SpecVer & Program Calculus 1–33
Dombrova Research 2021

On Collatz theorem

Report of – September 23, 2021

Grażyna Mirkowska
Faculty of Mathematics and Natural Sciences
UKSW Wóycickiego 1/3
01-938 Warszawa Poland
G.Mirkowska@uksw.edu.pl

Andrzej Salwicki
Dombrova Research
Partyzantów 19
05-092 Łomianki, Poland
salwicki@gmail.com

Abstract. Collatz conjecture has a proof.
We present a couple of observations. 1. The problem is of algorithmic nature, The conjecture states that Collatz algorithm Cl has the halting property. 2. There is an evidence of infinite execution of the program in a non-standard model of Peano arithmetic. 3. For every natural number n there exists numbers x, y, z such that the equation $n \cdot 3^x + y = 2^z$ holds. 4. Another algorithm IC computes on triples x, y, z. The consecutivve states of memory of any computation form monotone, descending sequences. 5. Hence, if a computation on triples is finite and successful then the corresponding computation of Collatz algorithm is finite too. 6. We construct an infinite set Z of elementary sentences that express the negation of halting proprty of Collatz algorithm. 7. The set Ax' of formulas that contains all axioms of elementary theory of addition of natural numbers and the set Z is consistent and has a model. Let $\mathfrak{M}$ denote any structure which is a model of axioms Ax'. 8. We show that the structure $\mathfrak{M}$ is not isomorphic to the standard structure of natural numbers with addition.
From this we infer that execution of Collatz algorithm in standard model of arithmetic is finite.

1. Introduction

The conjecture formulated by Lothar Collatz in 1937 is an algorithmic problem [1]. It should be shown that the following Cl program, executed in the standard structure $\mathfrak{N}$ of natural numbers with addition[2] has a finite computation for each $n \neq 0$.
We will be investigating the stop property of the following Cl program.

$$Cl : \left\{ \begin{array}{l} \textbf{while } n \neq 1 \textbf{ do} \\ \quad \textbf{if } even(n) \textbf{ then } n := \frac{n}{2} \textbf{ else } n := 3n+1 \textbf{ fi} \\ \textbf{od} \end{array} \right\}$$

In 2004, we noticed that there is a counterexample, see Appendix C (page 29). The two conclusions that can be made out of it, are:

- The formulation of the Collatz hypothesis requires clarification that the calculations are carried out in the standard model of natural numbers with addition . Note that, the operations of multiplication by 3 and division by 2 can be defined in Presburger arithmetic by means of addition.

- The Peano axioms, much less the Presburger axioms, are not sufficient to prove the conjecture.

And we found that if the conjecture is true, then it is a theorem of the algorithmic theory of natural numbers $\mathcal{ATN}$, (see the page 3).

2. Halting formula

Halting formula of a program K is any formula that expresses the finiteness of a computation of the program K.
Note, for many programs, their halting formulas are beyond the language of first-order logic. However, for every program its halting property can be expressed by an algorithmic formula of program calculus (i.e. algorithmic logic).

[1] It may be surprising that some people place this problem in the theory of dynamical systems.
[2] programmers may prefer another formulation: *execution of CL algorithm in unsigned integers of unlimited precision*

The following formula (halt) expresses the halting property of Collatz program Cl. It is to be shown that the formula is valid in the $\mathfrak{N}$ structure with addition natural numbers.

$$\left\{\begin{array}{l}\textbf{while } n \neq 1 \textbf{ do}\\ \quad \textbf{if } even(n) \textbf{ then } n := \frac{n}{2} \textbf{ else } n := 3n+1 \textbf{ fi}\\ \textbf{od}\end{array}\right\}(n=1) \qquad \textbf{(halt)}$$

Formula (LC) that asserts, there exists an iteration of the program if ... fi such that the condition $(n = 1)$ holds after is a halting formula of program Cl too.

$$\bigcup\left\{\begin{array}{l}\textbf{if } n \neq 1 \textbf{ then}\\ \quad \left[\begin{array}{l}\textbf{if } n \,\underline{\mathrm{mod}}\, 2 = 1\\ \qquad \textbf{then } n \leftarrow 3n+1\\ \qquad \textbf{else } n \leftarrow n \,\underline{\mathrm{div}}\, 2\\ \textbf{fi}\end{array}\right]\\ \textbf{fi}\end{array}\right\}(n=1) \qquad \textbf{(LC)}$$

Our goal is to show that the halting formula is valid in the standard structure $\mathfrak{N}$ of natural numbers or to prove the formula in algorithmic theory of natural numbers $\mathcal{ATN}$. For any model of the theory is isomorphic to the structure $\mathfrak{N}$. See [MS87], page 139.

$$\text{Axioms of } \mathcal{ATN} \left\{\begin{array}{l}\forall_x\, x+1 \neq 0\\ \forall_{x,y}\, x+1 = y+1 \implies x = y\\ \forall_x \{y := 0; \textbf{while } y \neq x \textbf{ do } y := y+1 \textbf{ od}\}(y = x)\end{array}\right\} \qquad \textbf{(ATN)}$$

Theory $\mathcal{ATN}$ can be extended, we can add definitions of useful operations $+, \times 3, \div 2$, and the parity unary relation. We shall use the following notation the atomic formula $e(n)$ reads n ***is even***, and formula $o(n)$ reads n ***is odd***.

Making use of axioms of calculus of programs $\mathcal{AL}$ and axioms of algorithmic theory of natural numbers $\mathcal{ATN}$ we can write a couple of formulas that are equivalent to the halting formula halt. Below

we presnt one of these formulas

$$\left\{\begin{array}{l}
(o(n) \wedge \mathbf{n}=1)\vee \\
(e(n) \wedge o(\frac{n}{2}) \wedge \mathbf{n}=2)\vee \\
(e(n) \wedge e(\frac{n}{2}) \wedge o(\frac{n}{4}) \wedge \mathbf{n}=4)\vee \\
(e(n) \wedge e(\frac{n}{2}) \wedge e(\frac{n}{4}) \wedge o(\frac{n}{8}) \wedge \mathbf{n}=8)\vee \\
(e(n) \wedge e(\frac{n}{2}) \wedge e(\frac{n}{4}) \wedge e(\frac{n}{8}) \wedge o(\frac{n}{16}) \wedge \mathbf{n}=16)\vee \\
\left(\begin{array}{l} e(n) \wedge e(\frac{n}{2}) \wedge e(\frac{n}{4}) \wedge e(\frac{n}{8}) \wedge e(\frac{n}{16}) \wedge o(\frac{n}{32}) \wedge \mathbf{n}=32)\vee \\ o(n) \wedge e(3n+1) \wedge e(\frac{3n+1}{2}) \wedge e(\frac{3n+1}{4}) \wedge e(\frac{3n+1}{8}) \wedge o(\frac{3n+1}{16}) \wedge \mathbf{n}=5 \end{array}\right)\vee \\
\left(\begin{array}{l} (e(n) \wedge e(\frac{n}{2}) \wedge e(\frac{n}{4}) \wedge e(\frac{n}{8}) \wedge e(\frac{n}{16}) \wedge e(\frac{n}{32}) \wedge o(\frac{n}{64}) \wedge \mathbf{n}=64)\vee \\ e(n) \wedge o(\frac{n}{2}) \wedge e(\frac{3n+1}{2}) \cdots \wedge \mathbf{n}=10 \end{array}\right)\vee \\
\left(\begin{array}{l} e(n) \wedge e(\frac{n}{2}) \wedge e(\frac{n}{4}) \wedge e(\frac{n}{8}) \wedge e(\frac{n}{16}) \wedge e(\frac{n}{32}) \wedge e(\frac{n}{64}) \wedge o(\frac{n}{128}) \wedge \mathbf{n}=128)\vee \\ o(n) \wedge e(3n+1) \wedge e(\frac{3n+1}{2}) \cdots \wedge \mathbf{n}=21) \\ e(n) \wedge o(\frac{n}{2}) \wedge e(\frac{3n+1}{2}) \cdots \wedge \mathbf{n}=20\vee \\ o(n) \wedge e(3n+1) \wedge o(\frac{3n+1}{2}) \cdots \wedge \mathbf{n}=3 \end{array}\right)\vee \\
\left\{\begin{array}{l} \mathbf{if}\ n \neq 1\ \mathbf{then} \\ \quad \left[\begin{array}{l} \mathbf{if}\ n \,\underline{\mathrm{mod}}\, 2 = 1 \\ \quad \mathbf{then}\ n \leftarrow 3n+1 \\ \quad \mathbf{else}\ n \leftarrow n \,\underline{\mathrm{div}}\, 2 \\ \mathbf{fi} \end{array}\right] \\ \mathbf{fi} \end{array}\right\}^{8} \bigcup \left\{\begin{array}{l} \mathbf{if}\ n \neq 1\ \mathbf{then} \\ \quad \left[\begin{array}{l} \mathbf{if}\ n \,\underline{\mathrm{mod}}\, 2 = 1 \\ \quad \mathbf{then}\ n \leftarrow 3n+1 \\ \quad \mathbf{else}\ n \leftarrow n \,\underline{\mathrm{div}}\, 2 \\ \mathbf{fi} \end{array}\right] \\ \mathbf{fi} \end{array}\right\} (n=1)
\end{array}\right\}$$

Based on these experiments, we can make some observations:

- There are many equivalent formulas that express the halting property of Collatz algorithm, among others, each formula δ_i from the sequence $\{\delta_i\}$, $i = 0.1.2.\ldots$

$$\bigvee_{j=0}^{i} \{K^j\}(n=1) \vee \{K^{i+1}\} \bigcup \{K\}(n=1) \qquad \textbf{gdzie } i = 0.1.2. \qquad (\delta_i)$$

  Evidently, the property can be expressed by another formula.

- istnieje łatwy przepis na utworzenie i-tej formuły tego ciągu

$$\delta_{i+1} = \{K\}\delta_i \quad \textbf{oraz } \delta_0 = (o(n) \wedge \mathbf{n} = 1)$$

- Let an algebraic structure $\mathfrak{A}$ has the signature consistent with the signature of elementary theory of natural numbers with

addition, let v be a valuation of variable n. From the definition of semantics we have

$$(\delta_i)_{\mathfrak{A}}(v) = \underset{i\in N}{l.u.b.}\left(\bigvee_{j=0}^{i}\{K^j\}(n=1)\right)_{\mathfrak{A}}(v)$$

- **the correspondence between the levels of the Collatz tree and the components of the halting formula is clearly visible,**
- **at higher levels of the Collatz tree more and more odd numbers appear - and the subsequent components of the halting formula are alternatives of longer and longer conjunctions of factors $o(\centerdot)$ or $e(\centerdot)$,**
- **Note, the alternative $(n = 128 \vee n = 21 \vee n = 20 \vee n = 3)$ may be written as follows $(n \cdot 3^0 + 0 = 2^7 \vee n \cdot 3^1 + 1 = 2^6 \vee n \cdot 3^1 + 4 = 2^6 \vee n \cdot 3^2 + 5 = 2^5)$.**

In the section 4 we shall use this observation.

3. Collatz tree

It is easy to notice that the set of those natural numbers for which the computation of the Collatz algorithm is finite, forms a tree.

Definition 3.1. ***Collatz tree*** $\mathcal{DC}$ **is a subset** $D \subset N$ **of the set** N **of natural numbers and the function** f **defined on the set** $D \setminus \{0,1\}$.

$$\mathcal{DC} = \langle D, f\rangle$$

where $D \subset N, 1 \in D, f\colon D \setminus \{0,1\} \to D$.
Function f **is determined as follows**

$$f(n) = \begin{cases} n \div 2 & \textbf{when } n \mod 2 = 0 \\ 3n+1 & \textbf{when } n \mod 2 = 1 \end{cases}$$

, the set D **is the least set containing the number 1 and closed with respect to the function** f**,**

$$D = \{n \in N : \exists_{i\in N}\ f^i(n) = 1\ \}\ .$$

As it is easy to see, this definition is highly entangled and the decision whether the set D contains every natural number is equivalent to the Collatz problem.

Remark 3.1. Set D has the following properties :

$$
\begin{aligned}
x \in D &\implies (x+x) \in D && (1)\\
x \in D \wedge \exists_y x = y+y &\implies y \in D && (2)\\
x \in D \wedge \exists_y x = y+y+1 &\implies (x+x+x+1) \in D && (3)\\
x \in D \wedge (\exists_e \exists_z e = z+z+1 \wedge x = e+e+e+1) &\implies e \in D && (4)
\end{aligned}
$$

Implications (1) and (4) show left and right son of element x. Similar, interesting properties has the complement of set D, if it is a non-empty set. Let $cD \stackrel{df}{=} N \setminus D$ denote the cmplement of set D.

Remark 3.2. If the complement $N \setminus D$ is a non-empty set, then it has similar properties:

$$
\begin{aligned}
x \in cD &\implies (x+x) \in cD && (5)\\
x \in cD \wedge \exists_y x = y+y &\implies y \in cD && (6)\\
x \in cD \wedge \exists_y x = y+y+1 &\implies (x+x+x+1) \in cD && (7)\\
x \in cD \wedge (\exists_e \exists_z e = z+z+1 \wedge x = e+e+e+1) &\implies e \in cD && (8)
\end{aligned}
$$

Note, both sets D and cD may be considered as graphs. Their structures are similar. However, the graph cD is not a tree .

Remark 3.3. *If Collatz conjecture is not true, then both sets D and $N \setminus D$ are infinite.*

From properties (6) and (7) follows the

Fact 3.1. If an x element does not belong to the Collatz tree then the computation of the Collatz algorithm starting with the state $v(n) = x$ is not finite.

4. Triples

The observations made when analyzing the halting formula of Collatz algorithm inspired us to introduce the notion of triple representing a given natural number n. Weshall also discuss the computations "on triples". Let us begin with the following remark

Fact 4.1. ***For every natural number*** $n \neq 0$ ***there exist many triples*** x, y, z ***of natural numbers such that the following equality holds***

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

We say that triple x, y, z **represents the number** n **and denote it by** $\langle x, y, z \rangle \asymp n$.

Proof:
The proof of this intuitive fact uses the law of Archimedes [3].

Let n be an arbitrary natural number. We choose a number x, it may be an arbitray natural number. Let the number $k = n \cdot 3^x$. Put $z = (\mu z)(2^z \geq k)$. and $y = 2^z - n \cdot 3^x$. Obviously the equality $n \cdot 3^x + y = 2^z$. holds. □

Example 4.1. Triple $\langle 1, 7, 6 \rangle$ represents number 19 for $19 \cdot 3 + 7 = 2^6$. Number 19 is represented by many moretriples.

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

Note, not every triple represents a number, consider $\langle 2, 4, 11 \rangle$, $\langle 2, 4044, 11 \rangle$.

Fact 4.1 is a theorem of elementary theory of natural numbers with addition $\mathcal{T}$, c.f. section 8.

Theorem 4.1. The sentence $\forall_n \exists_{,x,y.z}\, n \cdot 3^x + y = 2^z$ is a theorem of theory $\mathcal{T}$,

The proof is in section 8. Hence the theorem is valid in any model of theory $\mathcal{T}$,

[3]We recall that the law is not an elementary property, it can be formulated by an algorithmic formula $\forall_{0<a<b}\{z := a;\ \textbf{while}\ z \leq b\ \textbf{do}\ z := z + a\ \textbf{od}\}(z > b)\ \}$.

4.1. Proprties of triples

Definition 4.1. ***Dwie trójki*** $\langle x, y, z\rangle$ ***i*** $\langle u, v, t\rangle$ ***są <u>równoważne</u> gdy reprezentują tę samą liczbę naturalną***

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

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

Definition 4.2.

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

Z faktu 4.1 wynika, że dla kazdej liczby naturalnej n można wskazać kodującą ją trójkę, która ma liczbę y większą od dowolnie wybranej liczby naturalnej k.

Definition 4.3. *Nieparzystość* trójek wyznaczona jest przez nieparzystość liczby y

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

Kolejne spostrzeżenie

Fact 4.2. ***Liczba*** n ***jest nieparzysta wttw gdy reprezentująca ją trójka*** $\langle x, y, z\rangle$ ***jest nieparzysta.***

Definition 4.4. Jedynka

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

Definition 4.5. Klasa trójek równowaznych jest zbiorem trójek reprezentujących tę samą liczbę n.

Jedynka jest reprezentowana przez wiele trójek, np. $\langle 0, 0, 0\rangle, \langle 0, 1, 1\rangle, \langle 1, 1, 2\rangle, \langle 1, 5, 3\rangle, \langle 2,$

Na trójkach możemy określić dwie operacje

Definition 4.6. Operacja $div2$ przeprowadza trójkę parzystą $\langle x, y, z\rangle$ w trójkę $\langle x, y \div 2, z - 1\rangle$

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

Definition 4.7. Operacja $mult3$ określona jest na trójkach $\langle x, y, z \rangle$ takich, że trójka $\langle x, y, z \rangle$ jest nieparzysta i $x > 0$ i $y > 3^{x-1}$. W takim przypadku 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

Fact 4.3. ***Trójka*** $\langle x, y, z \rangle$ ***reprezentuje liczbę parzystą*** n ***wttw gdy trójka*** $\langle x, y \div 2, z - 1 \rangle$ ***reprezentuje liczbę*** $n \div 2$.
Ponadto $\langle x, y \div 2, z - 1 \rangle \prec \langle x, y, z \rangle$.
Jeśli liczba y ***jest nieparzysta i zachodzi*** $x > 0 \wedge y > 3^{x-1}$ ***to 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

Lemma 4.1. ***Jeśli program*** K***:*** {if odd then $mult3$ else $div2$ 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*** $\langle x, y, z \rangle$.

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

Proof:
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 algorytmu Collatza 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 dotychchczas pozwalają stwierdzić, że następujący diagram jest przemienny.

Fact 4.4. ***Działanie na trójkach prowadzi od trójki*** $\langle x, y, z \rangle$ ***kodującej liczbę*** n ***i spełnajacej przy tym warunek*** $\zeta : \big((y \mod 2 = 1) \Rightarrow (x > 0 \wedge y > 3^{x-1})\big)$***, do kolejnej trójki*** $\langle u, v, t \rangle$***, która reprezentuje wynik*** m ***następującej instrukcji warunkowej*** K : {if $odd(n)$ then $n := 3 * n + 1$ else $n := ndiv2$ fi}.

$$\zeta \Rightarrow \left| \begin{array}{ccc} n & \xrightarrow{K:\{\mathbf{if}\ \ odd(n)\ \ \mathbf{then}\ \ m{:=}3n{+}1\ \ \mathbf{else}\ \ m{:=}n/2\ \ \mathbf{fi}\}_{\mathfrak{N}}} & m \\ \uparrow{\scriptstyle m\cdot 3^x+y=2^z} & & {\scriptstyle m\cdot 3^u+v=2^t}\uparrow \\ \langle x,y,z\rangle & \xrightarrow{\bar{K}:\{\mathbf{if}\ \ odd(y)\ \ \mathbf{then}\ \ u,v,t:=x-1,y-3^{x-1},z\ \ \mathbf{else}\ \ u,v,t:=x,y/2,z-1\ \ \mathbf{fi}\}_{\mathfrak{T}}} & \langle u,v,t\rangle \end{array} \right|$$

Diagram ten jest przemienny pod warunkiem, że można wykonać operację odejmowania tzn. gdy y jest liczbą parzystą lub y jest nieparzyste i ($x > 0$ lub $y > 3^{x-1}$). Semantyczną treść opisaną powyższym diagramem można wyrazic następującą formuła

$$\big((n \cdot 3^x + y = 2^z \wedge odd(y)) \Rightarrow (x > 0 \wedge y > 3^{x-1}))\big) \Rightarrow \{K; \bar{K}\}(n \cdot 3^x + y = 2^z)$$

Pamiętajmy, że programy K oraz $\bar{K}$ są niezależne, a więc przemienne.

$$\big((n \cdot 3^x + y = 2^z \wedge odd(y)) \Rightarrow (x > 0 \wedge y > 3^{x-1}))\big) \Rightarrow \{\bar{K}; K\}(n \cdot 3^x + y = 2^z)$$

Przemienność tego diagramu pozwala nam dostrzec, że każdemu obliczeniu algorytmu Collatza Cl w strukturze $\mathfrak{N}$ liczb naturalnych, odpowiada obliczenie sprzężonego z Cl, algorytmu IC w strukturze trójek $\mathfrak{T}$.

Lemma 4.2. Jeśli spełniony jest warunek θ

$$\theta:\ (n \cdot 3^x + y = 2^z) \wedge \bigcap \left\{ \begin{array}{l} \mathbf{if}\ 3^x + y \neq 2^z \\ \mathbf{then} \\ \quad \mathbf{if}\ odd(y) \\ \quad \mathbf{then}\ x, y := x-1, y - 3^{x-1} \\ \quad \mathbf{else}\ z, y := z-1, y \div 2 \\ \quad \mathbf{fi} \\ \mathbf{fi} \end{array} \right\} (odd(y) \implies (x > 0 \wedge y > 3^{x-1}))$$

to następujący diagram jest przemienny

$$\begin{array}{ccc} n & \xrightarrow{\{\mathbf{while}\ \ n\neq 1\ \ \mathbf{do}\ \ \mathbf{if}\ \ odd(n)\ \ \mathbf{then}\ \ n:=3n+1\ \ \mathbf{else}\ \ n:=n/2\ \ \mathbf{fi}\ \ \mathbf{od}\}_{\mathfrak{N}}} & 1 \\ \uparrow{\scriptstyle n\cdot 3^x+y=2^z} & & {\scriptstyle 3^x+y=2^z}\uparrow \\ \langle x,y,z\rangle & \xrightarrow[\left\{ \begin{array}{l} \mathbf{while}\ 3^x+y\neq 2^z\ \mathbf{do} \\ \quad \mathbf{if}\ odd(y)\ \mathbf{then}\ x,y,z := x-1, y-3^{x-1}, z \\ \quad \mathbf{else}\ x,y,z := x, y/2, z-1\ \mathbf{fi} \\ \mathbf{od} \end{array} \right\}_{\mathfrak{T}}]{} & \langle x,y,z\rangle \end{array} \qquad \text{(CCI)}$$

Co wyraża się następującą formułą

$$\theta \Rightarrow \big(\{Cl\}(n = 1) \Leftrightarrow \{IC\}(3^x + y = 2^z)\big)\ . \qquad \text{(RCI)}$$

Proof:
gos by induction with respect to the number of iterations. Base of induction reduces to the commutativity of the preceding fact

$$\zeta \Rightarrow \left[\begin{array}{ccc} n & \xrightarrow{\{K\}_{\mathfrak{N}}} & m \\ {\scriptstyle n\cdot 3^x+y=2^z}\uparrow & & \uparrow{\scriptstyle m\cdot 3^u+v=2^t} \\ \langle x,y,z\rangle & \xrightarrow{\bar{K}_{\mathfrak{T}}} & \langle u,v,t\rangle \end{array}\right].$$

If the execution of algorithm is longer, say of length 1, then the following diagram applies

$$(\zeta \wedge \{\bar{K}\}\zeta) \Rightarrow \left[\begin{array}{ccccc} n & \xrightarrow{\{K\}_{\mathfrak{N}}} & n_1 & \xrightarrow{\{K\}_{\mathfrak{N}}} & n_2 \\ {\scriptstyle n\cdot 3^x+y=2^z}\uparrow & & {\scriptstyle n_1\cdot 3^{x_1}+y_1=2^{z_1}}\uparrow & & {\scriptstyle n_2\cdot 3^{x_2}+y_2=2^{z_2}}\uparrow \\ \langle x,y,z\rangle & \xrightarrow{\bar{K}_{\mathfrak{T}}} & \langle x_1,y_1,z_1\rangle & \xrightarrow{\bar{K}_{\mathfrak{T}}} & \langle x_2,y_2,z_2\rangle \end{array}\right].$$

By induction, we prove that for every natural number i the following diagram commutes

$$\bigwedge_{j=0}^{i}\{\bar{K}\}^j\zeta \Rightarrow \left|\begin{array}{ccccccccc} n & \xrightarrow{\{K\}_{\mathfrak{N}}} & n_1 & \xrightarrow{\{K\}_{\mathfrak{N}}} & n_2 & \xrightarrow{\cdots} & n_{i-1} & \xrightarrow{\{K\}_{\mathfrak{N}}} & n_i \\ {\scriptstyle n\cdot 3^x}\uparrow{\scriptstyle +y=2^z} & & {\scriptstyle n_1\cdot 3^{x_1}+y_1}\uparrow{\scriptstyle =2^{z_1}} & & {\scriptstyle n_2\cdot 3^{x_2}+y_2}\uparrow{\scriptstyle =2^{z_2}} & & \cdots\uparrow & & {\scriptstyle n_i\cdot 3^{x_i}+y_i=2^{z_i}}\uparrow \\ \langle x,y,z\rangle & \xrightarrow{\bar{K}_{\mathfrak{T}}} & \langle x_1,y_1,z_1\rangle & \xrightarrow{\bar{K}_{\mathfrak{T}}} & \langle x_2,y_2,z_2\rangle & \xrightarrow{\cdots} & \langle x_{i-1},y_{i-1},z_{i-1}\rangle & \xrightarrow{\bar{K}_{\mathfrak{T}}} & \langle x_i,y_i,z_i\rangle \end{array}\right|$$

Now, if for some $i \in N$ the number $n_i = 1$, then $\forall_k n_{i+k} = 1$. Thus, the formula RCI has a proof. □

Formuła RCI jest twierdzeniem algorytmicznej teorii $\mathcal{ATN}$ ale nie jest ŕownowazna twierdzeniu Collatza. W rozdziale 5 nawiążemy do tego twierdenia.

The same conclusion may be obtained by an almost formal proof of the formula

$$\theta \wedge \neg\{n := \varepsilon;\ Cl\}(n = 1) \Rightarrow \neg\{IC\}true$$

Myslę, że da się dowieść takiego zdania (z meta-teorii $\mathcal{ATN}$ dla każdej liczby naturalnej $i = o, 1, \ldots$ twierdzeniem teorii $\mathcal{ATN}$ jest zdanie o postaci

$$\varepsilon 3^x + y = 2^z \wedge y_i \wedge \{n := \varepsilon; K^i\}(n \neq 1) \Rightarrow \{\bar{K}^i\}(3^x + y \neq 2^z)$$

Jeśli tak jest to stosując dwukrotnie aksjomat Ax_{23} otrzymamy stwierdzenie, ze twierdzeniem teorii $\mathcal{ATN}$ jest każda formuła $(n3^x + y = 2^z \wedge \bigcap\{K\}(oddy \rightarrow) \wedge \neg\{Cl\}(n = 1) \Rightarrow \{\bar{K}\}^i(3^x + y \neq 2^z)$, gdzie $i = 0, 1, \ldots.$ Stad przez zastosowanie reguły wnioskowania otrzymamy
$n3^x + y = 2^z \wedge \bigcap\{K\}(oddy) \wedge \neg\{Cl\}(n = 1) \Rightarrow \neg\{IC\}(Err \vee 3^x + y = 2^z)$

4.2. Zbiory T_n

Każdej liczbie n przyporządkowujemy zbiór T_n trójek x, y, z takich, że zachodzi równość $n3^x + y = 2^z$.
Zbiór taki jest niepusty.
Zbiory T_n są parami rozłączne, tj. $n \neq m \implies \mathrm{T}_n \cap \mathrm{T}_m = \emptyset$.
Zbiór T_n jest zamknięty ze względu na operacje $o_1, o_2, o_3, o_4, o_5, o_6$

$$\begin{aligned}
o_1(\langle x,y,z\rangle) &= \langle x+1, 3y+2^z, z+2\rangle && (9)\\
o_2(\langle x,y,z\rangle) &= \langle x+1, 3y-2^z, z+1\rangle \quad \textbf{gdy } 3y > 2^z && (10)\\
o_3(\langle x,y,z\rangle) &= \langle x, y+2^z, z+1\rangle && (11)\\
o_4(\langle x,y,z\rangle) &= \langle x, y-2^{z-1}, z-1\rangle \quad \textbf{gdy } y > 2^{z-1} && (12)\\
o_5(\langle x,y,z\rangle) &= \langle x-1, \frac{y-2^{z-2}}{3}, z-2\rangle \quad \textbf{gdy } y > 2^{z-2} \textbf{ i } (y-2^{z-2})\mathrm{mod}3 = 0 && (13)\\
o_6(\langle x,y,z\rangle) &= \langle x-1, \frac{y+2^{z-1}}{3}, z-1\rangle && (14)
\end{aligned}$$

Wynika stąd, że zbiór T_n jest nieskończony i nieograniczony.

Zauważ związki

$o_5(o_1(\langle x, y, z\rangle) = \langle x, y, z\rangle$
$o_4(o_3(\langle x, y, z\rangle) = \langle x, y, z\rangle$
$o_6(o_2(\langle x, y, z\rangle) = \langle x, y, z\rangle$
$o_4(o_1(\langle x, y, z\rangle) = \langle x, y, z\rangle$
O trójkach zbędnych (niepotrzebnych) ...
Trójka wieksza od trójki przydatnej nie musi być przydatna, może być zbędna.
PRZYKŁAD.
Ale też od każdej trójki dobrej istnieje większa od niej trójka dobra.

4.3. Program IC

A więc można realizować obliczenia algorytmu Collatza "***na trójkach***". Rozważmy następujący program. Zakładamy, że program IC startuje z wartościami x, y, z spełniającymi warunel $n \cdot 3^x + y = 2^z \wedge \neg Err$. Jest to warunek ***wstępny*** obliczeń.

$$IC : \left\{ \begin{array}{l} \textbf{while } 3^x + y \neq 2^z \ \ (*\ tj.\ n \neq 1\ *) \textbf{ do} \\ \qquad \textbf{if } \big(odd(y) \wedge ((x = 0) \textbf{ or } (y < 3^{x-1}))\big) \textbf{ then } Err := true; \textbf{ exit fi}; \\ \qquad \textbf{if } odd(y) \textbf{ then } x := x - 1;\ y := y - 3^x;\ (*n := 3 * n + 1\ *) \\ \qquad \textbf{else } \ z := z - 1;\ y := y \operatorname{div} 2;\ (*\, n := \frac{n}{2}\ *) \textbf{ fi} \\ \textbf{od} \end{array} \right\}$$

Obliczenie w strukturze trójek jest zawsze skończone. Wynika to z następującego twierdzenia algorytmicznej teorii liczb naturalnych $\mathcal{ATN}$.

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

Co prawda, może się zdarzyć, że obliczenie algorytmu IC zakończy się niepowodzeniem, $Err = true$.

4.4. Własności programu IC

Fakt 1. ***Każde obliczenie programu*** IC ***jest skończone i albo jest sukces i osiągnięto jedynkę albo jest błąd i obliczenia algorytmu IC nie można kontynuować.***

$$\mathfrak{N} \models \forall n\,(n \cdot 3^x + y = 2^z) \implies \{IC\}\Big(\underbrace{(3^x + y = 2^z)}_{\langle x,y,z\rangle \asymp 1} \vee \underbrace{(odd(y) \wedge (x = 0 \vee y < 3^{x-1})}_{Error}\Big)$$

Błąd o jakim tu mowa to niemożność kontynuowania obliczeń. Jest tak gdy *y jest nieparzyste* **i** $y < 3^{x-1}$ **lub gdy** *y jest nieparzyste* **i** $x = 0$. **Pierwszy błąd łatwo wyeliminować, zauważ, że trójka** $\langle x, y, z\rangle$ **reprezentuje tę samą liczbę co trójka** $\langle x, y+2^z, z+1\rangle$. **Wystarczy więc w sytuacji gdy** *y jest nieparzyste* **i** $y < 3^{x-1}$ **zastąpić trójkę** $\langle x, y, z\rangle$ **przez trójkę** $\langle x, y+2^z, z+1\rangle$. **Zniknie też ryzyko wystąpienia błędu drugiego rodzaju. Upoważnia to nas do sformułowania następującej uwagi.**

Uwaga 2. **Jeśli obliczenie algorytmu** IC **rozpoczynające się od pewnej trójki** $\langle x, y, z\rangle$ **kończy sie po** k **iteracjach błędem** $Err = true$ **to obliczenie algorytmu** IC **rozpoczynającego się od trójki** $\langle x+1, 3y+2^z, z+2\rangle$ **będzie dłuższe o co najmniej dwie iteracje. Obie te trójki reprezentują tę sama liczbę** n.

4.5. Program IIC

Powyższe uwagi prowadzą do napisania programu poszukujacego dla liczby n **dobrej trójki ją reprentujacej.**

```
IIC:  read(n);
      Let z = (μr)(2^r ≥ n).
      x, x_s := 0;  z_s := z;  y, y_s := 2^z − n;
      Err := false;
      while 3^{x_s} + y_s ≠ 2^{z_s} do
   IC:   while 3^x + y ≠ 2^z do
            if odd(y) ∧ (x = 0 ∨ 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 · y_s;
           Err := false;
        else  exit fi;
      od
```

Nietrudno zauważyć, że program IIC **pomija trójki prowadzące do błedu".**

Remark 4.1. **Dla danej liczby** n **program** IIC **ma obliczenie nieskończone wtedy i tylko wtedy gdy program** Cl **ma obliczenie nieskończone.**

Program IIC dokonuje systematycznego przeszukiwania w zbiorze T_n i ewentualnie zwraca najmniejszą trójke "dobrą", tj. taką, że obliczenie programu IC jest udane.

5. Proof of Collatz theorem

Our plan may be summarized in four points:

(*i*) We proved that the sentence $\forall_n \exists_{x,y,z}\, n \cdot 3^x + y = 2^z$ is a theorem of the elementary theory $\mathcal{T}^+$ of natural numbers with addition (Presburger theory). See Appendix B, page 28. Hence, this sentence holds true in any model of the theory.

(*ii*) In Appendix C, page 29, we show infinite computations of Collatz's algorithm in a non-standard (non-Archimedean) model of Presburger arithmetic Ar. The model is computable and programmable. This example is not a complete counter-example against Collatz conjecture. It only shows that the Collatz conjecture is not a theorem of elementary theory of natural numbers with addition or any other elementary theory.

(*iii*) We proved that there is a model $\mathfrak{M}$ of the $\mathcal{T}^+$ theory such, that it contains an element ε ,for which the Collatz algorithm has an infinite computation, c.f. lemma 5.1. ***We do not assume*** that this model contains unreachable elements.

(*iv*) We show that in any model of $\mathcal{T}^+$ theory , if for a certain n element, the computation of the Collatz algorithm is infinite, then the model is not isomorphic to the standard model of natural numbers (for it contains unreachable elements).

From this we conclude, that if a model has no unreachable elements, then there are no infinite computations.

5.1. Structure in which some element has an infinite Collatz computation.

We have known that in the non-standard model of Presburger arithmetic $\mathfrak{M}$ unreachable elements have infinite Collatz computations. The model is described in the literature, see [Grz71]. We provide examples of infinite Collatz computations and a definition

of this structure in a programming language in section 9, Appendix C. Now, we are going to to show that there are non-Collatz elements without assuming that they are unreachable elements.
We construct an elementary theory $\mathcal{T}^+$ which is an extension of the theory of $\mathcal{T}$ (i.e. Presburger arithmetic) in a way that permits to show an element different from any number that occurs in Collatz tree.
The extension of theory $\mathcal{T}$ is made in three steps

1. (*language*) we add a new constant ε to the alphabet and correspondingly we extend the sets of term and of formulas of the language of theory $\mathcal{T}$.

2. (*logic*) the operation of consequence remains the same, remember the sets of terms and of formulas are bigger,

3. (axioms of data structure) to the set Ax of Presburger's axioms we add an infinite set of sentences Z.

Our intention is to prove the following fact. An assumption that for some element ε the Collatz computation is infinite, does not lead to a contradiction with axioms of elementary theory of addition of natural nymbers (i.e. the Presburger's theory).

As a natural reflex, we would like to add the negation of the instance of halting formula for $n = \varepsilon$ to the axioms of of $\mathcal{T}$ theory. However the formula

$$\neg\{n \leftarrow \varepsilon\} \left\{ \begin{array}{l} \textbf{while } n \neq 1 \textbf{ do} \\ \quad \textbf{if } even(n) \\ \quad \textbf{then } n \leftarrow n \,\textbf{div}\, 2 \\ \quad \textbf{else } n \leftarrow 3n+1 \\ \quad \textbf{fi} \\ \textbf{od} \end{array} \right\} (n = 1)$$

does not belong to the language of elementary theory of $\mathcal{T}$. Moreover, the following formula 15 that expresses the same looping property of computation of Collatz algorithm does not belong to

the language of first-order theory Ar.

$$\{n \leftarrow \varepsilon\} \bigcap \left\{ \begin{array}{l} \textbf{if } n \neq 1 \textbf{ then} \\ \quad \textbf{if } n \,\mathbf{mod}\, 2 = 0 \\ \quad \textbf{then } n \leftarrow n \textbf{ div } 2 \\ \quad \textbf{else } n \leftarrow 3n+1 \\ \quad \textbf{fi} \\ \textbf{fi} \end{array} \right\} (n \neq 1) \tag{15}$$

However, for every algebraic structure $\mathfrak{A}$ the above formula (15) is valid in $\mathfrak{M}$ if and only if every formula of the following scheme is valid in $\mathfrak{A}$

$$\begin{array}{l} \{n \leftarrow \varepsilon\}(n \neq 1), \\ \{n \leftarrow \varepsilon\} \left\{ \begin{array}{l} \textbf{if } n \neq 1 \textbf{ then} \\ \quad \textbf{if } even(n) \textbf{ then } n \leftarrow n \textbf{ div } 2 \\ \quad\quad \textbf{else } n \leftarrow 3n+1 \textbf{ fi} \\ \textbf{fi} \end{array} \right\} (n \neq 1), \\ \dots \\ \{n \leftarrow \varepsilon\} \left\{ \begin{array}{l} \textbf{if } n \neq 1 \textbf{ then} \\ \quad \textbf{if } even(n) \textbf{ then } n \leftarrow n \textbf{ div } 2 \\ \quad\quad \textbf{else } n \leftarrow 3n+1 \textbf{ fi} \\ \textbf{fi} \end{array} \right\}^i (n \neq 1), \\ \dots \end{array}$$

Each of these formulas is equivalent to certain first-order formula ϑ, such that the formula does not contain any algorithm. One can verify this claim making use of axioms of assignment instruction e.g. $\{n \leftarrow \varepsilon\}(n > 1) \equiv (\varepsilon > 1)\}$, conditional instruction and composition of programs. We illustrate our claim by the following equivalence.

$$\{n \leftarrow \varepsilon\}\{\textbf{if } P(n) \textbf{ then } n \leftarrow n \,\mathbf{div}\, 2 \textbf{ else } n \leftarrow 3n+1 \textbf{ fi}\}(n \neq 1) \equiv$$
$$\Big(\big(P(\varepsilon) \wedge \varepsilon \neq 2\big) \vee \underbrace{\big(\neg P(\varepsilon) \wedge 3\varepsilon + 1 \neq 1\big)}_{false} \Big)$$

Continuing, we obtain the following formulas

Formuła
$(o(\varepsilon) \wedge \varepsilon \neq 1)$
$(e(\varepsilon) \wedge o(\frac{\varepsilon}{2}) \wedge \varepsilon \neq 2)$
$(e(\varepsilon) \wedge e(\frac{\varepsilon}{2}) \wedge o(\frac{\varepsilon}{4}) \wedge \varepsilon \neq 4)$
$(e(\varepsilon) \wedge e(\frac{\varepsilon}{2}) \wedge e(\frac{\varepsilon}{4}) \wedge o(\frac{\varepsilon}{8}) \wedge \varepsilon \neq 8)$
$(e(\varepsilon) \wedge e(\frac{\varepsilon}{2}) \wedge e(\frac{\varepsilon}{4}) \wedge e(\frac{\varepsilon}{8}) \wedge o(\frac{\varepsilon}{16}) \wedge \varepsilon \neq 16)$
$\left(\begin{array}{l} e(\varepsilon) \wedge e(\frac{\varepsilon}{2}) \wedge e(\frac{\varepsilon}{4}) \wedge e(\frac{\varepsilon}{8}) \wedge e(\frac{\varepsilon}{16}) \wedge o(\frac{\varepsilon}{32}) \wedge \varepsilon \neq 32) \vee \\ o(\varepsilon) \wedge e(3\varepsilon+1) \wedge e(\frac{3\varepsilon+1}{2}) \wedge e(\frac{3\varepsilon+1}{4}) \wedge e(\frac{3\varepsilon+1}{8}) \wedge o(\frac{3\varepsilon+1}{16}) \wedge \varepsilon \neq 5 \end{array}\right)$
$\left(\begin{array}{l} (e(\varepsilon) \wedge e(\frac{\varepsilon}{2}) \wedge e(\frac{\varepsilon}{4}) \wedge e(\frac{\varepsilon}{8}) \wedge e(\frac{\varepsilon}{16}) \wedge e(\frac{\varepsilon}{32}) \wedge o(\frac{\varepsilon}{64}) \wedge \varepsilon \neq 64 \vee \\ e(\varepsilon) \wedge o(\frac{\varepsilon}{2}) \wedge e(\frac{3\varepsilon+1}{2}) \cdots \wedge \varepsilon \neq 10 \end{array}\right)$
$\left(\begin{array}{l} e(\varepsilon) \wedge e(\frac{\varepsilon}{2}) \wedge e(\frac{\varepsilon}{4}) \wedge e(\frac{\varepsilon}{8}) \wedge e(\frac{\varepsilon}{16}) \wedge e(\frac{\varepsilon}{32}) \wedge e(\frac{\varepsilon}{64}) \wedge o(\frac{\varepsilon}{128}) \wedge \varepsilon \neq 128 \vee \\ o(\varepsilon) \wedge e(3\varepsilon+1) \wedge e(\frac{3\varepsilon+1}{2}) \cdots \wedge \varepsilon \neq 21 \vee \\ e(\varepsilon) \wedge o(\frac{\varepsilon}{2}) \wedge e(\frac{3\varepsilon+1}{2}) \cdots \wedge \varepsilon \neq 20 \vee \\ o(\varepsilon) \wedge e(3\varepsilon+1) \wedge o(\frac{3\varepsilon+1}{2}) \cdots \wedge \varepsilon \neq 3 \end{array}\right)$ ...

Horizontal lines separate formulas corresponding to different levels of Collatz tree. We define the set Z as containing all formulas $\varepsilon \neq \mathrm{k}$ such„that the expression $n = \mathrm{k}$ occurs in halting formula of Collatz algorithm, k is number. Hence, the set Z contains the sentences $\{\varepsilon \neq 1, \varepsilon \neq 2, \varepsilon \neq 4, \varepsilon \neq 8, \varepsilon \neq 16, \varepsilon \neq 32, \varepsilon \neq 5, \varepsilon \neq 64, \varepsilon \neq 10, \varepsilon \neq 128, \varepsilon \neq 20, \varepsilon \neq 21, \varepsilon \neq 3, \dots\}$

Na zbiór Ax' aksjomatów nowej teorii składają się: aksjomaty Ax teorii Presburgera (por. Dodatek A, strona 26) oraz nieskończony zbiór Z.

$$Ax' = Ax \cup Z$$

Udowodnimy, że zbiór Ax' jest niesprzeczny. Zaczniemy od wykazania, że każdy skończony podzbiór Ax_0 zbioru Ax' jest niesprzeczny . Wykażemy mianowicie, że zbiór Ax_0 posiada model w standardowej strukturze $\mathfrak{N}$ liczb naturalnych z dodawaniem. Nasze zadanie ogranicza się do podania właściwej interpretacji stałej: ε.
Niech Z_0 będzie dowolnym skończonym podzbiorem zbioru Z.
Zbiór zdań $Ax \cup Z_0$ jest niesprzeczny. Aby to wykazać weźmy jako

wartość stałej ε jakąś liczbę l większą od każdej liczby k występującej w zbiorze zdań Z_0. Taki wybór zapewnia prawdziwośc każdego zdania ze zbioru Z_0.

We proved that every finite subset of the set Ax' is consistent. By the compactness theorem on first-order logic[4], we obtain

Lemma 5.1. The set Ax' is a consistentset of formulas.

Now, we apply the model existence theorem[5], which reads: ***for every consistent set S of first-order formulas there exists an algebraic structure $\mathfrak{A}$ such that it is a model of the set S, i.e. for every formula $\sigma \in S$ the formula is valid in the structure $\mathfrak{A}$.***
In this way we proved the following lemma.

Lemma 5.2. There is an algebraic structure $\mathfrak{M}$, such that every sentence of the set Ax' is valid in it.

Corollary 5.1. The execution of Collatz algorithm in structure $\mathfrak{M}$ that starts with value of variable n equal ε, $v(n) = \varepsilon$ is infinite.

For the structure $\mathfrak{M}$ is a model of the set Z.

Corollary 5.2. Element ε of structure $\mathfrak{M}$ does not belong to the Collatz tree DC.

It remains to be proved that every standard natural number has a finite Collatz computation.
This it is equivalent to the following statement ***for every element n if its Collatz computation is infinite than the element n is non-standard (unreachable) element of a model of elementary theory of natural number***.

[4]Compactness theorem *If every finite subset of a set S of formulas is consistent, then the set S is consistent too.*
[5]of first-order logic

5.2. Infinite Collatz computation require unreachable elements

Nasz dowód jest podobny do dowodu lematu 5.1 . Niech n będzie dowolnym elementem takim. że obliczenie algorytmu Collatza dla n jest nieskończone (tj. jest on nie-Collatzowy). Obliczenie algorytmu IIC dla elementu n jest nieskończone. Nie istnieje trójka $\langle x, y, z\rangle$ liczb naturalnych standardowych taka, że obliczenie algorytmu IC przebiega bez błędu Err. Najpierw udowodnimy, że istnieją trzy stałe c_x, c_y, c_ztakie, że zachodzi równość $n{\cdot}3^x{+}y = 2^z$ i obliczenie algorytmu IC przebiega beż błędu. Rozszerzamy język elementarnej teotii $\mathcal{T}^+$ dodając do niego cztery stałe: $\varepsilon, c_x, c_y, c_z$. Ustalamy, że wartością stałej ε jest element n. Wykażemy, że niesprzeczny jest zbiór $Ax' = Ax \cup Df \cup U \cup Y$ .Zbiory Ax oraz Z były opisane wczesniej. Zbiór U zawiera jedną formułe $\varepsilon{\cdot}3^x{+}y = 2^z$. Zbiór Df zawiera definicje przydatnych operacji i relacji $P2, P3, 3x, div2, even, odd$, por rozdział 7. Zbiór Y zawiera wszystkie zdania pierwszego rzędu równoważne zdaniom postaci

$$\left\{\begin{array}{l} x := c_x; \\ y := c_y; \\ z := c_z; \\ Err := false \end{array}\right\} \left\{\begin{array}{l} \textbf{if } 3^x + y \neq 2^z \textbf{ then} \\ \quad \textbf{if } (odd(y) \wedge (x = 0 \vee y < 3^{x-1})) \textbf{ then } Err := true \\ \quad \textbf{else} \\ \quad\quad \textbf{if } odd(y) \textbf{ then} \\ \quad\quad\quad x := x - 1;\ y := y - 3^x \\ \quad\quad \textbf{else} \\ \quad\quad\quad y := y \div 2;\ z := z - 1 \\ \quad\quad \textbf{fi} \\ \quad \textbf{fi} \\ \textbf{fi} \end{array}\right\}^i (\neg Err)$$

gdzie $i = 0, 1, 2, \ldots$.

Zdania zbioru Y możemy ustawić w ciąg $Y = \{y_0, y_1, y_2, \ldots\}$.

Zdanie y_i ze zbioru Y wyraża następującą własność: ***jeśli w obliczeniu trójkowym wykonano*** i ***kroków i osiągnięto stan*** $\langle x, y, z\rangle$***, to jeśli aktualna wartośćć trójki*** $\langle x, y, z\rangle$ ***nie reprezentuje jedynki, to można będzie wykonać krok*** $i + 1$***-szy. Inaczej mówiąc zdanie*** y_i ***wyklucza ryzyko błędu*** Err ***w*** $i + 1$***-szym kroku obliczenia*** .

Tak jak poprzednio pokażemy, ze każdy skończony podzbiór zbioru Ax' jest niesprzeczny. Przydadzą się nam obserwacje poczynione w rozdziale 4.5, dotyczące programu IIC. Niech Y_0 będzie dowolnym, skończonym podzbiorem zbioru Y. Niech i będzie największa liczbą iteracji wymienioną w tym zbiorze. Z własności programu

IIC wiemy, ze istnieje taka trójka $\langle x, y, z\rangle$ liczb naturalnych, że podczas wykonywania i iteracji programu $\bar{K}$ nie wystąpi błąd Err. A więc można zastosować rwierdzenie o zwartości i wywnioskować, że cały zbiór Ax' jest niesprzeczny. Z twierdzenia o pełności wnioskujemy, że istnieje taka trójka $\langle x, y, z\rangle$, że dla każdej liczby naturalnej i po wykonaniu i iteracji programu $\{\bar{K}\}^i$. stan trzech zmiennych x, y, z spełnia warunek $odd(y) \Rightarrow (x > 0 \wedge y > 3^{x-1})$. Czyli spełniony jest warunek θ występujący w lemacie 4.2. Przytoczmy go tu ponownie

$$\theta : \left(\begin{array}{l} (\varepsilon \cdot 3^{c_x} + c_y = 2^{c_z}) \wedge \\ \left\{ \begin{array}{l} x := c_x; \\ y := c_y; \\ z := c_z \end{array} \right\} \bigcap \left\{ \begin{array}{l} \textbf{if } 3^x + y \neq 2^z \\ \textbf{then} \\ \quad \textbf{if } odd(y) \\ \quad \textbf{then } x, y := x-1, y-3^{x-1} \\ \quad \textbf{else } z, y := z-1, y \div 2 \\ \quad \textbf{fi} \\ \textbf{fi} \end{array} \right\} \left(\begin{array}{l} odd(y) \implies \\ (x > 0 \wedge y > 3^{x-1}) \end{array} \right) \end{array} \right)$$

Oprócz tego warunku mamy też nieskończone obliczenie algorytmu Cl dla e;ementu ε. Koniunkcja ta pozwala wyprowadzić jako wniosek.

Corollary 5.3. If the condition θ holds and the execution of program Cl for $n = \varepsilon$ is infinite, then the computation of program IC starting with the triple $\langle c_x, c_y, c_z\rangle$ is infinite too .

Proof:
It suffices to show that the computation of the following program

$$\left\{ \begin{array}{l} x := c_x;\ y := c_y;\ z := c_z; Err := false; \\ \textbf{while } 3^x + y \neq 2^z \textbf{ do} \\ \quad \textbf{if } (odd(y) \wedge (x = 0 \vee y < 3^{x-1})) \textbf{ then } Err := true; exit \\ \quad \textbf{else} \\ \quad\quad \textbf{if } odd(y) \textbf{ then} \\ \quad\quad\quad x := x - 1;\ y := y - 3^x \\ \quad\quad \textbf{else} \\ \quad\quad\quad y := y \div 2;\ z := z - 1 \\ \quad\quad \textbf{fi} \\ \quad \textbf{fi} \\ \textbf{od} \end{array} \right\}.$$

is infinite, i.e. it does not terminate with ***Err=true***, nor with $3^x + y = 2^z$. We accept the following denotations: the symbol t denotes the triple $\langle c_x, c_y, c_z\rangle$. and the symbol $\bar{K}$ denotes the following program

$$\bar{K}: \left\{ \begin{array}{l} \textbf{if } 3^x + y \neq 2^z \textbf{ then} \\ \quad \textbf{if } (odd(y) \wedge (x = 0 \vee y < 3^{x-1})) \textbf{ then } Err := true \\ \quad \textbf{else} \\ \quad\quad \textbf{if } odd(y) \textbf{ then} \\ \quad\quad\quad x := x - 1;\ y := y - 3^x \\ \quad\quad \textbf{else} \\ \quad\quad\quad y := y \div 2;\ z := z - 1 \\ \quad\quad \textbf{fi} \\ \quad \textbf{fi} \\ \textbf{fi} \end{array} \right\}.$$

We know that the triple t represents the element ε. This is illustrated by the diagram

$$\begin{array}{ccccccccccc} \varepsilon & \xrightarrow{K_{\mathfrak{M}}} & \varepsilon_1 & \xrightarrow{K_{\mathfrak{M}}} & \varepsilon_2 & \xrightarrow{K_{\mathfrak{M}}} & \ldots\varepsilon_k & \xrightarrow{K_{\mathfrak{M}}} & \varepsilon_{k+1} & \xrightarrow{K_{\mathfrak{M}}} & \ldots \\ \Big\uparrow & & & & & & & & & & \\ t & & & & & & & & & & \end{array} \qquad \text{(Dia1)}$$

The upward arrow from t to ε means that the equality $\varepsilon \cdot 3^{c_x} + c_y = 2^{c_z}$ holds.
The sentence y_0 from the set Y is true. Hence there exists the triple $t_1 = \{\bar{K}\}^1_{T_{\mathfrak{M}}}(t)$.

$$\begin{array}{ccccccccccc} \varepsilon & \xrightarrow{K_{\mathfrak{M}}} & \varepsilon_1 & \xrightarrow{K_{\mathfrak{M}}} & \varepsilon_2 & \xrightarrow{K_{\mathfrak{M}}} & \ldots\varepsilon_k & \xrightarrow{K_{\mathfrak{M}}} & \varepsilon_{k+1} & \xrightarrow{K_{\mathfrak{M}}} & \ldots \\ \Big\uparrow & & & & & & & & & & \\ t & \xrightarrow[\succ]{\bar{K}_{T_{\mathfrak{M}}}} & t_1 & & & & & & & & \end{array} \qquad \text{(Dia2)}$$

We can add the arrow from t_1 to ε_1. It is justified by tthe Fact 4.4, page 9.

$$\begin{array}{ccccccccccc} \varepsilon & \xrightarrow{K_{\mathfrak{M}}} & \varepsilon_1 & \xrightarrow{K_{\mathfrak{M}}} & \varepsilon_2 & \xrightarrow{K_{\mathfrak{M}}} & \ldots\varepsilon_k & \xrightarrow{K_{\mathfrak{M}}} & \varepsilon_{k+1} & \xrightarrow{K_{\mathfrak{M}}} & \ldots \\ \Big\uparrow & & \Big\uparrow & & & & & & & & \\ t & \xrightarrow[\succ]{\bar{K}_{T_{\mathfrak{M}}}} & t_1 & & & & & & & & \end{array} \qquad \text{(Dia3)}$$

By induction, we obtain that *for every natural number k the following diagram is comutative*.

$$\begin{array}{ccccccccccc} \varepsilon & \xrightarrow{K_{\mathfrak{M}}} & \varepsilon_1 & \xrightarrow{K_{\mathfrak{M}}} & \varepsilon_2 & \xrightarrow{K_{\mathfrak{M}}} & \ldots\varepsilon_k & \xrightarrow{K_{\mathfrak{M}}} & \varepsilon_{k+1} & \xrightarrow{K_{\mathfrak{M}}} & \ldots \\ \Big\uparrow & & \Big\uparrow & & \Big\uparrow & & \Big\uparrow & & \Big\uparrow & & \\ t & \xrightarrow[\succ]{\bar{K}_{T_{\mathfrak{M}}}} & t_1 & \xrightarrow[\succ]{\bar{K}_{T_{\mathfrak{M}}}} & t_2 & \xrightarrow[\succ]{\bar{K}_{T_{\mathfrak{M}}}} & \ldots t_k & \xrightarrow[\succ]{\bar{K}_{T_{\mathfrak{M}}}} & t_{k+1} & \xrightarrow[\succ]{\bar{K}_{T_{\mathfrak{M}}}} & \ldots \end{array} \qquad \text{(Dia)}$$

The structure $\mathfrak{M}$ models the set Y of sentences, hence, after k-fold execution of assignment instruction {x:=x-1;y:=y-3^x} one can safely execute the same instruction again, without risk of *Err* error. Moreover, for every natural number i the equality $\varepsilon_i \cdot 3^{x_i} + y_i = 2^{z_i}$ holds, here the elements x_i, y_i, z_i are constituents of the triple t_i. Hence the computation of the program IC in the structure $\mathfrak{M}$ can be prolongued to any desired length.

Fact 5.1. Execution of program IC starting with the state $v : \frac{x|y\ |z}{c_x\ |c_y|c_z}$ is infinite.

Note, the sequence of triples in the lower row of the last diagram is decreasing! and infinite.

By earlier considerations we arrive to the conclusion that the structure $\mathfrak{M}$ contains unreachable elements.

□

From this immediately follows the following lemma.

Lemma 5.3. The structure $\mathfrak{M}$ contains unreachable elements.

We have proved that there exists an algebraic structure $\mathfrak{M}$ with the following properties:

- the structure is a model of Presburger arithmetic Ar,
- the structure contains an element ε such, that the computation of Collatz algorithm starting with ε is infinite,
- the structure contains unreachable elements.

dies 19 Septembri 2021

5.3. Collatz theorem

Let's summarize what we know:

- If the Collatz computation for an element n is finite, then there is a triple $\langle x, y, z\rangle$ such that $n \cdot 3^x + y = 2^z$ and the computation of algorithm IC for this triple is finite.

- If for some triple $\langle x, y, z\rangle$, the following equality holds $n \cdot 3^x + y = 2^z$ and the computation of algorithmIC is finite, then the computation of the Collatz algorithm for n is finite.
- if in a model $\mathfrak{M}$ of the elementary theory of natural numbers with addition (i.e. Presburger theory) an element n is unreachable, then the computation of the Collatz algorithm for n is infinite. item There is a structure $\mathfrak{M}$, model of Presburger arithmetic and an element ε such that, the computation of the Cl algorithm is infinite.
- Let an algebraic structure $\mathfrak{A}$ be a model of Presburger arithmetic. Let n be any element for which the computation of the Collatz algorithm is infinite. There are three elements: $\langle x, y, z\rangle$ such, that the equality $n \cdot 3^x + y = 2^z$ holds and the computation of the IC algorithm for this triple is infinite.

From these facts we derive the following proposition.

Theorem 5.1. (Collatz 1937)
Dla każdej liczby naturalnej n obliczenie algorytmu Collatza Cl jest skończone.

Proof:
It follows from the facts enumerated above that if for some element n of the $\mathfrak{A}$ structure, which is a model of the elementary theory of the addition of natural numbers, the calculation of the Collatz algorithm is infinite, then the structure is a non-standard model of the theory. Hence, by transposition, we obtain Collatz theorem.

□

6. Podsumowanie

Nietrudno zauważyć, że przedstawiony dowód jest okrężny. Nie potrafimy, na razie, przedstawić dowodu wyprowadzonego wprost w algorytmicznej teorii $\mathcal{ATN}$ z aksjomatów tej teorii lub np. z prawa Archimedesa, które jest twierdzeniem teorii $\mathcal{ATN}$.
Kolejne zadanie to oszacowanie kosztu algorytmu Collatza, wiemy, że obliczenia są skończone, ale nie potrafimy oszacować ich długości. Natomiast koszt odpowiedzi na pytanie czy liczba n jest elementem drzewa Collatza jest stały $O(1)$. Z twierdzenia Collatza potrafimy jednak wyprowadzić kilka ciekaych wniosków, o czym napiszemy osobno.

Podziękowania

Andrzej Szałas znalazł błąd w innej, wcześniejszej pracy na temat algorytmu Collatza. Wiktor Dańko zwrócił naszą uwagę na lukę w analizie obliczeń trójkowych. Ludwik Czaja i Marek Warpechowski przekazali nam swe wątpliwości, dzięki nim powstała obecna wersja pracy.
Wszystkie usterki i błędy są naszego autorstwa.

7. Dodatek A – Kilka faktów o elementarnej teorii dodawania

W rozdziale 4 zaobserwowaliśmy parę pożytecznych faktów o trójkach reprezentujących liczby naturalne.
Rozważać będziemy następującą teorię T_+, por. [Grz71] str. 239 i następne.

Definicja 1. Teoria $T_+ = \langle \mathcal{L}, \mathcal{C}, Ax \rangle$ jest układem trzech przedmiotów:

$\mathcal{L}$ jest językiem pierwszego rzędu. Na alfabet tego języka składaja się: zbiór V zmiennych, znaki operacji: $0, S, +$, znak relacji równości $=$, znaki funktorów logicznych i kwanrtfikatorów, znaki pomocnicze, m. in nawiasy..
Zbiór wyrażeń poprawnie zbudowanych to suma teoriomnogościowa zbioru termów T i zbioru formuł F.
Zbiór termów T jest to najmniejszy zbiór napisów zawierający zbiórzmiennych V i napis **0** i zamknięty ze względu na reguły: jeśli dwa napisy τ_1 oraz τ_2 są termami to termem jest też napis postaci $(\tau_1 + \tau_2)$, jeśli napis τ jest termem to napis $S(\tau)$ jest także termem.
Zbiór formuł jest najmniejszym zbiorem napisów zawierającym równośći tj. napisy postaci $(\tau_1 = \tau_2)$ i zamkniętym ze względu na reguły: jeśli napisy α oraz β są formułami to formułami są tez napisy postaci

$$(\alpha \vee \beta),\ (\alpha \wedge \beta),\ (\alpha \implies \beta),\ \neg\alpha$$

formułami sątież napisy postaci

$$\forall_x \alpha,\ \exists_x \alpha$$

gdzie x jest zmienną, a α jest formuła.

$\mathcal{C}$ jest operacją konsekwencji zdeterminowaną przez przyjęcie aksjomatów logiki pierwszego rzędu (rachunku predykatów) i reguł wnioskowania logiki pierwszego rzędu

Ax jest zbiorem formuł wyliczonych poniżej.

$$\forall_x \ x+1 \neq 0 \tag{a}$$
$$\forall_x \forall_y \ x+1 = y+1 \implies x = y \tag{b}$$
$$\forall_x \ x+0 = x \tag{c}$$
$$\forall_{x,y} \ (y+1)+x = (y+x)+1 \tag{d}$$
$$\{\Phi(0) \wedge \forall_x [\Phi(x) \implies \Phi(x+1)] \implies \forall_x \Phi(x) \tag{I}$$
$$\tag{16}$$

Tu napis $\Phi(x)$ należy zastąpić jakąkolwiek formułą. Ostatni wiersz jest więc schematem indukcji.
Do tego zbioru dodajemy aksjomaty definiujące dodatkowe pojęcia

$$Parz(x) \stackrel{df}{\equiv} \exists_y \, x = y+y \tag{p}$$
$$x \, div \, 2 = y \equiv (x = y+y \ \vee \ x = y+y+1) \tag{D2}$$
$$3x \stackrel{df}{=} x+x+x \tag{3x}$$

W teorii T_+ przeprowadzimy dowody paru faktów znanych z rozdziału 4. Dzięki temu upewnimy się, że fakty te prawdziwe ą także w każdym modelu teorii . Natomiast w rozdziale 5 posłużymy się teorią Presburgera

Definicja 2. Teoria $Ar = \langle \mathcal{L}, \mathcal{C}, Ax \rangle$ jest układem trzech przedmiotów:

$\mathcal{L}$ jest językiem pierwszego rzędu. Na alfabet tego języka składaja się: zbiór V zmiennych, znaki operacji: $0, +$, znak relacji równości $=$.
Zbiór wyrażeń poprawnie zbudowanych to unia zbioru termów T i zbioru formuł F. Zbiór termów T jest to najmniejszy zbiór napisów zawierający zbiórzmiennych V i napis 0 i zamknięty ze względu na reguły: jeśli dwa napisy τ_1 oraz τ_2 są termami to termem jest też napis postaci $(\tau_1 + \tau_2)$, jeśli napis τ jesttermem to napis $S(\tau)$ jest także termem.

$\mathcal{C}$ jest operacją konsekwencji zdeterminowaną przez przyjęcie aksjomatów rachunku predykatów i reguł wnioskowania logiki pierwszego rzędu

Ax jest zbiorem formuł wyliczonych poniżej.

$$\forall_x \ x+1 \neq 0 \tag{A}$$
$$\forall_x \ x \neq 0 \implies \exists_y x = y+1 \tag{B}$$
$$\forall_{x,y} \ x+y = y+x \tag{C}$$
$$\forall_{x,y,z} \ x+(y+z) = (x+y)+z \tag{D}$$
$$\forall_{x,y,z} \ x+z = y+z \implies x = y \tag{E}$$
$$\forall_x \ x+0 = x \tag{F}$$
$$\forall_{x,z} \ \exists_y \ (x = y+z \vee z = y+x) \tag{G}$$
$$\forall_x \exists_y \ (x = y+y \vee x = y+y+1) \tag{H2}$$
$$\forall_x \exists_y \ (x = y+y+y \vee x = y+y+y+1 \vee x = y+y+y+1+1) \tag{H3}$$

. . . .

$$\forall_x \exists_y \left(\begin{array}{l} x = \underbrace{y+y+\cdots+y}_{k} \vee \\ \quad x = \underbrace{y+y+\cdots+y}_{k} + 1 \vee \\ \qquad x = \underbrace{y+y+\cdots+y}_{k} + \underbrace{1+1}_{2} \vee \\ \qquad \ldots \\ \qquad\quad x = \underbrace{y+y+\cdots+y}_{k} + \underbrace{1+1+\cdots+1}_{k-2} \vee \\ \qquad\qquad x = \underbrace{y+y+\cdots+y}_{k} + \underbrace{1+1+\cdots+1}_{k-1} \end{array} \right) \tag{Hk}$$

. . .

Przypomnijmy parę faktów
F1. Teoria T_+ jest elementarnie równoważna teorii Ar.[Pre29, Sta84]
F2. Teoria Ar jest rozstrzygalna. [Pre29].
F3. Złożoność teorii Ar, czyli koszt udowodnienia, że dane zdanie α

jest twierdzeniem (lub jego negacją) jest rzędu podwójnie wykładniczego $O(2^{2^n})$. [FR79].
F4. Teorie T_+ oraz Ar mają modele niestandardowe, tj. modele zawierające elementy nieosiągalne, zob Dodatek C, strona 29.

8. Dodatek B– dowód twierdzenia 4.1

W tym rozdziale wykażemy, że zdanie *dla każdego* n *istnieją* x, y, z *takie, że* $n \cdot 3^x + y - 2^z$ jest twierdzeniem teorii T_+ dodawania. Elementarna teoria $\mathcal{T}_+$ dodawania liczb naturalnych została przypomniana w dodatku A.
Działania mnożenia i potęgowania są niedostępne w teorii T_+. Ale nie są niezbędne do osiągnięcia naszego celu.
Teorię T_+ wzbogacamy o dwie funkcje $P2(\cdot)$ oraz $P3(\cdot,\cdot)$. zdefiniowane w ten sposób

$$\begin{array}{l|l} P2(0) = 1 & P3(y,0) = y \\ P2(x+1) = P2(x) + P2(x) & P3(y,x+1) = P3(y,x) + P3(y,x) + P3(y,x) \end{array}$$

Lemat 3. Powyższe definicje są poprawne, tzn. twierdzeniami teorii T_+ wzbogaconej w ten sposób są zdania $\forall_x \exists_y\, P2(x) = y$ i $\forall_{x,y,z} P2(x) = y \wedge P2(x) = z \implies y = z$.

$$T_+ \vdash \forall_x \exists_y\, P2(x) = y \quad \textbf{i}$$

$$T_+ \vdash \forall_{x,y,z} P2(x) = y \wedge P2(x) = z \implies y = z.$$

Podobnie twierdzeniami wzbogaconej teorii T_+ są zdania $\forall_{y,x} \exists_z\, P3(y,x) = z$ i $\forall_{y,x,z,u} P3(y,x) = z \wedge P3(y,x) = u \implies z = u$.

Dowód przebiega przez indukcję względem zmiennej x.

Potrzebna nam będzie następująca definicja relacji mniejszości $a < b \stackrel{df}{=} \exists_c\, a + S(c) = b$. Wykorzystując definicje funkcji $P2$ oraz $P3$ napiszemy wyrazenie $P3(n,x) + y = P2(z)$.

Lemma 8.1. Następujące zdanie jest twierdzeniem wzbogaconej teorii T_+

$$\forall_n \exists_{x,y,z} P3(n,x) + y = P2(z)$$

Najpierw przez indukcję dowodzimy, że $T_+ \vdash \forall_n\, n < 2^n$. A dokładniej, $T_+ \vdash \forall_n\ n < P2(n)$. Łatwo sprawdzić, że $T_+ \vdash 0 < P2(0)$. Założmy, że $T_+ \vdash \forall_n \{n < P2(n)\}$. Nierówność $n+1 < P2(n+1)$ wynika z dwu

nierówności $T_+ \vdash n < P2(n)$ **i** $\mathcal{T} \vdash 1 < P2(n)$.

W podobny sposób uzyskamy $T_+ \vdash P3(n,x) < P2(z) \wedge (z = n + x + x)$
Wynika stąd, że $T_+ \vdash \forall_n \exists_{x,y,z}\ P3(n,x) + y = P2(z)$.
Właściwie wykazaliśmy, że $\forall_{n,x} \exists_{y,z} P3(n,x) + y = P2(z)$

Lemma 8.2. **Niech $\mathfrak{M}$ będzie jakimkolwiek modelem arytmetyki Presburgera. Element n jest osiągalny wtedy i tylko wtedy gdy istnieje taka trójka reprezentująca element n, tj. taka, że zachodzi równość** $P3(n,x) + y = P2(z)$ **czyli** $n \cdot 3^x + y = 2^z$ **i liczby** x, y, z **są osiągalne.**

Proof:
Jeśli spełnione są formuły
$\{q := 0;\ \mathit{while}\ q \neq x\ \mathit{do}\ q := q + 1\ \mathit{od}\}(x = q)$,
$\{q := 0;\ \mathit{while}\ q \neq y\ \mathit{do}\ q := q + 1\ \mathit{od}\}(y = q)$,
$\{q := 0;\ \mathit{while}\ q \neq z\ \mathit{do}\ q := q + 1\ \mathit{od}\}(z = q)$
i ponadto zachodzi równość
$P3(n,x) + y = P2(z)$ **to łatwo sprawdzic,że spełniona jest formuła** $\{t := 0;\ \mathit{while}\ n \neq t\ \mathit{do}\ t := t + 1\ \mathit{od}\}(t = n)$ □

9. Appendix C - an example of an infinite computation

At this point, we'll remind you of a few facts that are less known to the IT community. In Appendix A we described the Ar theory of addition of natural numbers. The only functor in the language of this theory is $+$, we also have two constants 0 and 1 and the predicate of equality $=$. Now, we will program the algebraic structure $\mathfrak{M}$, which is a model of this theory, i.e. all axioms of theory Ar are true in the structure $\mathfrak{M}$. First we will describe this structure as mathematicians do, then we will write a class (ie a program module) implementing this structure. medskip

9.1. Mathematical description of the structure

$\mathfrak{M}$ is an algebraic structure

$$\mathfrak{M} = \langle M; \mathbf{0}, \mathbf{1}, \oplus; = \rangle \qquad \text{(NonStandard)}$$

such that M is a set of pairs $\langle k, w\rangle$ where element $k \in Z$ is an integer, element w is a rational, non-negative number and the following requirements are lsatisfied:

(*i*) for each element $\langle k, w\rangle$ if $w = 0$ then $k \geq 0$,

(*ii*) the meaning of the constant **0** is $\langle 0.0\rangle$,

(*iii*) the meaning of constant 1 is $\langle 1.0\rangle$,

(*iv*) the operation $\oplus$ of addition is determined as follows

$$\langle k, w\rangle \oplus \langle k', w'\rangle \stackrel{df}{=} \langle k + k', w + w'\rangle.$$

Lemma 9.1. The algebraic structure $\mathfrak{M}$ is a model of Ar theory.

The reader will check that each axiom of the Ar theory is a sentence true in the structure $\mathfrak{M}$.
The structure $\mathfrak{M}$ is not a model of the $\mathcal{ATN}$, algorithmic theory of natural numbers. Elements of the structure $\langle k, w\rangle$. such as $w \neq 0$ are ***unreachable***. i.e. for each element $x_0 = \langle k, w\rangle$ such that $w \neq 0$ the following condition holds

$$\neg\{y := 0; \textbf{while } y \neq x_0 \textbf{ do } y := y + 1 \textbf{ od}\}(y = x_0)$$

The subset $\mathfrak{N} \subset \mathfrak{M}$ composed of only those elements for which $w = 0$ is a model of the theory $\mathcal{ATN}$. The elements of the structure $\mathfrak{N}$ are called ***reachable***. A very important theorem of the foundations of mathematics is

Fact 9.1. The structures $\mathfrak{N}$ and $\mathfrak{M}$ are not isomorphic. See [Grz71], p. 256.

As we will see in a moment, this fact is also important for IT specialists.

9.2. Definition in programming language

Perhaps you have already noticed that the $\mathfrak{M}$ is computable. The following is a class that implements the structure $\mathfrak{M}$. The implementation uses the integer type, we do not introduce rationalNumbers explicitly.

```
unit StrukturaM: class;
   unit Elm: class(k,li,mia: integer);
   begin
      if mia=0 then raise Error fi;
      if li * mia <0 then raise Error fi;
      if li=0 and k<0 then raise Error fi;
   end Elm;
   add: function(x,y:Elm): Elm;
   begin
      result := new Elm(x.k+y.k, x.li*y.mia+x.mia*y.li, x.mia*y.mia )
   end add;
   unit one : function:Elm;  begin   result:= new Elm(1,0,2)  end one;
   unit zero : function:Elm;  begin   result:= new Elm(0,0,2)  end zero;
   unit eq: function(x,y:Elm): Boolean;
   begin
      result := (x.k=y.k) and (x.li*y.mia=x.mia*y.li )
   end eq;
end StrukturaM
```

The following lemma expresses the correctness of the implementation

Lemma 9.2. The set of Elm objects with the *add* operation is a model of the Ar theory

9.3. Infinite Collatz algorithm computation

How to execute the Collatz algorithm in StructuraM? It's easy.

```
pref StrukturaM block
   var n: Elm;
   unit odd: function(x:Elm): Boolean; ... result:=(x.k mod 2)=1 ... end odd;
   unit div2: function(x:elm): Elm; ...
begin
   n:= new Elm(8,1,2);
    while  not eq(n,one)  do
       if  odd(n) then
          n:=add(n,add(n,add(n,one)))  else n:= div2(n)
       fi
    od
end block;
```

Below we present the computation of Collatz algorithm for $n = \langle 8, \frac{1}{2}\rangle$.

$$\langle 8, \frac{1}{2}\rangle,\ \langle 4, \frac{1}{4}\rangle, \langle 2, \frac{1}{8}\rangle,\ \langle 1, \frac{1}{16}\rangle,\ \langle 4, \frac{3}{16}\rangle,\ \langle 2, \frac{3}{32}\rangle,\ \langle 1, \frac{3}{64}\rangle,\ \langle 4, \frac{9}{64}\rangle,\ \langle 2, \frac{9}{128}\rangle, \cdots$$

None of the elements of the above sequence is a standard natural number. Each of them is unreachable. It is worth looking at an example of another calculation. Will something change when we assign n a different object? e.g. n: = new Elm (19,2,10)?

$$\langle 19, \tfrac{10}{2}\rangle,\ \langle 58, \tfrac{30}{2}\rangle, \langle 29, \tfrac{30}{4}\rangle,\ \langle 88, \tfrac{90}{4}\rangle,\ \langle 44, \tfrac{90}{8}\rangle,\ \langle 22, \tfrac{90}{16}\rangle,\ \langle 11, \tfrac{90}{32}\rangle,\ \langle 34, \tfrac{270}{32}\rangle,\ \langle 17, \tfrac{270}{64}\rangle,$$
$$\langle 52, \tfrac{810}{64}\rangle, \langle 26, \tfrac{405}{64}\rangle, \langle 13, \tfrac{405}{128}\rangle, \langle 40, \tfrac{1215}{128}\rangle, \langle 20, \tfrac{1215}{256}\rangle, \langle 10, \tfrac{1215}{256}\rangle, \langle 5, \tfrac{1215}{512}\rangle, \langle 16, \tfrac{3645}{512}\rangle, \langle 8, \tfrac{3645}{1024}\rangle,$$
$$\langle 4, \tfrac{3645}{2048}\rangle, \langle 2, \tfrac{3645}{4096}\rangle, \langle 1, \tfrac{3645}{8192}\rangle, \langle 4, \tfrac{3*3645}{8192}\rangle, \langle 2, \tfrac{3645*3}{2*8192}\rangle, \langle 1, \tfrac{3*3645}{4*8192}\rangle, \langle 4, \tfrac{9*3645}{4*8192}\rangle, \cdots$$

And one more computation.

$$\langle 19, 0\rangle,\ \langle 58, 0\rangle, \langle 29, 0\rangle,\ \langle 88, 0\rangle,\ \langle 44, 0\rangle,\ \langle 22, 0\rangle,\ \langle 11, 0\rangle,\ \langle 34, 0\rangle,\ \langle 17, 0\rangle, \langle 52, 0\rangle, \langle 26, 0\rangle,$$
$$\langle 13, 0\rangle, \langle 40, 0\rangle, \langle 20, 0\rangle, \langle 10, 0\rangle, \langle 5, 0\rangle, \langle 16, 0\rangle, \langle 8, 0\rangle,\ \ \langle 4, 0\rangle, \langle 2, 0\rangle, \langle 1, 0\rangle$$

Corollary 9.1. The structure $\mathfrak{M}$, which we have described in two different ways, is the model of the T_+ theory (you can also say that this structure implements the specification given by the axioms of the Ar theory), with the non-obvious presence of unreachable elements in it.

Another observation

Corollary 9.2. The halting property of the Collatz algorithm cannot be proved from the axioms of the T_+ theory, nor from the Ar theory.

The reader may wish to construct the computation that starts with $\langle 8, \frac{1}{7}\rangle$.

References

[FR79] Jeanne Ferrante and Charles W. Rackoff. The Computational Complexity of Logical Theories" Springer Verlag, Heidelberg, 1979.

[Grz71] Andrzej Grzegorczyk. Zarys Arytmetyki Teoretycznej. PWN, Warszawa, 1971,pp.311.

[MS87] Grażyna Mirkowska and Andrzej Salwicki. Algorithmic Logic. PWN,Warszawa,1987, pp.372. `http://lem12.uksw.edu.pl/wiki/Algorithmic_Logic`, 1987. [Online; accessed 7-August-2017].

[Pre29] Mojżesz Presburger. Über die Vollständigkeit eines gewissen Systems der Arit Comptes Rendus du 1-er Congres Mathematiciens des Pays Slaves, Varsovie, 1929, 95-101.

[Sta84] Ryan Stansifer. Presburger's Article on Integer Arithmetic: Remarks and Translation. Technical Report TR84-639, Cornell University, 1984.