On Collatz theorem

Report of – October 7, 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 enjoys 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.

Keywords
Collatz algorithm, halting property, program calculus, expressing and proving properties of algorithms, algorithmic theory of numbers

AMS classification
68Q25, 68Q60 , 03H15

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 26). 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 χ 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 structure $\mathfrak{N}$ of natural numbers with addition.

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

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

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

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.

———————————————- Axioms of $\mathcal{ATN}$ ——————

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

The last axiom says:*every element x is reachable*[3].

———————————————

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.

[3]Note, operations of addition, multiplication and any recursive function are definable in this theory.

Below we display 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} \textbf{if } n \neq 1 \textbf{ then} \\ \quad \left[\begin{array}{l} \textbf{if } n \,\underline{\mathrm{mod}}\, 2 = 1 \\ \quad \textbf{then } n \leftarrow 3n+1 \\ \quad \textbf{else } n \leftarrow n \,\underline{\mathrm{div}}\, 2 \\ \textbf{fi} \end{array}\right] \\ \textbf{fi} \end{array}\right\}^{8} \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 \\ \quad \textbf{then } n \leftarrow 3n+1 \\ \quad \textbf{else } n \leftarrow n \,\underline{\mathrm{div}}\, 2 \\ \textbf{fi} \end{array}\right] \\ \textbf{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. In particular (by axiom Ax_{16}[4], each formula δ_i from the sequence $\{\delta_i\}$, $i = 0.1.2.\ldots$

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

  is a halting formula of Collatz algorithm.
  Evidently, the property can be expressed by other formulas.

- There is an easy way to create the $i+1$-th formula of this sequence $\bigvee_{j=0}^{i}\{K^j\}(n = 1) \vee \{K^{i+1}\}(n=1) \vee \{K^{i+2}\}\bigcup\{K\}(n=1)$

[4] Ax_{16}. $\big(\bigcup\{M\}\alpha \equiv (\alpha \vee \{M\}\bigcup\{M\}\alpha)\big)$

- 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

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

, 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 $cD \overset{df}{=} N \setminus D$ denote the complement 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.

Let us terminate this section with the following observation

Remark 3.4. In a non-standard model of elementary theory of natural numbers with addition, the complement of Collatz tree is an infinite set.

For in this model there are unreachable elements, c.f. section 9.

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 [5].

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 **??**.

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 **??**. Hence the theorem is valid in any model of theory $\mathcal{T}$,

4.1. Properties of triples

Definition 4.1. *Triples $\langle x, y, z\rangle$ and $\langle u, v, t\rangle$ are* equivalent *if they represent the same natural number .*

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

Moreover, one can define a (lexicographical) order in the set of triples $\succ$.

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

Definition 4.2.

$$\langle u,v,t\rangle \succ \langle x,y,z\rangle \overset{df}{\Leftrightarrow} x<u \text{ or } x=u \text{ and } z<t \text{ or } x=u \text{ and } z=t \text{ and } y<v$$

It follows from the fact 4.1 that , for every natural number n there is a triple representing n such, that the number y is bigger than an arbitrarily chosen number k.

Definition 4.3. *The relation of parity of a triple is determined by the parity of number* y.

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

Another fact

Fact 4.2. *Number n is odd iff the triple $\langle x,y,z\rangle$ that represents n is odd .*

Definition 4.4. One

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

the number 1 is represented by many triples e.g. $\langle 0,0,0\rangle, \langle 0,1,1\rangle, \langle 1,1,2\rangle, \langle 1,5,3\rangle, \langle 2,7,4\rangle, ...$

We define two operations on triples

Definition 4.5. Operation $div2$ is defined on even triples $\langle x,y,z\rangle$ and results with triple $\langle x, y \div 2, z-1\rangle$.

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

Definition 4.6. Operation $mult3$ is defined on triples $\langle x,y,z\rangle$ such, that the argument is an odd triple $\langle x,y,z\rangle$ and condition $x>0 \wedge y>3^{x-1}$ holds.

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

Note

Fact 4.3. *triple $\langle x,y,z\rangle$ represents an even number n if and only if the triple $\langle x, y\div 2, z-1\rangle$ represents the number $n \div 2$.*
Moreover, $\langle x, y\div 2, z-1\rangle \prec \langle x,y,z\rangle$.
If the number y is odd and the condition $x>0 \wedge y>3^{x-1}$ is satisfied, then the triple $\langle x,y,z\rangle$ represents the number n if and only if the triple $\langle x-1, y-3^{x-1}, z\rangle$ represents the number $3n+1$.
Moreover, $\langle x-1, y-3^{x-1}, z\rangle \prec \langle x,y,z\rangle$.

Note

Lemma 4.1. *If the program K: $\{$**if** odd **then** $mult3$ **else** $div2$ **fi**$\}$ maps triple $\langle x, y, z\rangle$ into triple $\langle u, v, t\rangle$, then the starting triple $\langle x, y, z\rangle$ is bigger $\succ$ than triple $\langle u, v, t\rangle$.*

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

Proof:
If the number y is even, then we decrement the value of z and we divide y by 2. In the opposite case we decrement the value of x and we subtract 3^x from y. □

In each step of execution of Collatz algorithm , the expression $x + z$ is decremented by 1 , the value of y is also decremented accordingly.

4.2. Program IC

Consider the following program. We assume that the initial values of variables x, y, z satisfy the condition $n \cdot 3^x + y = 2^z \wedge \neg Err$.
It is the *precondition* of computations.

```
IC : {
  while $3^x + y \neq 2^z$ $(*\ tj.\ n \neq 1\ *)$ do
      if $\big(odd(y) \wedge ((x = 0)$ or $(y < 3^{x-1}))\big)$ then $Err := true$; exit fi;
      if $odd(y)$ then $x := x - 1;\ y := y - 3^x;\ (*n := 3 * n + 1\ *)$
      else $z := z - 1;\ y := y \,\mathrm{div}\, 2;\ (*\, n := \frac{n}{2}\ *)$ fi
  od
}
```

Any computation in the structure of triples is finite. It is a consequence of the following theorem of algorithmic theory of natural numbers $\mathcal{ATN}$.

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

However, it may happen that the execution of algorithm IC terminate with an error $Err = true$.

4.3. Properties of program IC

Fact 4.4. Each execution of the program IC is finite and either it is successful (reaches the number one) or it encounters an error Err.

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

Remark 4.1. If the execution of the algorithm IC starts with triple $\langle x, y, z\rangle$ and ends after k iterations with error $Err = true$, then another execution of algorithm IC that starts with the triple $\langle x + 1, 3y + 2^z, z + 2\rangle$ is longer by at least two iterations.
Both triples represent he same number n.

4.4. Program IIC

The remarks made earlier inspire us to write the following program that for a given number n searches a useful triple.

```
      read(n);
      Let $z = (\mu r)(2^r \geq n)$.
      $x, x_s := 0;\ z_s := z;\ y, y_s := 2^z - n$;
      Err := false;
      while $3^{x_s} + y_s \neq 2^{z_s}$ do
          while $3^x + y \neq 2^z$ do
            if $odd(y) \wedge (x = 0 \vee y < 3^{x-1})$
              then Err:=true; exit fi;
IIC:  IC:   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
```

One can easily remark that the program IIC omits triples that lead to an error Err.

Remark 4.2. For a given number n the program IIC has an infinite execution if and only if the execution of program Cl is infinite.

If the execution of program Cl for a number n is finite, then the program IIC finds a triple $\langle x, y, z \rangle$ such that the execution of program IC for this triple is free of error Err.

4.5. Two lemmas

Lemma 4.2. *Let K denote the instruction* {if $odd(n)$ then $n := 3 * n + 1$ else $n := n\,div\,2$ fi}. *If a given triple $\langle x, y, z \rangle$ satisfies the condition $\zeta : \big((y \mod 2 = 1) \Rightarrow (x > 0 \wedge y > 3^{x-1})\big)$, and reprents the number n , then the execution of program $\bar{K}$ results in a triple that represents the number m such that m is the result of one iteration stepof Collatz algorithm.*

$$\zeta \Rightarrow \left| \begin{tikzcd} n \arrow[r, "K:\{\textbf{if }odd(n)\textbf{ then }m:=3n+1\textbf{ else }m:=n/2\textbf{ fi}\}_{\mathfrak{N}}"] \arrow[d, "n\cdot 3^x+y=2^z"] & m \\ \langle x, y, z \rangle \arrow[r, "\bar{K}:\{\textbf{if }odd(y)\textbf{ then }u,v,t:=x-1,y-3^{x-1},z\textbf{ else }u,v,t:=x,y/2,z-1\textbf{ fi}\}_{\mathfrak{T}}"] & \langle u, v, t \rangle \arrow[u, "m\cdot 3^u+v=2^t"'] \end{tikzcd} \right|$$

The above diagram is commutative if the operations of decrementing x and subtracting 3^x from y are defined, i.e. if the value of y is odd and $(x > 0 \wedge y > 3^x)$.
he semantical property described by the diagram can be expressed by the following formula.

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

The programs K and $\bar{K}$ are independent, hence they commute.

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

This remark applies as well to any iteration of program K.

Lemma 4.3. If the following condition θ is satisfied

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

Then the following diagram commutes

$$\begin{tikzcd} n \arrow[rr, "Cl: \{\textbf{while } n\neq 1 \textbf{ do if } odd(n) \textbf{ then } n:=3n+1 \textbf{ else } n:=n/2 \textbf{ fi od}\}_{\mathfrak{N}}"] & & 1 \\ \langle x,y,z\rangle \arrow[u, "n\cdot 3^x+y=2^z"'] \arrow[rr, "IC: \left\{\begin{array}{l}\textbf{while } 3^x+y\neq 2^z \textbf{ do} \\ \quad \textbf{if } odd(y) \textbf{ then } x,y,z := x-1, y-3^{x-1}, z \\ \quad \textbf{else } x,y,z := x, y/2, z-1 \textbf{ fi} \\ \textbf{od}\end{array}\right\}_{\mathfrak{I}}"'] & & \langle x,y,z\rangle \arrow[u, "3^x+y=2^z"] \end{tikzcd} \quad \textbf{(CCI)}$$

This property is expresed by the formula

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

Proof:
We have to consider four cases.

Case a) If for a given number n there exist three numbers x, y, z such that 1° the equality $n \cdot 3^x + y = 2^z$ holds and 2° the execution of algorithm IC is finite, then the execution of Collatz algorithm Cl is finite.
Case b) If for a given number n there exist three numbers x, y, z such that 1° the equality $n \cdot 3^x + y = 2^z$ holds and 2° the execution of algorithm IC is infinite (and free of error), then the execution of Collatz algorithm Cl is infinite.
Case c) If for a given number n the execution of Collatz algorithm Cl is finite, then there exist

three numbers x, y, z such that 1° the equality $n \cdot 3^x + y = 2^z$ holds and 2° the execution of algorithm $IC$is finite.
Case d) If for a given number n the execution of Collatz algorithm Cl is infinite, then there exist three numbers x, y, z such that 1° the equality $n \cdot 3^x + y = 2^z$ holds and 2° , the execution of algorithm $IC$is infinite.

The proof of case c) goes by induction with respect to the number of iterations.
Base of induction reduces to the commutativity of the following diagram

$$\zeta \Rightarrow \left[\begin{tikzcd} n \arrow[r, "\{K\}_{\mathfrak{N}}"] & m \\ \langle x,y,z\rangle \arrow[u, "n\cdot 3^x+y=2^z"] \arrow[r, "\bar{K}_{\mathfrak{T}}"] & \langle u,v,t\rangle \arrow[u, "m\cdot 3^u+v=2^t"'] \end{tikzcd}\right].$$

If the execution of algorithm is longer, say of length 2, then the following diagram applies

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

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

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

Now, if for some $i \in N$ the number $n_i = 1$, then $\forall_k n_{i+k} = 1$.
Thus, in the case c) the formula **RCI** has a proof.
The proof of the cases a) and b) are easy. We leave them as an exercise.

It remains to prove the case d).
As in case c) we prove that for every natural number i the diagram commutes

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

and for every natural number i, the inequality $n_i \neq 1$ holds.
From this fact, we deduce that, for every natural number i the formula of the following form, is a theorem of algorithmic theory of natural numbers $\mathcal{ATN}$.

$$\underbrace{c_n \cdot 3^{c_x} + c_y = 2^{c_z}}_{\Upsilon} \wedge \left\{\begin{array}{l} x := c_x; \\ y := c_y; \\ z := c_z \end{array}\right\} \bigwedge_{j=0}^{i}\{\bar{K}^j\}\zeta \wedge \{n := c_n\} \bigwedge_{j=0}^{i}\{K^j\}(n \neq 1) \implies \left\{\begin{array}{l} x := c_x; \\ y := c_y; \\ z := c_z \end{array}\right\} \bigwedge_{j=0}^{i}\{\bar{K}^j\}(3^x + y \neq 2^z) \tag{9}$$

Now, we apply the axiom Ax_{17} of program calculus, see [MS87] , page 57 [6] and obtain that for every natural number i the formula of the following form is a theorem of $\mathcal{ATN}$ theory.

$$\Upsilon \wedge \left\{ \begin{array}{c} x := c_x; \\ y := c_y; \\ z := c_z \end{array} \right\} \bigcap \{\bar{K}\}\zeta \wedge \{n := c_n\} \bigwedge_{j=0}^{i} \{K^j\}(n \neq 1) \implies \left\{ \begin{array}{c} x := c_x; \\ y := c_y; \\ z := c_z \end{array} \right\} \bigwedge_{j=0}^{i} \{\bar{K}^j\}(3^x + y \neq 2^z) \tag{10}$$

By another application of axiom Ax_{17} we obtain that for every natural number i the formula of the following form is a theorem of $\mathcal{ATN}$ theory.

$$\Upsilon \wedge \left\{ \begin{array}{c} x := c_x; \\ y := c_y; \\ z := c_z \end{array} \right\} \bigcap \{\bar{K}\}\zeta \wedge \{n := c_n\} \bigcap \{K\}(n \neq 1) \implies \left\{ \begin{array}{c} x := c_x; \\ y := c_y; \\ z := c_z \end{array} \right\} \bigwedge_{j=0}^{i} \{\bar{K}^j\}(3^x + y \neq 2^z) \tag{11}$$

The latter formulas $(i = 0, 1, \dots)$ are equivalent to

$$\Upsilon \wedge \left\{ \begin{array}{c} x := c_x; \\ y := c_y; \\ z := c_z \end{array} \right\} \bigcap \{\bar{K}\}\zeta \wedge \neg\{n := c_n\}\{Cl\}(n = 1) \implies \left\{ \begin{array}{c} x := c_x; \\ y := c_y; \\ z := c_z \end{array} \right\} \bigwedge_{j=0}^{i} \{\bar{K}^j\}(3^x + y \neq 2^z) \tag{12}$$

We are able now to apply the inference rule $r5$ [7] and obtain

$$\underbrace{\Upsilon \wedge \left\{ \begin{array}{c} x := c_x; \\ y := c_y; \\ z := c_z \end{array} \right\} \bigcap \{\bar{K}\}\zeta \wedge}_{\theta} \neg\{n := c_n\}\{Cl\}(n = 1) \implies \left\{ \begin{array}{c} x := c_x; \\ y := c_y; \\ z := c_z \end{array} \right\} \bigcap \{\bar{K}\}(3^x + y \neq 2^z) \tag{13}$$

One more step, we introduce the program IC in the consequent of the implication

$$\theta \wedge \neg\{n := c_n\}\{Cl\}(n = 1) \implies \neg \left\{ \begin{array}{c} x := c_x; \\ y := c_y; \\ z := c_z \end{array} \right\} \{IC\}3^x + y = 2^z) \tag{14}$$

which completes the proof of the formula **RCI**. □

4.6. T$_n$ sets of triples

For every number n we define the set T_n of triples x, y, z such, that he equality $n3^x + y = 2^z$.
The set T_n is a non-empty set.
Sets T$_n$ are pairwise disjoint, i.e. $n \neq m \implies \mathrm{T}_n \cap \mathrm{T}_m = \emptyset$.

[6] Ax_{17}. $(\bigcap\{M\}\alpha \equiv (\alpha \wedge \{M\}\bigcap\{M\}\alpha))$

[7] rule $r5$: $\dfrac{\{\beta \implies KM^i\alpha\}_{i \in N}}{\beta \implies K\bigcap M\alpha}$

Each set T_n T_n is closed with respect to the following operations $o_1, o_2, o_3, o_4, o_5, o_6$

$$\begin{aligned}
o_1(\langle x,y,z\rangle) &= \langle x+1, 3y+2^z, z+2\rangle && (15)\\
o_2(\langle x,y,z\rangle) &= \langle x+1, 3y-2^z, z+1\rangle \quad \text{when } 3y>2^z && (16)\\
o_3(\langle x,y,z\rangle) &= \langle x, y+2^z, z+1\rangle && (17)\\
o_4(\langle x,y,z\rangle) &= \langle x, y-2^{z-1}, z-1\rangle \qquad \text{when } y>2^{z-1} && (18)\\
o_5(\langle x,y,z\rangle) &= \langle x-1, \frac{y-2^{z-2}}{3}, z-2\rangle \ \text{ when } \ (y-2^{z-2})\mathrm{mod}3=0 && (19)\\
o_6(\langle x,y,z\rangle) &= \langle x-1, \frac{y+2^{z-1}}{3}, z-1\rangle && (20)
\end{aligned}$$

As a consequence the set T_n is infinite set.

Note the following dependencies
$o_5(o_1(\langle x,y,z\rangle) = \langle x,y,z\rangle$
$o_4(o_3(\langle x,y,z\rangle) = \langle x,y,z\rangle$
$o_6(o_2(\langle x,y,z\rangle) = \langle x,y,z\rangle$
$o_4(o_1(\langle x,y,z\rangle) = \langle x,y,z\rangle$

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 **??**. Hence, this sentence holds true in any model of the theory.

(*ii*) In Appendix C, page 26, 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 21 that expresses the same looping property of computation of Collatz algorithm does not belong to the language of first-order theory Ar.

$$\{n \leftarrow \varepsilon\} \bigcap \left\{ \begin{array}{l} \textbf{if } n \neq 1 \textbf{ then} \\ \quad \textbf{if } n \bmod 2 = 0 \\ \quad \textbf{then } n \leftarrow n \textbf{ div } 2 \\ \quad \textbf{else } n \leftarrow 3n+1 \\ \quad \textbf{fi} \\ \textbf{fi} \end{array} \right\} (n \neq 1) \tag{21}$$

However, for every algebraic structure $\mathfrak{A}$ the above formula (21) is valid in $\mathfrak{M}$ if and only if every formula of the following scheme is valid in $\mathfrak{A}$

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

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.

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

Continuing, we obtain the following formulas

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

Horizontal lines separate formulas corresponding to different levels of Collatz tree.

We add four constants $\varepsilon, c_x, c_y, c_z$ to the alphabet of the language. We modify the definitions of sets of terms and formulas in a corresponding way.
The set Ax' of axioms of the new theory is a union of set Ax of axioms of Presburger theory (cf. section 7, page 23), an infinite set of sentences Z, an infinite set of sentences Y, the set Df of auxiliary definitions, the sentence $U : \varepsilon \cdot 3^{c_x} + c_y = 2^{c_z}$.

$$Ax' = Ax \cup Z \cup Y \cup Df \cup U$$

We define the set Z as containing all formulas $\varepsilon \neq \mathbf{k}$ such,,that the expression $n = \mathbf{k}$ occurs in halting formula of Collatz algorithm, $\mathbf{k}$ is number. Hence, the set Z contains the sentences $\{\varepsilon \neq 1, \varepsilon \neq 2, \varepsilon \neq 4, \varepsilon \neq 8, \varepsilon \neq 16, \varepsilon \neq 32, \varepsilon \neq 5, \varepsilon \neq 64, \varepsilon \neq 10, \varepsilon \neq 128, \varepsilon \neq 20, \varepsilon \neq 21, \varepsilon \neq 3, \ldots\}$

The set Y contains all sentences of first-order language that are equivalent to thealgorithmic sentences

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

where $i = 0, 1, 2, \ldots$.
The sentences of the set Y can be arranged into the sequence $Y = \{y_0, y_1, y_2, \ldots\}$.
The sentence y_i expresses the following property: if in a computation on triples performed i steps and the current state of memory is $\langle x, y, z\rangle$, and if $3^x + y \neq 2^z$ then the following condition holds $odd(y) \implies x > 0 \wedge y > 3^x$. In other words the sentence y_i excludes the risk of error in the $i+1$-th iteration of execution of program IC.
Let us see two examples

$$y_0 : \neg odd(c_y) \vee (c_x > 0 \wedge c_y > 3^{c_x - 1})$$

$$y_1 : \left\{\begin{array}{l} x := c_x; \\ y := c_y; \\ z := c_z \end{array}\right\} \left\{\begin{array}{l} \textbf{if } 3^x + y \neq 2^z \textbf{ then} \\ \quad \textbf{if } odd(y) \textbf{ then} \\ \quad\; x := x - 1;\ y := y - 3^x \\ \quad \textbf{else} \\ \quad\; y := y \div 2;\ z := z - 1 \\ \quad \textbf{fi} \\ \textbf{fi} \end{array}\right\} (odd(y) \implies (x > 0 \wedge y > 3^{x-1}))$$

The set Df contains definitions of auxiliary fuctions $P2$, $P3$, and paririty predicate ,
The set $U = \{P3(\varepsilon, c_x) + c_y = P2(c_z)\}$ contains just one entence.

We shall show that that the set Ax' is consistent. We begin by showing that every finite subset $Ax_0 \subset Ax'$ is consistent. Namely, we shall demonstrate that the set Ax_0 has a model in standard structure $\mathfrak{N}$ of natural numbers with addition. Our task reduces to finding proper values of of four constants $\varepsilon, c_x, c_y, c_z$.
The set $Ax \cup Df \cup U$ is consistent, cf. section 4 and section 7, page 22 and section **??** , page **??**, for the standard structure of natural numbers with addition is a model for it.
Let Z_0 be an arbitrary finite subset of the set Z, similarly, let Y_0 Y.
The set $Ax \cup Df \cup U \cup Z_0 \cup Y_0$ is consistent. Let l be a number greater than any number k that occurs in the set Z_0. This choice assures the validity of every sentence of the set Z_0. From the infinite set T_l of triples that represent the element l, we should choose a triple such that every sentence of the set Y_0 is true. Look at the properties of the algorithm IIC. Let j be the biggest index of the sentence y_i that belongs to the set Y_0. The sentence y_j says: if after execution of j iterations of algorithm IC the current state of memory is not represnting

the number 1, then the current values of variables x and y will b ssatisfying the condition $odd(y) \implies (x > 0 \wedge y > 3^x)$. It suffices that the initial value of variable $x = c_x$ be greater than the number j, we can put $c_x = j + 1 \& c_z = j + 2 \& c_y = 2^{j+2} - l \cdot 3^{j+1}$. In this way we assure the validity ofthe sentence U for $l \cdot 3^{c_x} + c_y = 2^{c_z}$ as well as the validitity of every sentence from the set Y_0.
We proved that every finite subset of the set Ax' is consistent.

By the compactness theorem on first-order logic[8], we obtain

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

Now, we apply the model existence theorem[9], 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.

The facts gathered till now, allow to apply the lemma **RCI**.
Hence,

Fact 5.1. The execution of the algorithm IC that begins with the values c_x, c_y, c_z is infinite.

5.2. Infinite Collatz computation require unreachable elements

In our proof we shall imitate the proof of lemma 5.1 . Let ε be an arbitrary element such, that the execution of Collatz algorithm isinfinite. (i.e. element ε is non-Collatz one). It is evident that the execution of algorithm IIC (c.f. 4.4) for element ε is infinite, too. For any triple $\langle x, y, z\rangle$ of standard, natural numbers the execution of program IC terminates with the error Err.
Yet, the execution of Collatz algorithm for the element ε is infinte. Our nearest goal is to prove that there exist three elements c_x, c_y, c_z such that 1° the equality $\varepsilon \cdot 3^{c_x} + c_y = 2^{c_z}$ holds and 2° the computation of algorithm IC is infinite.

We extend the language of the elementary theory $\mathcal{T}^+$ adding to it four constants: c_n, c_x, c_y, c_z.

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

We fix the value of constant c_n assuming that the equality $c_n = \varepsilon$ is an axiom of new theory. We aregoing to show that the following set of first-order sentences $Ax' = Ax \cup Df \cup U \cup Y$ is consistent. Sets Ax and Z were described earlier. Set U contains one formula $c_n \cdot 3^{c_x} + c_y = 2^{c_z}$. Set Df contains definitions of useful operations and relations $P2, P3, 3x, div2, even, odd$, c.f. section 7.
Set Y contains all first-order sentences quivalent to the algorithmic sentences of the form

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

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

In other words Y is a sequence $Y = \{y_0, y_1, y_2, \ldots\}$.

Sentence y_i expresses the following semantical property: *Let s be an execution of algorithm IC that starts with triple $\langle c_x, c_y, c_z\rangle$. If during this execution, after i steps and reached the state x, y, z then if the current state does not represents the number 1, then in the next step no errror Err will be raised.*
In other words the sentence y_i excludes the risk of error Err in $i+1$-th step of execution.

As we did it earlier, we show that every finite subset $Ax_0 \subset Ax'$ is consistent. We shall use observations made in section 4.4, that concern the algorithm IIC. Let Y_0 be any finite subset of the set Y of sentences. Let i be the largest number of iterations mentioned in the set . From the properties of algorithm IICwe know that there exists triple $\langle x, y, z\rangle$ of natural numbers such, that during exection of i iterations of program $\bar{K}$ no error Err will occur.
Hence, we can apply the compactness theorem and assert that the whole set Ax' is consistent.
By completeness theorem we infer that there is a triple $\langle x, y, z\rangle$ such ,that for every natural number i after execution of i iterations of the while intruction in algorithm IC the next iteration can be executed without risk of Err error. I.e. the condition θ mentioned in the lemma 4.3 is satisfied. Let us cite it here

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

Hence, two formulas hold: the formula θ and $\neg\{n := \varepsilon; Cl\}(n = 1)$. From this conjunction we deduce

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:
This is an immediate consequence of the lemma 4.3. $\square$

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.
- 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)
For every natural number n, the execution of Collatz algorithm is finite.

Proof:
It follows from the facts listed 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 $\mathfrak{A}$ is a non-standard model of the theory. Hence, by transposition, we obtain Collatz theorem. $\square$

6. Final remarks

It is not difficult to notice that the presented proof uses metamathematical methods. We are not able, for now, to present a proof derived directly in the algorithmic theory of $\mathcal{ATN}$ from the axioms of this theory or, for example, from Archimedes' law, which is a theorem of $\mathcal{ATN}$.
The next task is to estimate the cost of the Collatz algorithm, we know that the computations are finished, but we cannot estimate their length.

Acknowledgments

Andrzej Szałas found an error in another, earlier version of our work on Colllatz conjecture. Wiktor Dańko called our attention to a lacune in the analysis of computations on triples. Ludwik Czaja and Marek Warpechowski conveyed their doubts , we are owing them very ymuch.
The present authors are responsible for any errors and deficiencies that you may find.

7. Appendix A – Some facts on elementary theory of natural numbers with addition

In section 4 we observed a few of useful facts on triples of natural numbers that represent another number.
We shall consider the following theory $\mathcal{T}^+$, cf. [Grz71] p. 239 and following ones.

Definition 7.1. Theory $\mathcal{T}^+ = \langle \mathcal{L}, \mathcal{C}, Ax \rangle$ is the system of three elements:

$\mathcal{L}$ is a language of first-order. The alphabet of this language consist of: the set V of variables, symbols of operations: $0, S, +$, symbol of equality relation $=$, symbols of logical functors and quantifiers, auxiliary symbols as brackets ...
The set of well formed expressions is the union of te set T of terms and the set of formulas F.
The set T is the least set of expressions that contains the set V and constants 0 and 1 and closed with respect to the rules: if two expressions τ_1 and τ_2 are terms, then the expression $(\tau_1 + \tau_2)$.
The set F of formulas is the least set of expressions that contains the equalities (i.e. the expressions of the form $(\tau_1 = \tau_2)$) and closed with respect tothe following formation rules: if expressions α and β are formulas, then the aexpression of the form

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

are alsoformulas, moreover, the expressions of the form

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

where x is a variable and α is a formula, are formulas too.

$\mathcal{C}$ is the operation of consquence determined by axioms of first-order logic and the inference rules of the logic,

Ax is the set of formulas listed below.

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

The expression $\Phi(x)$ should be replaced by any formula. This is the induction scheme.
We augment the set of axioms adding the axioms that define a coiple of useful notions.

$$even(x) \overset{df}{\equiv} \exists_y\, x = y+y \tag{e}$$
$$odd(x) \overset{df}{\equiv} \exists_y\, x = y+y+1 \tag{o}$$
$$x\, div\, 2 = y \equiv (x = y+y \,\vee\, x = y+y+1) \tag{D2}$$
$$3x \overset{df}{=} x+x+x \tag{3x}$$

Proofs of some facts known from section 4 will be done in the framework of the theory $\mathcal{T}^+$. In this way we make sure that these facts are true in any model of the theory.
In the section 5 we shall use Presburger theory Ar.

Definition 7.2. Theory $Ar = \langle \mathcal{L}, \mathcal{C}, Ax\rangle$ is a system of three elements :

$\mathcal{L}$ is a language of first-order. The alphabet of this language contains the set V of variables, symbols of functors : $0, +$, symbol of equality predicate $=$.
The set of well formed-expressions is the union of set of terms 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.

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

. . . .

$$\forall_x\, \exists_y \begin{pmatrix} x = \underbrace{y+y+\cdots+y}_{k} \vee \\ x = \underbrace{y+y+\cdots+y}_{k} + 1 \vee \\ x = \underbrace{y+y+\cdots+y}_{k} + \underbrace{1+1}_{2} \vee \\ \dots \\ x = \underbrace{y+y+\cdots+y}_{k} + \underbrace{1+1+\cdots+1}_{k-2} \vee \\ x = \underbrace{y+y+\cdots+y}_{k} + \underbrace{1+1+\cdots+1}_{k-1} \end{pmatrix} \tag{Hk}$$

. . .

Let us recall some facts
F1. Theory $\mathcal{T}^+$ is elementarily equivalent to the theory Ar.[Pre29, Sta84]
F2. Theory Ar is decidable. [Pre29].
F3. The computational complexity of theory Ar, is double exponential $O(2^{2^n})$ this result belongs to Fisher and Rabin, see [FR79].
F4. Theories $\mathcal{T}^+$ and Ar have non-standard model, see section 9, p. 26.

8. Appendix B– proofs of useful facts

In this section we shall show that the sentence $\forall_n \exists_{x,y,z}\, n \cdot 3^x + y = 2^z$ is a theorem of the theory $\mathcal{T}^+$ of addition. The elementary theory $\mathcal{T}^+$ of natural numbers with addition was recalled in the preceding section.
Operations of multiplication and power are inaccessible in the theory $\mathcal{T}^+$. However, we do not need them.
We enrich the theory $\mathcal{T}^+$ adding two functions $P2(\cdot)$ and $P3(\cdot.\cdot)$. defined in this way

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

Lemma 8.1. The definitions given above are correct, i.e. the the following sentences aretheorems of the theory with two definitions

$$\mathcal{T}^+ \vdash \forall_x \exists_y\, P2(x) = y \text{ and}$$
$$\mathcal{T}^+ \vdash \forall_{x,y,z} P2(x) = y \wedge P2(x) = z \implies y = z.$$

Similarly, the sentences
$\forall_{y,x} \exists_z\, P3(y,x) = z$ and
$\forall_{y,x,z,u} P3(y,x) = z \wedge P3(y,x) = u \implies z = u$
are theorems of theory $\mathcal{T}^+$.

Proof goes by induction with respect to the value of variable x.

We shall use the following definition of the order relation $a < b \overset{df}{=} \exists_c\, a + S(c) = b$.
Making use of the definition of function $P2$ and $P3$ we shall write the formula $P3(n,x) + y = P2(z)$ as it exppresses the same content as expression $n \cdot 3^x + y = 2^z$.

Lemma 8.2. The following sentence is a theorem of of the theory $\mathcal{T}^+$ enriched by the definitions.
$\forall_n \exists_{x,y,z} P3(n,x) + y = P2(z)$

We begin proving by induction that $\mathcal{T}^+ \vdash \forall_n\, n < 2^n$. Namely, $\mathcal{T}^+ \vdash \forall_n\, n < P2(n)$. It is easy to see that $\mathcal{T}^+ \vdash 0 < P2(0)$. Assume that $\mathcal{T}^+ \vdash \forall_n \{n < P2(n)\}$. Inequality $n+1 < P2(n+1)$ follows from the two given below $\mathcal{T}^+ \vdash n < P2(n)$ and $\mathcal{T} \vdash 1 < P2(n)$.

In the similar manner, we obtain $\mathcal{T}^+ \vdash P3(n,x) < P2(z) \wedge (z = n + x + x)$
As a consequence we have $\mathcal{T}^+ \vdash \forall_n \exists_{x,y,z}\, P3(n,x) + y = P2(z)$.
Note, we have proved a somewhat stronger sentence $\forall_{n,x} \exists_{y,z} P3(n,x) + y = P2(z)$.

Lemma 8.3. 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
$\{q := 0;\ while\ q \neq x\ do\ q := q + 1\ od\}(x = q)$,
$\{q := 0;\ while\ q \neq y\ do\ q := q + 1\ od\}(y = q)$,
$\{q := 0;\ while\ q \neq z\ do\ q := q + 1\ od\}(z = q)$
i ponadto zachodzi równość
$P3(n, x) + y = P2(z)$ to łatwo sprawdzic,że spełniona jest formuła $\{t := 0;\ while\ n \neq t\ do\ t := t + 1\ od\}(t = n)$ □

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

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

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

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

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-2021].

[Pre29] Mojżesz Presburger. Über die Vollständigkeit eines gewissen Systems der Arithmetik ganzer Zahlen , in welchem die Addition als einzige Operation hervortritt . 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. `http://cs.fit.edu/~ryan/papers/presburger.pdf`. [Online; accessed 7-August-2021]. Technical Report TR84-639, Cornell University, 1984.