Na twierdzenie Collatza

Szkic dowodu

Grażyna Mirkowska
Dombrova Research
Partyzantów 19
05-092 Łomianki, POLAND
G.Mirkowska@uksw.edu.pl

Andrzej Salwicki
Dombrova Research
salwicki@mimuw.edu.pl

Streszczenie. Dowodzimy, że dla każdej liczby naturalnej n algorytm Collatza ma obliczenie skończone.

17 grudnia 2018

1. Wprowadzenie

Ten szkic jest raportem z aktualnego stanu pracy nad artykułem o twierdzeniu Collatza. Idea dowodu jest bardzo prosta. Pełny dowód wymaga uzupełnienia szczegółów.
Algorytmu Collatza przedstawiać nie trzeba.

$$C: \quad \{\mathbf{while}\ n \neq 1\ \mathbf{do\ if}\ odd(n)\ \mathbf{then}\ n := 3n+1\ \mathbf{else}\ n := n/2\ \mathbf{fi\ od}\}$$

Mamy zamiar opisać jego obliczenia w dwu, na pozór, różnych strukturach danych: $\mathfrak{N}$ oraz $\mathfrak{T}$.
Pierwsza struktura to standardowy zbiór liczb naturalnych N ze zwykłymi działaniami następnika i dodawania. To wystarcza by dobrze (tj. efektywnie) zdefiniować polecenia n:=3· n +1 oraz n:= n ÷ 2. Dalej zapisywać je będziemy jako polecenie $mult3$ i polecenie $div2$. W algorytmie Collatza występują też dwie formuły $n \neq 1$ oraz $odd(n)$. Zastąpimy je przez krótkie $equal1$ i odd. W algorytmie występuje tylko jedna zmienna, nie musimy jej wymieniać z nazwy.

Druga struktura $\mathfrak{T}$ to zbiór trójek liczb naturalnych z odpowiednio dobranymi poleceniami, zobacz poniżej. Stąd pomysł zapisania tego algorytmu w abstrakcyjny sposób.

$$\{\mathbf{while}\ \mathrm{not}\ equal1\ \mathbf{do\ if}\ odd\ \mathbf{then}\ mult3\ \mathbf{else}\ div2\ \mathbf{fi\ od}\}$$

Jedyna zmienna n występująca w tym programie przyjmuje bądź wartości będące liczbami naturalnymi(gdy obliczenia realizujemy w strukturze $\mathfrak{N}$ bądź trójkami liczb naturalnych (gdy obliczenia prowadzone sa w strukturze $\mathfrak{T}$). Polecenia $mult3$ i $div2$ to, odpowiednio: pomnóż n przez 3 i dodaj 1 oraz podziel n przez 2, bądź, gdy działania wykonywane są na trójkach polecenia te realizowane są w inny sposób.

2. Trójki

Zacznijmy od następującego spostrzeżenia

Fakt 1. *Dla każdej liczby naturalnej $n \neq 0$ istnieje nieskończenie wiele trójek liczb naturalnych x, y, z takich, że spełniona jest równość*

$$n \cdot 3^x + y = 2^z$$

Mówimy, że trójka x, y, z *reprezentuje* (albo, *koduje*) liczbę n.

Dowód:
Dowód tego intuicyjnego faktu wykorzystuje prawo Archimedesa[1]. Niech n będzie dowolnie ustaloną liczbą naturalną.
Wybierzmy liczbę x, może to być dowolnie duża liczba naturalna.
Wyznaczymy liczby naturalne y i z tak, by zachodziła równość $n \cdot 3^x + y = 2^z$. Niech liczba $k = n \cdot 3^x$.
Niech liczba 2^z będzie najmniejszą potęgą liczby 2 większą od k. Kładziemy $y = 2^z - k$. □

Przykład 2. Trójka $\langle 1, 7, 6\rangle$ reprezentuje liczbę 19 ponieważ $19 \cdot 3 + 7 = 2^6$.
Liczba 19 jest także reprezentowana przez inne trójki

$\langle 1, 7, 6\rangle$	$19 \cdot 3^{\mathbf{1}} + \mathbf{7} = 2^{\mathbf{6}}$
$\langle 2, 85, 8\rangle$	$19 \cdot 3^{\mathbf{2}} + \mathbf{85} = 2^{\mathbf{8}}$
$\langle 3, 511, 10\rangle$	$19 \cdot 3^{\mathbf{3}} + \mathbf{511} = 2^{\mathbf{10}}$
$\langle 4, 509, 11\rangle$	$19 \cdot 3^{\mathbf{4}} + \mathbf{509} = 2^{\mathbf{11}}$
$\langle 5, 3575, 13\rangle$	$19 \cdot 3^{\mathbf{5}} + \mathbf{3575} = 2^{\mathbf{13}}$
$\langle 6, 2533, 14\rangle$	$19 \cdot 3^{\mathbf{6}} + \mathbf{2533} = 2^{\mathbf{14}}$
$\langle 7, 23983, 16\rangle$	$19 \cdot 3^{\mathbf{7}} + \mathbf{23983} = 2^{\mathbf{16}}$
$\ldots$	

Natomiast, nie każda trójka liczb naturalnych reprezentuje jakąś liczbę naturalną, np. trójki $\langle 2, 4, 11\rangle$, $\langle 2, 4044, 11\rangle$ ...

Definicja 3. *Dwie trójki $\langle x, y, z\rangle$ i $\langle u, v, t\rangle$ są* równoważne *gdy reprezentują tę samą liczbę naturalną*

$$\langle x, y, z\rangle \equiv \langle u, v, t\rangle \quad \stackrel{\mathbf{df}}{=} \quad \frac{2^z - y}{3^x} \in N \wedge \frac{2^z - y}{3^x} = \frac{2^t - v}{3^u}$$

[1]Przypomnijmy, prawo Archimedesa jest własnością algorytmiczną, prawo to nie jest wyrażalne formułą pierwszego rzędu.

W zbiorze trójek możemy zdefiniować porządek leksykograficzny $\prec$.

Definicja 4.

$$\langle x,y,z\rangle \prec \langle u,v,t\rangle \overset{df}{\Leftrightarrow} z<t \text{ lub } z=t \text{ i } y<v \text{ lub } z=t \text{ i } y=v \text{ i } x<u$$

Z faktu 1 wynika, że dla kazdej liczby naturalnej n można wskazać kodującą ją trójkę, która ma liczbę y większą od dowolnie wybranej liczby naturalnej k. Z uwagi tej skorzystamy w dalszym ciągu naszego dowodu.

Definicja 5. *Nieparzystość* trójek wyznaczona jest przez nieparzystość liczby y

$$odd(\langle x,y,z\rangle) \overset{df}{\Leftrightarrow} \text{y jest nieparzyste}$$

Kolejne spostrzeżenie

Fakt 6. *Liczba n jest nieparzysta wttw gdy reprezentująca ją trójka $\langle x,y,z\rangle$ jest nieparzysta.*

Definicja 7. Jedynka

$$equal1(\langle x,y,z\rangle) \overset{df}{\Leftrightarrow} (2^z - y = 3^x)$$

Jedynka jest reprezentowana przez wiele trójek, np. $\langle 0,0,0\rangle, \langle 0,1,1\rangle, \langle 1,1,2\rangle, \langle 1,5,3\rangle, ...$
Na trójkach możemy określić dwie operacje

Definicja 8. Operacja $div2$ przeprowadza trójkę $\langle x,y,z\rangle$ w trójkę $\langle x, y \div 2, z-1\rangle$

$$\langle x,y,z\rangle \xrightarrow{\{div2\}} \langle x, y \div 2, z-1\rangle$$

Operacja $mult3$ przeprowadza trójkę $\langle x,y,z\rangle$ w trójkę $\langle x-1, y-3^{x-1}, z\rangle$

$$\langle x,y,z\rangle \xrightarrow{\{mult3\}} \langle x-1, y-3^{x-1}, z\rangle$$

Zauważ

Fakt 9. *Trójka $\langle x,y,z\rangle$ reprezentuje liczbę n wttw gdy trójka $\langle x, y \div 2, z-1\rangle$ reprezentuje liczbę $n \div 2$.*
Ponadto $\langle x, y \div 2, z-1\rangle \prec \langle x,y,z\rangle$.
Trójka $\langle x,y,z\rangle$ reprezentuje liczbę n wttw gdy trójka $\langle x-1, y-3^{x-1}, z\rangle$ reprezentuje liczbę $3n+1$.
Ponadto $\langle x-1, y-3^{x-1}, z\rangle \prec \langle x,y,z\rangle$.

Zauważmy z kolei

Lemat 10. *Jeśli jedna iteracja algorytmu Collatza, polegająca na wykonaniu instrukcji $\{\mathbf{if}\ odd\ \mathbf{then}\ mult3\ \mathbf{else}\ div2\ \mathbf{fi}\}$ przeprowadza trójkę $\langle x,y,z\rangle$ w trójkę $\langle u,v,t\rangle$, to wynikowa trójka $\langle u,v,t\rangle$ jest mniejsza $\prec$ od trójki $\langle x,y,z\rangle$.*

$$\langle x,y,z\rangle \xrightarrow{\{\mathbf{if}\ odd\ \mathbf{then}\ mult3\ \mathbf{else}\ div2\ \mathbf{fi}\}} \langle u,v,t\rangle \text{ implikuje } \langle u,v,t\rangle \prec \langle x,y,z\rangle$$

Dowód:
Jeśli liczba y jest parzysta to od z odejmujemy jedynkę i dzielimy y przez 2. W przeciwnym przypadku zmniejszana jest liczba x i od y odejmowana jest liczba 3^x.
W każdym kroku obliczenia wyrażenie $x + z$ zmniejsza swą wartość o 1, zmniejszana jest też wartość zmiennej y. □

Wynika stąd, że każde obliczenie w strukturze $\mathfrak{T}$ jest skończone.
Obserwacje poczynione dotychczas pozwalają stwierdzić, że następujący diagram jest przemienny.

Fakt 11. (Przemienność diagramu)
Działanie na trójkach prowadzi od trójki kodującej liczbę n do kolejnej trójki, która reprezentuje wynik m instrukcji warunkowej {if odd(n) then n:=3*n+1 else n:=n div 2 fi}.

$$\begin{array}{ccc} n & \xrightarrow{\{\textbf{if } odd(n) \textbf{ then } m{:=}3n{+}1 \textbf{ else } m{:=}n/2 \textbf{ fi}\}_{\mathfrak{N}}} & m \\ {\scriptstyle n\cdot 3^x+y=2^z}\downarrow & & \downarrow{\scriptstyle m\cdot 3^u+v=2^t} \\ \langle x,y,z\rangle & \xrightarrow{\{\textbf{if } odd(y) \textbf{ then } u,v,t{:=}x{-}1,y{-}3^{x-1},z \textbf{ else } u,v,t{:=}x,y/2,z{-}1 \textbf{ fi}\}_{\mathfrak{T}}} & \langle u,v,t\rangle \end{array}$$

Przemienność tego diagramu umożliwia nam dostrzeżenie, że każdemu obliczeniu algorytmu Collatza w strukturze $\mathfrak{T}$ trójek, odpowiada obliczenie tego samego algorytmu w strukturze $\mathfrak{N}$ liczb naturalnych.

Przyjmijmy oznaczenie K dla programu {if odd then mult3 else div2 fi}. Napis $K_{\mathfrak{N}}$ oznacza funkcję realizująca program K w strukturze $\mathfrak{N}$. Odpowiednio, napis $K_{\mathfrak{T}}$ oznacza funkcję realizująca program K w strukturze $\mathfrak{T}$.

Lemat 12. *Niech n będzie dowolną liczbą naturalną.*
Rozpatrzmy dwa obliczenia algorytmu Collatza realizowane w strukturze $\mathfrak{T}$ trójek rozpoczynające się, odpowiednio, od trójek q i r. Oba obliczenia są skończone. Litery $q, q_1, q_2, \ldots q_f$ oznaczają kolejne trójki pierwszego obliczenia. Ciąg $r, r_1, r_2, \ldots r_g$ to drugie obliczenie. Pionowe znaki równości przypominają, że trójka q_i koduje liczbę n_i.

$$\begin{array}{ccccccccccccc}
q & \xrightarrow[\succ]{K_{\mathfrak{T}}} & q_1 & \xrightarrow[\succ]{K_{\mathfrak{T}}} & q_2 & \xrightarrow[\succ]{K_{\mathfrak{T}}} & q_3\ldots & \xrightarrow[\succ]{K_{\mathfrak{T}}} & q_f & & & & \\
\| & & \| & & \| & & \| & & \| & & & & \\
n & \xrightarrow{K_{\mathfrak{N}}} & n_1 & \xrightarrow{K_{\mathfrak{N}}} & n_2 & \xrightarrow{K_{\mathfrak{N}}} & n_3\ldots & \xrightarrow{K_{\mathfrak{N}}} & n_f\ldots & \xrightarrow{K_{\mathfrak{N}}} & n_{g-1} & \xrightarrow{K_{\mathfrak{N}}} & n_g \\
\| & & \| & & \| & & \| & & \| & & \| & & \| \\
r & \xrightarrow[\succ]{K_{\mathfrak{T}}} & r_1 & \xrightarrow[\succ]{K_{\mathfrak{T}}} & r_2 & \xrightarrow[\succ]{K_{\mathfrak{T}}} & r_3\ldots & \xrightarrow[\succ]{K_{\mathfrak{T}}} & r_f\ldots & \xrightarrow[\succ]{K_{\mathfrak{T}}} & r_{g-1} & \xrightarrow[\succ]{K_{\mathfrak{T}}} & r_g
\end{array}$$

Oba obliczenia na ich części wspólnej realizują tę samą część obliczenia algorytmu realizowanego w strukturze liczb naturalnych. Co więcej, gdy $f < g$ to pomiędzy początkowymi trójkami tych obliczeń zachodzi relacja $q \prec r$.

A więc można realizować obliczenia algorytmu Collatza "*na trójkach*". Obliczenie w strukturze $\mathfrak{T}$ jest zawsze skończone. Wynika to z następującego twierdzenia algorytmicznej teorii liczb naturalnych $\mathcal{ATN}$.

$$\mathcal{ATN} \vdash \forall x\{\mathbf{while}\ x \neq 0\ \mathbf{do}\ x := x - 1\ \mathbf{od}\}(x = 0)$$

Co prawda, może się zdarzyć, że obliczenie takie zakończy się błędem, patrz Tabela 2. Ale wystarczy wybrać odpowiednio dużego reprezentanta x, y, z liczby n i rozpocząć obliczenie od tej trójki by błędu uniknąć. Zauważmy, że zawsze można wybrać taką trójkę reprezentującą liczbę n, że wszystkie liczby x, y, z będą tak duże jak zechcemy. W ten sposób gwarantujemy sobie, że obliczenie w strukturze $\mathfrak{T}$ będzie skończone i wolne od błędu (tj. *udane*). A stąd wynika, że skończone jest obliczenie dla liczby n w strukturze $\mathfrak{N}$.

W celu udowodnienia, że dla każdej liczby naturalnej n istnieje taka reprezentująca ją trójka $\langle x, y, z\rangle$, że obliczenie rozpoczynające się od tej trójki jest skończone i udane rozpatrzmy następujący algorytm CC.

$$CC: \left\{\begin{array}{l}
(*\ \text{do liczby } n \text{ dobierz liczby } x, y, z \text{ tak by } \ n \cdot 3^x + y = 2^z\ *) \\
\quad z := \lceil log\, n \rceil;\ zS := z; \\
\quad x := \text{największa liczba taka, że } 2^z > n * 3^x; \\
\quad y := 2^z - n * 3^x; \\
\quad nieznaleziono := true; Err := false; \\
\mathbf{while}\ nieznaleziono\ \mathbf{do} \\
\qquad (*\ \text{wykonaj algorytm IC dla trójki <x,y.z>}\ *) \\
IC: \left[\begin{array}{l}
\quad \mathbf{while}\ 3^x + y \neq 2^z\ \mathbf{do} \\
\quad\ \mathbf{if}\ x = 0 \text{ and y nieparzyste } \mathbf{then}\ Err := true;\ exit\ \mathbf{fi}; \\
\quad\ \mathbf{if}\ y\ parzyste\ \mathbf{then}\ z := z - 1;\ y := y\, div\, 2; \\
\quad\ \mathbf{else}\ x := x - 1;\ y := y - 3^x\ \mathbf{fi} \\
\quad \mathbf{od};
\end{array}\right] \\
\quad \mathbf{if}\ Err \\
\quad \mathbf{then}\ (*\ \text{dobierz nową trójkę}\ *) \\
\qquad z := zS + 1;\ zS := z; \\
Z: \quad [\ x := \text{największa liczba taka, że } 2^z > n * 3^x;\ xS := xS + 1\ ] \\
\qquad y := 2^z - n * 3^x;\ Err := false \\
\quad \mathbf{else} \\
\qquad nieznaleziono := false \\
\quad \mathbf{fi} \\
\mathbf{od}
\end{array}\right\}$$

Jest oczywiste, że

Fakt 13. *Jeśli obliczenie algorytmu CC w strukturze trójek $\mathfrak{T}$ jest skończone i wolne od błędu, to algorytm C wykonywany w standardowym modelu liczb naturalnych $\mathfrak{N}$ zatrzymuje się po skończonej liczbie*

kroków.

Można udowodnić parę faktów o algorytmie CC.

Fakt 14. *Każde obliczenie wychodzi z wewnętrznej pętli IC i albo jest sukces i osiągnięto jedynkę albo jest błąd i obliczenia nie można kontynuować.*

$$\mathfrak{N} \models \forall n\,(n \cdot 3^x + y = 2^z) \implies \{IC\}\big((3^x + y = 2^z) \vee (x = 0 \wedge y\ parzyste)\big)$$

Lemat 15. *Rozważmy dwa kolejne obliczenia w wewnętrznej pętli IC. Załóżmy, że początkową trójką pierwszego obliczenia jest $\langle x, y, z\rangle$, a drugie obliczenie zaczyna się od trójki $\langle x{+}1, y', z{+}1\rangle$. Przyjmijmy, że oba obliczenia są nieudane, w pierwszym obliczeniu wykonano f kroków, a w drugim obliczeniu wykonano g kroków. Pierwsze obliczenie kończy się trójką $\langle 0, 2k+1, z_f\rangle$, a drugie obliczenie kończy się trójką $\langle 0, 2m+1, z_g\rangle$.*
Jeśli spełnione są te założenia, to drugie obliczenie jest dłuższe o co najmniej dwa kroki i końcowa trójka drugiego obliczenia jest mniejsza od końcowej trójki pierwszego obliczenia

$$f + 2 \leq g \quad \text{oraz} \quad \langle 0, 2m+1, z_g\rangle \prec \langle 0, 2k+1, z_f\rangle.$$

Dowód:
Zacznijmy od dwu łatwych obserwacji:

- Z tego, że trójka $\langle x, y, z\rangle$ reprezentuje liczbę naturalną, wynika, że $y < 2^z$.
- Z tego, że obie trójki $\langle x, y, z\rangle$ oraz $\langle x+1, y', z+1\rangle$ reprezentują tę samą liczbę n wynika, że $y' = 3y - 2^z$.

A więc $3y - 2^z < 2y$.
Porównajmy dwa obliczenia

lp.	**Lewe**	**Prawe**
1	$\langle x, y, z\rangle$	$\langle x+1, 3y-2^z, z+1\rangle$
...	...	...
f	$\langle 0, 2k+1, z_f\rangle$	$\langle 1, 2p+1, z_f+1\rangle$
$f+1$		$\langle 0, 2p, z_f+1\rangle$
$f+2$		$\langle 0, p, z_f\rangle$

Prawe obliczenie może być dłuższe jeśli p jest liczbą parzystą, ale nie musi tak być. Stąd, $f + 2 \leq g$
Przypomnijmy, że $3y - 2^z < 2y$. Wobec tego $p < k$, bowiem w każdym kroku prawego obliczenia '*naśladujemy*' decyzje podejmowane w tym samym kroku obliczenia lewego, dzieląc przez 2 lub odejmując 3^x. Przez indukcję ze względu na długość lewego obliczenia otrzymujemy $2p+1 < 2(2k+1)$. Po kolejnych dwu krokach obliczenia mamy trójkę $\langle 0, p, z_f\rangle$. Jeśli jest to ostatnia trójka prawego obliczenia, tzn. gdy $p = 2m+1 \wedge g = f+2$, to zachodzi $\langle 0, 2m+1, z_g\rangle \prec \langle 0, 2k+1, z_f\rangle$. Jeśli prawe obliczenie jest jeszcze dłuższe, $g > f+2$, to tym bardziej końcowa trójka prawego obliczenia jest mniejsza od końcowej trójki lewego obliczenia, por. lemat 10. □

Lemat 16. *Rozważmy teraz dwa kolejne obliczenia w wewnętrznej pętli IC. Załóżmy, że początkową trójką pierwszego obliczenia jest $\langle x, y, z\rangle$, a drugie obliczenie zaczyna się od trójki $\langle x, y', z{+}1\rangle$. Przyjmijmy, że oba obliczenia są nieudane, w pierwszym obliczeniu wykonano f kroków, a w drugim obliczeniu wykonano g kroków. Pierwsze obliczenie kończy się trójką $\langle 0, 2k+1, z_f\rangle$, a drugie obliczenie kończy się trójką $\langle 0, 2m+1, z_g\rangle$. Jeśli spełnione są te założenia, to oba obliczenia są tej samej długości $f = g$.*

Dowód:
Oba obliczenia przebiegają podobnie. Oba zaczynają się od trójek reprezentujących tę samą liczbę n i w kazdym kroku ta własność zostaje zachowana. Gdy pierwsze obliczenie zakończy się na trójce $\langle 0, 2k+1, z_f\rangle$, to po tej samej liczbie kroków drugie obliczenie dochodzi do trójki $\langle 0, 2m+1, z_f+1\rangle$, ponieważ oba obliczenia zaczynają z tą samą wartością x i w każdym kroku, parzystości kolejnych trójek w tych dwu obliczeniach są takie same. Można tego dowieść przez indukcję ze względu na długość pierwszego obliczenia. Zobacz też diagram w lemacie 12. □

Chcielibyśmy, by własność zmniejszających się trójek kończących nieudane obliczenia instrukcji IC była niezmiennikiem programu CC, ale tak nie jest na co wskazuje poprzedzający to zdanie lemat 16. Nie przeszkodzi to nam w udowodnieniu następującego lematu.

Lemat 17. *Dla każdej liczby naturalnej $n > 0$ obliczenie algorytmu CC jest skończone.*

Dowód:
Należy wykazać, że istnieje taka trójka $\langle x, y, z\rangle$, że obliczenie wewnętrznej pętli IC będzie udane tzn. po skończonej liczbie kroków otrzymamy trójkę spełniającą warunek $n \cdot 3^x + y = 2^z$. Zauważmy, że w obliczeniu algorytmu CC możemy pominąć te fragmenty o jakich mówi lemat 16. Co prawda obliczenia algorytmu wewnętrznego IC kończą się trójką większą od poprzedniej, ale stanowią wierną imitację poprzedniego obliczenia algorytmu IC.
Czytelnik zechce samodzielnie dokonać odpowiednich zmian w algorytmie CC.
Jeśli rozpatrzymy dowolne obliczenie tak zmodyfkowanego algorytmu, to mamy do czynienia z sekwencją obliczeń algorytmu IC. Każde takie obliczenie jest coraz dłuższe i kończy się coraz mniejszą trójką. Wynika stąd, że pewien podciąg (tj. obliczenie algorytmu IC) kończy się trójką reprezentującą liczbę 1. □

2.1. Twierdzenie Collatza

Powyższe uwagi pozwalają nam sformułować następujące

Twierdzenie 18. (Collatza)
Dla dowolnej liczby naturalnej $n > 0$ obliczenie algorytmu Collatza jest skończone.

Dowód:
Dowód wynika z faktu, że każda kolejna trójka jest mniejsza $\prec$ od poprzedzającej ją trójki.
W celu uniknięcia błędu przerwania obliczeń "*na trójkach*" ...

wystarczy obliczenie rozpocząć od trójki z odpowiednio dużymi liczbami x, y, z, reprezentującej tę samą liczbę n. □

3. Wnioski

- Nietrudno zauważyć, że realizacja obliczeń w niestandardowym modelu arytmetyki dodawania może mieć obliczenie nieskończone.
  Jako wniosek z twierdzenia Collatza uzyskujemy następujący

  Fakt 19. Jeśli realizujemy algorytm Collatza w pewnym modelu teorii (Peano) pierwszego rzędu i jeśli w tej strukturze dla pewnego n jest obliczenie nieskończone, to struktura ta jest modelem niestandardowym.

  .

- Algorytm Collatza wyznacza pewną *permutację* zbioru N liczb naturalnych.

- Można opisać inny algorytm (odwrotny) szukający zadanej liczby n po kolei w warstwach, poczynając od warstwy S_0.

  Fakt 20. Algorytm odwrotny zawsze znajdzie liczbę n.

  Algorytm ten przyporządkowuje liczbie n numer i warstwy oraz położenie j liczby n w warstwie. W ten sposób mamy nową *funkcję pary*. Numerem pary liczb $\langle i, j\rangle$ jest j-ta liczba w warstwie S_i.

4. Nowe pytania

1. Jak określić funkcję kosztu?

2. Czy algorytm Collatza wyznacza najmniejszą trójkę x, y, z taką, że obliczenie zaczynające się od niej, będzie wolne od błędu?

3. Rozumowanie przytoczone powyżej wykazuje *prawdziwość* twierdzenia Collatza. Twierdzenie to nie jest zawarte w zbiorze twierdzeń arytmetyki Peano. (...)
   Jak przeprowadzić *dowód* tego twierdzenia w algorytmicznej teorii liczb naturalnych?

Tablica 1. Tabela obliczeń algorytmu Collatza dla n=1079

n	**x**	**y**	**z**
1079	15	1697398531	34
3238	14	1692615562	34
1619	14	846307781	33
4858	13	844713458	33
2429	13	422356729	32
7288	12	421825288	32
3644	12	210912644	31
1822	12	105456322	30
911	12	52728161	29
2734	11	52551014	29
1367	11	26275507	28
4102	10	26216458	28
2051	10	13108229	27
6154	9	13088546	27
3077	9	6544273	26
9232	8	6537712	26
4616	8	3268856	25
2308	8	1634428	24
1154	8	817214	23
577	8	408607	22
1732	7	406420	22
866	7	203210	21
433	7	101605	20
1300	6	100876	20
650	6	50438	19
325	6	25219	18
976	5	24976	18
488	5	12488	17
244	5	6244	16
122	5	3122	15
61	5	1561	14
184	4	1480	14
92	4	740	13
46	4	370	12
23	4	185	11
70	3	158	11
35	3	79	10
106	2	70	10
53	2	35	9
160	1	32	9
80	1	16	8
40	1	8	7
20	1	4	6
10	1	2	5
5	1	1	4
16	0	0	4
8	0	0	3
4	0	0	2
2	0	0	1
1	0	0	0

Tablica 2. Cztery obliczenia dla n=19

3	511	10	19
2	502	10	58
2	251	9	29
1	248	9	88
1	124	8	44
1	62	7	22
1	31	6	11
0	30	6	34
0	15	5	17
-1	Err		

4	509	11	19
3	482	11	58
3	241	10	29
2	232	10	88
2	116	9	44
2	58	8	22
2	29	7	11
1	26	7	34
1	13	6	17
0	12	6	52
0	6	5	26
0	3	4	13
-1	Err		

5	3575	13	19
4	3494	13	58
4	1747	12	29
3	1720	12	88
3	860	11	44
3	430	10	22
3	215	9	11
2	206	9	34
2	103	8	17
1	100	8	52
1	50	7	26
1	25	6	13
0	24	6	40
0	12	5	20
0	6	4	10
0	3	3	5
-1	Err		

6	2533	14	19
5	2290	14	58
5	1145	13	29
4	1064	13	88
4	532	12	44
4	266	11	22
4	133	10	11
3	106	10	34
3	53	9	17
2	44	9	52
2	22	8	26
2	11	7	13
1	8	7	40
1	4	6	20
1	2	5	10
1	1	4	5
0	0	4	16
0	0	3	8
0	0	2	4
0	0	1	2
0	0	0	1

Rozpoczęcie obliczeń od zbyt “małej” początkowej trójki reprezentującej liczbę n może zakończyć się błędem. Zauważ, dla większych trójek początkowych obliczenia są dłuższe, a końcowe trójki tworzą ciąg malejący.

$$\langle 0, 15, 5\rangle \succ \langle 0, 3, 4\rangle \succ \langle 0, 3, 3\rangle \succ \langle 0, 0, 0\rangle$$

Tablica 3. Dalsze obliczenia dla n=19

6	2533	14	19
5	2290	14	58
5	1145	13	29
4	1064	13	88
4	532	12	44
4	266	11	22
4	133	10	11
3	106	10	34
3	53	9	17
2	44	9	52
2	22	8	26
2	11	7	13
1	8	7	40
1	4	6	20
1	2	5	10
1	1	4	5
0	0	4	16
0	0	3	8
0	0	2	4
0	0	1	2
0	0	0	1

7	23983	16	19
6	23254	16	58
6	11627	15	29
5	11384	15	88
5	5692	14	44
5	2846	13	22
5	1423	12	11
4	1342	12	34
4	671	11	17
3	644	11	52
3	322	10	26
3	161	9	13
2	152	9	40
2	76	8	20
2	38	7	10
2	19	6	5
1	16	6	16
1	8	5	8
1	4	4	4
1	2	3	2
1	1	2	1

9	150311	19	19
8	143750	19	58
8	71875	18	29
7	69688	18	88
7	34844	17	44
7	17422	16	22
7	8711	15	11
6	7982	15	34
6	3991	14	17
5	3748	14	52
5	1874	13	26
5	937	12	13
4	856	12	40
4	428	11	20
4	214	10	10
4	107	9	5
3	80	9	16
3	40	8	8
3	20	7	4
3	10	6	2
3	5	5	1

14	43341317	27	19
13	41746994	27	58
13	20873497	26	29
12	20342056	26	88
12	10171028	25	44
12	5085514	24	22
12	2542757	23	11
11	2365610	23	34
11	1182805	22	17
10	1123756	22	52
10	561878	21	26
10	280939	20	13
9	261256	20	40
9	130628	19	20
9	65314	18	10
9	32657	17	5
8	26096	17	16
8	13048	16	8
8	6524	15	4
8	3262	14	2
8	1631	13	1

Wszystkie, odpowiednio “duże” trójki zapewniają obliczenie skończone i bez błędu. Końcowe trójki reprezentują liczbę 1.