じゃんけんの拡張 ～大人数でも秒で決着をつけたい～

筑波大学附属駒場高校１年　米田 優峻
開成高校１年　米田 寛峻

１．はじめに

「じゃんけん」というゲームは、誰が賞品をもらうか決めたり、学校で委員を決めたりする時によく使われており、ほとんどの日本人は知っていると思います。ある日、私たちは 10 人以上の人数でじゃんけんをする機会がありました。その時、優勝者を決めるのに 30 回以上のじゃんけんが必要だったのを覚えています。

あの時から、「人数が多くても素早く優勝者を決められるように、じゃんけんを改良する方法はあるのか」ということについて考え始めました。もし優勝者を素早く決められ、かつ実用的である改良方法があれば、間違いなく今までのような「大人数でのじゃんけんに苦しむ」ということはゼロになるのではないかと考え、研究をすることにしました。

２．研究の方法

研究の対象となる問題は、「じゃんけんを使って、素早く N 人の中から**優勝者**を決める」というものです。例えば、通常のじゃんけんにおいては、「8 人のうち 5 人がグーを出し、3 人がチョキを出す」という状況であれば 5 人が勝ち、さらにまた 5 人の中でじゃんけんを行い優勝者を決める必要があります。

しかし、プレイヤーの出す手の組み合わせに対して勝者を 1 人だけ出すことにすると、すぐに優勝者が決まるので、**「勝者 1 人が決まる」または「あいこ」**に限定することを考えます。つまり、各プレイヤーの出す手 $a_1, a_2, a_3, \ldots, a_N$ ($a_i \in$ {グー, チョキ, パー}) に対して、関数 $f(a_1, a_2, a_3, \ldots, a_N) = S$ が存在し、S は必ず 1 以上 N 以下の整数（誰が勝つか）または -1 （あいこ）であるということを意味します。**当然のごとく、各プレイヤーが最適な戦略を行ったときに全てのプレイヤーが平等な確率で勝つようなじゃんけんを考えなければなりません。**例えば、2 人で通常のじゃんけんを行う時、関数の値は以下のようになります。

$a_1 \setminus a_2$	グー	チョキ	パー
グー	-1	1	2
チョキ	2	-1	1
パー	1	2	-1

$f(a_1, a_2)$ の値

なお、3 人で通常のじゃんけんをやる場合では、勝者が 2 人いる場合がありますので、このようなことは考えないことにします。

今回の研究では、まず通常のじゃんけんが如何に時間がかかるかというのを説明したうえで、上記のようなルール（勝者が 1 人　またはあいこ）のもとで、仮に人数が 100 人・1000 人程度ととても多くても、早く決着がつきかつ実用的であるじゃんけんの方法を考案しました。

ただし前提として、じゃんけんをする人は計算がコンピューターのように速くないとします。つまりできるだけ少ない計算回数で勝敗を決めたいです。

3．普通のじゃんけんの場合 - 勝敗が決まらない！

まず、一般的にプレイされているじゃんけんについて、優勝者が決まるまでの「じゃんけんの回数」の期待値についての考察をします。普通のじゃんけんは、次のようなアルゴリズムで対戦されます。

a. 各プレイヤーが「グー」「チョキ」「パー」のどれかを出す。
b. 全体で「グー」「チョキ」だけが出ていたら「グー」を出した人だけが残る。「チョキ」「パー」だけが出ていたら「チョキ」を出した人だけが残る。「パー」「グー」だけが出ていたら「パー」を出した人だけが残る。それ以外の場合、全員残る。
c. 残ったプレイヤーが 2 人以上の場合、a. に戻る。1 人になったら優勝者が決定される。

各プレイヤーがランダムに手を出すと仮定して、何回で優勝者が決まるかを、検証してみることにしました。具体的には、プレイヤーの数が 2, 3, 4, ..., 17, 18 のときに、コンピューター同士で 10000 回対戦したときの回数の平均値と分散を、コンピュータープログラムを用いて求めます。この結果は以下のようになりました。(有効数字 5 桁)

n	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
平均	1.5043	2.2355	3.1849	4.4957	6.2659	8.5975	12.099	16.913	24.278	34.896	50.328	73.368	108.71	159.82	240.70	347.62	509.66
分散	.88283	1.2951	1.8123	2.6542	3.9550	5.8584	8.6693	12.473	19.341	28.940	43.198	64.484	97.235	147.72	224.49	332.36	493.01

これを見るに、10 人でじゃんけんで優勝者を決めるとき、30 回以上のじゃんけんをする場合が多くあることが分かります。18 人となると、1000 回以上のじゃんけんをすることもあります。

これはランダムに 10000 回の試合やった結果なので、平均を厳密に計算しているわけではなりません。ここで、平均を厳密に計算する漸化式を作りました。

ここで $C(n, k)$ を「n 個の中から k 個を選ぶ場合の数 (二項係数)」と定義します。

定理 3.1 a_N を「先ほどのじゃんけんのアルゴリズムで、N 人の中から優勝者を決めるときに、じゃんけんをする回数の期待値」とします。$N \geq 2$ のとき、$a_N = \frac{1}{2^N-2} \times \left(\sum_{k=1}^{N-1} C(N, k) \times a_k + 3^{N-1} \right)$ が成立する。

[証明] 1 回のじゃんけんで $k\,(1 \leq k \leq N-1)$ 人が残るのは、「k 人がグーを出して $N-k$ 人がチョキを出す場合の数」の 3 倍となるので、$3C(n, k)$ となります。$C(n, 0) + C(n, 1) + C(n, 2) + \ldots + C(n, n) = 2^n$ なので、$k\,(1 \leq k \leq N-1)$ 人が残る場合の数の合計は $3\left(2^N - 2\right)$ 通りになることから、「全員残る (あいこ)」の場合の数は $3^N - 3\left(2^N - 2\right)$ 通りになります。このことから、次の方程式が成り立ちます。

$$a_N = \frac{1}{3^N} \left[\sum_{k=1}^{N-1} 3C(N, k) \times a_k + \left(3^N - 3\left(2^N - 2\right)\right) a_N \right] + 1$$

この方程式を a_N について解くと、$a_N = \frac{1}{2^N-2} \times \left(\sum_{k=1}^{N-1} C(N, k) \times a_k + 3^{N-1} \right)$ が得られます。(証明終わり)

また、$a_1 = 0$ であることは明らかなので、これを用いてじゃんけんの回数の期待値を求めると、次のようになります。(有効数字 5 桁)

n	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
期待値	1.5000	2.2500	3.2143	4.4857	6.2198	8.6467	12.104	17.092	24.350	34.980	50.625	73.740	107.99	158.87	234.57	347.40	515.73	767.14	1142.9

定理 3.2 $\lim_{N\to\infty} \frac{a_N}{1.5^N} = \frac{1}{3}$ が成立する。

[証明]

帰納法で $a_k = O\left(1.5^k\right)^{[*1]}$ を示します。まず、$N=1$ のとき、$a_1 = 0$ よって $O(1.5^N)$ です。次に、$N \geq 2$ のときを考えます。$1 \leq k \leq N$ のとき $a_k = O(1.5^k)$ です。$\sum_{m=1}^{k} C(k+1,\ m) \times O(1.5^m) = O\left(2.5^{k+1}\right)$ であり、これは $(1.5x+1)^{k+1}$ の x^m の係数が $C(k+1,\ m) \times 1.5^m$ だからです。$O\left(2.5^{k+1}\right) + 3^k = O\left(3^{k+1}\right)$ となるので、これを定理 3.1 の漸化式の通りに $O\left(2^{k+1}\right) = 2^{k+1} - 2$ で割ると $a_{k+1} = O\left(1.5^k\right)$ が導けます。

$a_k = O\left(1.5^k\right)$ とし、N が十分大きいとき、$\sum_{m=1}^{k} C(k+1,\ m) \times O(1.5^m) = O\left(2.5^N\right) + 3^{N-1}$ は 3^{N-1} とほぼ等しく、これを $2^N - 2$ で割った値が $\frac{1}{3} \cdot 1.5^N$ とほぼ等しくなることから、題意が示されます。(証明終わり)

これより、人数が 1 増えるごとに約 1.5 倍ずつ「じゃんけんの回数」が増えていくことが分かりました。$N = 30$ 程度となったとき、平均で約 64000 回程度のじゃんけんをする必要が出てくるので、とても現実的に可能ではありません。4. 以降では、じゃんけんを一般化することによって、平均のじゃんけんの回数を 1 に近い値にします。

4．勝敗を決まりやすくする「新しいじゃんけん」 - 3 人・4 人対戦の場合

ここでは、じゃんけんを一般化することを考えます。例えば 3 人の場合、2. で述べたように一般化すると、3 人のプレイヤーの出す手 $a_1,\ a_2,\ a_3$ に対して、$f(a_1,\ a_2,\ a_3)$ を「その手の出し方の組み合わせに対して、プレイヤー 1, 2, 3 の誰が優勝者になるか、あるいはあいこ (その場合 -1 を返す) なのか」を決定する関数とします。

この関数をうまく決めれば、じゃんけんを効率化できると考えました。しかし、じゃんけんの試合として成立するためには、次の 3 条件を満たす必要があります。

a. プレイヤーがランダムに手を出したとき、どのプレイヤーも勝つ確率は等しい
b. すべてのプレイヤーについて、「ランダムに手を出す」よりも良い戦略はない
c. あいこの確率が 100% にならない

例えば、関数を「どの手の組み合わせに対しても、$f(a_1,\ a_2,\ a_3) = 1$」のように定義すると条件 b. と c. を満たしますが、a. を満たしません。また、$T(x)$ を x がグーのとき 0、チョキのとき 1、パーのとき 2 を返すように定義すると、関数を $f(a_1,\ a_2,\ a_3) = T(a_1) + 1$ のように定義すると、条件 a., c. を満たしますが、条件 b. を満たしません。また、関数を $f(a_1,\ a_2,\ a_3) = -1$ のように定義すると、条件 a., c. を満たしますが、c. は満たしません。

まず、3 人でじゃんけんをすることについて考えます。ここでは、$a \bmod b$ を「a を b で割った余り」と定義します。

例えば、$f(a_1,\ a_2,\ a_3) = (T(a_1) + T(a_2) + T(a_3) \bmod 3) + 1$ と定義することを考えます。実は 3 つの条件すべてが成立します。これを今から確認します。

まず、条件 a. について考えます。$0 \leq b_1, b_2, b_3 \leq 2$ (b_i は整数) の中で、$b_1 + b_2 + b_3$ を 3 で割った余りが 0, 1, 2 となるものは、それぞれ 9 個ずつあります。したがって、プレイヤーがランダムに手を打った場合、どのプレイヤーも勝つ確率は 3 分の 1 になります。

次に、条件 b. について考えます。プレイヤーが b_1, b_2, b_3 に対応する手 (i.e. $b_i = T(a_i)$) を出すとします。仮にプレイヤー 1 の出す手が b_1 から $(b_1 + 1) \bmod 3$ に変わっても、プレイヤー 2 が $(b_2 + 1) \bmod 3$、プレイヤー 3 が $(b_3 + 1) \bmod 3$ を出したときの結果と変わりません。b と $(b+1) \bmod 3$ は 1 対 1 対応しているので、プレイヤー 1 の出す手を変えても確率は変わらないことになります。

最後に条件 c. ですが、この関数にあいこである場合は存在しないので、明らかに満たします。したがって、この関数はじゃんけんの試合として成立し、1 回のじゃんけんで確実に優勝者が決まります。

次に、4 人でじゃんけんをすることについて考えます。関数を次のように定義すれば、じゃんけんの試合として成立する 3 つの条件を満たします。(ここでは証明は省略します)

優勝者	(b1, b2, b3, b4) の組み合わせ	個数
プレイヤー1	(0, 0, 0, 0), (0, 0, 0, 1), (0, 0, 0, 2), (0, 0, 1, 0), (0, 0, 1, 1), (0, 0, 1, 2), (1, 1, 1, 0), (1, 1, 1, 1), (1, 1, 1, 2), (1, 1, 2, 0), (1, 1, 2, 1), (1, 1, 2, 2), (2, 2, 0, 0), (2, 2, 0, 1), (2, 2, 0, 2), (2, 2, 2, 0), (2, 2, 2, 1), (2, 2, 2, 2)	18
プレイヤー2	(0, 0, 2, 0), (0, 0, 2, 1), (0, 0, 2, 2), (0, 1, 0, 0), (0, 1, 0, 1), (0, 1, 0, 2), (1, 1, 0, 0), (1, 1, 0, 1), (1, 1, 0, 2), (1, 2, 1, 0), (1, 2, 1, 1), (1, 2, 1, 2), (2, 0, 2, 0), (2, 0, 2, 1), (2, 0, 2, 2), (2, 2, 1, 0), (2, 2, 1, 1), (2, 2, 1, 2)	18
プレイヤー3	(0, 1, 1, 0), (0, 1, 1, 1), (0, 1, 1, 2), (0, 1, 2, 0), (0, 1, 2, 1), (0, 1, 2, 2), (1, 2, 0, 0), (1, 2, 0, 1), (1, 2, 0, 2), (1, 2, 2, 0), (1, 2, 2, 1), (1, 2, 2, 2), (2, 0, 0, 0), (2, 0, 0, 1), (2, 0, 0, 2), (2, 0, 1, 0), (2, 0, 1, 1), (2, 0, 1, 2)	18
プレイヤー4	(0, 2, 0, 0), (0, 2, 0, 1), (0, 2, 0, 2), (0, 2, 1, 0), (0, 2, 1, 1), (0, 2, 1, 2), (1, 0, 1, 0), (1, 0, 1, 1), (1, 0, 1, 2), (1, 0, 2, 0), (1, 0, 2, 1), (1, 0, 2, 2), (2, 1, 0, 0), (2, 1, 0, 1), (2, 1, 0, 2), (2, 1, 2, 0), (2, 1, 2, 1), (2, 1, 2, 2)	18
あいこ	(0, 2, 2, 0), (0, 2, 2, 1), (0, 2, 2, 2), (1, 0, 0, 0), (1, 0, 0, 1), (1, 0, 0, 2), (2, 1, 1, 0), (2, 1, 1, 1), (2, 1, 1, 2)	9

すると、1 回で優勝者が決まる確率が 9 分の 8 になります。優勝者が決まるまでのじゃんけんの回数の期待値 x は、$x=\frac{1}{9}x+1$ を解いて $x=\frac{9}{8}$ 回であることが分かります。

これにより、3 人や 4 人でやるとき、従来のじゃんけんより回数を大きく減らせることが分かりました。しかし、5 人以上になると表にするのも難しくなるので、一般の N に対して考えてみることにしましたが、これにはもう 1 つ工夫をする必要があります。これを 5. で書きます。

5. 勝敗を決まりやすくする「新しいじゃんけん」 - N 人の場合で一般化できるか？

一般化するためには、先ほどの 3 つの条件を、数学的な式に書き換えなければならないと思いました。5. の前半では条件を定式化すること、後半ではこの式を用いて一般の N に対する「じゃんけんの回数の期待値」の最小値を求めたいと思います。

$b_1, b_2, b_3, ..., b_N$ $(b_i \in \{0, 1, 2\})$ に対して、$f(b_1, b_2, b_3, ..., b_N) = f(a_1, a_2, a_3, ..., a_N)$ (ただし、$b_i = T(a_i)$ になるように $a_1, a_2, a_3, ..., a_N$ を定める) と定義します。そのとき、次の定理が成り立ちます。

定理 5.1 「次の 2 つの条件をすべて満たすこと」は、「じゃんけんの試合として成立する 3 条件を満たすこと」であるための十分条件である。

A. $f(b_1, b_2, b_3, ..., b_N) = k$ を満たす N-tuple $(b_1, b_2, b_3, ..., b_N)$ の集合を S_k $(1 \le k \le N)$ とするとき、$|S_1| = |S_2| = |S_3| = \cdots = |S_N| \neq 0$ を満たす。

B. $f(b_1, b_2, b_3, ..., b_N) = f((b_1 + 1) \bmod 3,\ (b_2 + 1) \bmod 3,\ (b_3 + 1) \bmod 3, ...,\ (b_N + 1) \bmod 3)$ を満たす。

[証明]

まず条件 a. について考える。ランダムに手を出した場合に、プレイヤー k が勝つ確率が $\left|S_k\right| \div 3^N$ であるから、A. より明らかに成立する。

次に条件 b. について考える。まず、プレイヤー 1 以外の全ての人がランダムに動くと仮定する。その時、プレイヤー 1 にはランダムより良い戦略はない。条件 B より、$f(0, b_2, b_3, ..., b_N)$ と $f(1, (b_2+1) \bmod 3, ..., (b_N+1) \bmod 3)$ と $f(2, (b_2+2) \bmod 3, ..., (b_N+2) \bmod 3)$ の帰結は同じだから、例えばプレイヤー 1 がグーを出したとして、プレイヤー 1 がチョキを出した場合に対応する組み合わせが 1 通りだけ存在する。これはプレイヤー 1 が他の手を出した場合でも同様である。

したがって、プレイヤー 1 がグー、チョキ、パーのどれを出してもプレイヤー 1 が勝つ $(b_2, b_3, ..., b_N)$ の組み合わせの数は変わらない。また、これはプレイヤー 1 以外の人に対しても同じように成り立つ

逆に、ランダムに動かないプレイヤーがいる場合は一人以上に有利な手／不利な手が存在することになる。（そのようなプレイヤーは有利な手しか出さない。）1. 2. より、全てのプレイヤーにおいてランダムより良い戦略が存在しないので、全員の戦略において有利な手／不利な手は存在しないと仮定しても矛盾しない。よって全ての人がランダムに動かすことが最適な戦略の一つである。

最後に条件 c. について考えるが、$\left|S_{-1}\right| = N - \left|S_1\right| - \left|S_2\right| - \left|S_3\right| - \cdots - \left|S_N\right|$ が成立すること、また条件 A. より、$\left|S_{-1}\right| \neq N$ が成立するので、条件 c. は明らかに満たされる。(証明終わり)

これが十分条件であるが、必要条件ではないということに注意します。実は、一般化されたじゃんけんの中で、2. 節で示したように結果を「優勝者が決まる」または「あいこ」に限定したものでは、1 回で優勝者が決まる確率の最大値が $\left[\frac{3^{N-1}}{N}\right] \cdot \frac{N}{3^{N-1}}$ になります。

定理 5.2 「一般化されたじゃんけん」のうち「1 回のじゃんけんの後に残る人が 1 人、または全員 (あいこ)」であるものの中で、1 回で優勝者が決まる確率の最大値は $\left[\frac{3^{N-1}}{N}\right] \cdot \frac{N}{3^{N-1}}$ である。ただし、$[x]$ を「x を超えない整数の最大値」とする。

[証明]

プレイヤー 1 がグーを出してプレイヤー 1 以外がランダムに行動した場合に、条件 a., b. よりそれぞれのプレイヤーが勝つ確率は等しくなければならない。このとき、全体の「出す手の組み合わせ」の個数は 3^{N-1} 個であり、この中でそれぞれのプレイヤーが勝つ確率を等しくせねばならないので、最大でもそれぞれのプレイヤーが勝つ組み合わせは $\left[\frac{3^{N-1}}{N}\right]$ 通り以下となる。なので、プレイヤー 1 がグーを出した場合に 1 回で優勝者が決まる組み合わせは $\left[\frac{3^{N-1}}{N}\right] \cdot N$ 通り以下である。プレイヤー 1 がチョキ・パーを出した場合も同様なので、1 回で優勝者が決まる確率の上限が $\left[\frac{3^{N-1}}{N}\right] \cdot \frac{N}{3^{N-1}}$ であることが示された。

次に、定理 5.1 の 2 条件が満たされていても、1 回で優勝者が決まる確率が $\left[\frac{3^{N-1}}{N}\right] \cdot \frac{N}{3^{N-1}}$ になるものがあることを示す。条件 B. で帰結 (あいこかどうか、またそうでなければ勝つプレイヤー) が等しくなっていなければならない組み合わせは

- $(0, b_2, b_3, b_4, ..., b_N)$
- $(1, (b_2+1) \bmod 3, (b_3+1) \bmod 3, (b_4+1) \bmod 3, ..., (b_N+1) \bmod 3)$
- $(2, (b_2+2) \bmod 3, (b_3+2) \bmod 3, (b_4+2) \bmod 3, ..., (b_N+2) \bmod 3)$

の 3 つだけである。これを「サイクル」ということにする。なぜなら、3 番目の組み合わせに関しては 1 番目の組み合わせと帰結が同じ、となっていればよいからである。

N 人のじゃんけんにおけるサイクルは 3^{N-1} 個あり、先ほどの条件 B. の言い換えと条件 A. より「それぞれのプレイヤーが優勝するサイクルの個数はすべて同じ」でなければならない。それ以外の制約はないから、それぞれの k に対して k が優勝するサイクルを $\left[\frac{3^{N-1}}{N}\right]$ 個になるように「分配」することが

でき、余ったサイクルをあいこにすれば、1回で優勝者が決まる確率が $\left[\frac{3^{N-1}}{N}\right]\cdot\frac{N}{3^{N-1}}$ となる。よって、題意は示された。(証明終わり)

この結果から、例えば人数が 2, 3, 4, 5, 6, 7, 8, 9 のときの、1回で優勝者が決まる確率の最大値*a と、じゃんけんの回数の期待値の最小値*b は、次の表のようになる。(*b の小数は、小数第6位を四捨五入)

N	2	3	4	5	6	7	8	9
*a	2/3	1	8/9	80/81	80/81	728/729	728/729	1
*b (分数)	3/2	1	9/8	81/80	81/80	729/728	729/728	1
*b (小数)	1.50000	1.00000	1.11111	1.01250	1.01250	1.00137	1.00137	1.00000

このように、じゃんけんを一般化したら、人数が多くても1回に限りなく近い回数でじゃんけんの優勝者が決められることが分かりました。

6．実用化するための課題

5. 節で証明した条件を使って、例えば $N=4$ の場合 27 サイクル中 6 個を「A の勝ち」、6 個を「B の勝ち」、6 個を「C の勝ち」、6 個を「D の勝ち」、残る 3 個を「あいこ」とするように無作為に関数を決定すればじゃんけんとして成立させることができます。

確かに、この方法だと人数が多くてもほとんど1回しかじゃんけんを行う必要がありません。しかし、N 人対戦だと 3^{N-1} 個のサイクルが存在するので、各々のサイクルに対して誰が勝つかをいちいち決めていたら状態数が大きすぎる値になります。例えば、$N=30$ 人でじゃんけんを行う場合、$3^{29}=68,630,377,364,883$ 個の状態に対して一個一個「誰が勝者か、またはあいこか」を決める必要があります。こんなルールは一般の人が覚えられるものではありません。

ですので、よりルールが簡単で覚えやすく計算しやすいじゃんけんを考えることにしました。「実用性」にはいくつかの観点があります。

1. ルールが簡単
2. ゲームを実行するのにかかる「計算回数」が少ない

観点 1. と 2. の両方に重点を置くために、「実用性の値」を「ルールを覚えるのに必要な計算回数と、じゃんけんの帰結を判定するために必要な計算回数の合計」とします。プレイヤーの番号を割り振る部分だけは「実用性の値」の計算回数に組み込まないとします。当然、「実用性の値」が少ないほど実用的です。各々のサイクルに対して誰が勝つかをいちいち決めていたら、ルールを覚えるのに必要な計算回数は 3^N オーダーとなり、実用的ではないことがすぐにわかります。

実用的な方法を2ステップで思いついたので、1ステップごとに紹介します。7. 節では「実用性の値」を $O(N)$ にして、8. 節ではこれを $O(\log N)$ にする方法を考えます。

7．実用化できそうな解法①

[方法]

じゃんけんのうち、定理 5.1 の 2 条件を満たすものについて考えます。

$c_1=0$, $c_2=(b_2-b_1)\ mod\ 3$, $c_3=(b_3-b_1)\ mod\ 3$, …, $c_N=(b_N-b_1)\ mod\ 3$ とします。このとき、$f(c_1, c_2, c_3, ..., c_N)=f(b_1, b_2, b_3, ..., b_N)$ となります。なぜなら、$(b_1, b_2, b_3, ..., b_N)$ と $(c_1, c_2, c_3, ..., c_N)$ は同じサイクルに属しているからです。なので、帰結は $(c_2, c_3, c_4, ..., c_N)$ に依存することになります。

そこで、$c_i \in \{0, 1, 2\}$ であることを利用して、$\overline{c_N c_{N-1} c_{N-2} \cdots c_2}$ を 3 進法の数として見ることを考えます。つまり、$X = c_2 + 3c_3 + 9c_4 + 27c_5 + \ldots + 3^{N-2}c_N$ として、X の値で帰結を決めることを考えます。例えば、勝つプレイヤーの番号が $(X \bmod N) + 1$ とすることを考えます。

しかし、この方法をそのまま使うと端数が出てしまいます。例えば 4 人対戦だと (c_2, c_3, c_4) は 27 通りあるので、X のとりうる値は 0 以上 26 以下の整数となります。したがって、人 1, 2, 3 が勝つ場合の数が 7 なのにも関わらず人 4 が勝つ場合の数が 6 であり、平等ではなくなってしまいます。

なので $H = 3^{N-1} \bmod N$ とし、最後の H 個（4 人の時 $H = 3$ より $(c_2, c_3, c_4) = (2,2,0), (2,2,1), (2,2,2)$ の 3 個）を「あいこ」にすると、各人が勝つ場合の数が同じになり、平等なじゃんけんになります。

以下の図は具体例で、2 人・3 人の場合で誰が勝つのかを表しています。

【2 人の場合】

優勝者	プレイヤー 1	プレイヤー 2	あいこ
各人の出目として考えられるもの	(0, 0) (1, 1) (2, 2)	(0, 1) (1, 2) (2, 0)	(0, 2) (1, 0) (2, 1)
{c2, c3, c4, ⋯, cN} として考えられるもの	{0}	{1}	{2}

【3 人の場合】

優勝者	プレイヤー 1	プレイヤー 2	プレイヤー 3	あいこ
各人の出目として考えられるもの	(0, 0, 0) (1, 1, 1) (2, 2, 2) (0, 1, 2) (1, 2, 0) (2, 0, 1) (0, 2, 1) (1, 0, 2) (2, 1, 0)	(0, 0, 1) (1, 1, 2) (2, 2, 0) (0, 1, 0) (1, 2, 1) (2, 0, 2) (0, 2, 2) (1, 0, 0) (2, 1, 1)	(0, 0, 2) (1, 1, 0) (2, 2, 1) (0, 1, 1) (1, 2, 2) (2, 0, 0) (0, 2, 0) (1, 0, 1) (2, 1, 2)	なし
{c2, c3, c4, ⋯, cN} として考えられるもの	{0, 0} {1, 2} {2, 1}	{0, 1} {1, 0} {2, 2}	{0, 2} {1, 1} {2, 0}	なし

例えば 4 人で対戦して、プレイヤー 1 から順に「チョキ、チョキ、パー、グー」を出したとき、つまり $(b_1, b_2, b_3, b_4) = (1, 1, 2, 0)$ のとき、$(c_2, c_3, c_4) = (0, 1, 2)$ となり、$X = \overline{c_4 c_3 c_2}$ を 3 進法の数としてみると $X = 21$ になります。これを 4 で割った余りは 1 であり、「あいこ」の部分でもないので、プレイヤー 2 が勝ちます。

[分析]

この方法を使うと、1 回で優勝者が決まる確率は、定理 5.2 で求めた最大値 $\left[\frac{3^{N-1}}{N}\right] \cdot \frac{N}{3^{N-1}}$ と同じになり、ほとんどの確率で 1 回で優勝者を決めることができます。

この方法で勝者を決めるためにかかる計算の回数、つまりどれくらいこの方法が実用的であるかについて考えてみました。

まず、$3^0, 3^1, 3^2, \ldots, 3^{N-1} \bmod N$ の値は、3 を掛けて N で割った余りを取ることを繰り返せば高々 N に比例する回数の計算でできます。また、$c_2, c_3, c_4, \ldots, c_N$ を求めるのも、b_i と b_1 の差分を取るだけなので N + (定数) に比例する回数の計算でできます。これらの値が求まれば、$c_2 + 3c_3 + 9c_4 + \ldots + 3^{N-2}c_N$ が N + (定数) に比例する回数の計算でできるのは自明です。

よって、勝敗を求めるためにかかる全体的な計算回数は N + (定数) 回に比例します。$N = 10$ くらいの大きさであっても人間には数十秒で計算できる範囲ではあるので、ある程度実用的だと言えます。

8．実用化できそうな解法② - さらに高速にじゃんけんをする方法

[方法]

基本的には、7. 節の方法を借用します。ただし、8. 節の方法のキーポイントは、「勝者が決まる確率が半分以上であれば平均で 2 回以内のじゃんけんで勝者が決まる」ということを利用したものです。これを利用して、計算時間を $\log N$ に比例する回数まで改善することを考えました。

さて、B を $\log_3 N$ 以上の最小の整数とします。そして、$C = c_2 + 3c_3 + 9c_4 + ... + 3^{B-1}c_{B+1}$ とします。C を N で割った余りによって勝者が決まります。（$C \bmod N = 0$ のとき人 1 が勝ち、1 のとき人 2 が勝ち… というようになります。）また、7. 節と同じように、このじゃんけんの改良版を平等にするために、端数の部分はあいこにします。

つまり、この方法は基本的なアイデアは 7. と同じであるが、「最初の $B+1$ 人の手しか気にする必要がない」というところだけが違う、ということです。

例えば、$N = 10$ で各人が {パー, チョキ, チョキ, グー, パー, チョキ, グー, パー, グー, チョキ} としましょう。このとき, $(c_2, c_3, c_4, c_5, c_6, ..., c_{10}) = (2, 2, 1, 0, 2, 1, 0, 1, 2)$ となります。$3^3 \geq 10 \geq 3^2$ より、$B = 3$ となります。つまり最初の 4 人の手のみを気にすることになります。$c_2 + 3c_3 + 9c_4 = 17$ です。これは「あいこ」ではなく、17 を 10 で割った余りは 7 なので、人 7+1 = 人 8 が勝ちます。

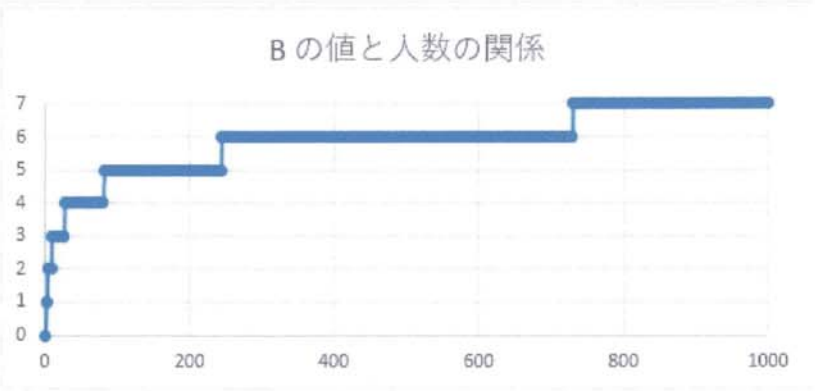
定理 8.1 この方法を用いるとき、決着が 1 回で付く確率は半分をを超える。

[証明] $(c_2, c_3, ..., c_{B+1})$ の組み合わせは 3^B 通りあります。よって、「あいこ」となる $(c_2, c_3, ..., c_{B+1})$ の組み合わせ数は $3^B \bmod N$ で、これは 0 以上 $N-1$ 以下です。$3^B \geq N$ は、「B が $\log_3 N$ 以上の最小の整数」であることより明らかに成立します。$3^B \geq 2N$ のとき、$3^B \bmod N < N \leq 3^B \div 2$ より「あいこ」の組み合わせが $3^B \div 2$ 通り 未満となります。また、$N \leq 3^B < 2N$ のとき、$N + (3^B \bmod N) = 3^B$ となり、$N > (3^B \bmod N)$ より「あいこ」の組み合わせが $3^B \div 2$ 通り未満となります。

よって、決着が 1 回のじゃんけんで付く組み合わせが $3^B - (3^B \div 2)$ 通りを超える - すなわち確率が半分を超えます。（証明終わり）

また、この方法において、$B = \log_3 N + 1$ を切り捨てたものなので、勝者を決めるためにかかる計算回数は $\log_3 N +$(定数) 回に比例します。すなわち、たとえ 100 人で改良版じゃんけんを行っても $B = 5$、世界の人口 76 億人で改良版じゃんけんを行っても $\log_3 7,600,000,000 < B$ より $B = 21$ となるので、計算回数もかなり現実的なものになります。

参考程度に、B と N の値は以下のようになります。



ですので、10 人、40 人、あるいは 100 人以上といった大人数で優勝者を平等に決めたいとき、この「改良版じゃんけん」の実用性は高そうだということが分かりました。

[おまけ - さらなる定数倍改善]

先ほどの方法でやると、例えば $N=41$ のとき $B=4$ となり、1 回で優勝者が決まる確率が 41/81、つまり平均で 81/41 = 約 2 回のじゃんけんが必要となってしまいます。しかし、$B=5$ に変更した場合には、平均で 243/205 = 約 1.19 回のじゃんけんしか必要ないです。

じゃんけんの回数の期待値を C として、計算回数の期待値が BC に比例します。$B=4$ のとき $BC=8$、$B=5$ のとき $BC=5.95$ となり、$B=log_3N+1$ 以上となる最小の整数にした方が計算回数が定数倍少なくなるといった場合があります。

一般的に、N が大きい場合は $B=log_3N+1$ にした方が良いという場合が多いです。なぜならその場合、あいこの割合が 4 分の 1 以下であることが証明できるからです。

定理 8.2 B を log_3N+1 以上の最小の整数とするとき、1 回で決着がつく確率が 4 分の 3 を超える。

[証明] $B \geq log_3N+1$ なので、$3^B \geq 3N$ を満たします。$3^B \geq 4N$ のとき、自明に「あいこの状態数」は $N-1$ 以下なので $3^B \div 4$ 未満になります。

$3N \leq 3^B < 4N$ のとき、$3^B \bmod N$ を E とするとき、$3N+E=3^B$ となり、$E \geq 3^B \div 4$ を仮定したとき、$3N+E \geq 3N+3^B \div 4 > 3^B$ (なぜなら、$4N > 3^B \Leftrightarrow 3N+3^B \div 4 > 3^B$ だから) となり $3N+E=3^B$ に矛盾します。したがって、あいこの状態数 E は $3^B \div 4$ 未満となります。

よって、$B=log_3N+1$ の場合「あいこ」の状態数は $3^B \div 4$ 以下、つまり確率が 4 分の 1 以下であることが証明できました。（証明終わり）

9．まとめ - フィナーレ

乱数生成器の無い状態で集団の中から平等に優勝者を決めたいとき、「じゃんけん」といった方法が用いられます。しかし、N 人まとめてじゃんけんをやると、およそ $\frac{1}{3} \times 1.5^N$ 回のじゃんけんが平均で必要となり、$N=100$ となると 10^{17} 回を超えてしまいます。その問題を解決するために、私たちは 7. 節あるいは 8. 節の方法を考案しました。特に 8. 節の方法は $log_3N \times$(定数) 回程度の計算で済み、仮に人数が 100 や 1000 と大きかったとしても人間が簡単に勝者を見い出せるので、乱数生成のない環境下では十分に実用的だと考えられます。

また、研究を行うにあたって、私たちはプログラミングやアルゴリズムに興味があったので、計算回数を具体的に分析をしたり、実際にプログラミングを用いて平均・標準偏差を求めたりしました。私たちは、普段様々な問題を解く時も計算回数などを考えたりすることが多く、この慣れがこの研究を目標までたどり着かせたのかもしれないと思っております。

ただし、この「改良版じゃんけん」では 1 人しか勝者を決めることが出来ません。私たちは、N 人の中から B 人の勝者を決めることのできるじゃんけんの改良版に関しても、今後考えてみようと思います。

[*1] … $O(1.5^N)$ は、どのような N に対しても $c \cdot 1.5^N$ 以下の値となるような定数 c が存在する、ということです。$f(n)=O(g(n))$ のとき、n が十分に大きな整数の時 $f(n) < cg(n)$ が成立します。（ただし c は定数です）例えば、$O(12n^3+27n^2-16n+100) = O(n^3)$、$O(2^n+n^{869120})=O(2^n)$ です。このような表記を「O 記法」「ビッグ・オー記法」といいます。アルゴリズム分野や競技プログラミングではよく使われます。

最後に、3-1 節で使った、ランダムにじゃんけんを行い、N 人のじゃんけんで勝者が決まるまでの回数の期待値と標準偏差を求めるプログラムを載せます。使ったプログラミング言語は C++ で、Visual Studio 2017 という環境で実行しました。

```
#include <iostream>
#include <vector>
#include <random>
#include <cmath>
using namespace std;

double avg = 0;
double stdev(vector<int>f) {
    double variance = 0;
    for (int i : f) variance += (avg - i) * (avg - i);
    variance /= f.size();
    return sqrt(variance);
}

int main() {
    mt19937 mt(123456789);
    uniform_int_distribution<int> p(0, 2);

    for (int t = 2; t <= 18; t++) {
        int samples = 10000;
        vector<int> dat(samples, 0); avg = 0;
        for (int q = 0; q < samples; q++) {
            int remaining = t;
            while (remaining >= 2) {
                int bit[3] = { 0, 0, 0 }; for (int i = 0; i < remaining; i++) bit[p(mt)]++;
                int V = (int)(bit[0] >= 1) + (int)(bit[1] >= 1) * 2 + (int)(bit[2] >= 1) * 4;
                if (V == 3) remaining = bit[0];
                if (V == 6) remaining = bit[1];
                if (V == 5) remaining = bit[2];
                dat[q]++; avg += 1.0 / samples;
            }
        }
        cout << "n = " << t << ": average = " << avg << ", standard deviation = " << stdev(dat) << '\n';
    }
    return 0;
}
```