Collatz conjecture becomes theorem

$3n+1$ **computations hide a paradox**

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] mimuw·edu.pl

Keywords: 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.
We are presenting the paradox, i.e. two theses T1 and T2 that contradict each other. Third thesis T3 solves the problem.

T1. We show that the Collatz conjecture "For every natural number n, the $3n+1$ computation is finite." is a *semantically valid statement*.

The sufficient and necessary criterion $\varphi(n)$ for termination of $3n+1$ computation is given 11, c.f. page 10.
We prove that, every instance $\varphi(n/r)$ of the criterion where the variable n is replaced by any natural number $r \neq 0$, is a *theorem* of Peano's arithmetic, Hence, the set $\{\varphi(n/r)\}_{r=0}^{\infty} \subset Th(\mathcal{PA})$ is a recursive subset of the set of theorems.

T2. Paradoxically, the Collatz conjecture itself, *is not a theorem* of number theory (Peano's arithmetic or a similar elementary theory).

It is so because, 1°the formula $\forall_n \varphi(n)$, obtained by putting the general quantifier $\forall_n$ in front of formula $\varphi(n)$, may obtain the value $\mathbb{F}$ - false, in a non-standard model of Peano's arithmetic and 2 °there is no way to bound the classical quantifier to the set of standard, reachable natural numbers.
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 11 is written as an algorithmic formula, c.f. (MainThm).

T3. 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 $\mathcal{ATN}$. Hence, every formula of the set St_3 has a proof. Making use of the infinitary inference rule R_4 (c.f. page 34) to the infinite set St_3 of premises we conclude the proof of the Main theorem.

$$\mathcal{ATN} \vdash \forall_{n>0} \left(\underbrace{\left\{ \begin{array}{l} q \leftarrow 1; \\ \textbf{while } n \neq q \textbf{ do} \\ \quad q \leftarrow q+1 \\ \textbf{od} \end{array} \right\} (n=q)}_{\mathbb{IF}\ n\ is\ a\ natural\ number} \implies \underbrace{\left\{ \begin{array}{l} m \leftarrow \rho(n); \\ \textbf{while } m \neq 1 \textbf{ do} \\ \quad m \leftarrow \rho(3m+1) \\ \textbf{od} \end{array} \right\} (m=1)}_{\mathbb{THEN}\ the\ computation\ for\ n\ is\ finite\ \mathbb{FI}} \right) \qquad \text{(MainThm)}$$

DEF. The function ρ is defined as $\rho(n) = (2j+1) \overset{df}{\Leftrightarrow} \exists_i \exists_j\, n = 2^i \cdot (2j+1)$.

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

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

seems always reach the value 1 He formulated the following conjecture

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

COMMENT. In the theory of recursive functions, one would write down this problem as follows,
define the function $f_C \colon N \to N$

$$f_C(n) = \begin{cases} f_C(n \div 2) & \text{when } n \text{ is even} \\ f_C(3n+1) & \text{when } n \text{ is odd and , } n \neq 1 \\ 1 & \text{when } n = 1 \end{cases}$$

and ask, whether for every natural number r, the value of the term $f_C(r)$ is well defined.

1.1. 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].

- By the Kleene's theorem [Kle36], the function defined bythe recurrence rec1 is calculated by the following program Cl

```
while n ≠ 1 do
   if even(n) then n ← n div 2  else n ← 3n+1 fi
od
```
(Cl)

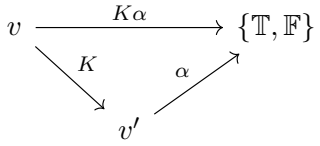
- Erwin Engeler [Eng93] in 1967 remarked that the halting property of program can be expressed by an infinite disjunction of quantifier free formulas.
  Next, he proved that algorithmic properties of programs executed in the structure of ordered real numbers are provable from the axioms of ordered, fields 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. In addition to the classical quantifiers $\forall, \exists$, we consider the iteration quantifiers $\bigcup, \bigcap$.
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].

1.2. 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							**Analysis**
m	**x**	**y**	**z**	**k**	$\mathbf{n3^x + y}$	$m \cdot 2^z$	For each i^{th}-row $0 \le i \le 6$
19	0	0	0	0	19	19	$k_{i+1} = \kappa(3m_i + 1)$
29	1	1	1	1	58	58	$m_{i+1} = (3m_i + 1) \div 2^{k_{i+1}}$
11	2	5	4	3	176	176	$x_i = i$ AND $z_i = \sum_{l=0}^{i} k_l$
17	3	31	5	1	544	544	ANND $y_i = \sum_{j=0}^{x_i - 1} 3^{x_i - 1 - j} \cdot 2^{z_j}$
13	4	125	7	2	1664	1664	invariants: $1^\circ\ n \cdot 3^{x_i} + y_i = m_i \cdot 2^{z_i}$
5	5	503	10	3	5120	5120	$2^\circ\ n \cdot \prod_{j=0}^{i-1}(3 + \frac{1}{m_j}) = m_i \cdot 2^{z_i}$
1	6	2533	14	4	16384	16384	two halting conditions: $1^\circ\ \exists_i m_i = 1$
1	7	23983	16	2	2^{16}	2^{16}	$2^\circ\ \exists_{i \in N}\ n \cdot 3^{x_i} + y_i = 2^{z_i}$
...		...	...		...	...	

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 approppriate, error-free triple.

1.3. 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*.

Theorem 1.1. 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.

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

(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.

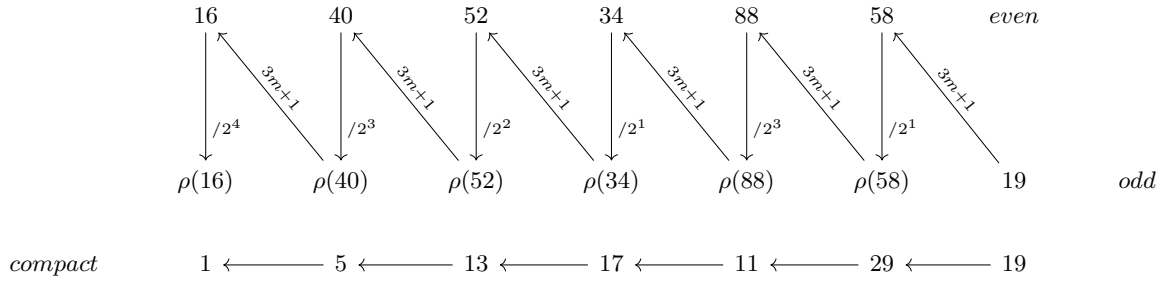


Figure 1. An example of compact computation

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 $m_{i+1} = \rho(3m_i + 1)$.

2. Collatz tree and halting criterion of $3n + 1$ computations

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 the Collatz tree.

Conjecture 1. The Collatz tree contains all natural numbers.

2.1. Halting criterion 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 showing the algorithm (Gr) equivalent to the algorithm Cl.
Next, we consider three algorithms $Gr1, Gr2, Gr3$ that are successive extensions of the eqGr algorithm.

Invariant of computation. The algorithm $Gr2$ will allow to formulate an *invariant* of $3n + 1$ computations.
The algorithm $Gr3$ will help to construct the necessary and sufficient condition for the termination of computations.

Lemma 2.1. 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)

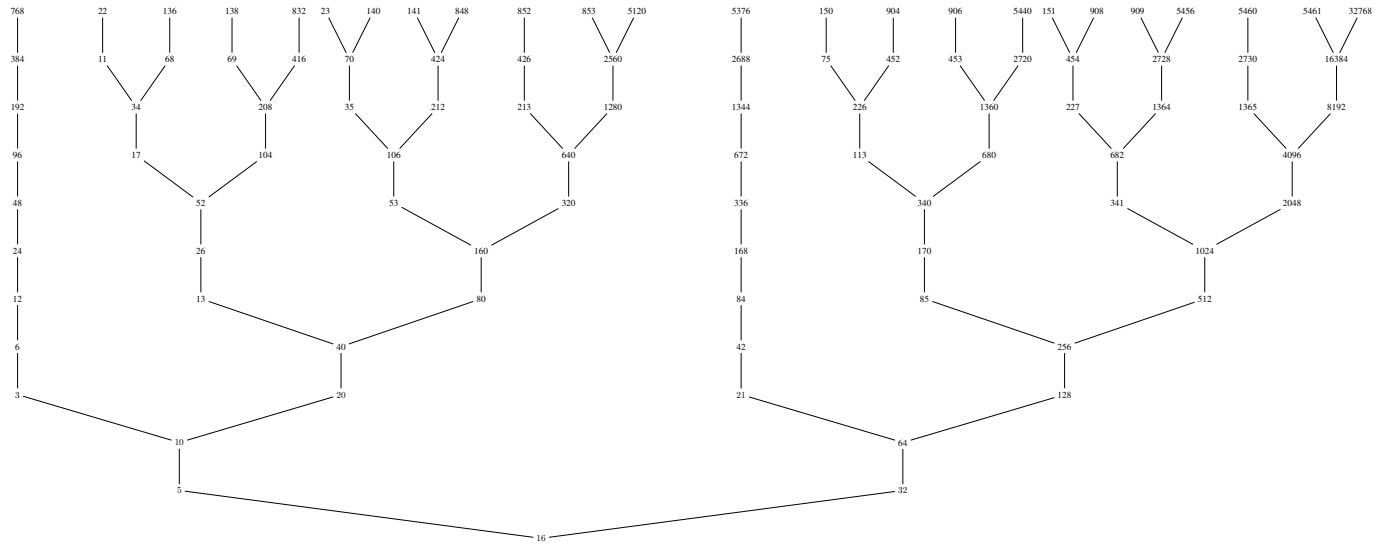


Figure 2. A fragment of Collatz tree

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?

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.

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

The definitions of the functions κ and ρ 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. □

Corollary 2.1. Every halting condition of the program Gr is also a halting condition for the program Cl.

Counting the operations of multiplication by 3 and division by 2. Next, we present the algorithm $Gr1$, an extension of algorithm Gr.

$$\begin{array}{|l} \textbf{var } n, aux, i : integer;\ k, m : \textbf{arrayof } integer; \\ \quad \Gamma_1 : \boxed{\ i := 0;\ k_i := \kappa(n);\ m_i := \rho(n);\ } \\ \textbf{while } m_i \neq 1 \textbf{ do} \\ \quad \Delta_1 : \boxed{\begin{array}{l} aux := 3 * m_i + 1;\ k_{i+1} := \kappa(aux); \\ m_{i+1} := \rho(aux);\ i := i+1 \end{array}} \\ \textbf{od} \end{array} \qquad \text{(Gr1)}$$

Lemma 2.2. 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, $m \in D$, initializes a new branch in the Collatz tree. 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 called the x-th *stratum* pf the set N of natural numbers.

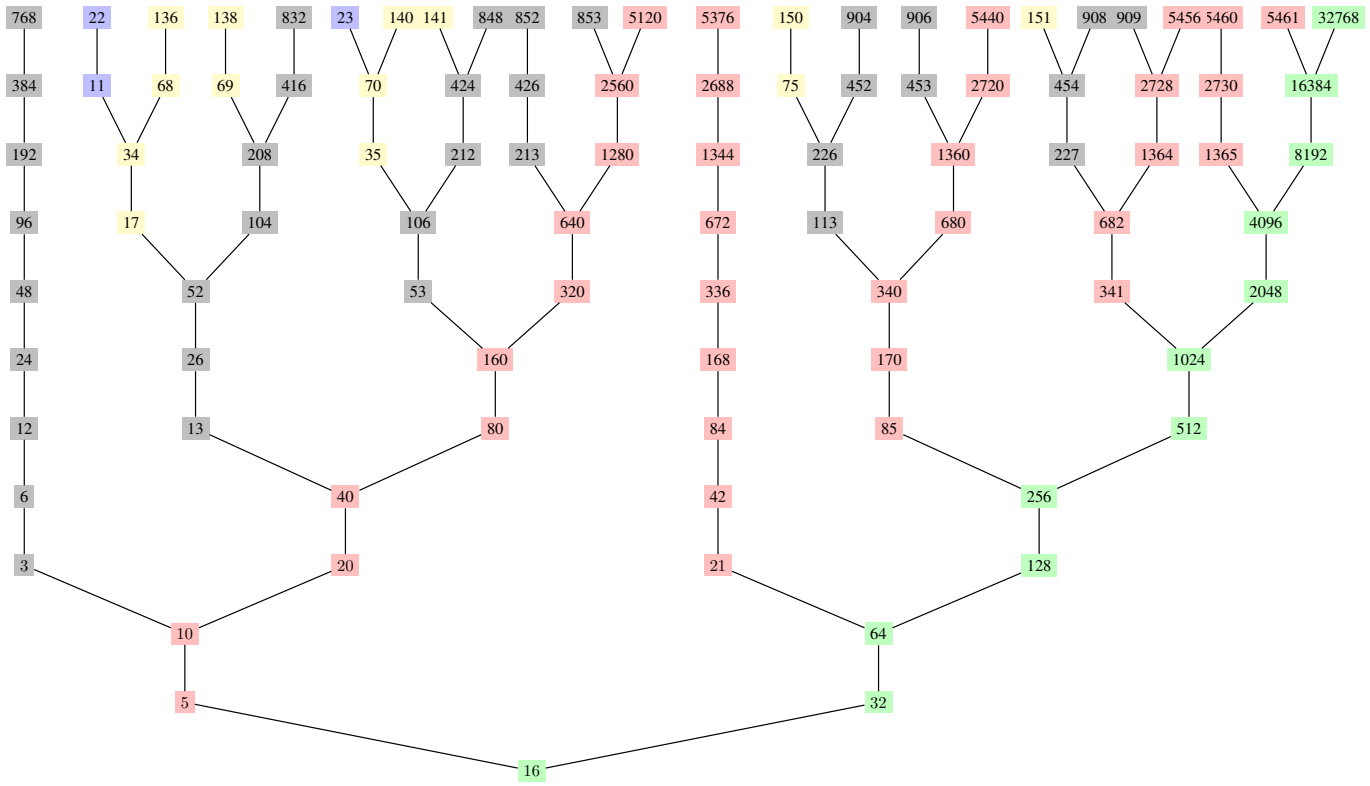


Figure 3. Strata $W_0 - -W_4$ of Collatz tree

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 post-condition $s \in W_i$.

Lemma 2.3. If the algorithm $Gr1$ halts for a number n then it computes the number i of stratum W_i that contains the number n,

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

Invariant of computation. Next, we present another algorithm Gr2 and a lemma 2.4.

var $n, aux, i, z, y : integer;\ k, m :$ **arrayof** $integer;$
$\Gamma_2 :$ $\boxed{i, y := 0;\ z, k_i := \kappa(n);\ m_i := \rho(n);}$
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 2.4. Algorithm $Gr2$ has the following properties:
(i) Both algorithms $Gr1$ and $Gr2$ are equivalent with respect to the halting property.
(ii) Formula $\varphi :$ $\boxed{n \cdot 3^i + y = m_i \cdot 2^z}$ 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:
One should conceive the formula φ as an abbreviation of an lengthy and obscure expression. Proofs of formulas (2), (3) are easy, it suffices to apply the axiom of assignment instruction Ax_{18}, see subsection 8.4. □

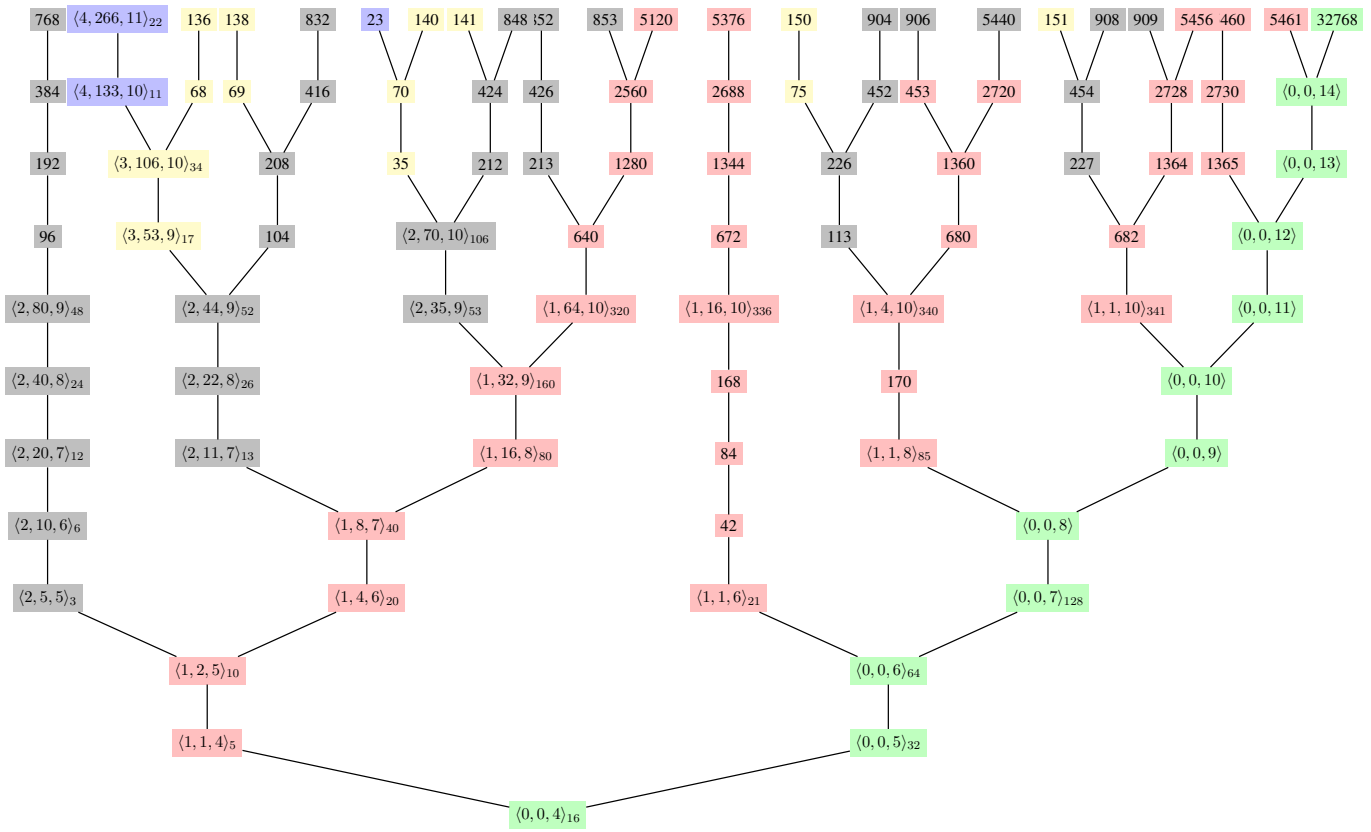


Figure 4. Tree of triples (strata 4 – 15)

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

$$\begin{array}{l}\textbf{var } n, aux, i : integer;\ k, m, x, y, z : \textbf{arrayof } integer;\\ \quad \Gamma_3 : \boxed{i, y_i := 0;\ z_i, k_i := \kappa(n);\ m_i := \rho(n);}\\ \textbf{while } n \cdot 3^i + y_i \neq 2^{z_i} \textbf{ do}\\ \quad \Delta_3 : \left|\begin{array}{l} aux := 3 * m_i + 1;\ k_{i+1} := \kappa(aux);\\ y_{i+1} := 3 * y_i + 2^{z_i};\ z_{i+1} := z_i + k_{i+1};\\ m_{i+1} := \rho(aux);\ i := i + 1; x_i := i\end{array}\right.\\ \textbf{od}\end{array} \tag{Gr3}$$

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

Lemma 2.5. 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

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

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

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

(*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 2.1. We can say *informally* that the algorithm $Gr3$ performs as follow

$i := 0;$
while $n \notin W_i$ **do** $i := i + 1$ **od**

Let us note an interesting information on route in graph

Corollary 2.2. 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 2.6. (*The sufficient and necessary criterion of 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 sentence 10 is valid in $\mathfrak{N}$

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

(*iii*) the following sentence 11 is valid in $\mathfrak{N}$

$$\{x := 0\} \bigcup \{x := x+1\} \left((r\cdot 3^x + y = 2^z) \wedge \left(\begin{array}{l} \left(y = \sum\limits_{j=0}^{x-1} 3^{x-1-j}\cdot 2^{\sum_{p=0}^{j} k_p} \right) \wedge \left(z = \sum\limits_{p=0}^{x} k_p \right) \wedge \\ \Big(k_0 = \kappa(r) \wedge m_0 = \rho(r) \Big) \wedge \\ \bigwedge\limits_{l=0}^{x-1} \Big(k_{l+1} = \kappa(3m_l+1) \wedge m_{l+1} = \rho(3m_l+1) \Big) \end{array} \right) \right). \tag{11}$$

Proof:
The proof is left to the reader. □

Perhaps you have noticed the difference between formulas 10 and 11.
Here we are offering a couple of *informal* explanations.
The formula 11 is the correct criterion for the finiteness of 3n+1 computations. The logical value of it says, *informally*, there exists x such that $x = 0 \vee x = 1 \vee x = 2 \vee$ ... and the conjunction contained in big parentheses is satisfied by x. On the other hand, the classical quantifier $\forall$ does not provide such comfort. It does not exclude the possibility that the value of x is non-standard. The formula 10 should only appear together with the reachability axiom (R) in the subsection 8.4.

The question: *is there a natural number n such that its computation is infinite?* arises in a natural way. The following lemma 2.7 brings a partial answer to this questions.

Lemma 2.7. Let $n \in N$ be an odd, natural number. The condition (*i*) implies the condition (*ii*)

(*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. page5, 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 (12)

$$m_j = \rho(3^j \cdot (2^{p-j+1} \cdot x - 1) + 1) \tag{12}$$

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 2.3. From the lemma 2.7 we learn the following fact:

- For any natural number $q \in Nat$ there exists a compact computation that contains the sequence of length q of consecutive, increasing odd numbers

2.1.1. Lemma on partial correctnes: $\boxed{\{Gr1\}\,(m=1) \implies \{Gr3\}\,(n\cdot 3^x + y = 2^z)}$

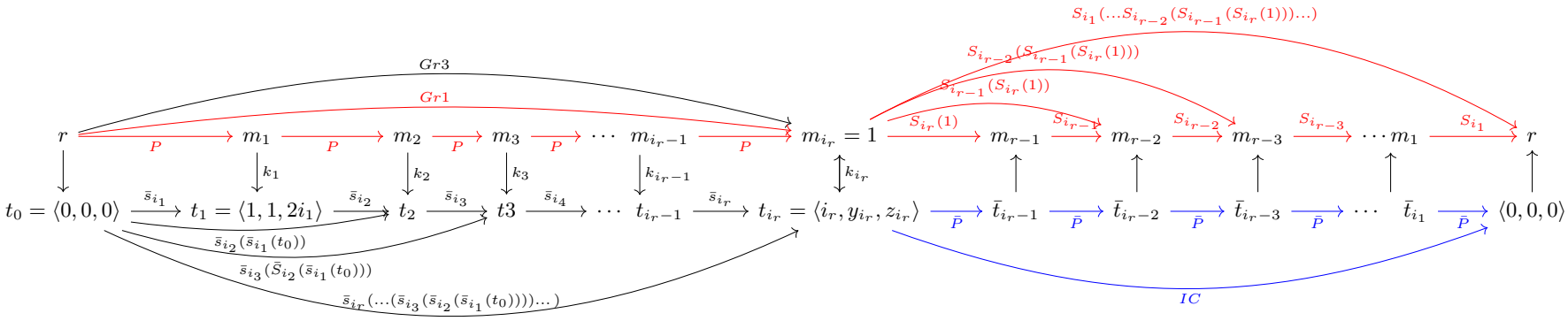


Figure 5. TWO ALGORITHMS TO FIND A PATH FROM 1 TO r OR FROM r TO 1

The numbers k_j and i_j are related as follows $k_j = 2i_j - \delta(m_{j-1})$. For the definition of $\delta(n)$ see page 14.

Lemma 2.8. (On the partial correctness of Gr3 program)
Let $r \neq 0$ be any natural number. The following conditions are equivalent

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

(*ii*) The algorithm (Gr3) halts and thepost-condition $r\cdot 3^x + y = 2^z$ holds $\{Gr2\}(r\cdot 3^x + y = 2^z)$

(*iii*)

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

The triple $\langle x, y, z\rangle$ computed by the program Gr2 permits the algorithm IC to calculate the path from 1 to r.

Proof:
The proof is left to te reader. □

3. Hotel Collatz $\mathcal{HC}$

Look at the figure 6. 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 3.1. (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 6 is only a small fragment of the infinite HC structure.
The figure 6 shows only a few red arrows. We drew only those red arrows that fit entirely on a page.

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.

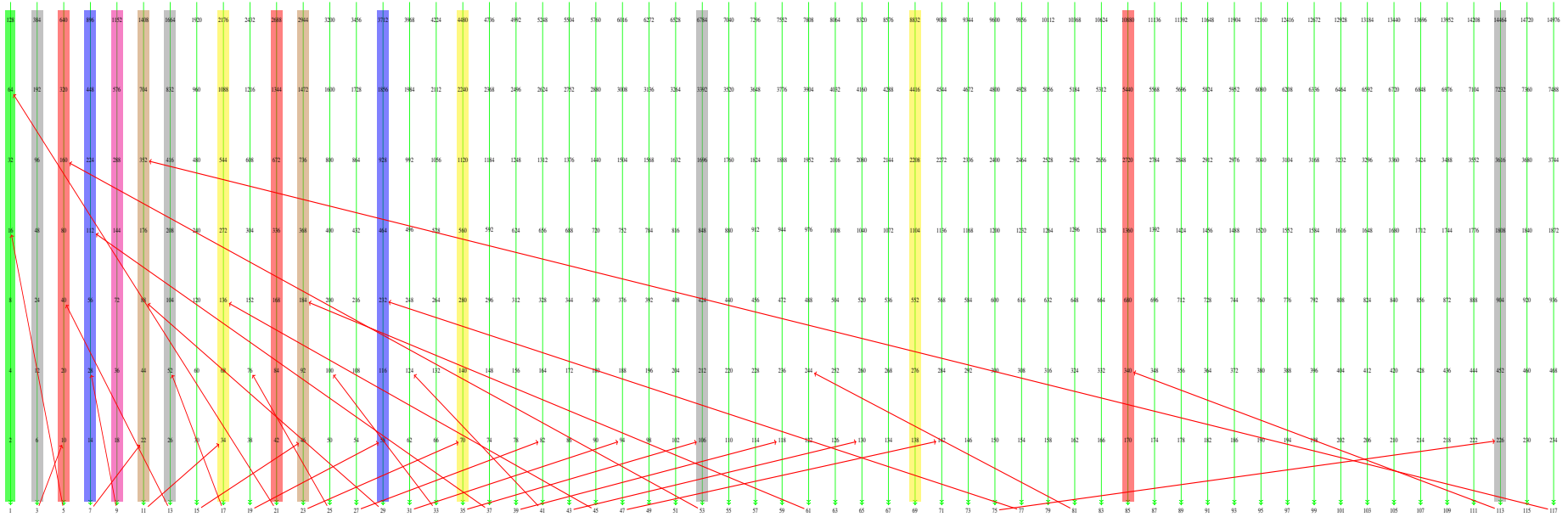


Figure 6. Hotel Collatz

Is the graph $\mathcal{HC}$ a tree? Instead of proving that the $Collatz$ tree contains every natural number we shall study another question: is the graph $\mathcal{HC}$ a tree ? An eventual positive answer solves the $3n+1$ problem.
This problem in turn boils down to the question: is the graph $\mathcal{G}$ defined by the set of all odd numbers, c.f. definition3.4, a subgraph of the graph $\mathcal{HC}$, a tree?
So, we need to prove that the graph $\mathcal{G}$ is acyclic and connected.

Corollary 3.1. 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 $p*2^k = 3*o+1$.

We are going to prove the following conjecture.

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.

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

We shall introduce the notions of successors and of predecessor of an odd, natural number. The odd numbers divisible by 3 have no successors. We shall use the following function δ to abbreviate the formulas. For very odd number not divisible by 3, we put $\delta(n) \stackrel{df}{=} (n \mod 3) - 1$.

Definition 3.2. [of successor $m = S_i(n)$]
The i-th successor of an odd number n is defined as follows

$$S_i(n) \stackrel{df}{=} \frac{n \cdot 2^{2i-\delta(n)} - 1}{3}, \qquad \text{where } i = 1, 2, \dots .$$

NOTE, the odd numbers divisible by 3 ($n \mod 3 = 0$) have no successors.
NOTE successor $S_1(1)$ is not defined.

Definition 3.3. (of predecessor $n = P(m)$)
The predecessor of an odd number m is the odd number $n = \frac{3*m+1}{2^{\kappa(3m+1)}}$.

We define the graph $\mathcal{G}$, To see an example subgraph of it, look at Fig. 7,

Definition 3.4. 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) $m = \cdot\frac{n \cdot 2^{2i-\delta(n)} - 1}{3}$ where $i \in \{1, 2, \dots\}$ and $\delta(n) \stackrel{df}{=} (n \bmod 3) - 1$.

W may conceive the graph $\mathcal{G}$ as an algebraic system

$$\mathfrak{G} = \langle V; 1, \{S_i\}_{i=0}^{\infty}, P, =\rangle$$

The following lemma gathers a couple of useful facts.

Lemma 3.1.

$$S_i(n) \neq 1 \tag{13}$$
$$S_i(n) \neq n \tag{14}$$
$$m = S_i(n) \Rightarrow P(m) = n \tag{15}$$
$$P(m) = n \Rightarrow \exists_{i>0}\, m = S_i(n) \tag{16}$$
$$S_i(n) = S_j(n) \implies i = j \tag{17}$$
$$S_i(n) = S_i(m) \implies n = m \tag{18}$$
$$i \neq j \implies S_j(S_i(n)) \neq S_i(S_j(n)) \tag{19}$$
$$i < j \implies S_i(n) < S_j(n) \tag{20}$$
$$\text{the value of } S_1(1) \text{ is undefined as well as the value of } P(1) \tag{21}$$

Proof:
We shall prove the formula (16). 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. □

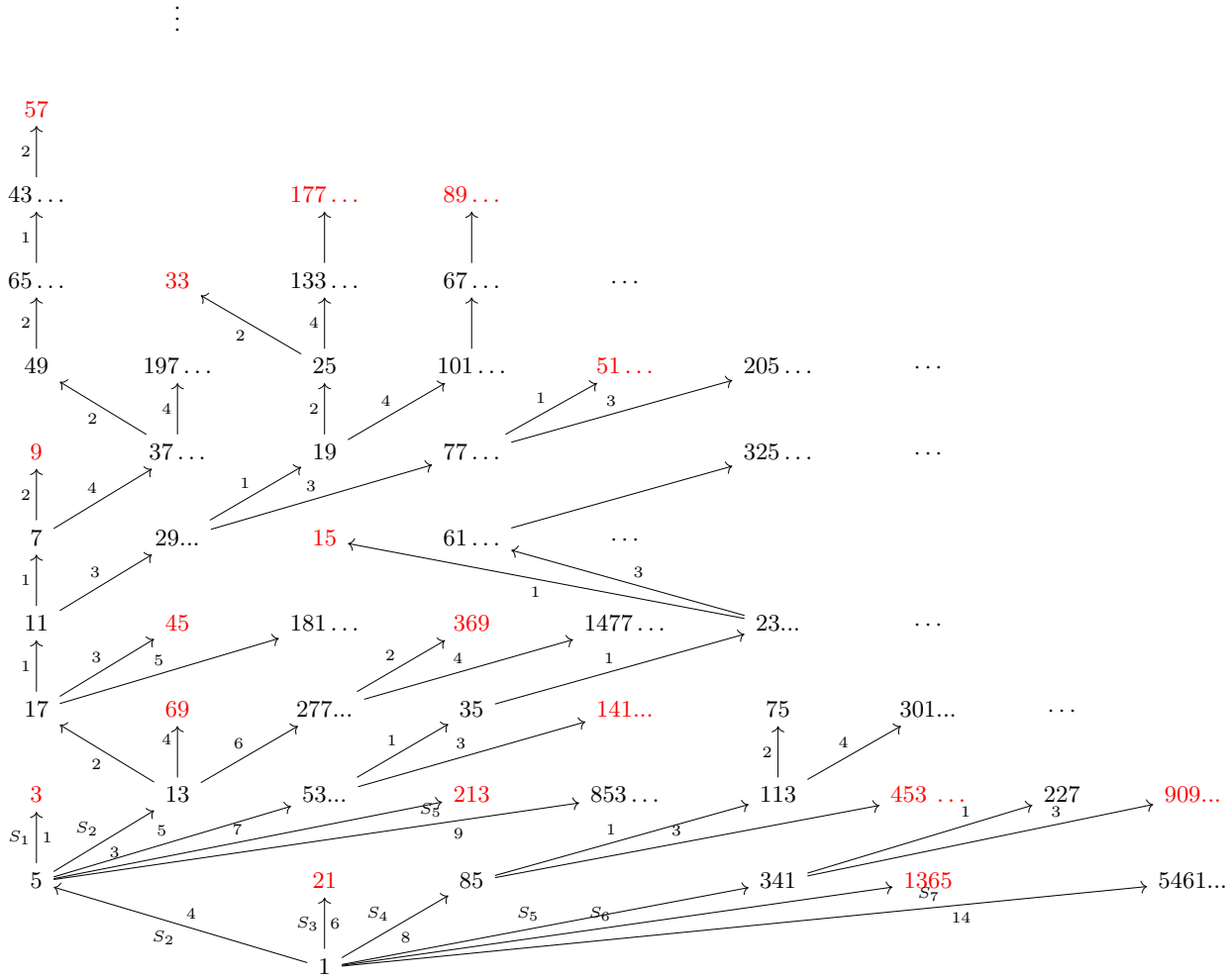


Figure 7. A fragment of the graph $\mathcal{G}$, The numbers k_i on the edges indicate the numbers ofsuccessors S_i, namely $i = \lceil \frac{k_i}{2} \rceil$

One can easily remark, that the graph $\mathcal{G}$ is a forest, i.e. it is acyclic. Is it a tree?

Lemma 3.2. The graph $\mathcal{G}$ is acyclic.

Proof:
The proof is easy, use the properties (13),(14),(17 - 21) listed in the lemma 3.1. □

It remains to be proved that the fraph $\mathcal{G}$ is connected.

3.2. The graph $\mathcal{G}$ is a tree.

The following lemma brings some crucial information.

Lemma 3.3. For every element n of the graph $\mathcal{G}$ such that $n \mod 3 \neq 0$ if there exist three natural numbers such that the following equality $n \cdot 3^x + y = 2^z$ holds, then for every positive, natural number $i > 0$ there exist three numbers x_i, y_i, z_i such that the equality $S_i(n) \cdot 3^{x_i} + y_i = 2^{z_i}$ holds.

Proof:

From the assumption we have

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

Let $i > 0$ be an odd natural numer. We multiply both sides of the equation by $2^{2i-\delta(n)}$

$$\begin{aligned}(n \cdot 2^{2i-\delta(n)}) \cdot 3^x + (y \cdot 2^{2i-\delta(n)}) &= 2^z \cdot 2^{2i-\delta(n)}\\ (n \cdot 2^{2i-\delta(n)} - 1) \cdot 3^x + (y \cdot 2^{2i-\delta(n)} + 3^x) &= 2^z \cdot 2^{2i-\delta(n)}\\ \frac{n \cdot 2^{2i-\delta(n)} - 1}{3} \cdot 3^{x+1} + (y \cdot 2^{2i-\delta(n)} + 3^x) &= 2^z \cdot 2^{2i-\delta(n)}\end{aligned}$$

We are putting $x_i = x + 1$ & $y_i = (y \cdot 2^{2i-\delta(n)} + 3^x)$ & $z_i = z + 2i - \delta(n)$ and obtain

$$S_i(n) \cdot 3^{x_i} + y_i = 2^{z_i}$$

□

We are going to address another interesting question: *is it possible to permute the columns of the Hotel Collatz in such a way that all red edges go to the left?* (and therefore the the odering $\prec$ of odd numbers determined by this permutation of columns, has the following property $\forall_{n \in Odd}\, 3 \nmid n \implies \forall_i\, n \prec S_i(n)$?)

Definition 3.5. (of subtrees D_c) For every ntural number c we define a finite tree $D_c = \langle V_c, E_c \rangle$, a subtree of the graph $\mathcal{G}$. Let $p(o)$ be the length of the path from the odd number o to 1. For every natural number c we put $V_c = \{n \in Odd\colon p(n) \leq c\}$. and $E_c = E \cap V_c \times V_c$

Example 3.1. A few subtrees contained in the graph $\mathcal{G}$

c	$D_c = \langle V_c, E_c \rangle$	numbers on diagonal c
0	$\langle \{1\}, \emptyset \rangle$	1
5	$\langle \{1,5\}, \{(1,5)\} \rangle$	5
7	$\langle \{1,5,21,3\}, \{(1,5),(1,21),(5,3)\} \rangle$	$21, 3$
9	$\langle \{1,5,21,,85,113\}, \{(1,5),(1,21),(5,3),(1,85),(5,13)\} \rangle$	$85, 13$
11	$\langle \{1,5,21,3,85,13,341,53\}, \{(1,5),(1,21),(5,3),(1,85),(5,13),(1,341),(5,53)\} \rangle$	$341, 53$
12	$\langle \{1,5,\ldots,17,113\}, \{(1,5),\ldots,(85,113),(13,17)\} \rangle$	$13, 17$

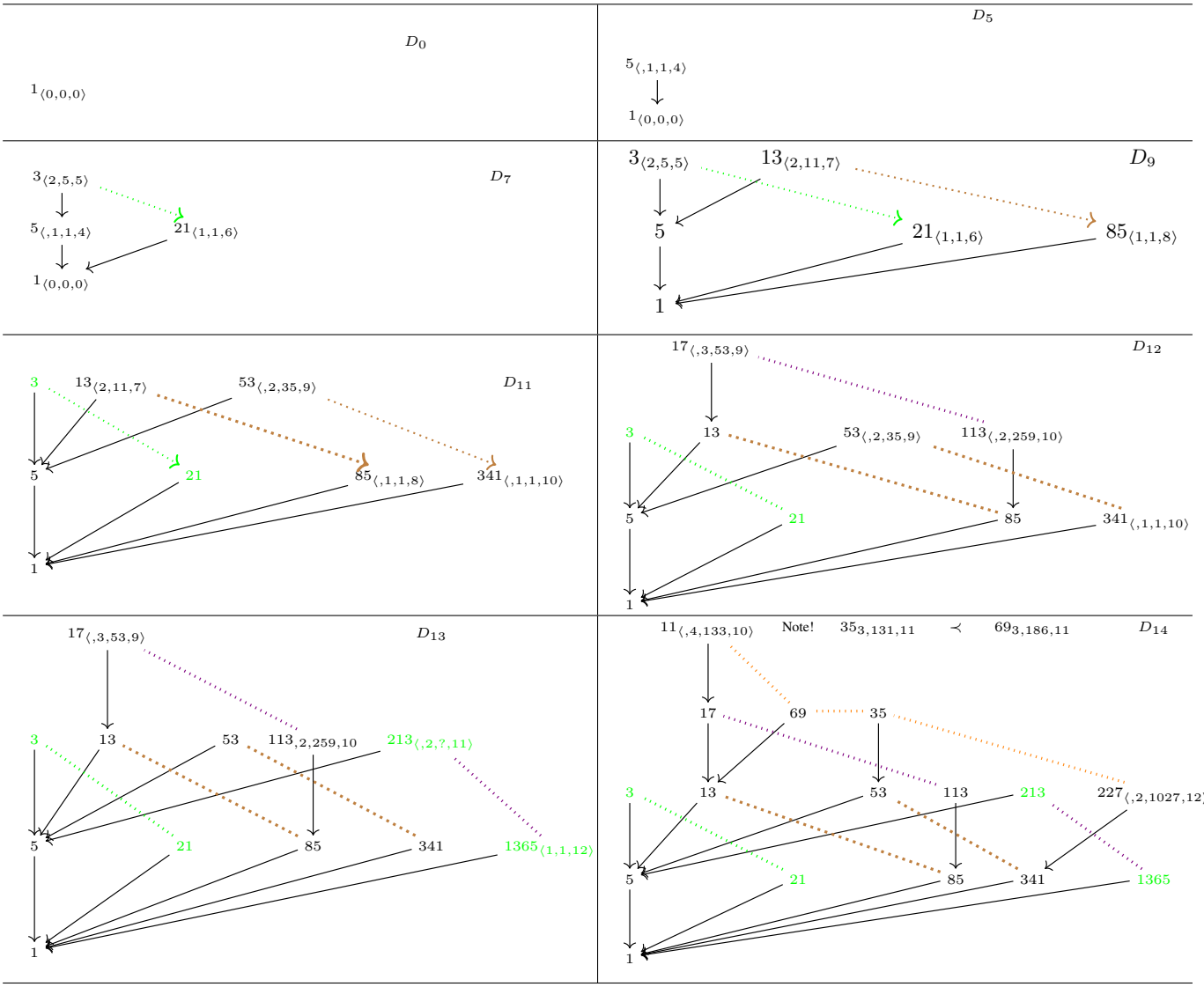


Figure 8. Couple of trees D_c

The trees form the increasing sequence $D_0 \subset D_5 \subset D_9 \subset D_{11} \subset D_{12} \subset D_{13} \subset D_{14}$.
Every tree is bounded by a diagonal $c = x + z$.
The figure 8 illustraes the trees $D_0 - D_{14}$.

Notice that each tree D_c, is the lower-left part of the graph G. For every natural number c the tree D_c contains all odd numbers that are not farther than c steps from 1. It means that for every node $o \in V_c$ the computation has no more than $c = x + z$ steps and the equation $n \cdot 3^x + y = 2^z$ holds.

End of Example 3.1

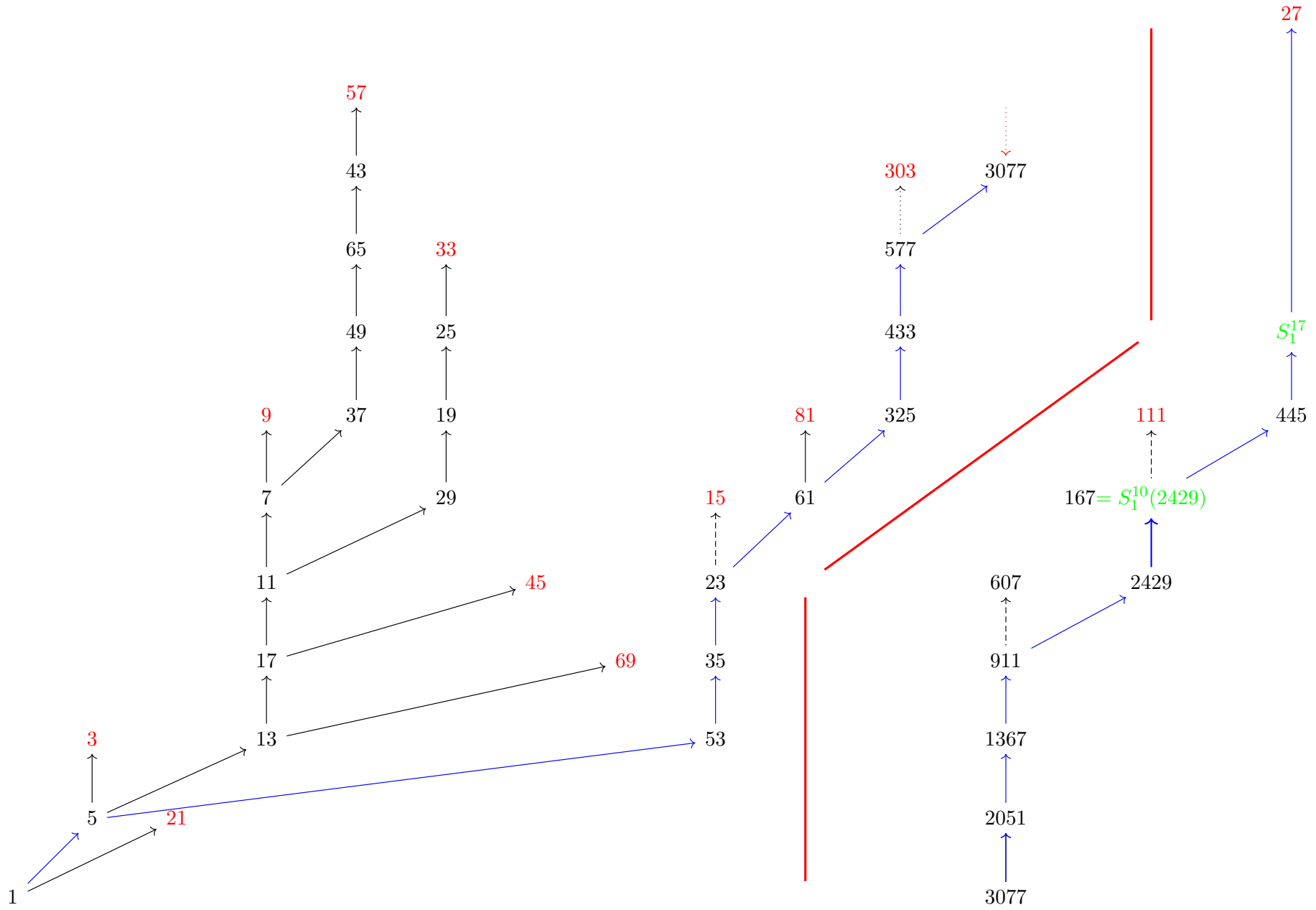


Figure 9. A subgraph of the graph $\mathcal{G}$ from another perspective $27 = S_1^{17}(S_2(S_1^{10}(S_2(S_1^3(3077)))))$ and $3077 = S_2(S_1^2(S_2^2(S_1^2(S_3(2(1))))))$

Lemma 3.4. The trees D_c have the following properties

(*i*) The trees $D_0, D_5, D_7, D_9, D_{11}$ are defined as shown in the example 3.1.

(*ii*) Trees $D_1, D_2, D_3, D_4, D_6, D_8, D_{10}$ are not defined.

(*iii*) For every $c \geq 11$ the following inclusion holds $D_c \subsetneq D_{c+1}$ For $c+1 = 5, 7, 9, 11$ trees D_{c+1} are constructed on the base of trees $D_1, D_5, , D_7, D_9$ respectively.

(*iv*) For every tree D_c and for every number $n \in D_c$ if a successor $S_i(n)$ has a path of length $c+1$ then the number n is in the tree D_c, i.e. $n \in D_c$

We shall order odd, natural numbers in a way similar to the rdering pairs of natural numbers defined by G. Cantor.

Definition 3.6. Let a and b be two odd, natural numbers. We assume that the number c is the least natural number that a is in the tree D_c. Similarly, we assume that $b \in D_d$ and for every number $d' < d$ the relation $b \notin D_{d'}$ holds. By the lemma 3.3 we know that exists two tiples $\langle x, y, z\rangle$ and $\langle u, v, t\rangle$ sch that two equations hold $a \cdot 3^x + y = 2^z$ and $b \cdot 3^u + v = 3^t$.
We put

$$a \prec b \stackrel{df}{\equiv} x + z < u + t \vee x + z = u + t \wedge x < u \vee x + z = u + t \wedge x = u \wedge y < v \tag{22}$$

Example 3.2. An initial segment of the sequence of natural numbers as ordered by the $\prec$ relation
$1 \prec 5 \prec 21 \prec 3 \prec 85 \prec 13 \prec 341 \prec 53 \prec 113 \prec 17 \prec 1365 \prec 213 \prec 227 \prec 35 \prec 69 \prec 11 \prec \ldots$

To every odd natural number n we assign its Collatz number $C(n)$

$$C\colon N \to N$$

Definition 3.7. C(n)$\stackrel{df}{=} \mathbf{card}(\{m\colon m \prec n\}) + 1$

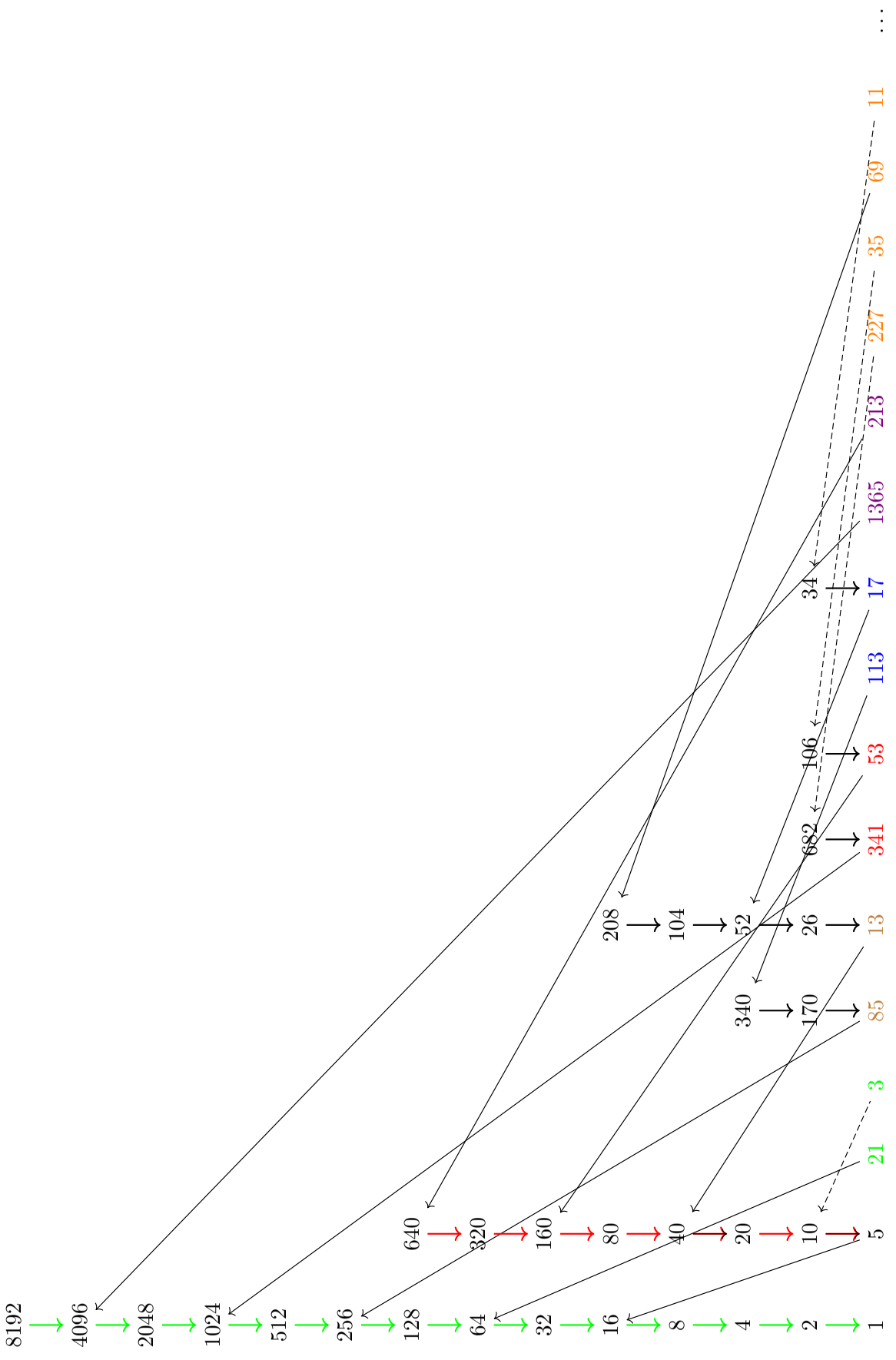


Figure 10. Hotel Collatz with columns ordered by the relation $\prec$. All diagonal arrows point to the left.

Lemma 3.5. The following properties hold

(*i*) The relation $\prec$ is ordering relation.

(*ii*) For every odd natural number n and for every natural number $i > 0$ the relation $n \prec S_i(n)$ holds.

(*iii*) For every odd, natural number n the relation $P(n) \prec n$ holds.

(*iv*) The function $C(n)$ defines a permutation of the set N of natural numbers.

(*v*) The graph $D = \bigcup_{x=0}^{\infty} D_c$ is equal to the graph $\mathcal{G}$, i.e. $D = \mathcal{G}$. Therefore the graph $\mathcal{G}$ is connected.

Proof:
A straightforward proof of properties $(i) - (iv)$is left to the reader. We shall prove the that the graph $\mathcal{G}$ is connected. Suppose, on the contrary, that the graph G is not connected. Hence there is a number c such that the tree D_{c+1} contains an element n_0 such that the element n_0 is not a successor of any element $n \in D_c$ (i.e. for every number $n \in D_c$ and for every number i, the element $n_0 \neq S_i(n)$). On the other hand the predecessor $m = P(n_0)$ of the number n_0 exists and $m \in D_c$. By the lemma 3.1(16) we have that there exists a number j such that $S_j(m) = n_0$. This is inconsistent with our assumption. Therefore the graphs D and $\mathcal{G}$ are equal, $D = \mathcal{G}$. □

From this and the lemma 3.2 we obtain

Lemma 3.6.

(*i*) The graph $\mathcal{G}$ is a tree.

(*ii*) The graph $\mathcal{HC}$ is a tree.

4. Main lemma

Lemma 4.1. Let r be a natural number. Every formula of the scheme ML_r is a theorem of algorithmic theory $\mathcal{ATN}$.

$$\mathcal{ATN} \vdash (n = r) \implies \{x := 0\} \bigcup \{x := x + 1\} \left(\left| \begin{array}{l} n \cdot 3^x + y = 2^z \wedge \\ \left(y = \sum_{j=0}^{x-1} 3^{x-1-j} \cdot 2^{\sum_{p=0}^{j} k_p} \right) \wedge \left(z = \sum_{p=0}^{x} k_p \right) \wedge \\ \underbrace{\left(\begin{array}{l} (k_0 = \kappa(n) \wedge m_0 = \rho(n)) \wedge \\ \left(\bigwedge_{l=0}^{x-1} \left(\begin{array}{l} k_{l+1} = \kappa(3m_l + 1) \wedge \\ m_{l+1} = \rho(3m_l + 1) \end{array} \right) \right) \end{array} \right)}_{\text{the length of computation } = x} \end{array} \right| \right) \qquad (ML_r)$$

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 of the form ML_r is recursive and it consists of theorems of $\mathcal{ATN}$ theory.

Proof:
The proof follows from the lemma 3.6 and the fact that the nodes of the tree $\mathcal{G}$ are standard, odd natural numbers. Hence no infinite computation exists. □

Example 4.1.

$$\left| \begin{array}{r} n = 5 \implies n \cdot 3^{\mathbf{1}} + 1 = 16 \\ n = 67 \implies n \cdot 3^{\mathbf{8}} + 84701 = 2^{19} \\ n = 5461 \implies n \cdot 3^{1} + 1 = 2^{14} \\ \\ \\ n = 227 \implies n \cdot 3^{2} + 5 = 2^{11} \end{array} \right| \quad \overbrace{\overbrace{(((227 * 3 + 1) \div 2^{1}) * 3 + 1) \div 2^{10} = 1}^{P(227)=341}}^{P(P(227))}$$

Corollary 4.1. 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 ML_r 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 4.2. In other words, every number r can be presented in the following form

$$\frac{\frac{\frac{\frac{\frac{2^{k_{i_r}}-1}{3} \cdot 2^{k_{i_r-1}}-1}{3} \cdot 2^{k_{i_r-2}}-1}{3} \cdot 2^{k_{i_r-3}}-1}{3} \cdots}{\cdots} \cdot 2^{k_0} - 1}{3} = r \tag{23}$$

or in yet another form

$$r = S_{k_0}\left(S_{k_1}\left(\cdots\left(S_{k_{i_r-1}}\left(S_{k_{i_r}}(1)\right)\right)\cdots\right)\right) \tag{24}$$

(25)

Example 4.2.

$$\left|\begin{array}{c} 227 = S_1(S_5(1)) \\ 61 = S_2(S_1^2(S_3(S_2(1)))) \\ 27 = S_1^{17}(S_2(S_1^{10}(S_2(S_1^3(S_2(S_1^2(S_2^2(S_1^2(S_3(S_2(1))))))))))) \end{array}\right|$$

4.1. The limitations of the Main lemma

Does the lemma 4.1 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.

It turns out that the criterion of termination of computation in its form 10 is *falsifiable* in any non-standard model of Peaono's arithmetic in spite of being valid in the standard structure of natural numbers $\mathfrak{N}$. ..
Two remarks are in order.

Remark 4.1. For every quadruple $\langle n, x, y, z\rangle$ of natural numbers the equation $n \cdot 3^x + y = 2^z$ can be treated as an abbreviation of a formula of Presburger's arithmic, c.f. subsection 8.2 and example 4.1.

This shows that the sentence 10 is not a theorem. For the details consult the subsection 8.1.

4.2. The positive consequnces of the main lemma.

Let Γ_3 and Δ_3 be parts of the program Gr3.
For every natural number $r > 0$, here exists the number $i(r)$such that the implication of the following form 26

$$n = r \implies \left\{\Gamma_3\,;(\textbf{if } m \neq 1 \textbf{ then } \Delta_3 \textbf{ fi})^{i(r)}\right\}(m = 1) \tag{26}$$

is a theorem of the algorithmic theory $\mathcal{ATN}$, c.f. the subsection 8.4 on page 35.

$$\mathcal{ATN} \vdash n = r \implies \left\{\Gamma_3\,;(\textbf{if } m \neq 1 \textbf{ then } \Delta_3 \textbf{ fi})^{i(r)}\right\}(m = 1) \tag{27}$$

Note, that the formula 26 can be replaced by an equivalent formula (provided that the variable l does not appear in Δ_3)

$$n = \underline{r} \implies \{\Gamma_3\,;\textbf{for } l := 1 \textbf{ to } i(r) \textbf{ do } \Delta_3 \textbf{ od}\}\,(m = 1). \tag{28}$$

Example 4.3.

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

5. Proof of the Collatz conjecture

Note, 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 $n = 1 \vee n = 2 \vee n = 3 \vee \ldots$. 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 5.1. (Main)
The following formula (Main Thm) is a theorem of formalized algorithmic theory $\mathcal{ATN}$

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

The theorem reads as follow, (the program in the consequent of the implication 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 the proof can be explained as follow:

1. We know that the set St_0, see page[22,(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 the elements of the set St_0 are theorems 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 $\{m := n \div 2^{\kappa(n)};\}$.
The sign Δ denotes the program $\{m := 3 \cdot m + 1;\ m := m \div 2^{\kappa(m)}\}$ or if you like

$$\{m := 3 \cdot m + 1;\ \mathbf{while}\ even(n)\ \mathbf{do}\ n := n \div 2\ \mathbf{od}\}$$

Proof:
We are resuming at the main lemma 4.1. 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 (29)

$$n = \underline{r} \implies \{\Gamma\}\,\{\mathbf{if}\ m \neq 1\ \mathbf{then}\ \Delta\ \mathbf{fi}\}^{i(\underline{r})}\,(m = 1) \tag{29}$$

is contained in St_1.

Lemma 5.1. 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.

$$St_1 \stackrel{df}{=} \left\{n = \underline{r} \implies \{\Gamma\}\,\{\mathbf{if}\ m \neq 1\ \mathbf{then}\ \Delta\ \mathbf{fi}\}^{i(\underline{r})}\,(m = 1)\right\}_{r=1}^{\infty} \tag{30}$$

In the proof of the lemma, we use the following axiom (Ax_{18}) of assignment instruction

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

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

and the axiom Ax_{20} of conditional instruction **if**.

$$\boxed{\mathbf{if}\ \gamma\ \mathbf{then}\ K\ \mathbf{else}\ M\ \mathbf{fi}\,\alpha \Leftrightarrow ((\gamma \wedge K\alpha) \vee (\neg M\alpha))} \tag{Ax_{20}}$$

Next, we consider the set St_2 that contain all formulas of the scheme shown in the equation 31 and only such formulas.

$$St_2 \stackrel{df}{=} \left\{n = \underline{r} \implies \{\Gamma;\ \mathbf{while}\ m \neq 1\ \mathbf{do}\ \Delta\ \mathbf{od}\}\,(m = 1)\right\}_{r=1}^{\infty} \tag{31}$$

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.

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

Hence the thesis T2) is justified.
Therefore, the set St_3 that consists of all formulas of the scheme (32).

$$St_3 \stackrel{df}{=} \left\{ \left\{ \begin{array}{l} q := 0; \\ \{\ q := q + 1\ \}^{\underline{r}} \end{array} \right\}(n = q) \implies \left\{ \begin{array}{l} \Gamma; \\ \mathbf{while}\ m \neq 1 \\ \mathbf{do} \\ \quad \Delta \\ \mathbf{od} \end{array} \right\}(m = 1) \right\}_{r=1}^{\infty} \tag{32}$$

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

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

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.

6. Final remarks.

Our message does not limit itself to the proof of Collatz theorem. We believe that the algorithmic logic is a proper tool in developing algorithmics, discrete mathematics and software engineering.
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.

Tigthening the gap
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.

The Gödel's theorem reveals a big gap between the set of valid properties of natural numbers and the set of theorems of Peano's arithmetic. We are hoping that the calculus of programs (*aka* algorithmic logic) will permit to solve many questions e.g. the Goldbach conjecture. There are many questions that remain open.

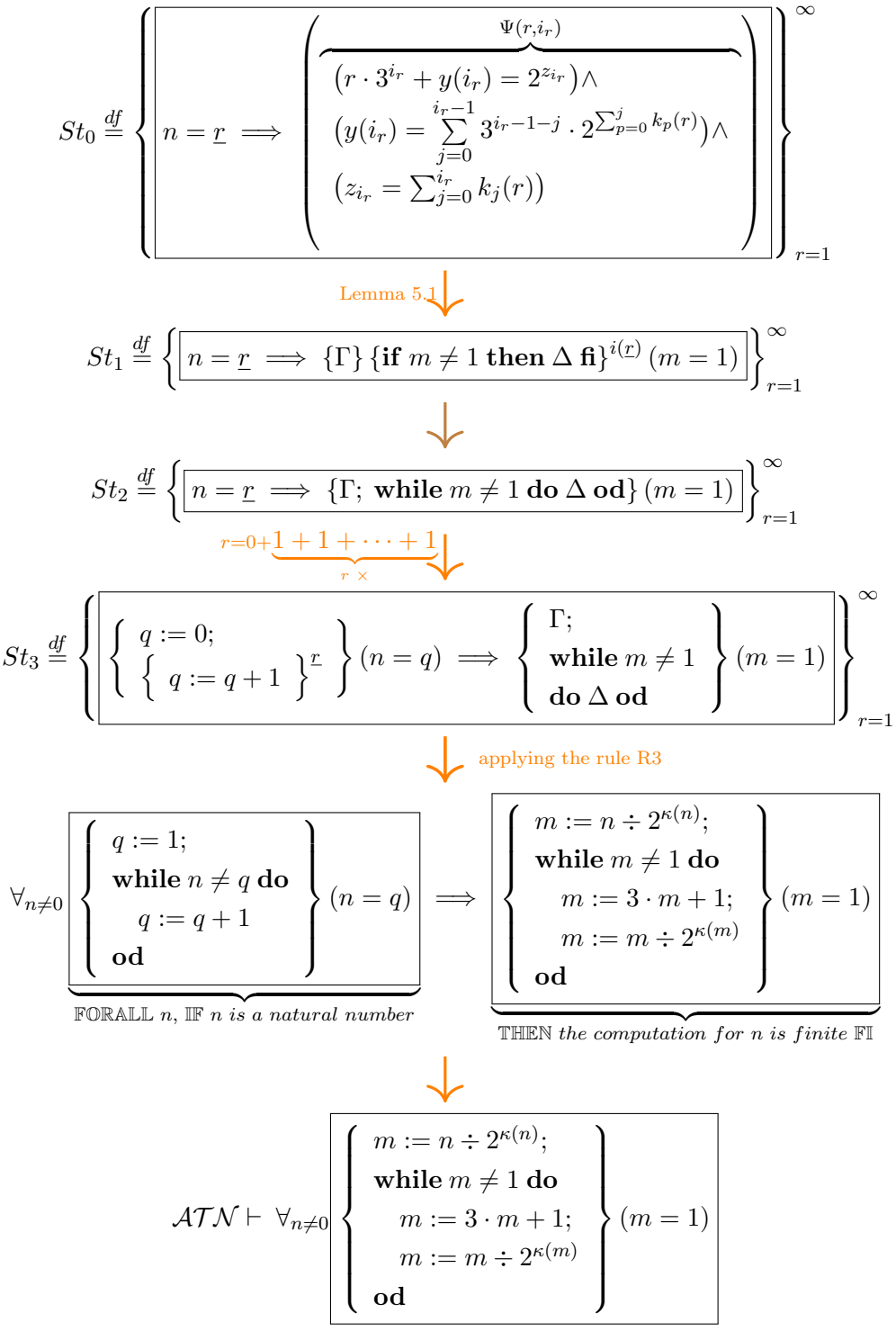


Figure 11. The structure of the proof of Main theorem

Efficiency problems
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 $O(2^{2^{2^n}})$
From the proof of lemma 3.4 one can deduce that the cost of the Collatz algorithm is polynomial.
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 6.1. One needs not to execute $3n+1$ computation. Just accept 1 as the *proven* result.

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.

7. 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.

[Kle36] Stephen C. Kleene General Recursive Functions of Natural Numbers. Mathematishes Annalen, 12, 1936, pages 727–742.

[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 ofComputer 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.

8. Suplements

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

8.1 The Jaśkowski's model of Presburger arithmetic for the use of programmers.

8.2 A short presentation of Presburger theory of addition.

8.3 A short presentation of calculus of programs *aka* algorithmic logic $\mathcal{AL}$.

8.4. A short presentation of the formalized algorithmic theory of natural numbers $\mathcal{ATN}$.

8.1. 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.
A programmer 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 8.1. 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 definition8.2, p.31), is a sentence true in the structure $\mathfrak{J}$, cf. next subsection 8.2.

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 8.4. 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 8.4. The elements of the structure $\mathfrak{N}$ are called *reachable*. A theorem of the foundations of mathematics states:

Lemma 8.2. 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 8.1. An element n is a standard natural number (i.e. is *reachable*) iff the program of adding ones to initial zero terminates

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

or, equivalently

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

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 8.1. 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

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

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

The mathematically oriented reader will easily accept our arguments. For the informaticians we offer the same thought in the software's disguise.

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 $n > m \stackrel{df}{=} \exists_{u \neq 0}\, m + u = n$. Invention of S. Jaśkowski (1929).

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

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 or li * mia <0 or 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 8.2) treated as a specification of a class (i.e. a module of program).

Lemma 8.3. 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;
```

The reader may experiment with our program. Note, it is easy to adapt to your favorite syntax (programming language). 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 8.1. 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 8.2. 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.

8.2. 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 8.2. 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:
  a) if two expressions τ_1 and τ_2 are terms, then the expression $(\tau_1 + \tau_2)$ is a term too.
  b) if an expression τ is a term, then the expression $S(\tau)$ 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 8.3. 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.

$\forall_x\ x+1 \neq 0$ (A)

$\forall_x\ x \neq 0 \implies \exists_y x = y+1$ (B)

$\forall_{x,y}\ x+y = y+x$ (C)

$\forall_{x,y,z}\ x+(y+z) = (x+y)+z$ (D)

$\forall_{x,y,z}\ x+z = y+z \implies x = y$ (E)

$\forall_x\ x+0 = x$ (F)

$\forall_{x,z}\ \exists_y\ (x = y+z \vee z = y+x)$ (G)

$\forall_x\ \exists_y\ (x = y+y \vee x = y+y+1)$ (H2)

$\forall_x\ \exists_y\ (x = y+y+y \vee x = y+y+y+1 \vee x = y+y+y+1+1)$ (H3)

. . . . . .

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

. . .

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 triple exponential $O(2^{2^{2^{cn}}})$. This result belongs to Oppen [Opo78] and it can be traced in the Presburger's paper [Pre29].

F4. Theories $\mathcal{T}$ and $\mathcal{AP}$ have non-standard model, see section 8.1, p. 28.

The reader may have doubts about the correctness of the formulas in our text that are using exponentiation. Of course, exponentiation does not occur in Presburger arithmetic.
We are offering the following explanations andcomments.

- Let n be a variable and x be a natural number (e.g., 3). Then, the notation n . $n \cdot 3^4$ should be understood as an abbreviation for the expression $(n+n+....+n)$ in which the variable n has been added 81 times.
- Let a, x, y, z be numerals or terms withou variables. The formula $a \cdot 3^x + y = 2^z$ comes as a neat abbreviation of some lengthy and obscure expression.
- In the section 5 we are operating within algorithmic theory $\mathcal{ATN}$. This theory is alike the Presburgers theory where the language admits algorithms (programs) as well formed expressions and instead of scheme of induction

we have the axiom of reachability (R), on page 35.
Consider the following program $P3(n,x)$ where n and x are some variables.

$$P3(n,x): \left\{ \begin{array}{l} q \leftarrow 0;\ w \leftarrow n; \\ \textbf{while } q \neq x \textbf{ do} \\ \quad w \leftarrow w+w+w; \qquad q \leftarrow q+1 \\ \textbf{od} \end{array} \right\}$$

If the initial value of the variable n is a number r and the initaila value of the variable x is a number a, i.e. $val: \frac{n \mid x}{r \mid a}$, then the execution of the program $P3$ is finite and the final value of the variable $w = r^a$.

8.3. 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

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

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

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

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.

Theorem 8.1. **(*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.

8.4. An outline of 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]

$$(R)\ \forall x\{q := 0;\ \mathbf{while}\ q \neq x\ \mathbf{do}\ q := s(q)\ \mathbf{od}\}(q = x)$$
$$(N)\ \forall x\ s(x) \neq 0$$
$$(J)\ \forall x \forall y\ s(x) = s(y)\ \Longrightarrow\ x = y$$
$$(D)\ \forall x\ \forall y \left\{ \begin{array}{ll} q := 0; w := x; & \\ \mathbf{while}\ q \neq y\ \mathbf{do} & q := s(q)\ ; w := s(w)\mathbf{od} \end{array} \right\} (x + y = w)$$

The termination property of the program in $(D)$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).

$$\mathcal{ATN} \vdash \exists_x \alpha(x) \Leftrightarrow \{x := 0\} \bigcup \{x := x + 1\}\alpha(x) \tag{33}$$

$$\mathcal{ATN} \vdash \forall_x \alpha(x) \Leftrightarrow \{x := 0\} \bigcap \{x := x + 1\}\alpha(x) \tag{34}$$

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

$$\mathcal{ATN} \vdash 0 < x < y \implies \{a := x;\ \mathbf{while}\ a < y\ \mathbf{do}\ a := a + x\ \mathbf{od}\}(a \geq y) \tag{35}$$

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

$$\mathcal{ATN} \vdash \Big(\alpha(x/0) \wedge \forall_x\big(\alpha(x) \implies \alpha(x/s(x))\big)\Big) \implies \forall_x \alpha(x) \tag{36}$$

Correctness of Euclid's algorithm

$$\mathcal{ATN} \vdash \left(\begin{array}{l} n_0 > 0 \wedge \\ m_0 > 0 \end{array}\right) \implies \left\{\begin{array}{l} n := n_0;\ m := m_0; \\ \mathbf{while}\, n \neq m\, \mathbf{do} \\ \quad \mathbf{if}\, n > m\, \mathbf{then}\ n := n \dot{-} m \\ \quad\quad \mathbf{else}\ m := m \dot{-} n \\ \quad \mathbf{fi} \\ \mathbf{od} \end{array}\right\} (n = gcd(n_0, m_0)) \tag{37}$$

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

Theorem 8.2. (*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 8.1.

As the simple corollary from the categoricity theorem we note
every formula α valid in the standard model of natural numbers $\mathfrak{N}$ has a proof in the theory $\mathcal{ATN}$.
We can formulate this remark as follows: *for every algorithmic formula α, if the formula is valid in the structure $\mathfrak{N}$ then there is a finite proof of it or a finite proof that the finite prof exists or a finite proof of the existence of a proof thea t finite proof exists or . . .*