形式言語とオートマトン

Note5　チューリングマシン
2020.5.12作成
2020.6.20 update 2021.5.28

中野眞一　群馬大学

チューリングマシン

コンピュータの動作原理は、それがどんなに大きくて複雑で高価なものであろうと、
ある簡単な機械の動作としてモデル化できる。

チューリングマシン --- (チューリング 1930年代)
コンピュータの能力は、特定の機種にとらわれることなく、抽象的かつ一般的に、考察できる。
(チューリングマシンは、実際には存在しません。数学的なモデルであることに注意。)
コンピュータに何ができるのか等の議論に用いる。
(実は、当時、このようなモデルは複数あったが、みな本質的には同じものであることが、後にわかった。)

チャーチ - マルコフ - チューリングの提唱(1930年代)
計算機のモデルは本質的に"ひとつ"である。(= チューリングマシンだけ考えればよい。)
　　チューリングマシンの動作 = アルゴリズム
　　コンピュータで解ける　= アルゴリズムがある
としよう。
(実は、アルゴリズムがない問題もあります。)

チューリングマシン

チューリングマシン(TM)とは、次の6つのものにより定義される(言語を表現するための)数学的モデルである。

(最強のオートマトンである。)

$TM = (Q, \Sigma, \Gamma, \delta, q_0, F)$

(1) Q は有限個の状態の集合である。

(2) Σ は有限個の入力記号の集合である。

(3) Γ は Σ を含む有限個の記号の集合である。テープ記号集合と呼ぶ。 空白を示す特別な記号B (ブランク)を含む。

(4) δ は $Q \times \Gamma$ から $Q \times \Gamma \times \{L,R\}$への関数である。

状態遷移関数と呼ぶ。

L と R はヘッドを左または右に1つ動かすことを示す。

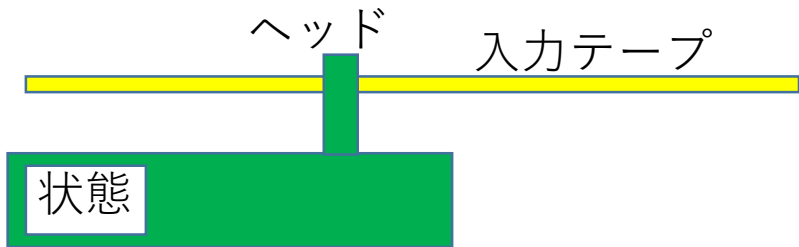
(例えば $\delta(a,b) = (c,d,R)$ならば、状態 $a \in Q$ のとき, テープ記号が $b \in \Gamma$ であれば、

ヘッドの下のテープ記号をdとし、

ヘッドを右に1つ動かし、状態を$c \in Q$とする。)

(5) q_0は、Qの1つの要素である。初期状態と呼ぶ。

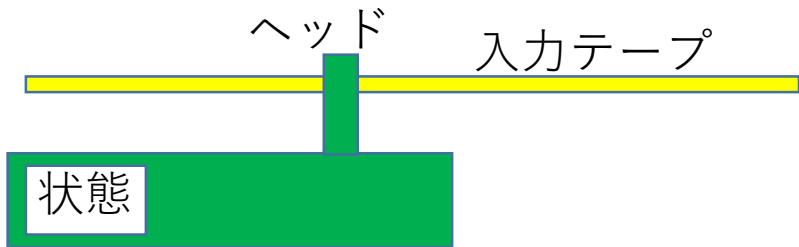
(6) FはQの部分集合である。受理状態と呼ぶ。



チューリングマシン

制御部と、一本の入出力テープを持つコンピュータであるとする。

テープ上に入力の文字列が置かれる。
ヘッドをこの文字列の左端に置く。
その後、下記を繰り返す。
　ヘッドの下のテープ記号を読み、状態を換え、ヘッドの下に記号を上書きし、
　ヘッドを左または右に1つ動かす。
特徴
"状態"という"有限個"の記憶の他に、
無限の長さの入出力テープを持つ。
テープは入力にも出力にも使用できる。
（オートマトンと違い）
テープの様々な部分を読み書きできる。



チューリングマシン

計算の例

略

計算不能

ある問題を解くc言語のプログラムがあるとき、その問題は計算可能であるという。

実は、計算可能でない問題 = 計算不能な問題 があります。
(今後、計算機がどんなに進歩しても解けない! ことに注意しよう!)

計算不能

入力	1	2	3	4	5	6	……
f_1	1	2	3	4	5	6	……
f_2	2	2	2	2	2	2	……
f_3	2	3	4	5	6	7	……
f_4	……						
f_{new}	2	3	5	……			

計算不能な関数があることの証明

正の整数を入力すると正の整数を出力する関数を考えよう。

関数は非可算無限個ある。しかし、プログラムは可算無限個しかないことを示す。

1つの整数を入力し、1つの整数を出力するプログラムで最短の長さのものは下記である。

```
Program p;
var     x: integer;
Begin
        readln(x);
        writeln(0);
End.
```

これの変数xを他の変数にしたり、出力を0から1,2,3,…9,に変更したものも同じ長さである。

これらを辞書順に並べてリストとしよう。

次に長さがもう1だけ長いすべてのプログラムを、すべてリストにする。

次に長さがもう1だけ長いすべてのプログラムを、すべてリストにする。

....

このようなリストが作成できるので、プログラムは可算無限個しかない。(整数の個数と同じ位しかない!)

これは、プログラムがない関数があることを意味する。

停止性問題

代表的な計算不能の例である停止性問題について解説する。

プログラムの中には、ある入力のときに無限ループに陥るものがある。

例（入力が11以上なら無限ループになる。）

```
program A
var  x:integer;
begin
      readln(x);
      while x > 10 do
          x:=x;
end.
```

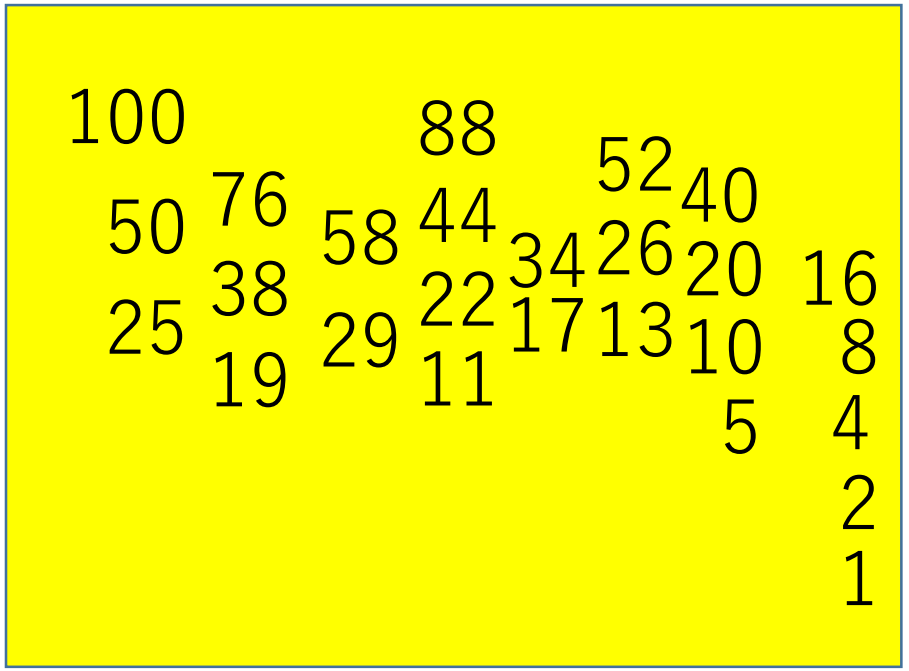
プログラムとその特定の入力があたえられたとき
無限ループになるかどうかを判定するプログラムを作ろう。停止性問題という。

停止性問題

一般にこのような停止性問題を解くプログラムの作成は非常に難しい!

下のプログラム は無限ループに陥いることがあるかな？）
（コラッツの予想, もしくは、角谷の予想といいます。あられ数とも。）

```
program D
begin
   readln(x);
   while x > 1 do
       if (x div 2)*2 = x then
            x:=x div 2              （偶数なら 2 で割る）
       else
            x:=3*x+1;               （奇数なら3倍+1に）
end.
```



停止性問題（証明）

停止性問題を解くようなプログラムBが存在しない
ことを背理法で証明する。

停止性問題（証明 つづき）

入力xをプログラムpに与えたとき、プログラムpが無限ループに陥ることなく
停止するかどうかを判定するプログラムhaltがあると仮定しよう。

```
procedure halt(p, x)
begin
    ....
    if .... then return='停止する'
    else         return='無限ループ'
end
```

haltを使用して次のプログラムselfhaltを作成する。
(プログラムpに自分自身を入力として与えたときに停止するかどうかを判断している。)

```
procedure selfhalt(p)
begin
  answer = halt(p,p);
  if   answer = '停止する'   then      return='停止する on 自分'
  else answer = '無限ループ' then  return='無限ループ on 自分'
end.
```

停止性問題（証明 つづき）

selfhaltを改造して次のプログラムを作成する。
(selfhaltの答が停止の場合は無限ループとなり、
無限ループの場合は停止することに注意)

```
procedure contrary(p)
begin
   answer = selfhalt(p)
   if answer = '停止する on 自分'  then
      while true do
            answer := true;   /*無限ループになる*/
   /*  answerが’無限ループ’なら停止する  */
end
```

停止性問題（証明 つづき）

さて、contraryにcontraryを入力したら何が起こるだろうか？？？
selfhalt の結果が 停止 でも 無限ループでも、いすれも、矛盾が生じる。

もしselfhaltの結果が"停止"ならば、if文はyesでありwhile文により、
contrary自身の実行は無限ループになる。
このとき、contraryにcontraryを入力すると、"無限ループ"となった。

もしselfhaltの結果が"無限ループ"ならば、if文はnoでありwhile文は
実行しないので、 contrary自身の実行は停止する。
このとき、contraryにcontraryを入力すると、"停止"となった。

どちらも矛盾！！！

同じような論理の矛盾の例

プログラムの動きを判定するプログラム。。。。

貼り紙禁止の貼り紙

ある小さな町に床屋が一人住んでいる。
彼は、この町の住人のうち、自分で自分のヒゲを剃らない全ての人のヒゲのみを剃る。
この記述には矛盾がある。床屋は自分のヒゲを剃るだろうか？

集合x は 集合xを要素としないならば通常集合と呼ぶ。
集合x は 集合xを要素とするならば 異常集合と呼ぶ。(つまり自分を要素とする。)
このとき、S = { x | x は通常集合 }
とすると S自身は通常集合であろうか、異常集合であろうか？

その本のタイトルが、本文中に現れる本を タイプA とする。
その本のタイトルが、本文中に現れない本を タイプB とする。
タイプBの本の完全リストを作り本にした。この本自身はどちらのタイプであろうか？

同じような論理の矛盾の例　（つづき）

法律の法律
手を上げる元気のない人は、手をあげて!
がんばらないことを、がんばる!
わからないことが、わからない
人生の目的は人生の目的を見つけることである。
すべての人に不公平という意味で、ある意味公平
僕の夢は夢を持つこと
気になることが、気になる
3Dプリンタで3Dプリンタを印刷する
（自己言及　それ自身に言及すること には注意！）

プログラム処理のプログラム、
アルゴリズム処理のアルゴリズム等の考察には注意が必要!

クラスP とクラスNP と クラスNP完全

計算可能な問題にも、
(1)　早く計算できる問題　＝　多項式時間のアルゴリズムがある問題　＝　クラス
　　$O(n^2)$とか
(2)　指数時間かかることがわかっている問題
　　$O(2^n)$とか
(3)　解の検証は多項式時間でできるが、解を求める多項式時間アルゴリズムは
　　今のところ知られていないもの　＝ クラスNP完全
の3種類があります。

解の検証が多項式時間でできる問題のクラス を クラス NP と呼びます。
解を求めることが多項式時間でできる問題のクラス を クラス P と呼びます。

(NP完全問題の例)
　　最大クリーク、 最大独立点集合、3SAT、　 2分割、点彩色、辺彩色、etc...
(P問題の例)
　　極大クリーク、 最大マッチング、2SAT、