Nimで遊ぼう

黒木玄

2018-09-03

- Repository: https://github.com/genkuroki/RecreationalMath/tree/master/MF2018 (https://github.com/genkuroki/RecreationalMath/tree/master/MF2018)

このファイルは次の場所でよりきれいに閲覧できる:

- nbviewer (http://nbviewer.jupyter.org/github/genkuroki/RecreationalMath/blob/master/MF2018/Nim.ipynb?flush_cache=true)

関連ウェブサイト:

- https://mathtrain.jp/nim (https://mathtrain.jp/nim)
- https://www.math.nagoya-u.ac.jp/~naito/lecture/high_school_1999/note.pdf (https://www.math.nagoya-u.ac.jp/~naito/lecture/high_school_1999/note.pdf)
- http://web.mit.edu/sp.268/www/nim.pdf (http://web.mit.edu/sp.268/www/nim.pdf)
- http://seisan.server-shared.com/524/524-2.pdf (http://seisan.server-shared.com/524/524-2.pdf)

チョコレートを折って食べるゲームは本質的にNimと同値である:

- http://chocolategame.jp/index.html (http://chocolategame.jp/index.html)

Julia言語: Julia言語 (https://www.google.co.jp/search?q=Julialang)のインストールについては

- WindowsへのJulia言語のインストール (http://nbviewer.jupyter.org/gist/genkuroki/81de23edcae631a995e19a2ecf946a4f)

を参照. このノートブックでは v0.6.4 を使用している. その理由は v1.0.0 はこれを書いている時点で Windows 上の Jupyter で利用できないから.

目次

1 組み合わせゲーム理論

組み合わせゲーム理論についてはツイッターで大量に解説をつぶやいたことがある. 以下のまとめを参照して欲しい.

- https://twitter.com/i/moments/1023916925154295809 (https://twitter.com/i/moments/1023916925154295809)
- https://twitter.com/i/moments/1023919157513834498 (https://twitter.com/i/moments/1023919157513834498)
- https://twitter.com/i/moments/1023920702749999104 (https://twitter.com/i/moments/1023920702749999104)

ポイントはゲーム全体に足し算の構造が入ることである. しかもその足し算は結合法則を満たし, 0 に対応するゲームが存在し, 足し算に関する逆元が存在する. それによってゲーム全体にはAbel群(可換群)の構造が入る.

ただし, 考えるゲームは2人が交互着手するタイプの常に有限時間で終わるゲームで自分の手番で合法的着手が不可能になった側が負けるというルールのゲームに限る.

組み合わせゲーム理論については囲碁を知っていると理解し易い:

- https://senseis.xmp.net/?CGT (https://senseis.xmp.net/?CGT)
- http://www.computer-go.jp/journal/vol5/vol5-5.pdf (http://www.computer-go.jp/journal/vol5/vol5-5.pdf)
- https://www.dumbo.ai.kyutech.ac.jp/~teigo/GoResearch/shibu2008.pdf (https://www.dumbo.ai.kyutech.ac.jp/~teigo/GoResearch/shibu2008.pdf)

2 Nim

有名な3山崩しの必勝法はゲーム全体の足し算の様子が3山崩しの場合に完全にわかってしまうことから得られる.

ゲームの足し算の定義は直観的には簡単である. 2つのゲームを合わせたものが2つのゲームを足したものになる. 3つ以上の足し算でも同様である.

例えば, a 個の石の山から交互に好きなだけ(1個以上の)石を取り去るゲームを考える. 自分の手番で取れる石が無ければ負ける.

石の山が1つしかなければ先手番がすべての石を取り去ってしまえば必ず勝てる. 相手の手番になったときに取れる石が無くなるからだ.

a 個の石の山と b 個の石の山のゲームを足して得られるゲームは

- 合法的着手はどちらか片方の山から好きなだけ(1個以上の)石を取り去ることとする

によって定義される. 2つ山のゲームでは自分の手番で2つの山の石の個数が異なっていれば必ず勝てる. 必勝法は2つの山の石の数が同じになるように石を取り去ることである. そのように石を取り去っておけば自分の手番になったときに必ず2つの山の石の数は異なる. ゆえに必ずどちらかの山に石が残っている. これを繰り返して行けばいつかは相手の手番でどちらの山の石の数も0になってしまう.

a, b, c 個の石の山のゲームを全部足して得られるゲームは

- 合法的着手はどれか1つの山から1個以上の石を取り去ることとする

と定義される. 山の個数が4つ以上でも同様である.

同じ場面における2人のプレーヤーの合法的着手の集合が等しいゲームは impartial game と呼ばれている. 以上の石取りゲームは impartial game の典型例である.

2人のプレーヤーの合法的着手が1つもないゲームを 0 と書く. 例えば, 1つ山の石取りゲームで石の個数が 0 ならばその石取りゲームは 0 になる.

$A + 0$ と A は同じゲームとみなせる. なぜならばゲーム 0 の側で2人のプレーヤーは何も着手できず, ゲーム $A + 0$ は実質的にゲーム A になってしまうからである.

2つの impartial games A, B が同値であるとは, それらの和 $A + B$ が後手必勝であることだと定義できる. (この定義は impartial ではないゲームにも拡張されている. この定義はその制限になっている.)

A が後手必勝のゲームならば $A + 0 = A$ となるので $A + 0$ は後手必勝になり, A と 0 は同値になる. 逆に A と 0 が同値ならば $A = A + 0$ が後手必勝になる. A が後手必勝になることと, A が 0 に同値なことは A が後手必勝なことの必要十分条件になる.

ゲームの足し算は以上で説明したゲームの同値関係と整合的である. ゲームの足し算をするときに, ゲームの同値をまるで等号のように扱ってよい. 以下ではゲームの同値 $\equiv$ と書くことにする.

ゲーム $A + A$ では後手必勝になる. 必勝法は2つ山の石取りゲームの場合と同様である. なぜならば, 先手が2つの A のどちらかで着手したら, 後手はもう一方の A で同じ着手をすることを繰り返せばよいからである. これで $A + A \equiv 0$ が示された. これは $-A \equiv A$ を意味する.

石の個数が a, b, c の石取りゲームをそれぞれを A, B, C と書くことにする. $A + B + C \equiv 0$ (これは $A + B \equiv C$ と論理的に独日)が成立する必要十分条件がわかれば, 石取りゲームの和 $A + B$ がどういうものであるかがわかる. 以下の結果がよく知られている.

0 以上の整数 a, b を2進数表示した結果のすべての桁を $0 \oplus 0 = 1 \oplus 1 = 0, 0 \oplus 1 = 1 \oplus 0 = 1$ というルール(2進数の各桁ごとのxor)で足し上げた結果を2進数表示として持つ 0 以上の整数を $a \oplus b$ と書く. 例えば2進数表示を最初に 0b を付けて表すことにすると

$$7 = 0\mathrm{b}0111, \quad 9 = 0\mathrm{b}1001, \quad 7 \oplus 9 = 0\mathrm{b}1110 = 14$$

$\oplus$ は結合法則, 交換法則, $a \oplus 0 = 0$, $a \oplus a = 0$ を満たす. $\oplus$ を**Nim和**と呼ぶことにする.

定理: $a \oplus b \oplus c = 0$ と $A + B + C \equiv 0$ は同値である. □

これは石の数のNim和 $a \oplus b \oplus c$ が 0 になることと, 3山の石取りゲーム $A + B + C$ が後手必勝であることが同値であることを意味する. だから, 自分の手番で $a \oplus b \oplus c \neq 0$ ならばどれかの山から適切に取り去ってNim和を 0 できれば必勝である.

この定理は実際にそのように石を取り去れることを示すことによって証明される. $a \oplus b \oplus c = r$ が 0 でないと仮定する. r の2進表示に現われる 1 と同じ a, b, c の桁のうち奇数個が 1 になっている. 特に r の2進表示の最高桁と同じ桁が 1 になっているものが a, b, c の中に存在する. それを x と書く. そのとき $x \oplus r$ は x より真に小さくなる. ゆえに x 個を減らして $x \oplus r$ 個にできる. そのとき, 全体のNim和は $r \oplus r = 0$ となる. これで $a \oplus b \oplus c \neq 0$ ならば A, B, C の山のどれかから石を取り去ってNim和を0にできる.

例: $(a, b, c) = (2, 4, 5)$ の場合. $2 = 0\mathrm{b}010, 4 = 0\mathrm{b}100, 5 = 0\mathrm{b}101$ なので $2 \oplus 4 \oplus 5 = 0\mathrm{b}11$ となる. これの最高桁と同じ桁が 1 になっているのは $2 = 0\mathrm{b}10$ である. $2 \oplus 0\mathrm{b}11 = 0\mathrm{b}01 = 1$ なので 2 の山を 1 に減らせば必勝である. □

In [1]:
```
bin(n) = string(n, base=2)
endof(a) = lastindex(a)
```

Out[1]:
```
endof (generic function with 1 method)
```

In [2]:
```
a, b, c = 2, 4, 5
@show bin.((a, b, c))
r = xor(a, b, c)
@show Int(r), bin(r)
@show y = xor(a, r)
@show xor(y, b, c)
```

```
bin.((a, b, c)) = ("10", "100", "101")
(Int(r), bin(r)) = (3, "11")
y = xor(a, r) = 1
xor(y, b, c) = 0
```

Out[2]:
```
0
```

In [3]:
```
# Nimの必勝手を表示してくれる函数

function NimStrategy(a)
    S = typeof(a)[]
    r = xor(a...)
    l = length(bin(r))
    iszero(r) && return S
    for i in 1:endof(a)
        b = bin(a[i])
        if length(b) ≥ l && b[end+1-l] == '1'
            x = vcat(a[1:i-1], xor(a[i], r), a[i+1:end])
            push!(S, x)
            println("$a -> $x")
        end
    end
    S
end
```

Out[3]:
```
NimStrategy (generic function with 1 method)
```

In [4]:
```
NimStrategy([2,4,5]);
```

```
[2, 4, 5] -> [1, 4, 5]
```

In [5]:
```
NimStrategy([15,7,4,9]);
```

```
[15, 7, 4, 9] -> [10, 7, 4, 9]
[15, 7, 4, 9] -> [15, 2, 4, 9]
[15, 7, 4, 9] -> [15, 7, 1, 9]
```

3 取れる石の個数に制限がある場合

取れる石の個数が最大 m 個までの同様のゲームを考えよう.

このとき, 相手が $m+1$ 個以上の山から石 k 個を取り去ったとき, 自分の手番で $m+1-k$ 個の石を同じ山から取れる. だから, もしもその山が $m+1$ 個減った状態で自分が必勝ならば, そうなる前の時点で自分が必勝である. したがって, すべての山から $m+1$ 個の倍数個の石を最大限取り去って, どの山の石の個数も m 個以下になった状態で自分が必勝ならば, そうする前からすでに必勝である.

どの山の石の個数も m 個以下ならば取れる石の個数が最大 m 個という制限が効いてくることはなくなる.

その場合の必勝法は上の方で述べた制限無しの石取りゲームと同じである.

In [6]:
```
# 制限付きNimの必勝手を表示してくれる函数

function NimStrategy(A, m)
    S = typeof(A)[]
    a = mod.(A, m+1)
    r = xor(a...)
    l = length(bin(r))
    iszero(r) && return S
    for i in 1:endof(A)
        b = bin(a[i])
        if length(b) ≥ l && b[end+1-l] == '1'
            x = vcat(A[1:i-1], xor(a[i], r), A[i+1:end])
            push!(S, x)
            println("$A -> $x")
        end
    end
    S
end
```

Out[6]: NimStrategy (generic function with 2 methods)

In [7]:
```
NimStrategy([2,4,5], 5);
```

```
[2, 4, 5] -> [1, 4, 5]
```

In [8]:
```
NimStrategy([2,4,5], 4);
```

```
[2, 4, 5] -> [2, 2, 5]
```

In [9]:
```
NimStrategy([2,4,5], 3);
```

```
[2, 4, 5] -> [1, 4, 5]
```

In [10]:
```
NimStrategy([2,4,5], 2);
```

```
[2, 4, 5] -> [2, 0, 5]
```

In [11]:
```
NimStrategy([15,7,4,9], 3);
```

```
[15, 7, 4, 9] -> [2, 7, 4, 9]
[15, 7, 4, 9] -> [15, 2, 4, 9]
[15, 7, 4, 9] -> [15, 7, 4, 0]
```

In [ ]:
```

```