K
1:   4 6 8 T Q A 3 5 7 9 J K
2: 3 6 9 Q 2 5 8 J A 4 7 T K
3:       3 7 J 2 6 T A 5 9 K
0: 7-2 8-0
1: J-1
2: K-3
====================================================
8 : A

0: A 2 3 4 5 6 7 8 9 T J Q K
1:   4 6 8 T Q A 3 5 7 9 J K
2: 3 6 9 Q 2 5 8 J A 4 7 T K
3:       3 7 J 2 6 T A 5 9 K
0: 7-2 8-0 A-3
1: J-1
2: K-3
====================================================
...
====================================================
18 : K
```

```
0:     3 4 5 6 7 8 9 T J Q K
1:   4 6 8 T Q A 3 5 7 9 J K
2:   6 9 Q 2 5 8 J A 4 7 T K
3:           7 J 2 6 T A 5 9 K
0: 7-1 8-2 A-3 6-3 8-0 8-1
1: J-1 5-1 T-2
2: K-3 K-1
======================================================
19 : 7

0:     3 4 5 6 7 8 9 T J Q K
1:   4 6 8 T Q A 3 5 7 9 J K
2:   6 9 Q 2 5 8 J A 4 7 T K
3:           7 J 2 6 T A 5 9 K
0: 7-1 8-2 A-3 6-3 8-0 8-1
1: J-1 5-1 T-2 7-2
2: K-3 K-1
======================================================
...
======================================================
26 : 6

0:     3 4 5 6 7 8 9 T J Q K
1:   4 6 8 T Q A 3 5 7 9 J K
2:     9 Q 2 5 8 J A 4 7 T K
3:           7 J 2 6 T A 5 9 K
0: 7-2 8-0 A-1 6-0 8-2 8-1 2-3
1: J-1 5-3 T-2 7-1 Q-0 T-3 5-1 J-0
2: K-3 K-1 9-3
======================================================
27 : 2

0:     3 4 5 6 7 8 9 T J Q K
1:   4 6 8 T Q A 3 5 7 9 J K
2:     9 Q 2 5 8 J A 4 7 T K
3:           7 J 2 6 T A 5 9 K
0: 7-1 8-2 A-3 6-3 8-0 8-1 2-3 2-2
1: J-1 5-1 T-2 7-2 Q-0 T-0 5-3 J-2
2: K-3 K-1 9-1
======================================================
....
======================================================
47 : 7

0:       4 5 6 7 8 9 T J Q K
1:       8 T Q A 3 5 7 9 J K
2:       Q 2 5 8 J A 4 7 T K
3:             2 6 T A 5 9 K
0: 7-1 8-1 A-3 6-3 8-2 8-0 2-3 2-2 6-0 Q-2 5-0
1: J-1 5-1 T-2 7-2 Q-1 T-1 5-3 J-0 T-0 T-3 A-2 J-2 9-0 7-0
2: K-3 K-1 9-1 K-0 3-1 A-1 Q-0 K-2
======================================================
48 : 9

0:       4 5 6 7 8 9 T J Q K
1:       8 T Q A 3 5 7 9 J K
2:       Q 2 5 8 J A 4 7 T K
3:             2 6 T A 5 9 K
0: 7-1 8-1 A-3 6-3 8-2 8-0 2-3 2-2 6-0 Q-2 5-0
1: J-1 5-1 T-2 7-2 Q-1 T-1 5-3 J-0 T-0 T-3 A-2 J-2 9-0 7-0
2: K-3 K-1 9-1 K-0 3-1 A-1 Q-0 K-2 9-3
======================================================
49 : 4

0:                     T J Q K
1:       8 T Q A 3 5 7 9 J K
2:           5 8 J A 4 7 T K
3:               6 T A 5 9 K
0: 7-1 8-1 A-3 6-3 8-2
1: J-1 5-1 T-2 7-2 Q-1 T-1 5-3 J-0 T-0 T-3 A-2 J-2
2: K-3 K-1 9-1 K-0 3-1 A-1 Q-0 K-2 9-3
======================================================
50 : 4

0:                     T J Q K
1:       8 T Q A 3 5 7 9 J K
2:           5 8 J A 4 7 T K
3:               6 T A 5 9 K
0: 7-1 8-1 A-3 6-3 8-2
1: J-1 5-1 T-2 7-2 Q-1 T-1 5-3 J-0 T-0 T-3 A-2 J-2
2: K-3 K-1 9-1 K-0 3-1 A-1 Q-0 K-2 9-3 4-2
======================================================
51 : 5

0:
1:
2:
3:
0:
1:
2:
======================================================
```