連載 あなたの知らないプログラミングの世界
～プログラミングがこんなに面白いって知っていましたか？～

基般

アルゴリズムってこんなに楽しい
～体験で学ぶコンピュータの数理～

秋葉拓哉（国立情報学研究所 情報学プリンシプル研究系）

並び替えゲーム：選択ソート ～小さい順にカードを並べ替えよう！

優れたプログラムを作るために必要不可欠な「アルゴリズム」について，ゲームをしながら学んでいきたいと思います．用意する道具は，表に数字が書かれたカードです．トランプでも構いません．これをシャッフルして，数字が分からないように，まずは4枚伏せて置いておきます．このカードを，数字を見ないで小さい順に並び替えてもらいます．回答者（ビットくん）は2枚を選んで質問者（秋葉先生）にどちらのカードが小さいかと聞くことができます．

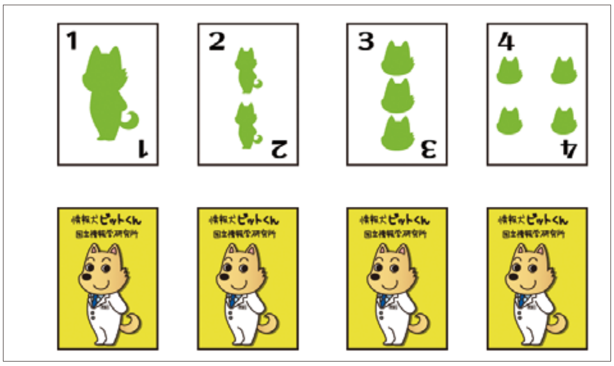


図-1　ビットカード

それでは早速，やってみましょう．

～～～～～～～～～～～～～～～～～～～～～～～～～～～～

秋葉　「今日するゲームは『並び替えゲーム』です．まずは，表に1から4の数が書いてある，4枚のカードを使います．これをシャッフルして伏せて置きます．ビットくんには，これを，数字が小さい順に並び替えてもらいます．でも，数字を見ちゃだめです」

ビット「え？　数字を見ないでどうやって並び替えるんですか？」

秋葉　「ビットくんの代わりに，僕がカードを見ます．でも，僕が教えるのは『どっちのカードが小さいか』だけです．2枚のカードを僕に渡すと，僕が『こっちのほうが小さかったよ』とだけ答えます」

ビット「大小の関係だけ教えてもらって，並び替えるんですね」

秋葉　「そうです．早速やってみましょうか」

～～～～～～～～～～～～～～～～～～～～～～～～～～～～

ビット「うーんと，それならできそう．まずこの2枚」

秋葉　「はい，こっちのほうが小さいです」

ビット「次に，これとこれ」

秋葉　「こっちのほうが小さいです」

ビット「次は，これとこれ」

秋葉　「こっちのほうが小さいです」

ビット「ということは，1のカードはこれのはず！」

秋葉　「正解！　ここに置かれていたのがいわば『チャンピオン』．毎回，チャンピオンに挑戦者が現れて，小さかった方，つまり『勝った』方が新たなチャンピオンになる．全員挑戦し終わったらそいつが真のチャンピオン，つまり一番小さいカードの1ですね．流石です」

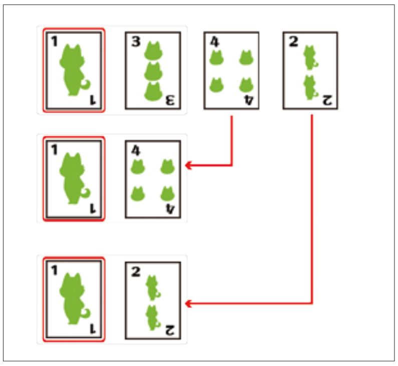


図-2　比べる順番

ビット「わーい！」

秋葉　「では，こいつは先頭で確定です．次の

2のカードは？」
ビット「あ，残っている3枚で同じことをすれば分かりますね」
秋葉　「そうです」
ビット「そうか，それで残り2枚になって，最後にその2つを比べれば，ちゃんと並べられる！」
秋葉　「その通り！というわけで早速やってみましょう」
～～～～～～～～～～～～～～～～～～～～～～～～
ビット「できました！」
秋葉　「はい，ばっちりです」
ビット「やったー」
秋葉　「この決まったやり方なら，このゲーム，簡単ですよね？」
ビット「はい．機械的なので，自信を持って，何度でもできそうです」
～～～～～～～～～～～～～～～～～～～～～～～～

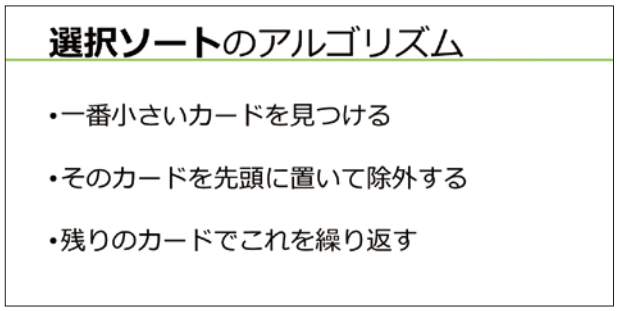


図-3　選択ソート

こういった作業を実現するための手順，これが『アルゴリズム』です．ここで行った手順は，『選択ソート』という名前のついた有名なアルゴリズムです．ソートというのは並び替えのことで，『選択ソート』という名前は，毎回，一番小さいカードを選択する，ということを繰り返すからついた名前です．今回はビットくんが自分でやりましたが，これくらい機械的な手順ならコンピュータにもプログラミングを通じてやってもらえそうですよね．

コンピュータにやってもらいたい仕事は，たくさんあります．たとえば今回のような，数字の並び替えをしてほしいとしましょう．『並び替えてほしいな』となんとなく思っただけでは，プログラムにはできません．どうやって並び替えるか，という具体的な手順を考えないとプログラムにはなりません．その具体的なやり方が『アルゴリズム』です．

プログラムは，コンピュータに対して命令をするものです．一方，アルゴリズムは考え方です．同じアルゴリズムでも違うコンピュータにやらせるためには違うプログラム言語でプログラムを作りなおさないといけないかもしれません．でも，考え方であるアルゴリズムは同じものが使えます．

並び替えゲーム：マージソート ～効率の良い並べ替えにトライ！

次は，同じく4枚のカードを小さい順に並べ替えます．ただし，カードを1回確認してもらうためには，質問者（秋葉）にコインを1枚渡す必要があります．回答者（ビット）にはあらかじめ，5枚のコインを渡しておきます．コインがなくなってしまったら，ゲームオーバーです．

それでは早速，やってみましょう．

～～～～～～～～～～～～～～～～～～～～～～～～
ビット「やっぱりコイン足りないんだびっと」
秋葉　「そうですね．でも，実はコイン5枚でも，必ず並び替えを完成させられる方法があるんです．さっきと違う方法を考えてみましょう．最初に1番目（A）のカードと2番目（B）のカードをまず比較し小さい順に縦に並べます（小さい方をAとし，大きい方をBとする）．次に，3番目のカード（C）と4番目（D）のカードを比較し，小さい順に縦に並べでみましょう（小さい方をDとし，大きい方をCとする）」＜コイン2枚消費＞
秋葉　「さて，一番小さいカードはどーれだ．」
ビット「それは，これ（A）とこれ（D）のうち小さい方が1のカードです．でも，これってさっきと同じじゃ？」＜コイン1枚消費＞
秋葉　「違うのは次からです．じゃー2番目のカードはどれだ？　実はコイン1枚で分かるよ．なんでこのカード上に置いたんだっけ？」
ビット「あ，BかDの小さいほうだ！」
秋葉　「正解～．というわけでこれが2のカード」

＜コイン１枚消費＞

ビット「やった！　残り２枚．残っているコインは１枚だけど，こいつらを比べて終わりだ」＜コイン１枚消費＞

秋葉　「その通り．こっちが小さくてこっちが大きいので，こっちが３で，こっちが４ですね．この方法なら，コイン５枚で必ず並び替えられるよ」

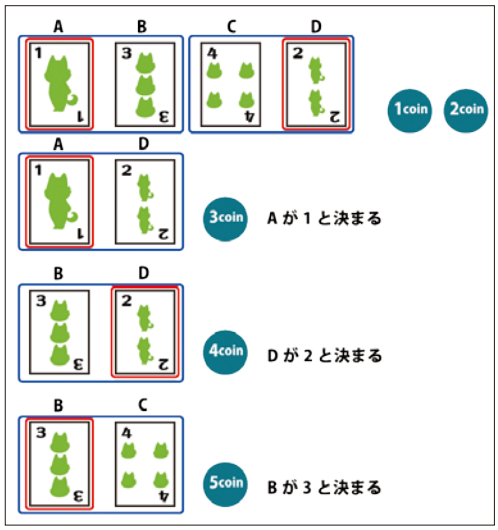


図 -4
5 つのコインで決まる

～～～～～～～～～～～～～～～～～～～～～～～～～～～～

ポイントは，最初に２つのグループに分けたことです．２枚のカードを比較して順番に並べました．２つの小さい順に並んでいるグループから，１つの小さい順に並んでいるグループを作ります．先頭に来るカードは，２つのグループの先頭のどちらかになるので，比較して，先頭を確定します．その次にくるカードも，残っているカードの先頭のどちらかということになります．比較して確定という感じです．

さて次は同じ要領で８枚のカードとコイン 17 枚でやってみましょう．

～～～～～～～～～～～～～～～～～～～～～～～～～～～～

ビット「一気に増えましたびっと……」

秋葉　「大ヒントを出すよ．まず，４枚ずつに分けて，さっきのやり方でそれぞれを小さい順に並べてみよう」

ビット「はーい」

（早送り）

秋葉　「よしよし．残っているコインは７枚．さて，この２つのグループから，小さい順に並べてみましょう．さっきと同じように，並び替えを完成させられるかな？　まずは１のカードはどれかな？」

ビット「これとこれの小さい方」

秋葉　「……その通り．次は？」

ビット「今度はこれとこれの小さい方」

秋葉　「ですね．次は？」

ビット「これとこれ」

秋葉　「そう．後はこれを繰り返せば」

（早送り）

ビット「できた！　17 枚ぴったり！」

秋葉　「そうです！」

～～～～～～～～～～～～～～～～～～～～～～～～～～～～

では復習してみましょう．８枚をまず２つのグループに分けました．その２つのグループで，さきほど４枚でやったときと同じ方法で並び替え，最後に小さい方から比べていきました．もし 16 枚だったら，今度は８枚ずつに分けて比べます．大分頭がこんがらがってきますが，８枚をそれぞれ並び替えた後で，小さい方から取っていけば，16 枚も並び替えることができますね．32 枚も同様です．

マージソートのアルゴリズム

- カードを 2 つのグループに分ける
- 2 つのグループを別々に並び替える
- 2 つのグループを 1 つにまとめる

図 -5　マージソート

このアルゴリズムは『マージソート』という名前を持ったアルゴリズムです．マージソートのアルゴリズムは，まずカードを２つのグループに分けます．そして２つのグループをそれぞれで並び替えたあとで，同じアルゴリズム，すなわち「自分自身」を適用します．この面白いところは，２つめのステップです．２つのグループを別々に並び替えるということをやっていますが，この並び替える部分でも

マージソートをします．たとえば 8 枚を並び替えるのであれば 4 枚 4 枚に分けたあと，4 枚でマージソートを行っています．16 枚だと，8 枚 8 枚に分けて，8 枚をマージソートし，さらに 4 枚 4 枚に分けてマージソートをする．こうやってアルゴリズムの中で自分自身を使うことを，『再帰処理』といいます．

ここまで，選択ソートとマージソートのアルゴリズムを勉強しました．並び替えという目的に対して，複数のアルゴリズムがあるということが分かりました．これが面白いところです．1 つの目的でもやりかたは 1 つではないということです．

使うコインの枚数

カードの枚数	選択ソートのコイン	マージソートのコイン
4 枚	3+2+1 = 6 枚	5 枚
8 枚	7+6+...+1 = 28 枚	17 枚
16 枚	15+14+...+1= 120 枚	49枚

図 -6　使うコインの枚数

選択ソートは，4 枚のカードを並び替えるのに，3 ＋ 2 ＋ 1 ＝ 6 枚のコインが必要でした．マージソートは 5 枚のコインが必要でしたね．これが，アルゴリズムの効率の違いです．このようにアルゴリズムの性能・効率というのものが，やり方によって変わってきます．

面白いことに，カードの枚数が増えていくほど，差が開いていきます．8 枚のカードでは，選択ソートは 7 ＋ 6 ＋ 5 ＋ 4･･･＝ 28 枚のコインが必要です．マージソートは 17 枚でしたね．16 枚のカードだと，選択ソートは 15 ＋・・・＝ 120 枚のコインが必要で，一方，マージソートは 49 枚のコインで済みます．マージソートは本当にとても効率の良い並び替えのアルゴリズムで，実際にもプログラムのいたるところで使われています．

というわけで，今回は並び替えゲームを題材にして，並び替えのアルゴリズムについて体験して学んでもらいました．アルゴリズムとそのパズル的な面白さを理解してもらえていれば幸いです．

脱出迷路：難しい迷路に迷わないように右手でサバイバルしよう！

今回使うのは，迷路です．壁は通れません．ペンで経路をたどってみてください．

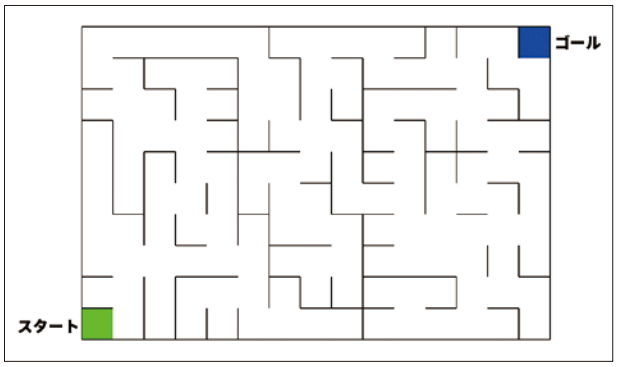


図 -7　迷路

～～～～～～～～～～～～～～～～～～～～～～～～～～～～

秋葉　「ビットくん，なかなかできませんねぇ」

ビット「悩むんだびっと……」

秋葉　「今回は，絶対に混乱せずに迷路をクリアできる必勝法を伝授します！　その名も『右手法』という名前のアルゴリズムです」

ビット「右手法……右手を使うんですか」

秋葉　「はい，いたってシンプルなやり方です．自分がここ，スタート地点に立っているとしましょう．どちらでもいいのですが，こちら向きにスタートするとします．右手を壁につけます．そのまま，右手をついたままで進み続けましょう．角でも絶対に右手を離さず，角に沿って曲がります」

ビット「やってみます」

(早送り)

ビット「できましたびっと！！！」

秋葉　「簡単でしょ？　右手を壁につけて移動すると絶対に解ける．これが右手法というやりかたです」

右手法のアルゴリズム

- 壁に右手をついたまま進む
- ゴールにたどり着いたら完成
- ゴールにたどり着けなければインチキ迷路

図 -8　右手法

ビット「なんで右手法で迷路が解けるんですか？」
秋葉　「スタート地点が壁につながっているし，ゴールも壁につながっている．つまり，壁を伝っていけば，必ずゴールに行けるはずなんです．右手をついていくと，実は全部の壁を一周するんですね．なので，右手をついたまま移動していけば，必ずゴールにたどり着くことができます」
ビット「じゃぁ，たとえば大きな迷路に閉じ込められちゃったとしても，この右手法を使えば脱出できるんですね」
秋葉　「はい．右手法を使えば，安心して出られる，というわけです．まだここで終わりじゃないです．次は，ゴールまで，一番短い時間（距離？）でたどり着く方法は分かりますか？」
ビット「さっきのだとちょっと遠回りな感じだびっと．うーん．こうですか？」
秋葉　「それが本当に一番短い経路だと，自信をもって言えますか？」
ビット「なんとなく，ですから，分からないですね」
秋葉　「なんとなくで一番短い経路を見つけるのは，結構難しいです．なので，絶対に一番短い行き方を見つける方法を教えます」
ビット「絶対，ですか」

～～～～～～～～～～～～～～～～～～～～～～～～～～～～

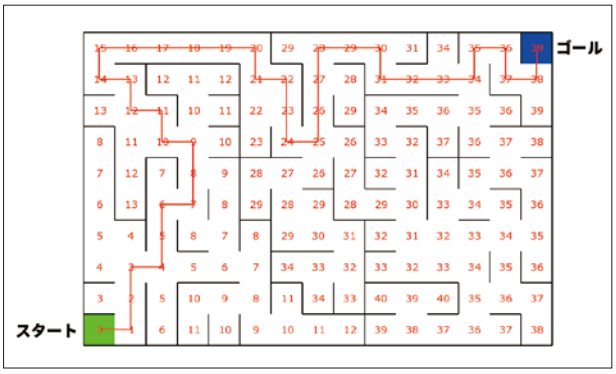


図 -9　幅優先探索の例

絶対に迷わず，かつ一番短い経路を見つけるやり方です．このやり方も結構簡単です．まず，スタート地点に 0 を書きます．次に，0 の隣に 1 を書きます．続いて隣に 2 を書きます．そして 2 に隣り合っている場所に 3 を書く……．これを繰り返すだけです．このように数字をゴールまで書いていきましょう．

ゴールまで行ったら，後は数字が，減る方向に経路を伝っていくと，それが一番短い経路ということになります．このスタート地点に書いた数字は何かというと，スタート地点から各地点までの数字が距離を表すのです．1 と書いてあるところには，1 より短く行けない．ここに書いてある数字というのは，その地点までに一番短く行く距離，になるのです．それをゴールまでやると，そこまで行くのに必要な一番短い距離になるのです．

幅優先探索のアルゴリズム

- スタートに 0 と書く
- 0 に隣り合うマスに 1 と書く
- 1 に隣り合うマスに 2 と書く
- ……
- ゴールから，数字が減る方向に辿る

図 -10　幅優先探索

闇雲に進むよりもずっとたしかな経路です．このアルゴリズムにも名前があります．『幅優先探索』というものです．たとえば，カーナビなどで使われている経路の検索は幅優先探索を発展させたアルゴリズムです．交差点ごとに距離を計算してく，というような方法で計算されています．アルゴリズムは，案外身近にあるんですね．

石取りゲーム：最後に石を取った人が負け！

引き続き，ゲームをしましょう．今回のゲームは，石取りゲームです．用意する道具は，石です．最初に石が置いてあります，石は 3 個のブロック，5 個のブロック，7 個のブロックに分けてあります．この石を変わりばんこに 1 つ以上取っていきます．石は 1 つ以上であれば何個でも取れます．ですが，どこか 1 つのブロック内のものを取る必要があります．同じブロック内であれば，2 個でも，全部取っても OK です．でも，違うブロックから同時に石を取ってはいけません．お互い取っていって，最後

の石を取った人が，勝ちです．

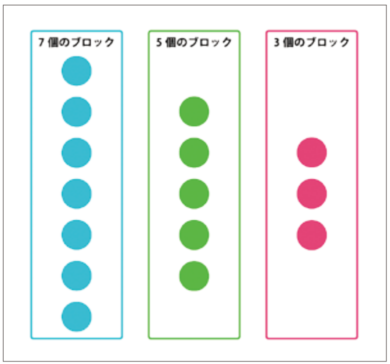


図 -11
石のブロック説明図

〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜

秋葉　「先手でも後手でもいいですよ．まずは先手でやってみますか」

ビット「頑張るんだぴっと！」

（早送り）

ビット「う～ん，全然勝てない！　先生，何かずるいことしてませんか？」

秋葉　「ばれましたか．このゲーム，実は，必ず勝つ方法があるんです．ビットくんに必勝法を教えてあげますよ．そのために，まずは『2 進数』のお勉強をしましょう」

ビット「はーい」

〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜〜

みなさんは，10 進数と 2 進数という言葉は聞いたことありますか？　コンピュータは 0 と 1 の世界だ，とか，数字を 2 進数で扱う，ってなんとなく聞いたことありませんか？　僕らが普段使う数字の表現，それが 10 進数です．

10 進数：我々が使う数の表現

- $25 = 2 \times 10^1 + 5 \times 10^0$
- $154 = 1 \times 10^2 + 5 \times 10^1 + 4 \times 10^0$

図 -12　10 進数

10 進数はこういうふうに書きますね．たとえば，25 という数が書いてあったら，2 が十の位で，5 が一の位です．2 × 10 足す 5 で，25 です．同じように 154 と書いてあれば，1 が百の位，5 が十の位，4 が一の位，10 ごとに桁が上がり数が増えていきます．

2 進数：コンピュータが使う数の表現

- 2進数 $11001 = 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 25$ 10進数
- 2進数 $10011010 = 1 \times 2^7 + 1 \times 2^4 + 1 \times 2^3 + 1 \times 2^1 = 154$ 10進数

図 -13　2 進数

そしてこれが 2 進数です．コンピュータは 0 や 1 でモノゴトを扱うといいますが，それがまさに 2 進数の考え方です．基本的には 10 進数と同じ感じですが，繰り上がる数が，10 ではなく，2 になっているという点，これが 2 進数です．

たとえば，11001 という数は，一番下が一の位というのは同じですが，次の位は 10 の位ではなくて，二の位です．次は四の位，八の位となります．11001 は，16 足す 8 足す 1 で 25 ということになります．同じように 154 も，2 進数で表すと，10011010 となります．

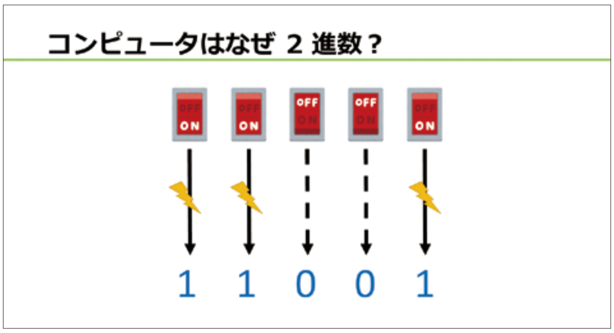


図 -14　コンピュータは 2 進数

なぜコンピュータでは 2 進数が使われるかというと，電気信号の処理と 2 進数の相性がいいからです．電気がオンのときに 1，オフのときに 0 を表すと考えると，コンピュータが 2 進数を表すのがとても簡単です．たとえば 8 桁の数を表したければ，8 本電気信号を置いておく．オンのときに 1，オフのときに 0 としておけば，コンピュータの電気信号で 2 進数で数が表現できるようになるんですね．

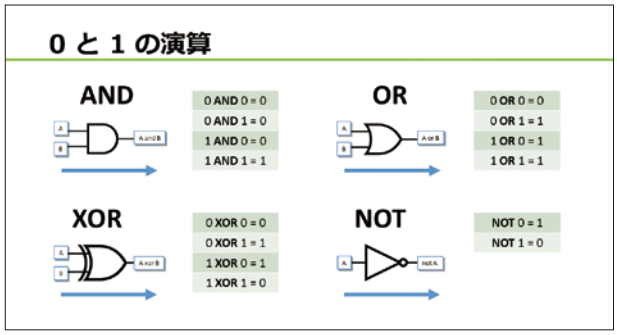
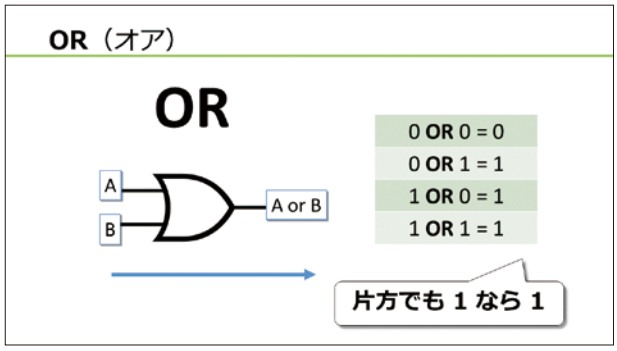
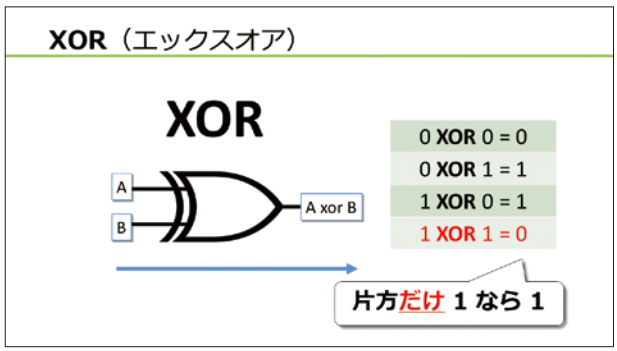


図 -15　0 と 1 の演算の例

さて，僕らは 10 進数で足し算，引き算，掛け算，割り算，みたいな計算を習いましたよね．2 進数では，2 進数に特有の演算があります．たとえばアンド，オア，エックスオア，ノットです．この中で，今日使うのはエックスオアです．オアというのは，片方でも 1 となれば 1 となるものです．エックスオアはオアの仲間で，片方だけ 1 なら 1 になるものです．





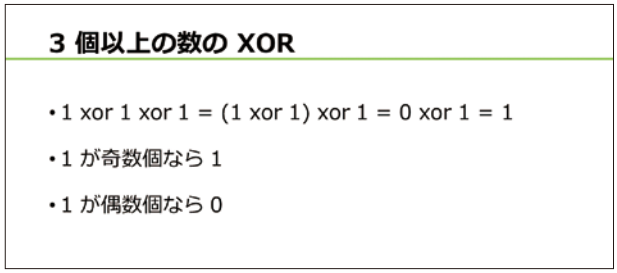


図 -16　OR と XOR

エックスオアのエックスは，エクスクルーシブという意味で，どちらかじゃないとダメという意味です．ですので，0 と 0 のときは 0 になります．0 と 1 のときには 1 になります．1 と 0 のときも 1．だけど 1 と 1 のときは 0 になってしまいます．これがエックスオアです．

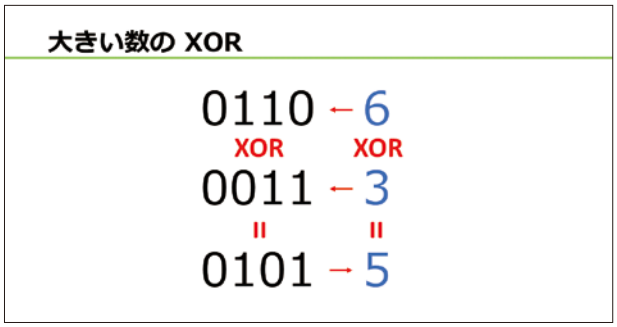


図 -17　大きい数の XOR

たとえば 4 桁のエックスオアを計算したければ，このように桁ごとに計算します．6 と 3 のエックスオアを計算したければ，6 を二進数で書くと 0110．3 を 2 進数で書くと 0011．そのあとで，桁ごとに計算して，0101 となり，5 となりますね．

ちょっと変わった演算ですが，でも，石取りゲームの必勝法においてはこれが主役になってきます．

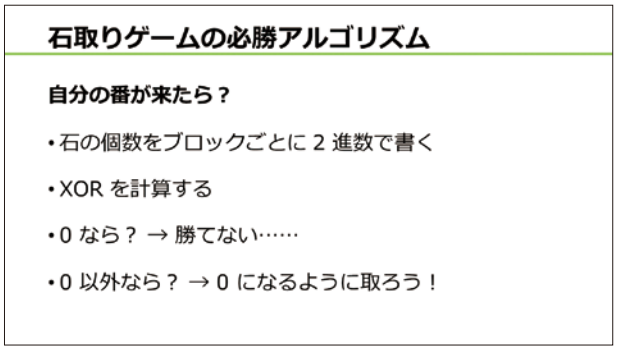


図 -18　石取りゲームの必勝アルゴリズム

では，石取りゲームの必勝法を説明します．まず，石の個数をブロックごとに 2 進数で書きます．そのエックスオアを計算する．計算したエックスオアが 0 でなければ勝てます．相手のターンでエックスオアが 0 になるように，自分が石を取ってあげればいいのです．これを繰り返せば，勝てるのです．

～～～～～～～～～～～～～～～～～～～～～～～～～～～～

秋葉　「よし，じゃあやってみましょうか．ビットくんが先手でいいですよ．まずは今の XOR を計算してみましょう」

ビット「まず，石が7つある，5つある，3つある……．2進数で書くと，7は，111．5は101．3は011．だからこれを桁ごとにXORを計算すると……（上から順番に計算）．一の位は，1と1で1，そして1と1で1．二の位は，1と0で1，そして1と1で0．四の位は，1と1で0，そして0と0で0！（ふぅ） 結果は，001になりますね！」

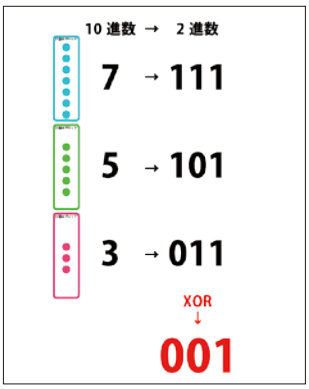


図-19（a）
7-5-3 上記計算の経過

秋葉 「はい．今の状態のXORは，1です．勝つためには，このXORが，0になるように，石を取ってやれば，ビットくんの勝ちです！」

ビット「なるほど！」

秋葉 「よし，このXORを0にするように石を取れますか？」

ビット「うーん．四の位と二の位はもう0なので，一の位が0になればいいんですよね？じゃあ，どこかのブロックから1つ取ってくると，XORが0になりますね？」

秋葉 「そのとおりですね」

ビット「じゃぁ7のブロックから石を1つ取ります！」

秋葉 「いいですね，ではこのXORを計算してみてください」

ビット「さっき7だったところが6になったから，6の2進数は，110．そうすると一の位のXORが，0になります」

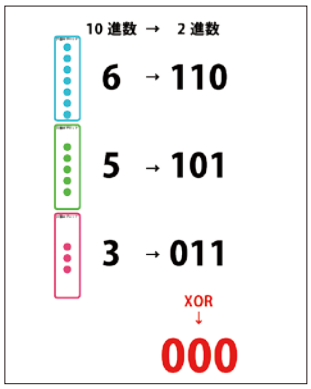


図-19（b）
6-5-3 上記計算の経過

秋葉 「僕の番ですね．でも実は，僕がここから何をどれだけ取ってもXORは0にできません．つまり，僕は，ビットくんが間違えてくれない限り勝てない状況です．なので，なんとなくですが，7のブロックから全部取ってみます」

ビット「じゃぁまたXORを計算します．6だったところが全部なくなっちゃったから0．2進数も000．そうすると，XORは，110ですね」

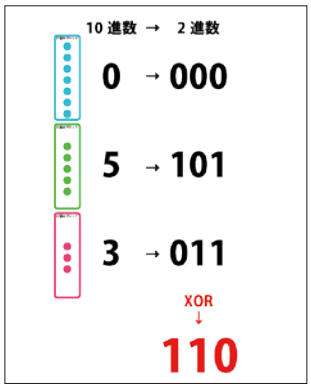


図-19（c）
0-5-3 上記計算の経過

秋葉 「これはちょっと難しいパターンですね．ヒントです．四の位のXORが1ですね．ということは，5のブロックの1を0にすれば，XORが0になる．なので，5のブロックから何か取ればいいということが分かります．つぎに，二の位のXORも1だから，打ち消さないといけない．たとえば，5のブロックを011にしてあげちゃえばいい」

ビット「2進数の011は……，10進数で3？ だから，5のブロックから，2つ取ってあげればいいんだ」

秋葉 「はい，ではXORを計算してみましょうか」

ビット「5だったところが3になって，011．XORは，000」

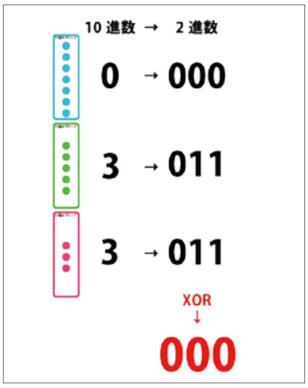


図 -19（d）
0-3-3 上記計算の経過

秋葉　「また，XOR が 0 になっちゃいました．これを続けていくと，ビットくんが勝てるのです」

～～～～～～～～～～～～～～～～～～～～～～～～～～

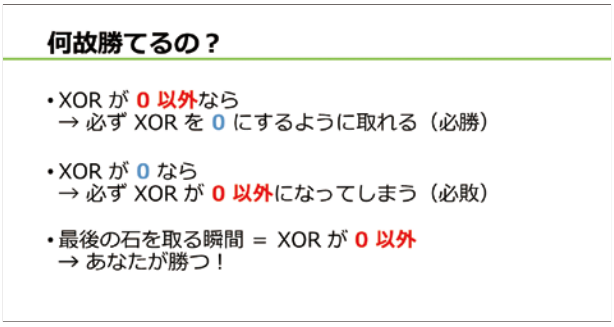


図 -20　必勝法が必ず勝てる理由

この方法がなぜ必勝法なのかを簡単に説明します．XOR が 0 ではないときは，自分が石を取ることで必ず 0 にできます．逆に，XOR が 0 のときに相手が石を取ると絶対 XOR が 0 以外になるんですね．そして，最後の 1 個を取る瞬間は，XOR が 0 以外から 0 になります．つまり，自分のターンで XOR が 0 でない状態をキープしていれば，絶対に最後の石を取ることができるんですね．

というわけで今回は，石取りゲームの必勝法を題材にして，コンピュータ内部の数の表現である 2 進数と，その演算の例として XOR について勉強してもらいました．プログラミングを動かす背景に，こういった理論やアルゴリズムが大切になってくるんですね．今回は，身近な題材を使って，コンピュータの背後にある数理のエッセンスを学んでみました．

（2016 年 7 月 25 日受付）

秋葉拓哉（正会員）■ akiba@preferred.jp
2016 年より（株）Preferred Networks 所属．