Algorytmy wokól Collatzowe

Grażyna Mirkowska & Andrzej Salwicki

Instytut Informatyki UKSW &
Dombrova Research , Dąbrowa Leśna

7 września 2021

Streszczenie

Przedstawiamy główne punkty dowodu twierdzenia Collatza.

Część I - Obserwacje

Algorytm *Cl* Collatza

```
var n: Nat ;
        (* n jest liczbą naturalną >0, *)
         (* działania są z Nat - ale co to jest Nat? *)
read(n);
```

Cl:

```
while n≠ 1 do
   if Parzyste(n) then n:=n ÷ 2 else n:= 3n+1 fi
od
```

Własności struktury Nat – przypomnienie

$$Nat = \langle N, +, 0, 1; =, <, Parzyste \rangle$$

N jest zbiorem,

$+ : N \times N \to N$ funktor $+$ oznacza operację dodawania,

$0, 1$ wyróżnione elementy zbioru N

$=$ relacja równości

$<$ relacja mniejszości

parzyste relacja parzystości

i prawdziwe są zdania

$$\forall_n \; n + 0 = n \tag{1}$$

$$\forall_{n,m} \; n + 1 = m + 1 \Rightarrow n = m \tag{2}$$

ponadto, dla każdej formuły Φ

$$\big(\Phi(0) \wedge \forall_n \, [\Phi(n) \to \Phi(n+1)]\big) \to \forall_n \, \Phi(n) \tag{3}$$

Algorytm *CI* Collatza powtórzony

Powtórzene nie zaszkodzi. Program CI jest idempotentny.

```
read (m);
n:=m;
CI: while n≠ 1 do
      if Parzyste(n) then n:=n÷ 2 else n:= 3n+1 fi
    od ;
```

```
n:=m;
while n≠ 1 do
  if Parzyste(n) then n:=n÷2 else n:= 3n+1 fi
od
```

Program Collatza jest równoważny programowi Cl_1

while Parzyste(n) do n:=n/2 od;
while n$\neq$ 1 do

K:
```
n:=3n+1;
while Parzyste(n) do n:=n/2 od;
```

od

Lemat

Programy Cl i Cl_1 są równoważne.

Zastępujemy

```
read(m);
n:=m;
while Parzyste(n) do n:=n÷ 2 od;
while n≠ 1 do
    K: n:=3n+1;
       while Parzyste(n) do n:=n÷2 od;
od ;
```

```
n:=m;
while n≠ 1 do
   if Parzyste(n) then n:=n÷2 else n:= 3n+1 fi
od
```

Nie zaszkodzi policzyc iteracje

```
read(m);
x:=0; n:=m;
while Parzyste(n) do n:=n/2 od;
while n≠ 1 do
   n:=3n+1; x:=x+1;
   while Parzyste(n) do n:=n/2 od;
od ;
```

```
n:=m;
while n≠ 1 do
   if Parzyste(n) then n:=n÷2 else n:= 3n+1 fi
od
```

Policzmy też liczbę dzieleń

```
read)m);
x:=0; z:=0; n:=m;
while Parzyste(n) do n:=n/2 ; z:=z+1 od;
while n≠ 1 do
   n:=3n+1; x:=x+1;
   while Parzyste(n) do n:=n/2; z:=z+1 od;
od ;
```

```
n:=m;
while n≠ 1 do
   if Parzyste(n) then n:=n÷2 else n:= 3n+1 fi
od
```

Obserwacja

Fakt

$z > x$

i niecałkiem z powietrza wzięte[1]

Obserwacja

$2^z > n \cdot 3^x$

[1] G. Mirkowska,A.Salwicki, *On Collatz theorem*, 2021, url: https://dabrowa.research.pl

Popatrzmy

```
read(m);
x:=0; z:=0; n:=m;
while Parzyste(n) do n:=n/2; z:=z+1 od;
while n≠ 1 do
   n:=3n+1; x:=x+1;
   while Parzyste(n) do n:=n/2; z:=z+1 od;
od ;
```

```
n:=m;
while n≠ 1 do
   if Parzyste(n) then n:=n÷2;z:=z-1 else n:= 3n+1; x:=x-1 fi
od
```

Teraz, po zkończeniu programu $x = 0$ i $z = 0$.

obliczmy y= $2^z - n \cdot 3^x$

read(m);
x:=0; z:=0; n:=m;
while do n:=n/2; z:=z+1 od;
while n$\neq$ 1 do
  n:=3n+1; x:=x+1;
  while Parzyste(n) do n:=n/2; z:=z+1 od;
od ;

---

n:=m; y:=$2^z - n \cdot 3^x$;
while n$\neq$ 1 do
  if Parzyste(n) then n:=n/2; z:=z-1;y:=y/2
  n:=3n+1; y:=y-3^{x-1}; x:=x-1; fi
od ;

Teraz $x = 0 \wedge z = 0 \wedge y = 0$ nieprawdaż?
Zauważ, niezmiennie $n \mod 1 \equiv y \mod 1$

Niezmiennik

Niezmiennikiem drugiej instrukcji while jest $\boxed{2^z = n \cdot 3^x + y}$

```
read(m);
x:=0; z:=0; n:=m;
while do n:=n/2; z:=z+1 od;
while n≠ 1 do
   n:=3n+1; x:=x+1;
   while Parzyste(n) do n:=n/2; z:=z+1 od;
od ;
```

```
n:=m;  y:=2^z − n · 3^x;
while n≠ 1 do (* 2^z = n · 3^x + y *)
   if Parzyste(n) then n:=n/2; z:=z-1;y:=y/2 (* 2^z = n · 3^x + y *)
 else n:=3n+1; y:=y-3^(x−1); x:=x-1; (* 2^z = n · 3^x + y *) fi
od ; (* 2^z = n · 3^x + y *)
```

W drugiej pętli while można zapomnieć o n

read(m); n:=m;

Cl':
```
:=0; z:=0;
while do n:=n÷2; z:=z+1 od;
while n≠ 1 do
    n:=3n+1; x:=x+1;
    while Parzyste(n) do n:=n/2; z:=z+1 od;
od ;
y:=2^z − n · 3^x;
```

n:=m;

IC:
```
while 3^x + y ≠ 2^z do
    if Parzyste(y) then z:=z-1;y:=y/2
    else y:=y-3^(x−1); x:=x-1;
od
```

Co z tego wynika?

Część 2

Cztery lematy i twierdzenie.

Lematy 1 i 2 wynikają wprost z poprzedniego slajdu. Własność stopu tych programów *CI* i *IC* jest prawie udowodniona. Pozostaje do wykazania, że zapętlenie jednego programu ma miejsce wtedy i tylkowtedy gdy drugi program sie zapętla.

Lemat 1 - Obliczeniu Collatza towarzyszy udane obliczenie trójkowe

Niech IC oznacza następujący program

IC:

```
while 3^x + y ≠ 2^z do
    if ¬ Parzyste(y) ∧(x = 0 ∨ y < 3^(x−1)) then Err:=true; exit fi;
    if Parzyste(y) then z:=z-1;y:=y/2
    else y:=y-3^(x−1); x:=x-1; fi
od ;
```

Fakt

Każde obliczenie algorytmu IC jest skończone.

Lemat 1

Jeśli dla liczby n obliczenie algorytmu Cl jest skończone, to istnieją takie liczby naturalne x, y, z, że $n \cdot 3^x + y = 2^z$ i obliczenie olgorytmu IC jest wolne od błędu.

Lemat 2 - Obliczenie trójkowe, bez błędu => skończone obliczenie Collatza

Lemat 2

Jeśli dla pewnych liczb n, x, y, z zachodzi równość $n \cdot 3^x + y = 2^z$ i obliczenie algorytmu IC jest wolne od błędu Err,
to obliczenie algorytmuu Collatza dla n jest skończone.

Lemat 3 nieskończone obliczenie trójkowe => nieskończone obliczenie Collatza

Nlech $\mathfrak{M}$ oznacza nie-standardowy model aksjomatów teorii dodawania.

$$\mathfrak{M} = \langle \mathbb{Z} \times \mathbb{Q}^+; +, \underbrace{(0;0)}_{0}, \underbrace{(1;0)}_{1}; = \rangle$$

Czyli uniwersum tej struktury to zbiór par $\langle k, w \rangle$ takich, że $k \in \mathbb{Z}$ jest liczbą całkowitą, a $w \in \mathbb{Q}^+$ to liczba wymierna dodatnia. Uwaga, gdy $w = 0$ to $k \geq 0$.
Operacja dodawania jest określona tak
$\langle k, w \rangle + \langle k', w' \rangle = \langle k + k', w + w' \rangle$.
Jedność to $\langle 1, 0 \rangle$, zero to $\langle 100 \rangle$.
Gdy $w \neq 0$ to element $\langle k, w \rangle$ jest *nieosiagalny* z zero przez dodawanie skończonej liczby jedynek.

Lemat 3 - nieosiagalne $\subset$ nie-Collatzowe

Lemat 3

Dla każdego elementu nieosiągalnego w strukturze $\mathfrak{M}$ obliczenie algorytmu Collatza jest nieskończone.

Zamiast dowodu rozpatrz przykład obliczenia Collatza dla $n_\varepsilon = (5; \frac{1}{2})$. (Pamiętaj, działania w $\mathfrak{M}$ odbywają się oddzielnie na składowych par.) $(5; \frac{1}{2}) \overset{\cdot 3+1}{\to} (16; \frac{3}{2}) \overset{/2}{\to} (8; \frac{3}{4}) \overset{/2}{\to} (4; \frac{3}{8}) \overset{/2}{\to} (2; \frac{3}{16}) \overset{/2}{\to} (1; \frac{3}{32}) \overset{\cdot 3+1}{\to}$
$(4; \frac{9}{32}) \overset{/2}{\to} (2; \frac{9}{64}) \overset{/2}{\to} (1; \frac{9}{128}) \overset{\cdot 3+1}{\to} (4; \frac{27}{128}) \overset{/2}{\to} (2; \frac{27}{256}) \overset{/2}{\to} (1; \frac{27}{512}) \overset{\cdot 3+1}{\to} \ldots$
Spróbuj z parą (5;0) i porównaj.

nieskończone obliczenie trójkowe

lemat

Jeśli dla pewnej trójki $\langle x, y, z \rangle$ obliczenie algorytmu IC jest nieskończone, to dla elementu n reprezentowanego przez tę trójke obliczenia algorytmu Collatza jest nieskończone.

Lemat 4 – nie-Collatzowe $\subset$ nieosiagalne

Lemat 4

Jeśli dla pewnego elementu ε obliczenie algorytmu Collatza jest nieskończone, to element ten jest nieosiągalny.

algorytm IIC poszukujący trójki wolnej od błędu

Niech element n będzie elementem nie-Collatzowym, przyjmijmy $z = (\mu l)(2^l > n)$.

IIC:

read(n);
$x, x_s := 0;\ z_s := z;\ y, y_s := 2^z - n$; Err := false;
while $3^{x_s} + y_s \neq 2^{z_s}$ do

IC:

while $3^x + y \neq 2^z$ do
if $odd(y) \wedge (x = 0 \vee y < 3^{x-1})$
then Err:=true; exit fi;
if $odd(y)$ then $y := y - 3^{x-1}; x := x - 1$
else $y := y/2; z := z - 1$ fi;
od;

if Err then
$x, x_s := x_s + 1;\ z, z_s := z_s + 2;\ y, y_s := 2^{z_s} + 3 \cdot y_s$; Err := false;
else exit fi;
od

Uwaga

Jeśli element n ma obliczenie Collatzowe nieskończone, to algorytm IIC ma obliczenie nieskończone.

algorytm B4 z kolejką trójek

Daną dla algorytmu B4 jest liczba n, algorytm buduje drzewo Collatza i zwraca trójkę t, która reprezentuje liczbę n. Ale co gdy $n \notin DC$?.

```
unit F4: class(m,x,y,z: Nat); end F4;
```

B4:

```
read(n);   x:=0; y:=0; z:=0;m:=1; p:=new Kolejka;
while n ≠ m  do
   p:=put(new F4(⟨2 * m, x, 2 * y, z + 1⟩),p);
   if m  mod 3 = 1 ∧ m ≠ 4 then
        if ((m − 1) ÷ 3)  mod 2 = 1 then
          p:=put(new F4((m − 1) ÷ 3, x + 1, y + 3^x, z),p);
        fi
   fi;
   t := first(p); m:=t.m;x:=t.x;y:=t.y;z:=t.z; p:=usun1z(p);
od
```

Fakt

Jeśli obliczenie Collatza dla n jest nieskończone to trójka $\langle x, y, z \rangle$ taka , że obliczenie algorytmu IC jest nieskończone, jest nieosiągalna.

Dowód lematu 4

Nie znamy , na razie, nieskomplikoanego dowodu tego lematu.
Zajrzyj tutaj: https://lem12.uksw.edu.pl/wiki/Collatz .

Twierdzenie Collatza

Twierdzenie Collatza

Dla każdej standardowej liczby naturalnej n, algorytm Collatza Cl **kończy** obliczenie z wartością $n = 1$.

Obliczenie algorytmu Collatza dla n jest albo skończone albo nieskończone.
Na to by zaszedł pierwszy przypadek, potrzeba i wystarcza by istniała taka trójka liczb naturalnych, która reprezentuje n i by obliczenie trójkowe dla tej trójki było udane.
Na to by obliczenie Collatza było nieskończone, potrzeba i wystarcza by istniała taka trójka, że obliczenie trójkowe jest wolne od błędu i nieskończone.
W tym przypadku liczby n, x, y, z są nieosiągalne, nie należą do N..

A więc dla każdej liczby naturalnej $n \in N$ obliczenie algorytmu Collatza jest skończone!