Collatz conjecture becomes theorem

Grażyna Mirkowska
UKSW University, Warsaw
Institute of Informatics
G.Mirkowska23 [a] gmail.com

Andrzej Salwicki
Dombrova Research
Partyzantów 19, 05-092 Łomianki, POLAND
A.Salwicki [a] UKSW·edu.pl

Key words: algorithm of Collatz, halting property, Collatz tree, calculus of programs,algorithmic theory of numbers.

ACM-class: F.3.1; D.2.4
MSC-class: 03D02 (Primary) 68Q02 (Secondary)

Abstract. **I**. We are showing that the following phrase of ethnic language "*For every n, if n is a natural number then $3n+1$ computation (aka Colatz computation) for n is finite.*" is a semantically valid statement. To do so we need a sufficient and necessary criterion of halting of $3n+1$ computation, see lemma 3.6. Next, the Main lemma 5.1 asserts that every instance of the criterion where the variable n is replaced by a natural number $\underline{r}$ is a statement *valid* in the standard data structure $\mathfrak{N}$ of natural numbers. A corollary of the lemma says: every instance of the cnjecture where the variable n is replaced by any natural number $\underline{c} \neq 0$, is a *theorem* of elementary arithmetic, in which the addition is the only operation.

II, Paradoxically, the Collatz conjecture itself is not a theorem of number theory (Peano's arithmetic), nor any mathematical theory that uses the first-order language and the classical predicate logic. It is so because, 1°every formalized, first-order theory has models that are non-isomorphic to the standard model of natural numbers, and °the infinite computations can be n encountered in a non-standard computable model of the elementary theory of natural numbers with addition.

To avoid the paradox, we will conduct our considerations in the formalized algorithmic theory $\mathcal{ATN}$ of natural numbers. The logical consequence operation of the theory is determined by the calculus of programs $\mathcal{AL}$, which is an extension of the predicate calculus. The halting condition of the Collatz computations is written as an algorithmic formula (Main Thm).

III. We are proving that, four infinite sets St_0, St_1, St_2, St_3 of formulas, are the *recursive sets* of *theorems* of the theory $\mathcal{ATN}$. We conclude our proof of the Main theorem, c.f. page 22, making use of the inference rule R_3(on page 34) to the infinite set St_3 of premises. Note, every premise has a proof.

$$\mathcal{ATN} \vdash \forall_{n\neq 0} \underbrace{\left\{\begin{array}{l} q := 1; \\ \textbf{while } n \neq q \textbf{ do} \\ \quad q := q + 1 \\ \textbf{od} \end{array}\right\}(n = q)}_{\text{FORALL } n, \text{ IF } n \text{ is a natural number}} \Longrightarrow \underbrace{\left\{\begin{array}{l} m := n \div 2^{\kappa(n)}; \\ \textbf{while } m \neq 1 \textbf{ do} \\ \quad m := 3 \cdot m + 1; \\ \quad m := m \div 2^{\kappa(m)} \\ \textbf{od} \end{array}\right\}(m = 1)}_{\text{THEN } \textit{the computation for } n \textit{ is finite} \text{ FI}} \qquad \text{(Main thm)}$$

The antecedent of this implication is the axiom A_1) of the $\mathcal{ATN}$ theory, c.f. page 35. Hence, we can cut it off.

1. Introduction

The $3n+1$ problem remained open for over 80 years. It has been formulated in 1937 by Lothar Collatz, c.f. [Lag10]. The problem became quite popular due to its wording, for it is short and easy to comprehend.
Collatz remarked that for any given natural number $n > 0$, the sequence $\{n_i\}$ defined by the following recurrence rec1

$$\left.\begin{array}{l} m_0 = n \\ m_{i+1} = \begin{cases} m_i \div 2 & \text{when } m_i \text{ is even} \\ 3 \cdot m_i + 1 & \text{when } m_i \text{ is odd} \end{cases} \quad \text{for } i \geq 0 \end{array}\right\} \tag{rec1}$$

seems always reach the value 1.

He formulated the following conjecture

$$\text{for all } n \text{ exists } i \text{ such that } m_i = 1 \tag{Collatz conjecture}$$

Note, the recurrence rec1 defines an infinite sequence of natural numbers. If an element $m_i = 1$ then all subsequent odd numbers are also equal to 1. One may say: the nature of $3x + 1$ problem concerns the stabilizationof the subsequence that contains all odd numbers of the sequence $\{m_i\}$.

The number of papers devoted to the problem is very high, c.f. [Lag10] . In 2024 alone, 230 preprints dealing with the 3n+1 problem were announced. It is worthwhile to consult social media: Wikipedia, youtube etc, there you can find technical analysis of the problem as well as some surprising ideas how to prove the Collatz conjecture.
Computers are used and are still crunching numbers in the search of an eventual counterexample to the Collatz conjecture. The reports on progress appear each year.
We claim that the counterexample approach is pointless, i.e. the computers can be turned off. Namely, we shall prove that any program that searches a counterexample will never reach its goal.

Our goal will be achieved if we prove that for each natural number n the computation of the following algorithm Cl in the algebraic domain $\mathfrak{N}$ of natural numbers is finite.

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

In this way we replaced the problem of stabilization of infinite sequences by the problem of termination of computations.

A bit of history

Below, we recall some important theorems, a couple of not too difficult remarks and a paradox.

- The formulation of the Collatz problem contains the phrase "for every n in the set N of natural numbers". Note, that the data structure of natural numbers can not be axiomatized by any set of first-order formulas. It is a corollary of Gödel's incompleteness theorem. Consquently, no proof of Collatz conjecture exists in any first-order theory.

- Note, that many important semantical properties: "*the Archimedean property*", "*the halting property of program*", et al., can not be expressed as first-order formulas [Kar64, TMR65].

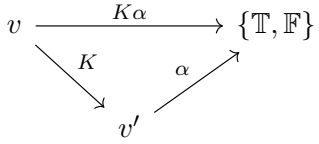
- Erwin Engeler [Eng93] in 1967 remarked that the halting property of program can be expressed by an infinite disjunction of quantifier free formulas.
  Next, he proved that algorithmic properties of programs executed in the structure of ordered real numbers are provable from the axioms of ordered, fields plus the Archimedean property (understood as an infinite disjunction).

- Programs are finite texts. The infinite disjunctions considered by Engeler have regularities.

- Motivated by this observation A. Salwicki (1969) introduced algorithmic formulas and the logical calculus (of programs) $\mathcal{AL}$ with iteration quantifiers [MS87]. The $\mathcal{AL}$ calculus allows to specify algorithmic properties and to construct the proofs that verify the validity of properties.

- The following expression $\boxed{\{\mathbf{while}\ \gamma\ \mathbf{do}\ \ x := 3x+1\ \mathbf{od}\ \}\,\alpha(x)}$ (it consists of a program and a formula) is an example of simple algorithmic formula. Every pair consisting of a program and a formula is an algorithmic formula. The set of all formulas contains all classical formulas, all simple algorithmic formulas and is closed with respect to the boolean operations: disjunction, $\vee$ conjunction $\wedge$, negation $\neg$ and two pairs of infinite boolean operations: the classical quantifiers $\exists.\forall$ and two iteration quantifiers $\bigcap\{K\}\,\alpha$ to be read as *the formula α holds after any iteration of the program K*. An existential iteration quantifier $\bigcup\{K\}\,\alpha$ reds:*there is an iteration of program K such that the formula $\{K^i\}\,\alpha$ is satisfied.*

- The logical value of the formula $K\alpha$ at a given valuation v of variables is determined as follows:



run the program K at v to arrive at another valuation $v' = K(v)$, then evaluate the value of the formula α at the valuation v', i.e. $val(K\alpha, v) = val(\alpha, v')$. If the result valuation v' is not defined then $val(K\alpha, v) = \mathbb{F}$.

- We introduced two countable families of infinite operations.
The meaning of the formula $\bigcap\{K\}\,\alpha$ is the greatest lower bound g.l.b. of the meanings of the formulas of the set $\{\alpha, K\alpha, K^2\alpha, K^3\alpha, \dots\}$. The meaning of the formula $\bigcup\{K\}\,\alpha$ is the least upper bound l.u.b. of the meanings of the formulas $\{\alpha, K\alpha, K^2\alpha, K^3\alpha, \dots\}$.
Algorithmic formulas of the form $\boxed{\mathbf{while}\ \gamma\ \mathbf{do}\ K\ \mathbf{od}\alpha}$ also define infinite operations on logical values.

- The calculus of programs enjoys the *completeness* property.
The proof of the completeness theorem uses of the Rasiowa-Sikorski lemma, c.f. [MS87].

1.1. An illustration of some regularities.

Look at the table 1.

Table 1. CASE n=19 of (compact) computation

1-st column =m_i of $Collatz$ computation, next columns: x =no of multiplications, z=no divisions, y=code of path from n to current m_i. In next columns 6 and 7 we check invariant $n \cdot 3^x + y \stackrel{?}{=} m_i \cdot 2^z$.

Computation						
m	**x**	**y**	**z**	**k**	$\mathbf{n3^x + y}$	$m \cdot 2^z$
19	0	0	0	0	19	19
29	1	1	1	1	58	58
11	2	5	4	3	176	176
17	3	31	5	1	544	544
13	4	125	7	2	1664	1664
5	5	503	10	3	5120	5120
1	6	2533	14	4	16384	16384
1	7	23983	16	2	2^{16}	2^{16}
...		...	...		...	...

Analysis

For each i^{th}-row $0 \le i \le 6$

$k_{i+1} = \kappa(3m_i + 1)$	5^{th} column
$m_{i+1} = (3m_i + 1) \div 2^{k_{i+1}}$	1^{st} column
$x_i = i$ AND $z_i = \sum_{l=0}^{i} k_l$	colums 2 and 4
ANND $y_i = \sum_{j=0}^{x_i - 1} 3^{x_i - 1 - j} \cdot 2^{z_j}$	column3

invariants: $1^\circ\ \ n \cdot \prod_{j=0}^{i-1}(3 + \frac{1}{m_j}) = m_i \cdot 2^{z_i}$

$2^\circ\ \ n \cdot 3^{x_i} + y_i = m_i \cdot 2^{z_i}$

two halting conditions:

$1^\circ\ \exists_i m_i = 1$ OR $2^\circ\ \exists_{i \in N}\ n \cdot 3^{x_i} + y_i = 2^{z_i}$

An analysis of the algorithm Cl leads to lengthy, awkward formulas that are error prone.
However, it helped us in noticing, that every computation of the 3n+1 algorithm is accompanied by a computation on the triples $\langle x, y, z\rangle$. Here the value of x shows how many operations of multiplications were done. Similarly, the value of z is the number of division by 2 executed. We remark that the value of y is a code of path starting from n. Look at table 1.
The computation on triples always terminate. For some triples it ends with error. The problem reduces to the task of proving that for every number n there exists an appropppriate, error-free triple.

1.2. Properties of the structure $\mathfrak{N}$ of the natural numbers.

It is tacitly assumed that computations are performed in the standard structure of natural numbers $\mathfrak{N}$.

$$\mathfrak{N} = \langle N, 0, 1, s, +, *, =, odd\rangle$$

where N is the set $\{0, 1, s(1), s(s(1)), \dots s^p(1), \dots\}$.
The meaning of the functor $+$ is the standard operation of addition. The sign $=$ denotes the identity relation. The t predicate $odd(x) = \mathbb{T}$ iff x is odd number. Note, $odd(x) \stackrel{df}{=} \exists_z x = z + z + 1$
In fact, we do not need multiplication operation, for the term $3*x$ can be replacded by $x+x+x$. Similarly $2*x = x+x$ and $x \div 2 = z$ iff $\exists_z x = z + z \vee x = z + z + 1$.

A theory that completely specifies the structure $\mathfrak{N}$. We are going to formulate and prove a theorem of termination property of the Collatz algorithm.
Therefore we need to choose a theory that 1°) Specifies the structure $\mathcal{N}$. 2°) Allows to express the termination property of algorthm. 3°) Provides enough tools to prove such formula.

Two traditional candidate theories are the Peano's arithmetic or Presburger theory of addition (Remember our computations need not multiplication operation.).
However, we are using a third theory: the formalized, algorithmic theory of natural numbers $\mathcal{ATN}$, [MS87, MS21]. For it comes together with the following *categoricity property*.

Metatheorem 1. Every model of the $\mathcal{ATN}$ theory is isomorphic to the standard structure $\mathfrak{N}$.

Note, The operations of multiplication by 3 and division by 2 can be defined by simple programs with addition as the only operation. This observation will be used in our considerations.

1.3. Properties of computations

Every natural number n may be presented in the form $n = 2^i \cdot (2j + 1)$. C.f. the notion of the *pairing function*.
In the sequel we use two functions $\kappa, \rho\colon N \to N$ defined as foolows

Definition 1.1. $\kappa(2^i \cdot (2j + 1)) \stackrel{df}{=} i$ $\rho(2^i(2j + 1)) \stackrel{df}{=} 2j + 1$

(In many texts the function κ is written as $exp(n, 2)$.) Note that function κ can be defined by an appropriate formula of Presburger arithmetic.

We shall present and analyse computations in their *compact* form. The figure 1 illustrates the concept.

Definition 1.2. A *compact* form of Collatz computation for a given number n is a subsequence that contains odd numbers only. More precisely, it is the sequence of numbers $\{m_i\}_{i\in I}$ such that $m_0 = n \div 2^{\kappa(n)}$ and for $i > 0$ the element $m_{i+1} = (3m_i + 1) \div 2^{\kappa(3m_i+1)}$, or simply $m_{i+1} = \rho(3m_i + 1)$.

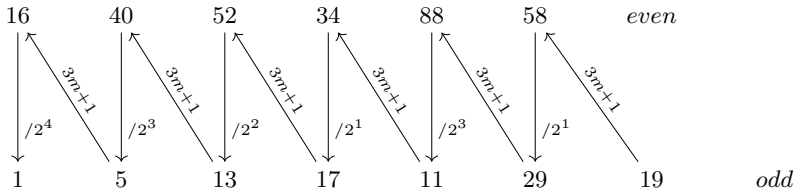


Figure 1. An example of compact computation

2. Collatz tree

A universal (generic) definition of tree reads as follows

Definition 2.1. A set D of finite sequences of natural numbers is a *tree* iff it satisfies the conditions (*i*) and(*ii*)

(*i*) if a sequence $s = i_1, i_2, \ldots, i_{l-1}, i_l \in D$ then its non-empty prefix $s' = i_1, i_2, \ldots, i_{l-1} \in D$ also belongs to D,

(*ii*) the empty sequence $\emptyset \in D$. It is the root of the tree.

According to this definition the set of finite computations of the program Cl written in the reverse order is a tree. We call it the Collatz tree. The figure 2 illustrates a fragment of Collatz tree.

Conjecture 1. The Collatz tree contains all natural numbers.

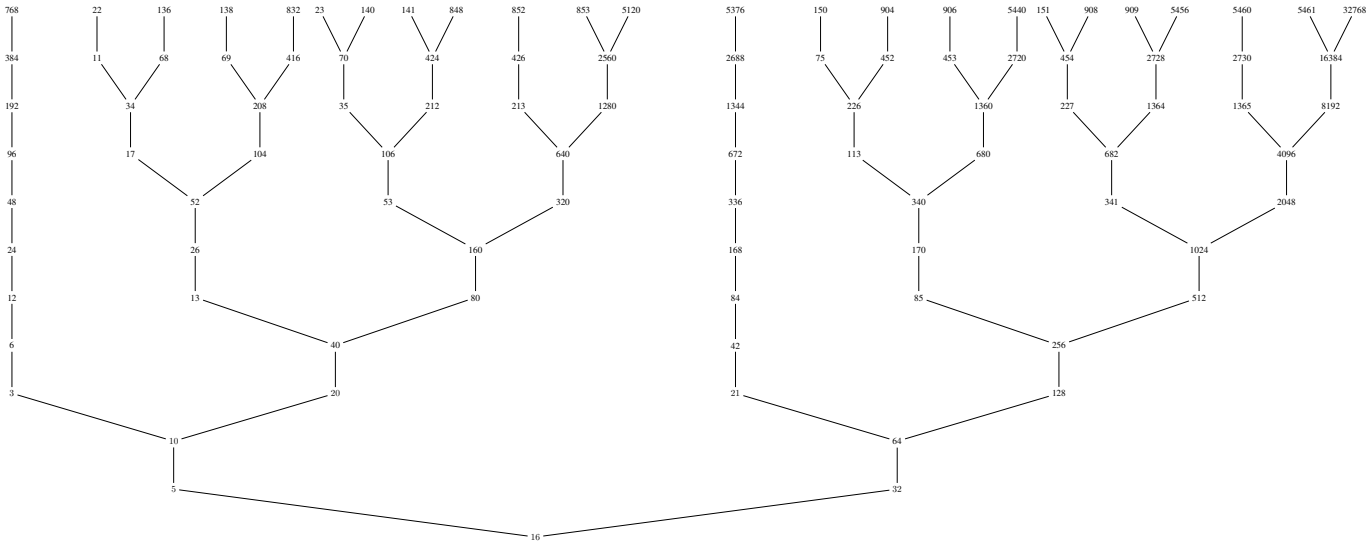


Figure 2. A fragment of Collatz tree

Shown levels 4-15. It does not include levels 0-3, they consist of elements 1 — 2 — 4 — 8 — .
Does every natural number n belong to the Collatz tree?

3. Halting conditions for Collatz's computations

Our aim, in this section, is to find a halting formula for the $3n + 1$ computations that will exhibit the properties of the computations and therefore will facilitate the proof. Note, that the obvious candidate, i.e. the formula $\{Cl\}(m = 1)$

leads to an infinite disjunction of quickly lengthening (and obscure) conjunctions. Therefore we shall consider other programs that are equivalent to the Cl program.
We begin with the algorithm (Gr) equivalent to the algorithm Cl. Next, we consider three algorithms $Gr1, Gr2, Gr3$ that are successive extensions of the eqGr algorithm.

Lemma 3.1. The following algorithm Gr is equivalent to Collatz algorithm Cl.

```
while even(n) do n := n ÷ 2 od;
while n ≠ 1 do
   n := 3 * n + 1;
   while even(n) do n := n ÷ 2 od
od;
```

or shorter

```
n := ρ(n);
while n ≠ 1 do
   n := 3 * n + 1;
   n := ρ(n);
od;
```

(Gr)

Proof:
The equivalence of the algorithms Cl and Gr is intuitive. Compare the recurrence of Collatz (rec1) and the following recurrence (rec2) that is calculated by the algorithm Gr.

$$\left.\begin{array}{rcll} k_0 = \kappa(n) & \wedge & m_0 = \rho(n) & \\ k_{i+1} = \kappa(3m_i + 1) & \wedge & m_{i+1} = \rho(3m_i + 1) & \text{for } i \geq 0 \end{array}\right\} \quad \text{(rec2)}$$

The definitions of the functions κ and $= \rho$ are contained in Definition 1.1, on page 4. Note, the following equivalence holds

$$\big(\kappa(n) = l' \wedge \rho(n) = m'\big) \Leftrightarrow \{\, l := 0;\ m := n;\ \textbf{while}\ even(m)\ \textbf{do}\ m := m \div 2;\ l := l + 1\ \textbf{od}\}\ \big(m = m' \wedge l = l'\big). \quad (1)$$

One can say the algorithm Gr is obtained by the elimination of **if** instruction from the Cl algorithm. However, construction of a formal proof is a non-obvious task. We are encouraging the reader to fill the details. □

Corollary 3.1. Every halting condition of the program Gr is also a halting condition for the program Cl.

Next, we present the algorithm $Gr1$, an extension of algorithm Gr.

```
var n, aux, i : integer;  k, m : arrayof integer;
  Γ1 : [ i := 0; k_i := κ(n);  m_i := ρ(n); ]
while m_i ≠ 1 do
  Δ1 : [ aux := 3 * m_i + 1;  k_{i+1} := κ(aux);
         m_{i+1} := ρ(aux);  i := i + 1 ]
od
```

(Gr1)

Lemma 3.2. Algorithm Gr1 has the following properties:
(*i*) Algorithms Gr and Gr1 are equivalent with respect to the halting property.
(*ii*) The sequences $\{m_i\}$ and $\{k_i\}$ calculated by the algorithm Gr1 satisfy the recurrence rec2.

Proof:
Both statements are very intuitive. Algorithm Gr1 is an extension of algorithm Gr. The inserted instructions do not interfere with the halting property of algorithm Gr. Second part of the lemma follows easily from the remark that $k_0 = \kappa(\mathrm{n})$ and $m_0 = \rho(n)$ and that for all $i > 0$ we have $k_{i+1} = \kappa(3 * m_i + 1)$ and $m_{i+1} = \rho(3 \cdot m_i + 1)$. □

Each odd number m in Collatz tree, $m \in D$, initializes a new branch. Let us give a color number $x + 1$ to each new branch emanating from a branch with color number x. Note, for every natural number$p$ the set of branches of the color p is infinite. Let W_x denote the set of natural numbers that obtained the color x. The set W_x will be calles the x-th

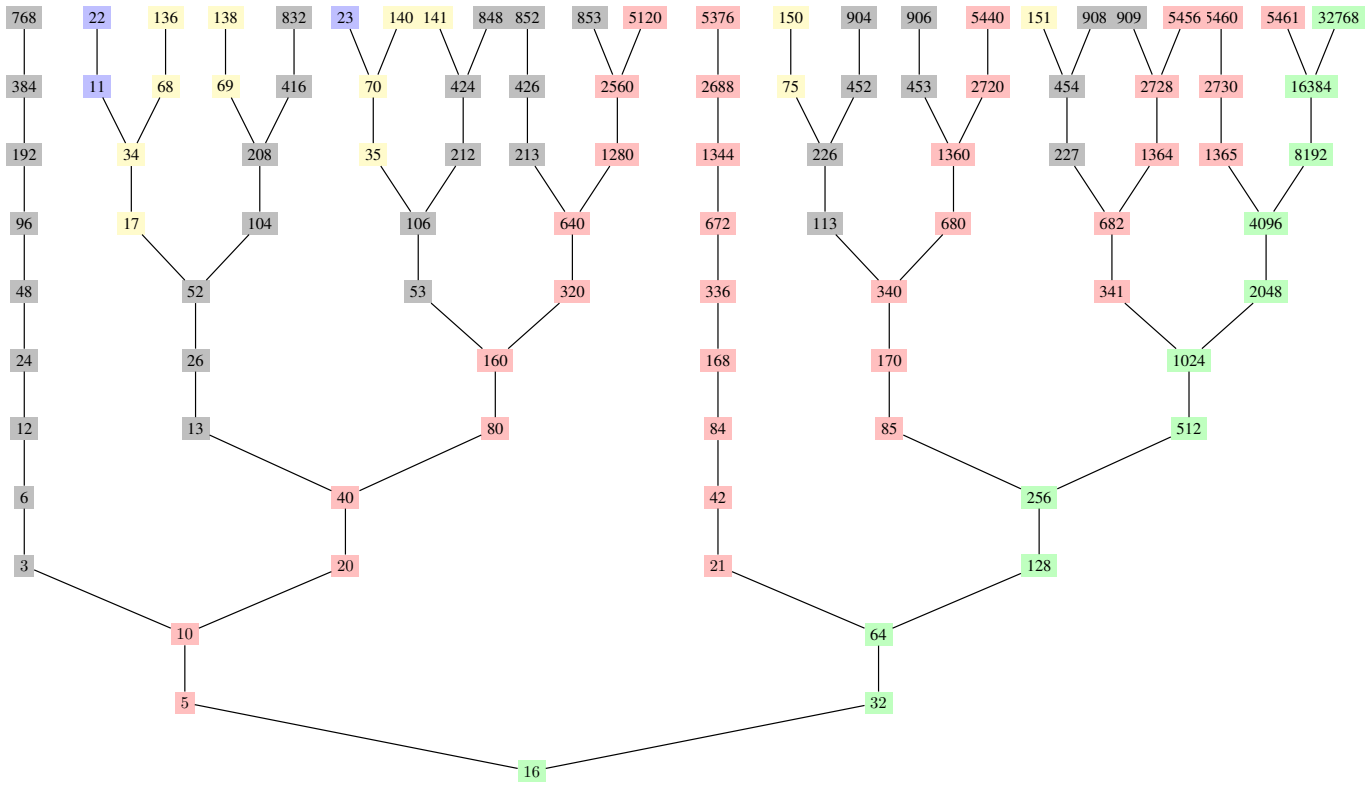


Figure 3. Strata $W_0 - -W_4$ of Collatz tree

stratum pf the set N of natural numbers.

Let s be a variable not occurring in algorithm $Gr1$. The following lemma states the partial correctnes of the algorithm $Gr1$ w.r.t. precondition $s = n$ and postcondition $s \in W_i$.

Lemma 3.3. If the algorithm $Gr1$ halts for a number n then it computes the number i of stratum W_i that contains the number n,

$$\{Gr1\}(true) \implies \big((s = n) \implies \{Gr1\}(s \in W_i)\big)$$

Next, we present another algorithm Gr2 and a lemma.

var $n, aux, i, z, y : integer;\ k, m :$ **arrayof** $integer;$
$\Gamma_2 :$ $i, y := 0;\ z, k_i := \kappa(n);\ m_i := \rho(n);$
while $m_i \neq 1$ **do**
$\Delta_2 :$ $aux := 3 * m_i + 1;\ k_{i+1} := \kappa(aux);$
$y := 3 * y + 2^{z_i};\ z := z + k_{i+1};$
$m_{i+1} := \rho(aux);\ i := i + 1$
od (Gr2)

Lemma 3.4. Algorithm $Gr2$ has the following properties:
(*i*) Both algorithms $Gr1$ and $Gr2$ are equivalent with respect to the halting property.
(*ii*) Formula $\varphi :$ $n \cdot 3^i + y = m_i \cdot 2^z$ is an invariant of the program $Gr2$ i.e. the formulas (2) and (3)

$$\{\Gamma_2\}\ \big((n \cdot 3^i + y = m_i \cdot 2^z) \wedge i = 0\big) \tag{2}$$

$$\big((n \cdot 3^i + y = m_i \cdot 2^z) \wedge i = x\big) \implies \{\Delta_2\}\big((n \cdot 3^i + y = m_i \cdot 2^z) \wedge i = x + 1\big) \tag{3}$$

are theorems of the algorithmic theory of numbers $\mathcal{ATN}$.

Proof:

Proofs of formulas (2), (3) are easy, it suffices to apply the axiom of assignment instruction Ax_{18}, see subsection 9.4. □

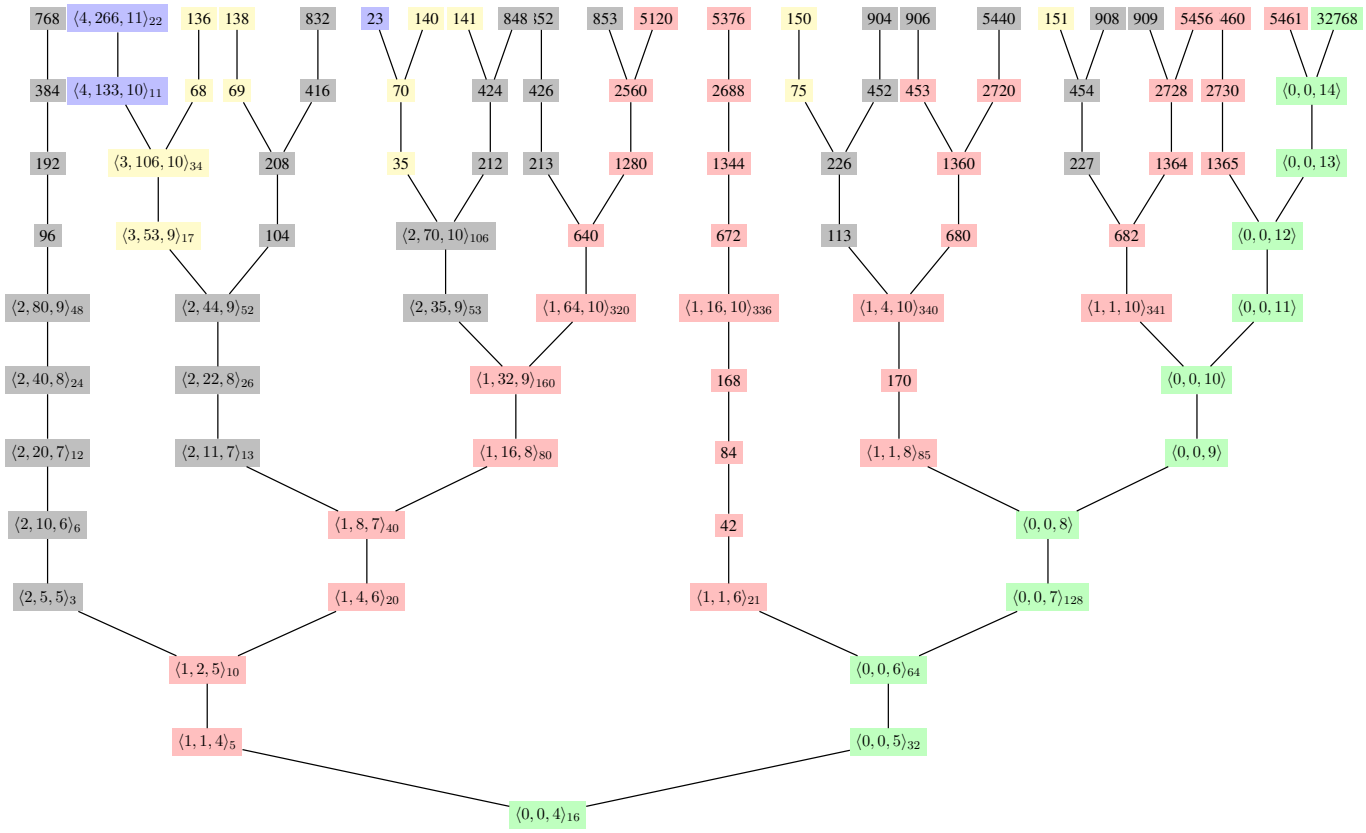


Figure 4. Tree of triples (strata 4 – 15)

Subsequent algorithm $Gr3$ exposes the history of the calculations of x, y, z.

var $n, aux, i : integer;\ k, m, x, y, z :$ **arrayof** $integer;$
$\Gamma_3 :$ $i, y_i := 0;\ z_i, k_i := \kappa(n);\ m_i := \rho(n);$
while $n \cdot 3^i + y_i \neq 2^{z_i}$ **do**
$\Delta_3 :$ $aux := 3 * m_i + 1;\ k_{i+1} := \kappa(aux);$
$y_{i+1} := 3 * y_i + 2^{z_i};\ z_{i+1} := z_i + k_{i+1};$
$m_{i+1} := \rho(aux);\ i := i + 1; x_i := i$
od (Gr3)

The algorithm $Gr3$ has a couple of interesting properties.

Lemma 3.5. Some semantical properties of the program Gr3 are:

(i) Both algorithms $Gr2$ and $Gr3$ are equivalent with respect to the halting property.

(*ii*) For every element n after each i-th iteration of algoritm $Gr3$, the following four formulas are satisfied

$$\begin{array}{|ll|}\hline \varphi:\ n\cdot 3^i+y_i=m_i\cdot 2^{z_i} & \xi: x_i=i \\ \zeta: z_i=\sum\limits_{j=0}^{i} k_j & \upsilon: y_i=\sum\limits_{j=0}^{i-1}\left(3^{i-1-j}\cdot 2^{z_j}\right) \\ \hline \end{array} \tag{4}$$

and the sequences $\{m_i\}$and $\{k_i\}$ satisfy the equalities of the recurrence (rec2).

(*iii*) In other words, the following formula is valid in the structure $\mathfrak{N}$ of natural numbers

$$\mathfrak{N}\models \Gamma_3\ \ \bigcap\{\textbf{if } m_i\neq 1 \textbf{ then } \Delta_3 \textbf{ fi}\}\,(\varphi\wedge\xi\wedge\zeta\wedge\upsilon) \tag{5}$$

(*iv*) for every element n the sequence of triples $\langle x_i, y_i, z_i\rangle$ calculated by algorithm $Gr3$ is increasing, monotone as it can be seen from the following formula.

$$(m_i\neq 1\wedge k_i=\kappa(3m_i+1))\implies\{\Delta_3\}\,(x_{i+1}=i+1\wedge y_{i+1}=3y_i+2^{z_i}\wedge z_{i+1}=z_i+k_i) \tag{6}$$

(*v*) Hence, for every element n the sequence of triples $\langle X_i\,(=i), Y_i, Z_i\rangle$ calculated by algorithm $Gr3$ is increasing, monotone as it can be seen from the following formula (7).

$$(m_i\neq 1\wedge k_i=\kappa(3m_i+1))\implies\{\Delta_3\}\,(X_{i+1}=i+1\wedge Y_{i+1}=3Y_i+2^{Z_i}\wedge Z_{i+1}=Z_i+k_i) \tag{7}$$

(*vi*) after each i-th iteration of subprogram Δ_3 the equivalence (8) holds

$$\mathfrak{N}\models\boxed{\{\Gamma_3\}\,;\{\Delta_3\}^i\,(m_i\neq 1\Leftrightarrow n\cdot 3^i+y_i>2^{z_i})} \tag{8}$$

Remark 3.1. We can say *informally* that the algorithm $Gr3$ performs as follow

$$\begin{array}{|l|}\hline i:=0; \\ \textbf{while } n\notin W_i \textbf{ do } i:=i+1 \textbf{ od} \\ \hline \end{array}$$

Let us note an interesting information on route in graph

Corollary 3.2. For every natural number n, for every natural number $p\in N$ the triple $\langle p, y_p, z_p\rangle$ computed by the program $\{\Gamma_3;\textbf{for } i:=1 \textbf{ to } p \textbf{ do } \Delta_3 \textbf{ od}\}$ determines a path from the number n to number m_p

$$\mathfrak{N}\models\boxed{\{\Gamma_3;\textbf{for } i:=1 \textbf{ to } p \textbf{ do } \Delta_3 \textbf{ od}\}\,(n\cdot 3^p+y_p=m_p\cdot 2^{z_p})} \tag{9}$$

Observe the following criterion

Lemma 3.6. (The sufficient and necessary criterion for termination of $3n+1$ computations)
Let r be a natural number (i.e. a numeral). The following conditions are equivalent

(*i*) the $3n+1$ computation for the number r is finite,

(*ii*) the following formula 10 is valid

$$\exists_x\,(r\cdot 3^x+y=2^z)\wedge\left(\begin{array}{l}\left(y=\sum\limits_{j=0}^{x-1}3^{x-1-j}\cdot 2^{\sum_{p=0}^{j}k_p}\right)\wedge\left(z=\sum\limits_{p=0}^{x}k_p\right)\wedge \\ \qquad\left(k_0=\kappa(n)\wedge m_0=\rho(n)\right)\wedge \\ \left(\bigwedge\limits_{l=0}^{x-1}\left(k_{l+1}=\kappa(3m_l+1)\wedge m_{l+1}=\rho(3m_l+1)\right)\right)\end{array}\right) \tag{10}$$

We draw the reader's attention to the fact that the formula (10) (in the criterion 3.6) is a formula of the *inessential extension* of Presburger's theory – i.e. the elementary theory of addition of natural numbers, c.f. the lemma 9.5 in the subsection 9.2.

4. Hotel Collatz $\mathcal{HC}$and the diagram $\mathfrak{D}$

Look at the figure 5. Imagine that the hotel $\mathcal{HC}$ consists of infinitely many towers. For each tower, the number of the room on the ground floor of the tower is odd. There is one room on each floor. Its number is twice the number of the room located on the floor below. Let $n = 2^i \cdot (2j+1)$. It means that the room number n is located in j^{th}-tower on the floor number i . Each tower is equipped with an elevator (shown as a green line). Moreover, each tower is connected to another by a staircaise that connects numbers $k = 2j+1$ and $3k+1$. This is shown as a red arrow $\overrightarrow{\langle k, 3k+1\rangle}$.

Definition 4.1. (Hotel Collatz)
The graph $\mathcal{HC} = \langle V, E\rangle$ is defined as follows

$V = N$ i.e. the set of vertices is the set of standard, reachable, natural numbers

$E = \{\overrightarrow{\langle k, p\rangle} : \exists_p\, k = p + p\} \cup \{\overrightarrow{\langle k, 3k+1\rangle} : \exists_p k = p + p + 1\}$ are edges of the graph

Note. Don't forget, our drawing 5 is only a small fragment of the infinite HC structure. The figure 5 shows a small part of red arrows. We drew only those red arrows that fit entirely on a page.

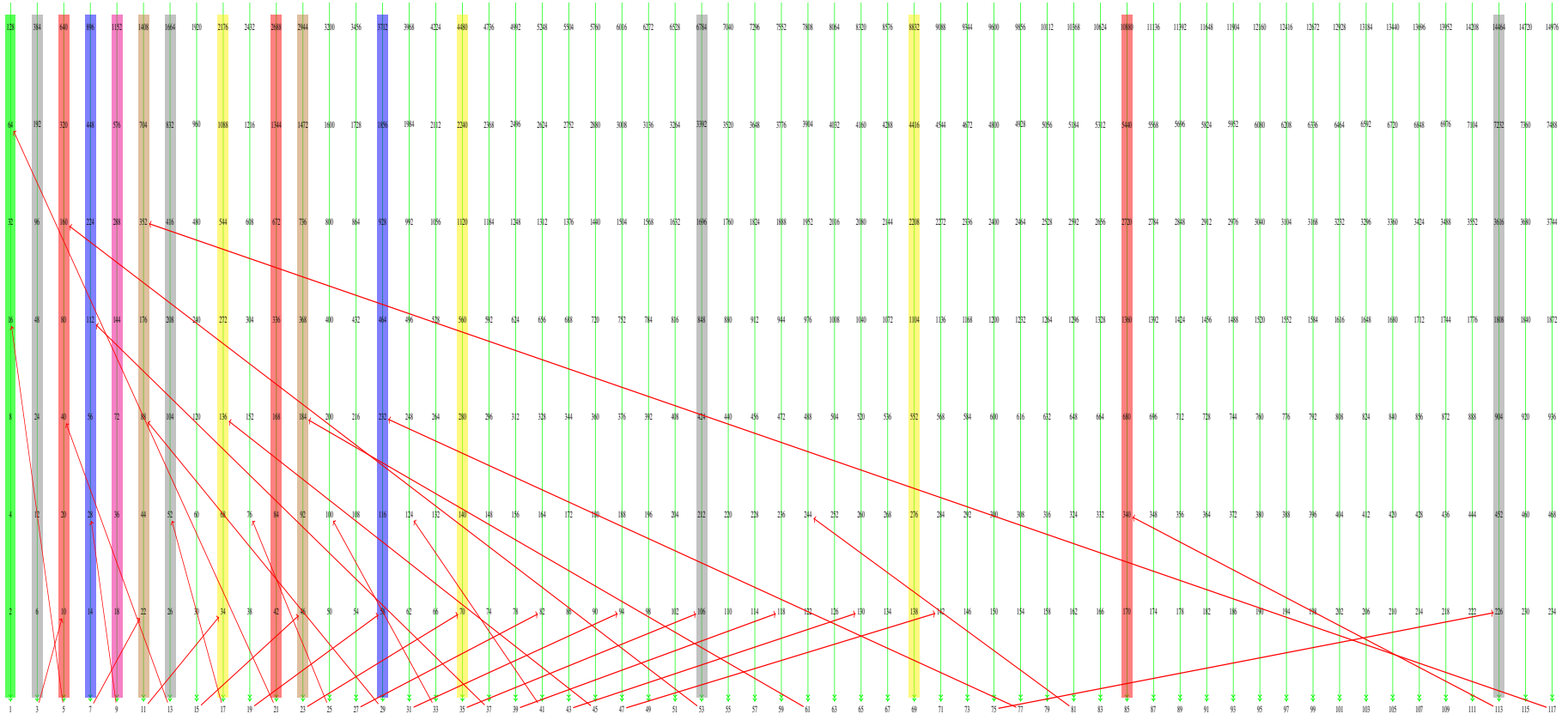


Figure 5. Hotel Collatz

It's easy to see that the sloping (red) edges of the $\mathcal{HC}$ graph change direction to the left or right. An odd-numbered edge with an odd-numbered start goes to the right. An even-numbered edge with an even-mumbered origin goes to the left.

Is it possible to find a cycle? Look at he figure **??**. This tree contains only odd numbers. If there is a cycle then it must be located farther then the largest number of the tree.
So, the problem reduces to the question: *is there an infinite path?*.

Is it possible to discern an infinite, increasing sequence of numbers m_i?

The following lemma 4.1 brings a partial answer to this questions.

Lemma 4.1. Let $n \in N$ be an odd, natural number. The following conditions are equivalent:

(*i*) The number n satisfies te equality $n+1 = 2^{p+1} \cdot (2j+1)$ where p, j are natural numbers,

(*ii*) the sequence $n = m_1 < m_2 < \cdots < m_p$ of p consecutive numbers in the compact computation is monotone, increasing, and the next computed number is smaller $m_p > m_{p+1}$.

Proof:
For every $j = 0, \ldots, p$, by the definition of compact Collatz computation, c.f. page4, the following equality holds

$$m_{j+1} = \rho(3 \cdot m_j + 1)$$

(BASE) Let us look at $m_1 = \rho(3 \cdot m_0 + 1)$. Note, $3 \cdot m_0 + 1 = 3 \cdot (2^{p+1} \cdot x - 1) + 1 = 3 \cdot (2^{p+1} \cdot x) - 2 = 2 \cdot (3 \cdot 2^p \cdot x - 1)$ and the value of the term $3 \cdot (2^p \cdot x - 1)$ is an odd number, because $expo(2 \cdot 3 \cdot (2^p \cdot x - 1), 2) = 1$. Hence $m_1 = 3 \cdot (2^p \cdot x - 1)$.
(INDUCTION STEP) In the same way we show that for every $j = 2, \ldots, p$ the subsequent number m_j satisfies the equality (11)

$$m_j = \rho(3^j \cdot (2^{p-j+1} \cdot x - 1) + 1) \tag{11}$$

and is odd.

It is obvious that the sequence $n = m_0, m_1, \ldots, m_p$ is increasing.
It is evident that $m_{p+1} < m_p$ because the number of the odd number m_p is even, for the number $3^{p+1} \cdot x$ is odd. □

Corollary 4.2. From the lemma 4.1 we learn two facts:

- For any natural number $q \in Nat$ there exists a compact computation that contains the sequence of length q of consecutive, increasing odd numbers
- There is no infinite increasing computation of standard (reachable) odd numbers .

The above lemma 4.1 shows that there is no infinite computation with all edges going right. For every natural number n the sequence of consecutive edges going right is no longer than $\kappa(n+1)$.

Corollary 4.3. It suffices to analyze the set of all odd, natural numbers.
Imagine the graph $\mathcal{HC}$ in three dimensional space. Every tower stands over an odd number o. Any two odd numbers o and p are connected by an edge iff there exists a natural number k such that the follwing equality holds $p * 2^k = 3 * o + 1$.

Conjecture 2. The hotel Collatz is an infinite, connected, acyclic graph, i.e. it is a tree. Number 1 is the root of the tree.

4.1. The tree $\mathcal{F}$ of triples.

In this subsection we shall study properties of the structure of triples of natural numbers. Some triples represent natural numbers, some do not. We say that a triple $\langle x, y, z\rangle$ represents a number n iff the following equality holds $n = (2^z \dot{-} y) \div 3^x$. Note, $n = (2^z \dot{-} y) \div 3^x \Leftrightarrow n \cdot 3^x + y = 2^z$.

Definition 4.2. (of triple's i-th successor $\bar{S}_i$)
We shall comnsider only triples that are representing natural numbers, i.e. that the formula $\exists_n n \cdot 3^x + y = 2^z$ holds. A triple $\langle x, y, z\rangle$ has successors iff the formula $\forall_n\, n \cdot 3^{x+1} + y \neq 2^z$ is satisfied, i.e. if $n \mod 3 \neq 0$.

The triple $\left\langle \underbrace{x+1}_{x'}, \underbrace{y * 2^k + 3^x}_{y'}, \underbrace{z+k}_{z'} \right\rangle$ is the i-th successor $\bar{S}_i$ of the triple $\langle x, y, z\rangle$, i.e. $\bar{S}_i \langle x, y, z\rangle \stackrel{df}{=} \langle x', y', z'\rangle$

where $i = 1, 2, \ldots$ and $k = 2i - \delta(\frac{2^z - y}{3^x})$. Note, the successor $\bar{S}_1(\langle 0, 0, 0\rangle)$ of the triple $\langle 0, 0, 0\rangle$ is not defined.

Definition 4.3. (of triple's predecessor $\bar{P}\,\langle x, y, z\rangle$)
The predecessor of the triple $\langle x, y, z\rangle$ is the triple
$\bar{P}\,\langle x, y, z\rangle \stackrel{df}{=} \left\langle x \dot{-} 1, (y \dot{-} 3^{x \dot{-} 1}) \div 2^{\kappa(y \dot{-} 3^{x \dot{-} 1})}), z \dot{-} \kappa(y \dot{-} 3^{x \dot{-} 1}) \right\rangle$

Definition 4.4. Let F be the set of triples, such that, it contains the triple $\langle 0, 0, 0\rangle$ and is closed with respect to the operations $\bar{S}_i$ of successors and $\bar{P}$ of predecessor.

The following lemma 4.4 is a collection of useful facts about the successors and the predecessor of the triples.

Lemma 4.4.

$$\bar{S}_i(t) \neq \langle 0, 0, 0\rangle \tag{12}$$
$$\bar{S}_i(t) \neq t \tag{13}$$
$$\exists_{m \in F} \exists_{i \in N}\; m = \bar{S}_i(t) \Rightarrow \bar{P}(m) = t \tag{14}$$
$$\bar{P}(m) = t \Rightarrow \exists_i\; m = \bar{S}_i(t) \tag{15}$$
$$\bar{S}_i(t) = \bar{S}_j(t) \implies i = j \tag{16}$$
$$\bar{S}_j(\bar{S}_i(t)) \neq \bar{S}_i(\bar{S}_j(t)) \tag{17}$$
$$i \neq j \implies \bar{S}_i(t) \neq \bar{S}_j(t) \tag{18}$$
$$\langle 0, 0, 0\rangle \neq t \implies \exists_i\, \bar{S}_i(\bar{P}(t)) = t \tag{19}$$
$$\text{the value of } \bar{S}_1(\langle 0, 0, 0\rangle\, 1) \text{ is undefined as well as the value of } \bar{P}(\langle 0, 0, 0\rangle) \tag{20}$$

Every formula appearing in the above lemma 4.4 may be easily verified. □

Definition 4.5. The graph $\mathcal{F}$ is the system

$$\mathcal{F} = \langle F, E\rangle$$

where set F is the set of nodes and the set E (of edges) is the set of all pairs of triples $(t, \bar{S}_i(t))$, $i = 1.2.\cdots$.
Alternatively we may consider $\mathfrak{F}$ an algebraic system of triples with one constant $\langle 0, 0, 0\rangle$ and infinitely many successors and one predecessor operation and one predicate of equality =

$$\mathfrak{F} \stackrel{df}{=} \left\langle F; \langle 0, 0, 0\rangle, \{\bar{S}_i\}_{i=1}^{\infty}, \bar{P}, = \right\rangle$$

You may look at the figure 6.

We define the the family $\{T_l\}_{l=0}^{\infty}$ of subsets of the set F of triples.

Definition 4.6. $T_0 \stackrel{df}{=} \{\langle 0, 0, 0\rangle\}$
$T_{l+1} \stackrel{df}{=} \{\langle x, y, z\rangle \colon \exists_{i \in N} \exists_{\langle x', y', z'\rangle \in T_l}\, \bar{S}_i \langle x', y', z'\rangle = \langle x, y, z\rangle\ \}$

Note the following facts

Fact 4.1.

(*i*) The graph $\mathcal{F}$ is cyclic-free.

(*ii*) The graph $\mathcal{F}$ is connected.

(*iii*) Hence, the graph $\mathcal{F}$ is a tree.

(*iv*) A triple $\langle x, y, z\rangle$ is a leaf iff it represents a number n divisible by 3.

(*v*) Every branch of the tree $\mathcal{F}$ is finite.

(*vi*) The family $\{T_l\}_{l=0}^{\infty}$ is a partition of te set F.

Let us observe the following scheme of natural induction with infinitely many successors

Lemma 4.5. (scheme of natural induction with infinitely many successors)
Let $\Psi(t)$ be an algorithmic formula in which t is a free variable of type triple of natural numbers. The formula of the scheme(NIS) is a theorem of algorithmic theory of triples.

$$\Psi(t/\langle 0,0,0\rangle) \wedge [(\Psi(t) \wedge \neg t \text{ is a leaf}) \implies \forall_{i\in N}\Psi(t/\bar{S}_i(t))] \implies \forall_{t\in F}\Psi(t) \tag{NIS}$$

The proof goes by the induction with the respect to the levels l of the family $\{T_l\}_{l=0}^{\infty}$.

Lemma 4.6. For every triple $t \in F$ the following algorithm 21 terminates when executed in the structure $\mathfrak{F}$

$$\{\mathbf{while}\ t \neq \langle 0,0,0\rangle\ \mathbf{do}\ t := \bar{P}(t)\ \mathbf{od}\} \tag{21}$$

The proof is by an easy induction.

Lemma 4.7. For every triple $t \in F$ there exists a sequence of natural numbers $j_1, j_2, \ldots j_{r-1}, j_r$ such that the following equality holds

$$t = \bar{S}_{j_r}(S_{j_{r-1}}(\ldots \bar{S}_{j_2}(\bar{S}_{j_1}(\langle 0,0,0\rangle))\ldots))$$

Proof:
The proof follows easily from the previous lemma 4.6. A look at the figures 6 and 7 and lemma4.8 may help too. □

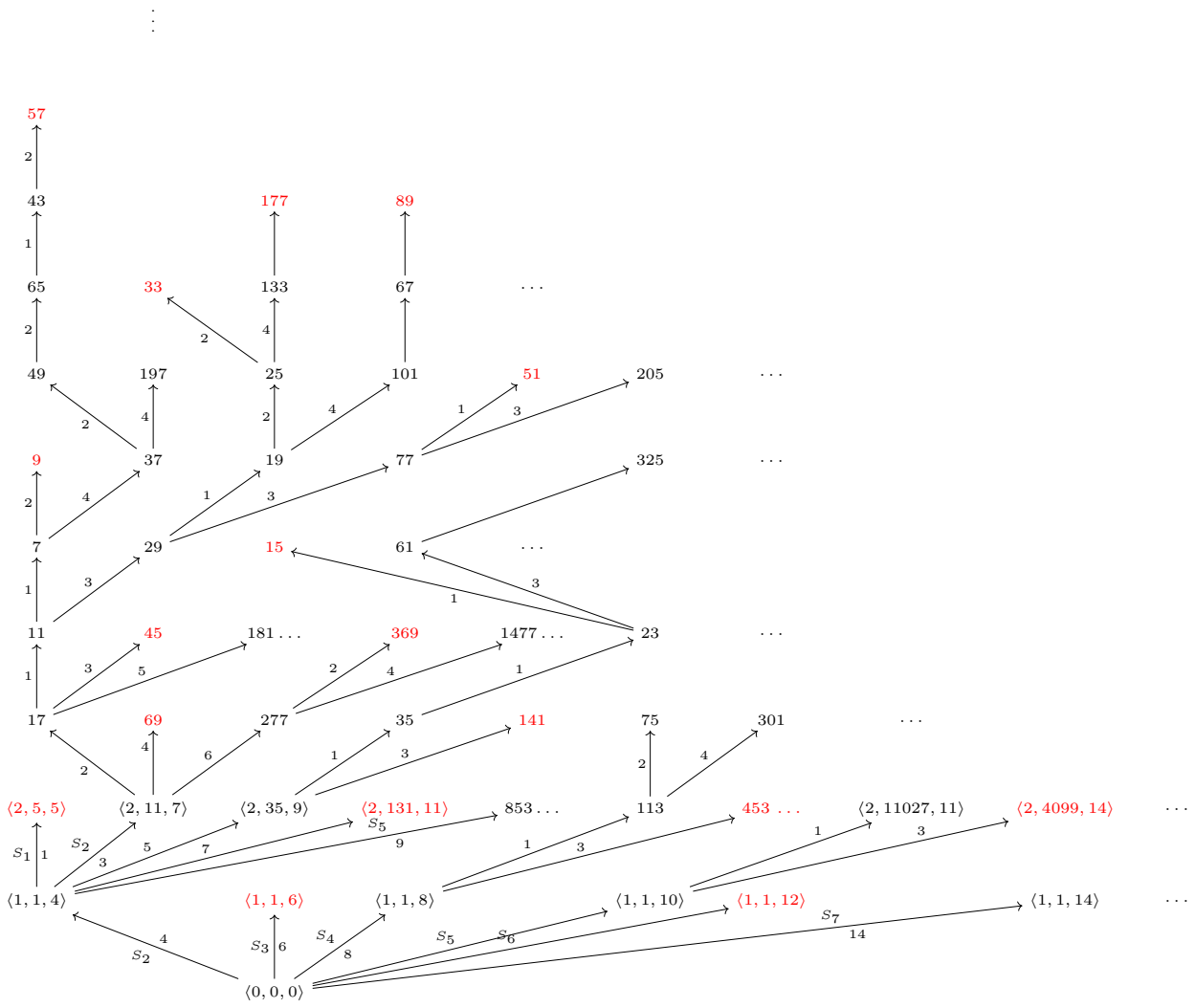


Figure 6. A fragment of the tree $\mathcal{F}$

A couple of equivalent conditions, that relate computations on numbers and computations on triples.

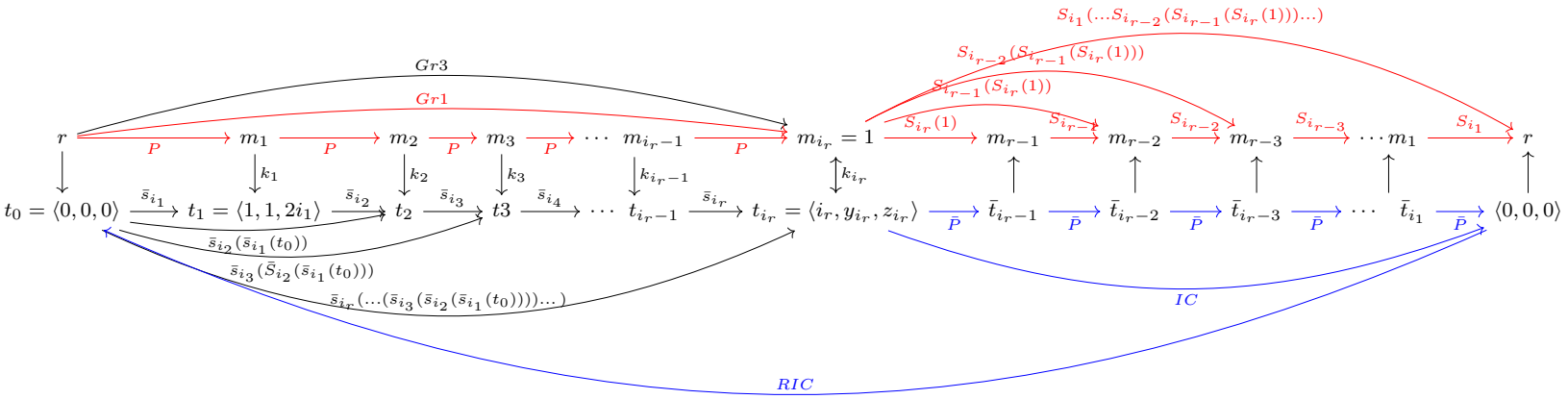


Figure 7. TWO ALGORITHMS TO FIND A PATH FROM 1 TO r OR FROM r TO 1

The numbers k_j and i_j are related as follows $k_j = 2i_j - \delta(m_{j-1})$.

Lemma 4.8. (The case of finite computation)
Let $r \neq 0$ be any natural number. The following conditions are equivalent

(*i*) The algorithm (Cl) halts.

$$\{Cl\}(m = 1)$$

(*ii*) The algorithm (Gr2) halts and thepost-condition $r \cdot 3^x + y = 2^z$ holds

$$\{Gr2\}(r \cdot 3^x + y = 2^z)$$

(*iii*)

$$\{Gr2\}(n \cdot 3^x + y = 2^z) \implies \underbrace{\left\{ \begin{array}{l} x' := x; \quad y' := y; \qquad z' := z; \quad m := r; \\ \textbf{while } x' + y' + z' \neq 0 \textbf{ do} \\ \quad k := \kappa(y');\ m := P(m);\ x' := x' - 1;\ y' := (y' \dot{-} 3^{x'}) \div 2^k;\ z' := z' - k \\ \textbf{od} \end{array} \right\}}_{IC} (x' + y' + z' = 0) \tag{22}$$

4.2. The graph $\mathcal{G}$ of odd, natural numers

We shall introduce the notions of successors and of predecessor of an odd, natural number. Consider the set V of all odd, natural numbers. We are going to define a graf $\mathcal{G}$, see figure. 8. This diagram shows odd natural numbers only. [1]

Definition 4.7. For every odd, natural number n, not divisible by 3, we put

$$\delta(n) \stackrel{df}{=} n \mod 3 - 1$$

$$\text{i.e. } \delta(n) = \begin{cases} 0 & \text{when } n \bmod 3 = 1 \\ 1 & \text{when } n \bmod 3 = 2 \end{cases}.$$

We define the graph $\mathcal{G}$, look at Fig. 8,

Definition 4.8. Graph $\mathcal{G}$ of odd numbers is the system of two sets

$$\mathcal{G} \stackrel{df}{=} \{V, E\}$$

The set V (of *nodes*) is the set of all odd natural numbers.
The set E (of *edges*) is the set of all pairs $\langle n, m\rangle$ such that

c1) n, m are odd natural numbers, $E \subset V \times V$,

c2) the number n is indivisible by 3, $n \bmod 3 \neq 0$,

c3) $m = \cdot\dfrac{n \cdot 2^{2i-\delta(n)} - 1}{3}$ where $i \in \{1, 2, \dots\}$ and $\delta(n) \stackrel{df}{=} n \bmod 3 - 1$.

Definition 4.9. (of successor $m = S_i(n)$)
The i-th successor of an odd number n is defined as follows $S_i(n) \stackrel{df}{=} \dfrac{n \cdot 2^{2i-\delta(n)} - 1}{3}$, where $i = 1, 2, \dots$.

NOTE, the odd numbers divisible by 3 ($n \mod 3 = 0$) have no successors, i,e, they are *leaves*.

NOTE successor $S_1(1)$ is not defined.

Definition 4.10. (of predecessor $n = P(m)$)
The predecessor of an odd number m is the odd number $n = \dfrac{3 * m + 1}{2^{\kappa(3m+1)}}$.

The following lemma gathers a couple of useful facts.

Lemma 4.9.

$$S_i(n) \neq 1 \tag{23}$$
$$S_i(n) \neq n \tag{24}$$
$$m = S_i(n) \Rightarrow P(m) = n \tag{25}$$
$$P(m) = n \Rightarrow \exists_{i>0}\, m = S_i(n) \tag{26}$$
$$S_i(n) = S_j(n) \implies i = j \tag{27}$$
$$i \neq j \implies S_j(S_i(n)) \neq S_i(S_j(n)) \tag{28}$$
$$i < j \implies S_i(n) < S_j(n) \tag{29}$$
$$\text{the value of } S_1(1) \text{ is undefined as well as the value of } P(1) \tag{30}$$

Proof:
We shall prove the formula (26). If $P(m) = n$ then $n \cdot 2^{\kappa(3\cdot m+1)} = 3 \cdot m + 1$. Hence $i = (\kappa(3 \cdot m + 1) + \delta(n)) \div 2$.
The proof of the remaining formulas is left to the reader. □

[1] To see the even numbers imagine that you are looking at the Hotel Collatz from below. Every tower of the hotel i.e. the set of all numbers $\{2^i(2t+1)\}, i = 0, 1, 2, \dots$ is hidden behind its number $2t + 1$.

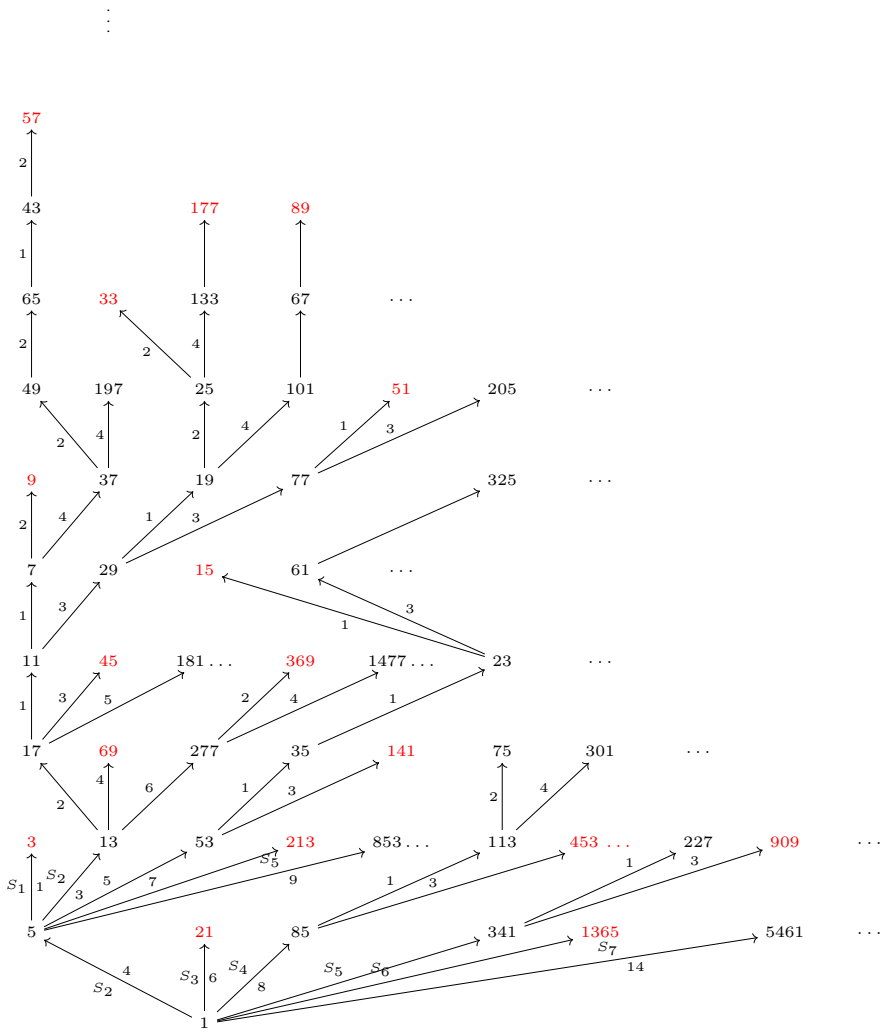


Figure 8. A fragment of the graph $\mathcal{G}$, The numbers k_i on the edges indicate the numbers ofsuccessors S_i, namely $i = \lceil \frac{k_i}{2} \rceil$

4.3. The diagram $\mathfrak{D}$ of the $3n+1$ structure of odd numbers

magine a three dimensional graph $\mathcal{FG}$. It consists of two graphs $\mathcal{F}$ and $\mathcal{G}$ allocated on their own planes. The nodea of graphs are joined by arrows $\vec{f}$. We put $\vec{f}\langle 0,0,0\rangle = 1$ and in general $\vec{f}\langle x,y,z\rangle = \dfrac{2^z - y}{3^x}$.
It is easy to verify that the mapping $\vec{f}$ is an injection.
Is it a bijection?

Lemma 4.10. For every triple $t \in F$ of the graph $\mathcal{F}$, the diagram on the figure 10 commutes

$$\forall_{t\in F}\ S_i(f(t)) = f(\bar{S}_i(t)) \tag{31}$$

and the function f establishes an isomorhism of graphs $\mathcal{F}$ and $\mathcal{G}$.

Proof:
Consider $t_0 = \langle 0,0,0\rangle$. Obviously, $f(t_0) = 1$. Put $g(1) \stackrel{df}{=} \langle 0,0,0\rangle$. Hence $g(f(t_0)) = t_0$

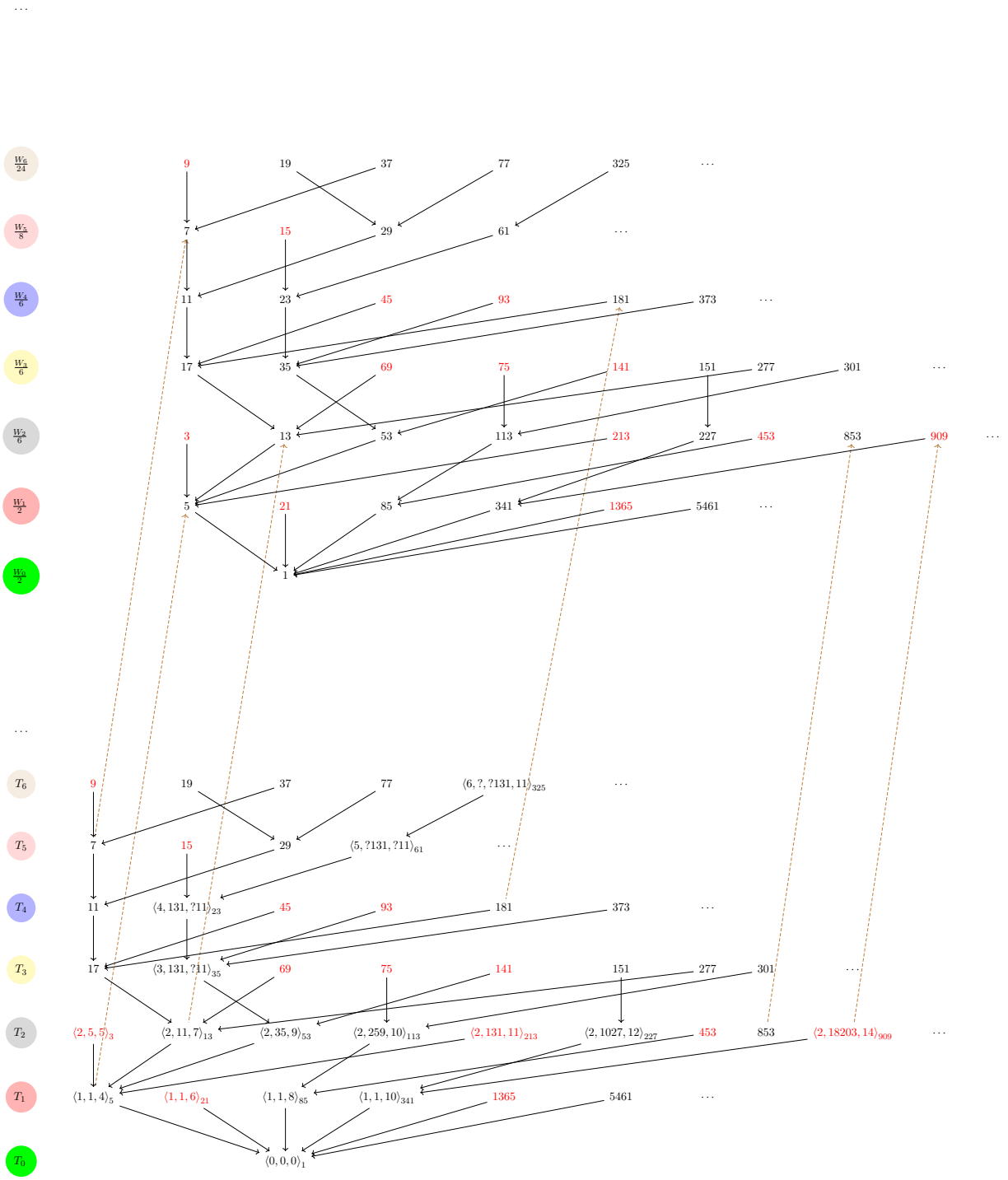


Figure 9. graphs $\mathcal{F}$ and $\mathcal{G}$ are isomorphic

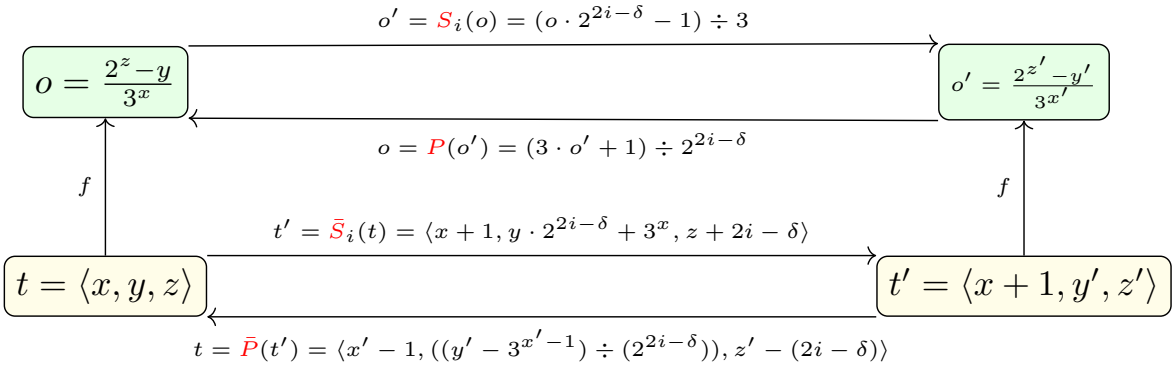


Figure 10. Commutativity of successors S_i (on odd numbers) and $\bar{S}_i$ (on triples)

For every number $i = 2, 3, \ldots$ the triple t_0 has the successor $\bar{S}_i(t_0) = \langle 1, 1, 2i \rangle$. Put $g(f(\langle 1, 1, 2i \rangle)) \stackrel{df}{=} \langle 1, 1, 2i \rangle$. It is easy to veify that, $1^\circ \forall_{i \geq 2}\, S_i(f(t_0)) = f(\bar{S}_i(t_0))$ and $2^\circ\ g(f(\bar{S}_i(t_0))) = \bar{S}_i(t_0))$.

We shall prove INDUCTION THESIS:

> for every natural number $l > 1$ and for every triple $t \in T_l$ such that the number $o = f(t)$ is not divisible by 3($3 \nmid o$) for every number $i \in N$ the equality $f(\bar{S}_i(t)) = S_i(f(t))$ holds and
> the equality $g(f(\bar{S}_i(t))) = \bar{S}_i(t))$ is satisfied too .

Assume that the thesis holds for $k \leq l$. Consider a triple $t \in T_l$ and the number $o = f(t)$.
We can assume that $3 \nmid o$ for the case $3 \mid o$ is not interesting.
Repeating, *mutatis mutandis*, the arguments used above, we verify that for every number $i \in N$ the equality $S_i(f(t)) = f(\bar{S}_i(t))$ holds and we can put $g(f(\bar{S}_i(t))) = \bar{S}_i(t))$.
Therefore, the thesis holds for every number $l \in N$.

It remains to be proved that the value $g(n)$ of the function g is defined for every odd, natural number n.
Suppose that for certain odd number n_0 for every triple $t \in F$ the inequality $f(t) \neq n_0$ holds.
Every odd umber has a predecessor. Let $m_0 = P(n_0)$.
By the lemma 4.9 (26) we now that $\exists_{i_0}\, S_{i_0}(m_0) = n_0$.
W.l.g. we can assume that the function g is efined for the argument m_0. Let $g(m_0) = t$.
We have the following equalities $f(t) = m_0 \wedge S_{i_0}(m_0) = n_0 \wedge t' = \bar{S}_{i_0}(t)$.
From the equality 31 we infer that $n_0 = f(t')$ and $g(n_0 = t')$. This proves that $g = f^{-1}$. □

From the lemma 4.10 we deduce the following

Lemma 4.11. Let n be an arbitrary odd number and the triple $t = \langle x, y, z \rangle = g(n)$, the following equalities (32) hold

$$(n \cdot 3^x + y = 2^z) \wedge \left(\begin{array}{c} \left(y = \sum_{j=0}^{x-1} 3^{x-1-j} \cdot 2^{\sum_{p=0}^{j} k_p} \right) \wedge \left(z = \sum_{p=0}^{x} k_p \right) \wedge \\ \left(k_0 = \kappa(n) \wedge m_0 = \rho(n) \right) \wedge \\ \left(\bigwedge_{l=0}^{x-1} \left(k_{l+1} = \kappa(3m_l + 1) \wedge m_{l+1} = \rho(3m_l + 1) \right) \right) \end{array} \right) \tag{32}$$

Proof:
The straightforward proof goes by an induction with respect to the levels of the tree G and is left to the reader. □

Note, the formula 32 is identical with the termination criterion 10.

We complete our considerations by the following

Lemma 4.12. For every natural number $n \in N$, the execution of the program (Gr3) is finite.

Proof:
Every $3n+1$ computation is loop-free.
Every natural number n belongs to the graph $\mathcal{G}$. The proof goes by induction on the levels of diagram $\mathfrak{D}$. We use also the commutativity lemma 4.10. □

Consider the following program (Ptr)

$$\left\{\begin{array}{l}\textbf{while } m \neq 1 \textbf{ do}\\ \quad m := P(m)\\ \textbf{od}\end{array}\right\} \tag{Ptr}$$

Let $\underline{\mathbf{n}}$ be a numeral, i.e. an expression that contains only constants, operators and parentheses (no variables).
Examples: $\underline{\mathbf{3}} = 1+1+1$, $\underline{\mathbf{227}} = (\underline{\mathbf{2048}} \dot{-} \underline{\mathbf{1027}}) \div \underline{\mathbf{9}}$,

Lemma 4.13. Let $\underline{\mathbf{n}}$ be a numeral representing a natural number. Every fornula of the following scheme (33)

$$\{m := \underline{\mathbf{n}};\ Ptr\}\,(m = 1) \tag{33}$$

is valid, i.e. the program Ptr terminates when executed with the initial valuation of the variable $m = \underline{\mathbf{n}}$.

Proof:
The thesis follows immediately from the properties of the tree $\mathfrak{G}$, c.f. page **??** □

5. Main lemma

Lemma 5.1. For every natural number r the formula 34 holds

$$\{x := 0\}\bigcup\{x := x+1\}\left(\begin{array}{l}(r\cdot 3^x + y = 2^z) \wedge \\ \left|\begin{array}{l}\left(y = \sum\limits_{j=0}^{x-1} 3^{x-1-j}\cdot 2^{\sum_{p=0}^{j} k_p}\right) \wedge \left(z = \sum\limits_{p=0}^{x} k_p\right) \wedge \\ \qquad \left(k_0 = \kappa(n) \wedge m_0 = \rho(n)\right) \wedge \\ \left(\bigwedge\limits_{l=0}^{x-1}\left(k_{l+1} = \kappa(3m_l+1) \wedge m_{l+1} = \rho(3m_l+1)\right)\right)\end{array}\right|\end{array}\right) \tag{34}$$

The lemma states that for every natural number r, the $3n+1$ computation terminates. More exacly, it states that the set St_0 of formulas is recursive and it consists of theorems of $\mathcal{ATN}$ theory.

Proof:
By an easy induction on the levels of the tree $\mathcal{G}$, c.f. Fig. **??** we verify that every node of the tree satisfies the termination criterion (10), □

Example 5.1.

$$\left|\begin{array}{r}5\cdot 3+1 = 16\\ 67\cdot 3^8 + y = 2^z\\ 5461\cdot 3+1 = 2^{14}\\ \\ \\ 227\cdot 3^2+5 = 2^{11}\end{array}\right|$$

to calculate y and z- you may use this sequence $67, 101, 19, 29, 11, 17, 13, 5, 1$

$$\overbrace{\overbrace{(((227*3+1)\ \div\ 2^1)}^{P(227)=341}*3+1)\ \div\ 2^{10}}^{P(P(227))} = 1$$

or $227*3^2 + (3*2^0 + 2^1) = 2^{11}$

Corollary 5.1. Let St_0 be the set of sentences of the form (S_0)

(*i*) every sentence of the set St_0 is a theorem of $\mathcal{T}'$ theory (i.e. of an inessential extension of Presburger theory) and hence it is the theorem of $\mathcal{ATN}$ algorithmic theory of natural numbrs as well.

(*ii*) the set St_0 is a recursive set.

Let $r > 0$ be a natural number, let i_r be the number such that the formula 34 is valid in the structure $\mathfrak{N}$.

- The set of equations St_0 that contains all equalities of the form (S_0) is a recursive set.

$$St_0 \stackrel{df}{=} \left\{ \boxed{r \cdot 3^{i_r} + \underbrace{\sum_{j=0}^{i_r-1} 3^{i_r-1-j} \cdot 2^{\sum_{p=0}^{j} k_p(r)}}_{y} = 2^{\underbrace{\sum_{j=0}^{i_r} k_j(r)}_{z}}} \right\}_{r=1}^{\infty} \tag{S_0}$$

- Every element of the set St_0 is a theorem of the theory $\mathcal{ATP}$ (Preburer's arithmetic).

The lemma states that 1°) the infinite set St_0 of expressions is accompanied by an algorithm that decides whether a given formula φ belongs to it, $\varphi \in St_0$ and 2°) every element φ of the set St_0 is a theorem of (inessentially etended) first-order theory of addition of natural numbers. Note, the proof of statement φ is done by performing the computation of algorithm.

Corollary 5.2. In other words, every number r can be presented in the following form

$$\frac{\frac{\frac{\frac{2^{k_{i_r}}-1}{3} \cdot 2^{k_{i_r-1}}-1}{3} \cdot 2^{k_{i_r-2}-1}}{3} \cdot 2^{k_{i_r-3}-1} \cdots}{\cdots} \cdot 2^{k_0} - 1}{3} = r \tag{35}$$

or in yet another form

$$r = S_{k_0}\left(S_{k_1}\left(\cdots\left(S_{k_{i_r-1}}\left(S_{k_{i_r}}(1)\right)\right)\cdots\right)\right) \tag{36}$$

or in even shorter form

$$P^{i_r}(r) = 1 \tag{37}$$

5.1. The limitations of the Main lemma

Does the lemma 5.1 solve the problem stated by Lothar Cllatz?
It seems so, since for every natural number r the computation of the algorithm Cl in the structure $\mathfrak{N}$ is finite.

One can write the statement

$$\forall_n \exists_{x,y,z}\, n \cdot 3^x + y = 2^z.$$

Is it a theorem?
Is the statement (38) a theorem?

$$\forall_n \left\{ \begin{array}{l} \textbf{while } n \neq 1 \textbf{ do} \\ \quad \textbf{if } odd(n) \textbf{ then } n := 3 \cdot n + 1 \textbf{ else } n := n \div 2 \textbf{ fi} \\ \textbf{od} \end{array} \right\} (n = 1) \tag{38}$$

In 1929 Stanisław Jaśkowski[Tar34] remarked that, a substructure $\mathfrak{J} \subset \mathfrak{C}$ of the field of complex numbers, provides a counterexample(s). This shows that the sentence (38) is not a theorem. For the details consult the subsection 9.1.

The algorithmic formula $\{q := 0; \mathbf{while}\ n \neq q\ \mathbf{do}\ q := q + 1\ \mathbf{od}\}\,(n = q)$ is satisfied by all elements that are natural numbers and it is not satisfied by other non-standard elements.

5.2. The positive consequnces of main lemma.

For every $r > 0$ be a natural number, a numeral representing this number will be denoted by $\underline{r}$. There exists the number $i(r)$such that formula of the form

$$n = \underline{r} \implies \left\{\Gamma_3\,; \mathbf{if}\ m \neq n\ \mathbf{then}\ \Delta_3\ \mathbf{fi}^{i(r)}\right\}(m = 1) \tag{39}$$

is a theorem of the theory $\mathcal{ATN}$.
Note, that the formula 39 can be replaced by an equivalent formula (provided that the variable l does not appear in Δ_3)

$$n = \underline{r} \implies \{\Gamma_3\,; \mathbf{for}\ l := 1\ \mathbf{to}\ i(r)\ \mathbf{do}\ \Delta_3\ \mathbf{od}\}\,(m = 1). \tag{40}$$

Example 5.2.

$$\mathfrak{N} \models n = 67 \implies \left\{m := n;\ \text{for } l := 1 \text{ to } 8 \text{ do } m := \frac{3 * m + 1}{2^{\kappa(3*m+1)}} \text{ od}\right\}(m = 1)$$
$$\mathfrak{N} \models \{m := 373;\ \text{for } l := 1 \text{ to } 4 \text{ do } m := \rho(3 * m + 1) \text{ od}\}(m = 1)$$
$$\mathfrak{N} \models n = 1367 \implies \{m := n;\ \text{for } l := 1 \text{ to } 11 \text{ do } m := \rho(3 * m + 1) \text{ od}\}(m = 1)$$

6. Proof of the Collatz conjecture

We claim that the formula (Main Thm) expresses the conjecture of Collatz. Look at the implication in the formula (Main Thm). The *antecedent* of this implication says: "n *is a natural number*" for it has the value $\mathbb{T}$ (i.e. true) iff the $n = 1 \vee n = 2 \vee n = 3 \vee \ldots$. Similarly, the *consequent* of the implication takes the value $\mathbb{T}$ iff the program in the consequent terminates and the final value of the variable m is 1.

Theorem 6.1. (Main)
The following formla (Main Thm) is a theorem of formalized algorithmic theory $\mathcal{ATN}$

$$\mathcal{ATN} \vdash \forall_{n\neq 0} \underbrace{\left[\left\{\begin{array}{l} q := 1; \\ \mathbf{while}\ n \neq q\ \mathbf{do} \\ \quad q := q + 1 \\ \mathbf{od} \end{array}\right\}(n = q)\right]}_{\mathbb{FORALL}\ n,\ \mathbb{IF}\ n\ is\ a\ natural\ number} \implies \underbrace{\left[\left\{\begin{array}{l} m := n \div 2^{\kappa(n)}; \\ \mathbf{while}\ m \neq 1\ \mathbf{do} \\ \quad m := 3 \cdot m + 1; \\ \quad m := m \div 2^{\kappa(m)} \\ \mathbf{od} \end{array}\right\}(m = 1)\right]}_{\mathbb{THEN}\ the\ computation\ for\ n\ is\ finite\ \mathbb{FI}} \tag{Main Thm}$$

The theorem reads as follow, (the program in the consequent is abbreviated as Gr):
for every n, *if* $n \neq 0$ *is a natural number,*
then the computation of the program Gr *is finite and final value of the variable* $m = 1$.

The idea of te proof can be explained as follow:

1. We know that the set St_0 p.21,(S_0), is recursive and consists of theorems of elementary theory of addition of natural numbers (i.e. the Presburger's arithnetic with helpful but inessential extensions), and hence of the algorithmic theory of numbers $\mathcal{ATN}$.

2. We transform this set to three consecutive sets $St_0 \to St_1 \to St_2 \to St_3$ in such a way that every set $St_i, i = 1, 2, 3$ is recursive and consists of theorems of algorithmic theory of natural numbers. Every formula φ of the set St_{i+1} has a proof from some formula $\psi \in St_i$.

3. In the last step we apply the inference rule R_3, see page 34, of infinitely many premises.

We use a couple of denotations.
Let r be a natural number.
The sign Γ abbreviates the program $\{m := n \div 2^{\kappa(n)};\}$ or if you like $\{m := n;\ \mathbf{while}\ even(n)\ \mathbf{do}\ n := n \div 2\ \mathbf{od}\}$.
The sign Δ abbreviates the program $\{m := 3 \cdot m + 1;\ \mathbf{while}\ even(n)\ \mathbf{do}\ n := n \div 2\ \mathbf{od}\}$
or if you like$\{m := 3 \cdot m + 1;\ m := m \div 2^{\kappa(m)}\}$.

Proof:
We are resuming at the main lemma 5.1, for every natural number r exists number i_r such that $m_{i_r} = 1$.
Consider the set St_1 of formulas such that for every natural number r the formula of the scheme (41)

$$n = \underline{r} \implies \{\Gamma\}\{\mathbf{if}\ m \neq 1\ \mathbf{then}\ \Delta\ \mathbf{fi}\}^{i(\underline{r})}(m = 1) \tag{41}$$

is contained in St_1.

Lemma 6.1. The following conditions hold

(i) The set St_1 is a recursive set.

(ii) For every formula $\psi \in St_1$ there is a formula $\varphi \in St_0$ such rhat the equivalence $\varphi \equiv \psi$ is a theorem of program calculus.

$$St_1 \stackrel{df}{=} \left\{n = \underline{r} \implies \{\Gamma\}\{\mathbf{if}\ m \neq 1\ \mathbf{then}\ \Delta\ \mathbf{fi}\}^{i(\underline{r})}(m = 1)\right\}_{r=1}^{\infty} \tag{42}$$

T1a) the set St_1 is a recursive set.

T1b) Every formula φ from the set St_1 is a theorem of algorithmic theory $\mathcal{ATN}$ of natural numbers.
In the proof we use the following axiom (Ax_{18}) of assignment instruction

$$\boxed{\{x := \tau\}\,\alpha(x) \Leftrightarrow \alpha(x/\tau)} \tag{Ax_{18}}$$

Lefthandsize is an algorithmic formula where $\{x := \tau\}$ is assigbment instruction, and $\alpha(x)$ is a formula.
Righthandsize is the formula that arises from $\alpha(x)$ by replacing all free occurencies of variable x in α by expression τ.

Next, we consider the set St_2 that contain all formulas of the scheme shown in the equation 43 and only such formulas.

$$St_2 \stackrel{df}{=} \left\{n = \underline{r} \implies \{\Gamma;\ \mathbf{while}\ m \neq 1\ \mathbf{do}\ \Delta\ \mathbf{od}\}(m = 1)\right\}_{r=1}^{\infty} \tag{43}$$

Our next observation says:

T2) Every formula ϕ from the set St_2 is a theorem of algorithmic theory $\mathcal{ATN}$ of natural numbers.

Each formula of the set St_2 is proved from a corresponding formula φ in the set St_1 using the following theorem (ThIf) of calculus of programs.

$$\{M; \mathbf{if}\ \gamma\ \mathbf{then}\ K\ \mathbf{fi}^i\}(\alpha \wedge \neg\gamma) \implies \left\{\begin{array}{l} M; \\ \mathbf{while}\ \gamma \\ \mathbf{do}\ K\ \mathbf{od} \end{array}\right\}(\alpha \wedge \neg\gamma) \tag{ThIf}$$

Hence the thesis T2) is justified.
Therefore, the set St_3 that consists of all formulas of the scheme (44).

$$St_3 \stackrel{df}{=} \left\{ \left\{\begin{array}{l} q := 0; \\ \left\{\ q := q+1\ \right\}^{\underline{r}} \end{array}\right\}(n = q) \implies \left\{\begin{array}{l} \Gamma; \\ \mathbf{while}\ m \neq 1 \\ \mathbf{do}\ \Delta\ \mathbf{od} \end{array}\right\}(m = 1) \right\}_{r=1}^{\infty} \tag{44}$$

and no other formulas is the set of theorems of the theory $\mathcal{ATN}$.
Now, we apply the following inference rule R_3 of calculus of programs

$$\frac{\{M; \mathbf{if}\ \gamma\ \mathbf{then}\ K\ \mathbf{fi}^i\}(\neg\gamma \wedge \alpha) \implies \beta\}_{i=0}^{\infty}}{\{M; \mathbf{while}\ \gamma\ \mathbf{do}\ K\ \mathbf{od}\}(\neg\gamma \wedge \alpha) \implies \beta\}} \tag{R_3}$$

to obtain the theorem Main Thm of the algorithmic theory $\mathcal{ATN}$ of natural numbers. □

COMMENT, the proof of the theorem Main Thm is an infinite tree, all of its branches are finite, all leafs are axioms (or some eaerlier proved theorems). Obviously,such a proof can not be written in finite time. Instead, we have proved that the proof exists. For the definition of proof in the calculus of programs consult [MS87] Definition II.5.2, p.58.

7. Final remarks.

Our message does not limit itself to the proof of Collatz theorem.
Namely, we are presenting a solid argument that the algorithmic language of program calculus is indispensable for expressing the semantic properties of programs. Halting property of program, correctness property, axiomatic specification of data structure of natural numbers,etc., can not be expressed by (sets) of first-order formulas.
We show the potential of calculus of programs as a tool for

- specification of semantical properties of software and
- verification of software against some specifications.

We hope the reader will forgive us for a moment of insistence (is it a propaganda?).
Calculus of programs $\mathcal{AL}$ is a handy tool. For there are some good reasons to use the calculus of programs

(*i*) The language of calculus $\mathcal{AL}$ contains algorithms (programs) and *algorithmic formulas* besides terms and first-order formulas.

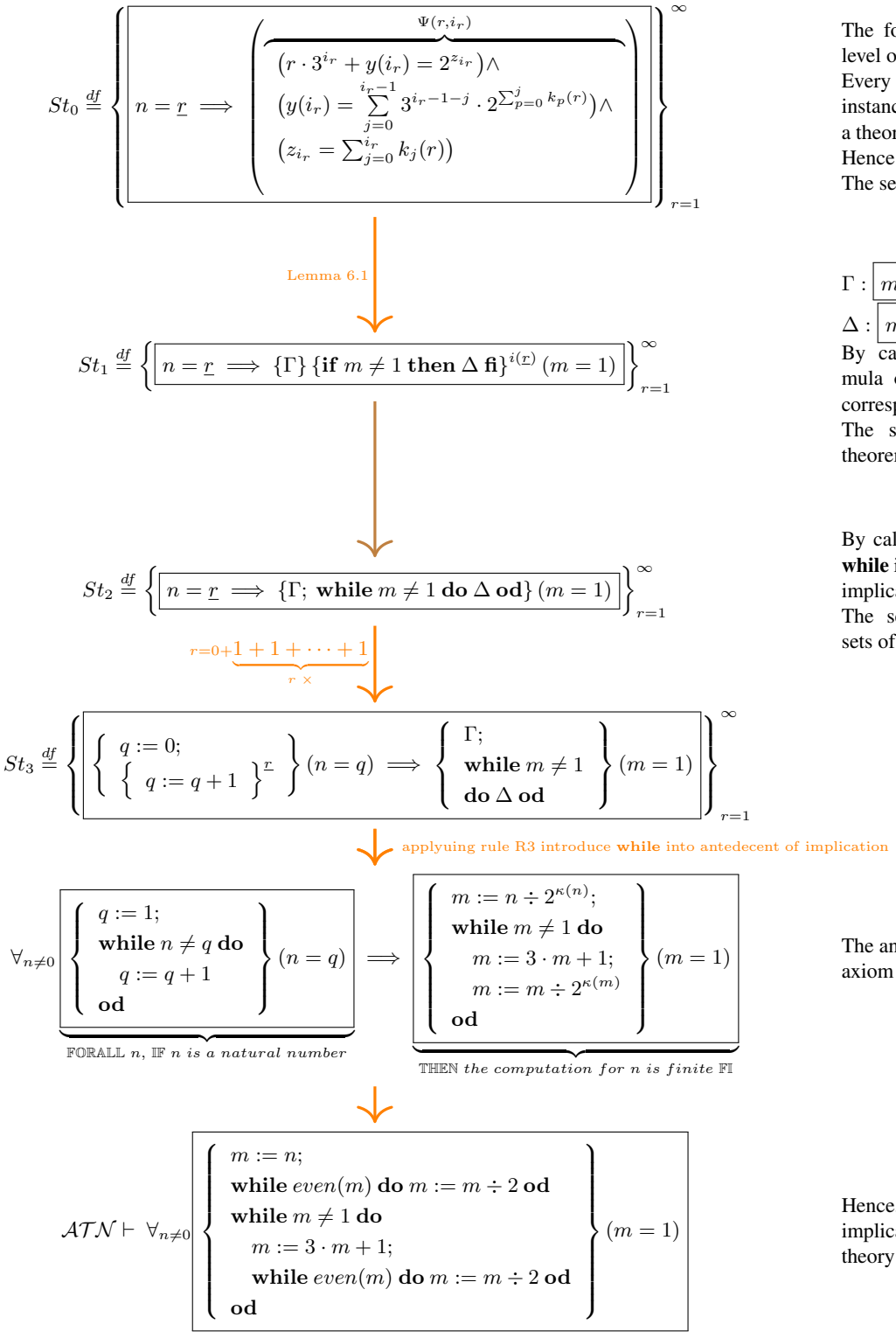
(*ii*) Any semantical property of an algorithm can be *expressed* by an appropriate algorithmic formula. Be it termination, correctness or other properties.

(*iii*) Algorithmic formulas enable to create complete, categorical *specifications* of data structures in the form of algorithmic theories.

(*iv*) Calculus of programs $\mathcal{AL}$ offers the *complete* set of tools for proving theorems of algorithmic theories.

Another contribution that this paper offers is an original way of proving theorems. Our proof of the Collatz conjecture is made in two stages. First, we prove that there are some recursive sets of formulas consisting of theorems of the $\mathcal{ATN}$ theory. Second, we reason on the sets of theorems much like one reasons on formulas and apply the infinitary rules of inference to achieve the goal.

There are many questions that remain open.



The formula $\Psi(r, i_r)$ says: i_r is the level of the number r in the tree $\mathcal{G}$. Every formula of the set St_0 is an instance of MAIN LEMMA, it is also a theorem of Presburger arithmetic. Hence it is theorem of $\mathcal{ATN}$ theory. The set St_0 is recursive.

Γ : $m := n \div 2^{\kappa(n)}$

Δ : $m := 3 \cdot m + 1;\ m := m \div 2^{\kappa(m)}$

By calculus of programs every formula of the set St_1 is equivalent to corresponding formula of the set St_0. The set St_1 is a recursive set of theorems.

By calculus of programs we introduce **while** instruction into the consequent of implication. The sets St_1, St_2, St_3 are recursive sets of theorems of $\mathcal{ATN}$ theory.

The antecedent of this implication is an axiom of the $\mathcal{ATN}$ theory.

Hence, the antecedent of the above implication is an axiom of the $\mathcal{ATN}$ theory and has been cut off.

Figure 11. Structure of the proof

7.1. Computational complexity

From the papers of M. Presburger [Pre29, Sta84] and D. Cooper [Coo72] one can infer an estimation of the pessimistic cost of a $3n+1$ computaion is $O(2^{2^{2^n}})$.
We have the following conjecture and a remark

Conjecture 3. For every natural number n its $3n+1$ computation consists of no more than $2n$ multiplications by 3 and no more than $3n$ divisions by 2.

Remark 7.1. One needs not to execute $3n+1$ computation. Just accept 1 as the *proven* result.

7.2. Principle of structural induction with infinitely many successors

Structural induction is a useful variant of the mathematical induction. Here we formulate its non-trivial version (IS).

Let $\alpha(x)$ be an algorithmic formula with a free variable x. Note, that when one replaces all the free occurrences of the variable x in the formula $\alpha(x)$ by a term τ then the resulting expression $\alpha(x/\tau)$ is a formula too.

Lemma 7.1. (Scheme of structural induction with infinitely many successors)
If
B) for every natural number $p \in N$, the formula $\alpha(x/2^p)$ is a theorem of the formalized, algorithmic theory $\mathcal{ATN}$,
and
K) for every odd, natural number x such that $x \bmod 3 \neq 0$ the following implication
$\alpha(x) \implies \forall_{i,p}\alpha(x/S_i(x)\cdot 2^p)$ is a theorem of the theory $\mathcal{ATN}$,
then
the formula $\forall_{n\in N}\alpha(n)$ is a theorem of the algorithmic theory $\mathcal{ATN}$
too.

Below, we present two variants of the scheme, one, with the iteration quantifiers.

$$\boxed{\begin{array}{l}\{i:=0\}\bigcap\{i:=i+1;\ x:=2^i\}\wedge\alpha(x)\wedge \\ \{x:=1\}\bigcap\{x:=x+2\}\,[(\alpha(x)\wedge 3\nmid x)\Rightarrow\{i:=0\}\bigcap\{i:=i+1\}\{p:=0\}\bigcap\{p:=p+1\}\{x:=S_i(x)\cdot 2^p\}\alpha(x)] \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \Rightarrow\{x:=1\}\bigcap\{x:=x+2\}\left(\{i:=0\}\bigcap\{i:=i+1\}\alpha(x\cdot 2^i)\right)\end{array}} \qquad \text{(IS)}$$

And another written with the classical quantifiers. In the latter case the axiom of reachability is needed.

$$\boxed{\forall_{i\in N}\alpha(x/2^i)\ \wedge\ \left[\forall_{Odd(x)}\ (\alpha(x)\wedge 3\nmid x)\implies\forall_i\forall_{p\in N}\ \alpha(x/S_i(x)\cdot 2^p)\right]\Rightarrow\forall_{n\in N}\ \alpha(n)} \qquad \text{(IS2)}$$

Proof:
The proof was already sketched in section 6. See also the figure 5.

□

Acknowlegments

Andrzej Szałas has shown to us the lacunes in our earlier proofs. Antek Ciaputa helped in calculations and drawing of Hotel Collatz. Hans Langmaack, Wiktor Dańko, Paweł Gburzyński and Marek Warpechowski sent a couple of useful comments.

8. References

[Coo72] D.C. Cooper Theorem Proving in Arithmetic without Multiplication.. Machine Intelligence , 7:91-99,1972.

[Eng93] Erwin Engeler Algorithmic Properties of Structures. World Scientific, Singapore, 1993.

[Grz13] Andrzej Grzegorczyk. An Outline of Mathematical Logic: Fundamental Results and Notions Explained with All Details. Springer, Netherlands, 2013.

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

[Kar64] Carol Karp. Languages with Expressions of Infinite Length. North Holland, 1964.

[Lag10] Jeffrey C. Lagarias, editor. The Ultimate Challenge: The 3x+1 Problem. American Mathematical Society, Providence R.I., 2010.

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

[MS21] G. Mirkowska and A. Salwicki. On Collatz theorem. http://lem12.uksw.edu.pl/images/2/27/On-Collatz-thm-11-10-21.pdf. [Online: accessed 17 December 2021], 2021.

[Opo78] Derek c. Oppen . "A $O(2^{2^{2^{cn}}})$ Upper Bound on the Complexity of Presburger Arithmetic. Journal of Computer and System Science, 16, 1978, pages 323–332

[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-wer Congres de Math. Pays Slav. pages 92–101,395, 1929.

[Sal23] Andrzej Salwicki. A new proof of Euclid's algorithm. https://arxiv.org/abs/2311.01474. [Online: accessed 19 December 2023], 2023.

[Sko34] Thoralf Skolem. Über die nicht-charakterisierbarkeit der zahlenreihe mittels endlich oder abzählbar unendlich vieler aussagen mit ausschließlich zahlenvariablen. Fundamenta Mathematicae, 23:150–161, 1934.

[Sta84] Ryan Stansifer. Presburger's Article on Integer Arithmetic: Remarks and Translation. Cornell University, Technical Report TR84-639, 1984. http://cs.fit.edu/~ryan/papers/presburger.pdf. [Online; accessed 6-August-2021].

[Tar34] Alfred Tarski. Bemerkung der Redaktion. Fundamenta Mathematicae, 23:161, 1934.

[TMR65] Alfred Tarski, Andrzej Mostowski, and Raphael M. Robinson. Undecidable Theories. North Holland, 1965.

9. Suplements

In this section, for the reader's convenience, we have collected some definitions, facts and statements. some useful theorems.

9.1. A structure with counterexamples

where Collatz computations may be of infinite length

Here we present some facts that are were discovered around 1929 by two students of Alfred Tarski: Mojżesz Presburger [Pre29, Sta84] and Stanisław Jaśkowski [Tar34]. These facts are less known to the IT community.
These facts may seem strange. The reader may doubt the importance of those facts. Yet, it is worth considering, non-standard data structures do exist, and this fact has ramifications. Strange as they seem, still it is worthwhile to be aware of their existence.

Now, we will expose the algebraic structure $\mathfrak{J}$, which is a model of the theory Ar, i.e. all axioms of theory Ar are true in the structure $\mathfrak{J}$. First we will describe this structure as mathematicians do, then we will write a class (i.e. a program module) implementing this structure.

Mathematical description of Jaśkowski's structure

$\mathfrak{J}$ is an algebraic structure

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

such that $M \subset \mathbb{C}$ is a set of complex numbers $k + \imath w$, i.e. of pairs $\langle k, w \rangle$, where element $k \in \mathbb{Z}$ is an integer, and element $w \in \mathbb{Q}^+$ is a rational, non-negative number $w \geq 0$ and the following requirements are satisfied:

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

(*ii*) $\underline{\mathbf{0}} \stackrel{df}{=} \langle 0 + \imath 0 \rangle$,

(*iii*) $\underline{\mathbf{1}} \stackrel{df}{=} \langle 1. + \imath 0 \rangle$,

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

$$(k + \imath w) \oplus (k' + \imath w') \stackrel{df}{=} (k + k') + \imath(w + w').$$

(*v*) the predicate = denotes as usual identity relation.

Lemma 9.1. The algebraic structure $\mathfrak{J}$ is a model of first-order arithmetic of addition of natural numbers $\mathcal{T}$.

The reader may check that every axiom of the $\mathcal{T}$ theory (see definition9.2, p.31), is a sentence true in the structure $\mathfrak{J}$, cf. next subsection 9.2.

The substructure $\mathfrak{N} \subset \mathfrak{J}$ composed of only those elements for which $w = 0$ is also a model of the theory $\mathcal{T}$.
It is easy to remark that elements of the form $\langle k, 0 \rangle$ may be identified with natural numbers k, $k \in N$. Have a look at table 2

The elements of the structure $\mathfrak{N}$ are called *reachable*, for they enjoy the following algorithmic property

$$\forall_{n \in N} \{y := \mathbf{0}; \mathbf{while}\ y \neq n\ \mathbf{do}\ y := y + \mathbf{1}\ \mathbf{od}\}(y = n)$$

The structure $\mathfrak{J}$ is not a model of the $\mathcal{ATN}$, algorithmic theory of natural numbers, cf . subsection 9.4. Elements of the structure $\langle k, w \rangle$. such as $w \neq \mathbf{0}$ are *unreachable*. i.e. for each element $x_0 = \langle k, w \rangle$ such that $w \neq 0$ the following condition holds

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

The substructure $\mathfrak{N} \subset \mathfrak{J}$ composed of only those elements for which $w = 0$ is a model of the theory $\mathcal{ATN}$ c.f. subsection 9.4. The elements of the structure $\mathfrak{N}$ are called *reachable*. A theorem of the foundations of mathematics states:

Lemma 9.2. The structures $\mathfrak{N}$ and $\mathfrak{J}$ are not isomorphic.

For the proof see [Grz71], p. 256. As we will see in a moment, this fact is also important for IT specialists.

An attempt to visualize structure $\mathfrak{M}$ is presented in the form of table 2. The universe of the structure $\mathfrak{J}$ decomposes onto two disjoint subsets (one green and one red). Every element of the form $\langle k, 0 \rangle$ (in this case $k > 0$) represents the natural number k. Such elements are called *reachable* ones. Note,

Definition 9.1. An element n is a standard natural number (i.e. is *reachable*) iff the program of adding ones to initial zero terminates

$$n \in N \stackrel{df}{\Leftrightarrow} \{q := \mathbf{0};\ \mathbf{while}\ q \neq n\ \mathbf{do}\ q := q + \mathbf{1}\ \mathbf{od}\}(q = n)$$

or, equivalently

$$n \in N \stackrel{df}{\Leftrightarrow} \{q := \mathbf{0}\} \bigcup \{\mathbf{if}\ n \neq q\ \mathbf{then}\ q := q + \mathbf{1}\ \mathbf{fi}\}(q = n)$$

Table 2. Model $\mathfrak{J}$ of Presburger arithmetic consists of complex numbers $a + \imath\, b$ where $b \in Q^+$ and $a \in Z$, additional condition: $b = 0 \Rightarrow a \geq 0$. Definition of order $n > m \stackrel{df}{\equiv} \exists_{u \neq 0}\, m + u = n$. Invention of S. Jaśkowski (1929).

STANDARD (*reachable*) elements	*Unreachable* (INFINITE) elements						
				$\cdots$			
	$-\infty \cdots$	$-11 + \imath 2$	$-10 + \imath 2$	$\cdots$	$0 + \imath 2$	$1 + \imath 2$	$2 + \imath 2$ $\cdots \infty$
				$\cdots$			
	$-\infty \cdots$	$-11 + \imath \frac{53}{47}$	$-10 + \imath \frac{53}{47}$	$\cdots$	$0 + \imath \frac{53}{47}$	$1 + \imath \frac{53}{47}$	$2 + \imath \frac{53}{47}$ $\cdots \infty$
				$\cdots$			
	$-\infty \cdots$	$-11 + \imath \frac{28}{49}$	$-10 + \imath \frac{28}{49}$	$\cdots$	$0 + \imath \frac{28}{49}$	$1 + \imath \frac{28}{49}$	$2 + \imath \frac{28}{49}$ $\cdots \infty$
				$\cdots$			
	$-\infty \cdots$	$-11 + \imath \frac{3}{47}$	$-10 + \imath \frac{3}{47}$	$\cdots$	$0 + \imath \frac{3}{47}$	$1 + \imath \frac{3}{47}$	$2 + \imath \frac{3}{47}$ $\cdots \infty$
				$\cdots$			
0 1 2 $\cdots$ 101 $\cdots$ ∞							

Note that the subset that consists of all non-reachable elements is well separated from the subset of reachable elements. Namely, every reachable natural number is less that any unreachable one. Moreover, there is no least element in the set of unreachable elements. I.e. the principle of minimum does not hold in the structure $\mathfrak{M}$.
Moreover, for every element n its computation contains either only standard, reachable numbers or is composed of only unreachable elements. This remark will be of use in our proof.

Remark 9.1. For every element n the whole Collatz computation is either in green or in reed quadrant of the table 2.

Elements of the structure $\mathfrak{M}$ are ordered as usual

$$\forall_{x,y}\; x < y \stackrel{df}{=} \exists_{z \neq \mathbf{0}}\; x + z = y.$$

Therefore, each reachable element is smaller than every unreachable element.
The order defined in this way is the lexical order. (Given two elements p and q, the element lying higher is bigger, if both are of the same height then the element lying on the right is bigger.)
The order type is $\omega + (\omega^* + \omega) \cdot \eta$

Remark 9.2. The subset of unreachable elements (red ones on the table 2) does not obey the principle of minimum.

Definition in a programming language

Perhaps you have already noticed that the $\mathfrak{M}$ is a computable structure. The following is a class that implements the structure $\mathfrak{M}$. The implementation uses the integer type, we do not introduce rational numbers 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 with respect to the axioms of Presburger arithmetic $\mathcal{AP}$ (c.f. subsection 9.2) treated as a specification of a class (i.e. a module of program).

Lemma 9.3. The structure $\mathfrak{E} = \langle E, add, zero, one, eq\rangle$ composed of the set $E = \{o \text{ object} : o\ in Elm\}$ of objects of class Elm with the *add* operation is a model of the $\mathcal{AP}$ theory,

$$\mathfrak{E} \models \mathcal{AP}$$

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; ...
  unit 3xp1: function(n: Elm): Elm; . . . result:=add(n,add(n,add(n,one))); . . . end 3xp1;
begin
  n:= new Elm(8,1,2);
  Cl:  while  not eq(n,one)  do
         if  odd(n) then
           n:=3xp1(n)  else n:= div2(n)     (* a version of algorithm Cl that uses class Elm *)
       fi
       od
end block;
```

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

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

Note, the computation of algorithm Gr for the same argument, looks simpler

$$\langle 8, \frac{1}{2}\rangle, \langle 4, \frac{1}{4}\rangle, \langle 2, \frac{1}{8}\rangle, \langle 1, \frac{1}{16}\rangle, \langle 1, \frac{3}{64}\rangle, \langle 1, \frac{9}{256}\rangle, \cdots$$

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 $\mathcal{AP}$ theory with the non-obvious presence of unreachable elements in it.

Corollary 9.2. The halting property of the Collatz algorithm cannot be proved from the axioms of the $\mathcal{T}$ theory, nor from the axioms of $\mathcal{AP}$ theory.

9.2. Presburger's arithmetic

Presburger's arithmetic is another name of elementary theory of natural numbers with addition.
We shall consider the following theory , cf. [Pre29],[Grz71] p. 239 and following ones.

Definition 9.2. 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 $(\tau_1 + \tau_2)$ is a term too.
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 to the 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 also formulas, 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 $\Phi(x)$ may be replaced by any formula. The result is an axiom of theory This is the induction scheme.
We augment the set of axioms adding four axioms that define a coiple of useful notions.

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

The theory $\mathcal{T}'$ obtained in this way is a conservative extension of theory $\mathcal{T}$.

Below we present another theory $\mathcal{AP}$ c.f. [Pre29], we shall use two facts: 1) theory $\mathcal{AP}$ is complete and hence is decidable, 2) both theories are elementarily equivalent.

Definition 9.3. Theory $\mathcal{AP} = \langle \mathcal{L}, \mathcal{C}, AxP \rangle$ 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 and set of formulas F. The set of terms T is the least set of expressions that contains the set of variables V and the expression 0 and closed with respect to the following two rules: 1) if two expressions τ_1 and τ_2 are terms, then the expression $(\tau_1 + \tau_2)$ is also a term, 2) if the expression τ is a term, then the expression $S(\tau)$ is also a term.

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

AxP The set of axioms of the $\mathcal{AP}$ theory is listed below.

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

. . . . . .

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

. . .

Let us recall a couple of useful theorems

F1. Theory $\mathcal{T}$ is elementarily equivalent to the theory $\mathcal{AP}$.[Pre29] [Sta84]

F2. Theory $\mathcal{AP}$ is decidable. [Pre29].

F3. The computational complexity of theory $\mathcal{AP}$, is double exponential $O(2^{2^n})$ this result belongs to Fisher and Rabin, see [**?**].

F4. Theories $\mathcal{T}$ and $\mathcal{AP}$ have non-standard model, see section 9.1, p. 27.

Now, we shall prove a couple of useful theorems of theory $\mathcal{T}$.

First, 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. 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

Definition 9.4. Two functions are defined $P2$ (of oneargument) and $P3$ (of two-arguments).

$P2(0) \stackrel{df}{=} 1$	$P3(y,0) \stackrel{df}{=} y$
$P2(x+1) \stackrel{df}{=} P2(x) + P2(x)$	$P3(y,x+1) \stackrel{df}{=} P3(y,x) + P3(y,x) + P3(y,x)$

Lemma 9.4. The definitions given above are correct, i.e. 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 $\mathcal{T}$.

An easy proof goes by induction with respect to the value of variable x.

Therefore the theory $\mathcal{T}'$ is an inessential extension of the theory $\mathcal{T}$. We can write 2^x instead of $P2(x)$ and $y \cdot 3^x$ instead of $P3(y,x)$.
In the proof of the lemma 9.5 , below, we shall use the definition of the order relation

$$a < b \stackrel{df}{=} \exists_{c \neq 0}\, a + c = b.$$

Making use of the definition of functions $P2$ 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 9.5. The following sentence is a theorem of the theory $\mathcal{T}$ enriched by the definitions of $P2$ and $P3$ functions.

$$\forall_n \exists_{x,y,z} P3(n,x) + y = P2(z) \tag{45}$$

Proof:
We begin proving by induction that $\mathcal{T} \vdash \forall_n\, n < 2^n$. It is easy to see that $\mathcal{T} \vdash 0 < P2(0)$. We shall prove that $\mathcal{T} \vdash \forall_n (n < P2(n) \implies (n+1 < P2(n+1))$. Inequality $n+1 < P2(n+1)$ follows from the two following inequalities $\mathcal{T} \vdash n < P2(n)$ and $\mathcal{T} \vdash 1 < P2(n)$. Hence the formula $n+1 < P2(n) + P2(n))$ is a theorem of theory $\mathcal{T}$. By definition $P2(n) + P2(n) = P2(n+1)$.
In the similar manner, we can prove the formula $\mathcal{T} \vdash \forall_n \forall_x\, P3(n,x) < P2(n+x+x)$
As a consequence we have $\mathcal{T} \vdash \forall_n \exists_{x,y,z}\, P3(n,x) + y = P2(z)$. □

Remark that the equation 45 can be read as

$$\forall_n \exists_{x,y,z} n \cdot 3^x + y = 2^z \tag{46}$$

9.3. An introduction to the calculus of programs $\mathcal{AL}$

For the convenience of the reader we cite the axioms and inference rules of calculus of programs i.e. algorithmic logic $\mathcal{AL}$.

Note. *Every axiom of algorihmic logic is a tautology.*
Every inference rule of $\mathcal{AL}$ is sound. [MS87]

Axioms

axioms of propositional calculus

$Ax_1)\ ((\alpha \Rightarrow \beta) \Rightarrow ((\beta \Rightarrow \delta) \Rightarrow (\alpha \Rightarrow \delta)))$
$Ax_2)\ (\alpha \Rightarrow (\alpha \vee \beta))$
$Ax_3)(\beta \Rightarrow (\alpha \vee \beta))$
$Ax_4)((\alpha \Rightarrow \delta)\ \Rightarrow ((\beta \Rightarrow \delta)\ \Rightarrow\ ((\alpha \vee \beta) \Rightarrow \delta)))$
$Ax_5)((\alpha \wedge \beta) \Rightarrow \alpha)$
$Ax_6)((\alpha \wedge \beta) \Rightarrow \beta)$
$Ax_7)((\delta \Rightarrow \alpha) \Rightarrow ((\delta \Rightarrow \beta) \Rightarrow (\delta \Rightarrow (\alpha \wedge \beta))))$
$Ax_8)((\alpha \Rightarrow (\beta \Rightarrow \delta)) \Leftrightarrow ((\alpha \wedge \beta) \Rightarrow \delta))$
$Ax_9)((\alpha \wedge \neg\alpha) \Rightarrow \beta)$
$Ax_{10})((\alpha \Rightarrow (\alpha \wedge \neg\alpha)) \Rightarrow \neg\alpha)$
$Ax_{11})(\alpha \vee \neg\alpha)$

axioms of predicate calculus

$Ax_{12})\ ((\forall x)\alpha(x) \Rightarrow \alpha(x/\tau)))$
$Ax_{13})\ (\forall x)\alpha(x) \Leftrightarrow \neg(\exists x)\neg\alpha(x)$

axioms of calculus of programs

$Ax_{14})\ K((\exists x)\alpha(x)) \Leftrightarrow (\exists y)(K\alpha(x/y))$
$Ax_{15})\ K(\alpha \vee \beta) \Leftrightarrow ((K\alpha) \vee (K\beta))$
$Ax_{16})\ K(\alpha \wedge \beta) \Leftrightarrow ((K\alpha) \wedge (K\beta))$
$Ax_{17})\ K(\neg\alpha) \Rightarrow \neg(K\alpha)$
$Ax_{18})\ ((x := \tau)\gamma \Leftrightarrow (\gamma(x/\tau) \wedge (x := \tau)true))$
$Ax_{19})\ \mathbf{begin}\ K; M\ \mathbf{end}\alpha \Leftrightarrow K(M\ \alpha)$
$Ax_{20})\ \mathbf{if}\ \gamma\ \mathbf{then}\ K\ \mathbf{else}\ M\ \mathbf{fi}\ \alpha \Leftrightarrow ((\gamma \wedge K\alpha) \vee (\neg M\alpha))$
$Ax_{21})\ \mathbf{while}\ \gamma\ \mathbf{do}\ K\ \mathbf{od}\ \alpha \Leftrightarrow$
$\quad (\neg\gamma \wedge \alpha) \vee \big(\gamma \wedge K\ \mathbf{while}\ \gamma\ \mathbf{do}\ K\ \mathbf{od}(\neg\gamma \wedge \alpha))\big)$
$Ax_{22})\ \bigcap K\alpha \Leftrightarrow (\alpha \wedge (K \bigcap K\alpha))$
$Ax_{23})\ \bigcup K\alpha \equiv (\alpha \vee (K \bigcup K\alpha))$

Inference rules

propositional calculus

$$R_1\ \frac{\alpha, (\alpha \Rightarrow \beta)}{\beta}$$

predicate calculus

$$R_6\ \frac{(\alpha(x)\ \Rightarrow\ \beta)}{((\exists x)\alpha(x)\ \Rightarrow\ \beta)} \qquad R_7\ \frac{(\beta\ \Rightarrow\ \alpha(x)}{(\beta \Rightarrow (\forall x)\alpha(x))}$$

calculus of programs AL

$$R_2\ \frac{(\alpha \Rightarrow \beta)}{(K\alpha \Rightarrow K\beta)} \qquad R_3\ \frac{\{s(\{\mathbf{if}\ \gamma\ \mathbf{then}\ K\ \mathbf{fi}\}^i\ \alpha) \Rightarrow \beta\}_{i\in N}}{s(\{\mathbf{while}\ \gamma\ \mathbf{do}\ K\ \mathbf{od}\}\ \alpha) \Rightarrow \beta}$$

$$R_4\ \frac{\{(K^i\alpha \Rightarrow \beta)\}_{i\in N}}{(\bigcup K\alpha \Rightarrow \beta)} \qquad R_5\ \frac{(\alpha \Rightarrow K^i\beta)_{i\in N}}{(\alpha \Rightarrow \bigcap K\beta)}$$

In the rules R_6 and R_7, it is assumed that x is a variable which is not free in β, i.e. $x \notin FV(\beta)$. The rules are known as the rule for introducing an existential quantifier into the antecedent of an implication and the rule for introducing a universal quantifier into the successor of an implication. The rules R_4 and R_5 are algorithmic counterparts of rules R_6 and R_7. They are of a different character, however, since their sets of premises are infinite. The rule R_3 for introducing a **while** into the antecedent of an implication of a similar nature. These three rules are called ω-rules. The rule R_1 is known as *modus ponens*, or the *cut*-rule. In all the above schemes of axioms and inference rules, α, β, δ are arbitrary formulas, γ and γ' are arbitrary open formulas, τ is an arbitrary term, s is a finite sequence of assignment instructions, and K and M are arbitrary programs.

Thesis 9.1. (*theorem on completeness of the calculus* $\mathcal{AL}$)
. Let $\mathcal{T} = \langle \mathcal{L}, \mathcal{C}, \mathcal{A}x \rangle$ be a consistent algorithmic theory, let $\alpha \in \mathcal{L}$ be a formula. The following conditions are equivalent

(*i*) Formula α is a theorem of the theory T, $\alpha \in \mathcal{C}(\mathcal{A}x)$,

(*ii*) Formula α is valid in every model of the theory T, $\mathcal{A}x \models \alpha$.

The proof may be found in [MS87] Theorem III.2.5 p.94. □

9.4. An introduction to the algorithmic theory of natural numbers $\mathcal{ATN}$

The language of algorithmic theory of natural numbers $\mathcal{ATN}$ is very simple. Its alphabet contains one constant 0 *zero* , one one-argument functor s and predicate = of equality. Axioms of $\mathcal{ATN}$ were presented in the book [MS87]

$$
\begin{array}{lll}
A_1) & \forall x \{q := 0;\ \textbf{while}\ q \neq x\ \textbf{do}\ q := s(q)\ \textbf{od}\}(q = x) & (R) \\
A_2) & \forall x\ s(x) \neq 0 & (N) \\
A_3) & \forall x \forall y\, s(x) = s(y) \implies x = y & (J) \\
A_4) & \forall x\, \forall y \left\{ \begin{array}{l} q := 0; w := x; \\ \textbf{while}\ q \neq y\ \textbf{do} \quad q := s(q)\ ; w := s(w) \textbf{od} \end{array} \right\} (x + y = w) & (D)
\end{array}
$$

The termination property of the program in A_4 is a theorem of $\mathcal{ATN}$ theory as well as the formulas $x + 0 = x$ and $x + s(y) = s(x + y)$.
Note, the following formulas are not theorems of first-order arithmetic of natural numbers (Peano's theory).

$$\mathcal{ATN} \vdash \exists_x\, \alpha(x) \Leftrightarrow \{x := 0\} \bigcup \{x := x + 1\} \alpha(x) \tag{47}$$

$$\mathcal{ATN} \vdash \forall_x\, \alpha(x) \Leftrightarrow \{x := 0\} \bigcap \{x := x + 1\} \alpha(x) \tag{48}$$

The axiom of of Archimedes – asserts the Archimedean property of natural numbers

$$\mathcal{ATN} \vdash 0 < x < y \implies \{a := x;\ \textbf{while}\ a < y\ \textbf{do}\ a := a + x\ \textbf{od}\}(a \geq y) \tag{49}$$

Scheme of induction – let $\alpha(x)$ be an algorithmic formla

$$\mathcal{ATN} \vdash \Big(\alpha(x/0) \wedge \forall_x \big(\alpha(x) \implies \alpha(x/s(x))\big)\Big) \implies \forall_x \alpha(x) \tag{50}$$

Correctness of Euclid's algorithm

$$\mathcal{ATN} \vdash \left(\begin{array}{l} n_0 > 0 \wedge \\ m_0 > 0 \end{array} \right) \implies \left\{ \begin{array}{l} n := n_0;\ m := m_0; \\ \textbf{while}\, n \neq m\, \textbf{do} \\ \quad \textbf{if}\, n > m\, \textbf{then}\ n := n \dot{-} m \\ \quad\quad \textbf{else}\ m := m \dot{-} n \\ \quad \textbf{fi} \\ \textbf{od} \end{array} \right\} (n = gcd(n_0, m_0)) \tag{51}$$

The theory $\mathcal{ATN}$ enjoys an important property of categoricity.

Thesis 9.2. (*meta*-theorem on categoricity of $\mathcal{ATN}$ theory)
Every model $\mathfrak{A}$ of the algorithmic theory of natural numbers is isomorphic to the structure $\mathfrak{N}$.

Compare this theorem with the fact that first-order theory of natural numbers has non-standard model, c.f. subsection 9.1.