WANTED: AN ALGORITHMIC PROOF OF HALTING PROPERTY OF 3X+1 PROGRAM

ANDRZEJ SALWICKI

Abstract. Please help to construct a proof that the following program P stops for every natural number n > 0
while n > 1 do if even(n) then n := n div 2 else n := 3n + 1 fi done.
in the framework of algorithmic logic. The prize: a box of champagne is
waiting for the author of the first proof.

1. Introduction

In this note we present a problem, a prize and some hints.

2. Problem

To prove (or to disprove) the following

Conjecture 1. The following formula is a theorem of algorithmic theory of natural numbers

\[(\forall n > 0) \bigcup \{\text{if even}(n) \text{ then } n := n \div 2 \text{ else } n := 3 \times n + 1 \} \big| (n = 1)\]

The conjecture is equivalent to well known conjecture: for every natural number
n > 0 the following program P terminates its execution.

\[P : \text{while } n \neq 1 \text{ do if even}(n) \text{ then } n := n \div 2 \text{ else } n := 3 \times n + 1 \text{ fi done}\]

Recently I have indicated that the set of theorems of Peano Arithmetic does
not contain any sentence saying that the above program P halts for every natural
number n.

3. The History of the Problem

The problem has more than 60 years. Many people worked on it, without suc-
cess, J. Lagarias[1] gives a presentation of the problem and the results concerning
it. The electronic version of the paper contains bibliography till 1996.
A special conference devoted to the problem was held in Eichstaett[2].
Information about numerical records obtaind till today is to be found [3].


The language of the theory consists of three subsets: the set of formulas, the set
of programs and the set of terms.
The alphabet of the theory contains the following symbols: individual variables \( x, y, z, \ldots \), constant 0, one-argument functor \( s \), the equality predicate \( = \), logical functors, quantifiers and iteration quantifiers.

The set \( T \) of terms is the smallest set of expressions which contains the expression 0, any individual variable and is closed with respect to the following rule: if an expression \( \tau \) is a term then the expression of the form \( s(\tau) \) is also a term.

The set \( F \) of formulas is the smallest set of expression which contains the expressions of the form \( \tau = \nu \) where \( \tau, \nu \) are terms, and which is closed with respect to the following set of rules:

- if \( \alpha \) and \( \beta \) are formulas then the following expressions are also formulas:
  \[ (\alpha \lor \beta), \quad (\alpha \land \beta), \quad \neg \alpha, \quad (\alpha \Rightarrow \beta) \]

- If \( \alpha \) is a formula and \( x \) is an individual variable then the expressions
  \[ (\exists x)\alpha \quad \text{and} \quad (\forall x)\alpha \]
  are formulas.

- If \( \alpha \) is a formula and \( K \) is a program then the expressions
  \[ K\alpha, \quad \bigcup K\alpha, \quad \bigcap K\alpha \]
  are formulas.

The set of programs is the smallest set of expressions such that
- if \( x \) is an individual variable and \( t \) is a term then the expression
  \[ x := \tau \]
  is a program.
- if \( K \) and \( M \) are programs then the expression
  \[ \text{begin } K; M \text{ end} \]
  is a program.
- If \( K \) and \( M \) are programs and \( \gamma \) is an open formula then the expressions
  \[ \text{while } \gamma \text{ do } K \text{ done} \quad \text{if } \gamma \text{ then } K \text{ else } M \text{ if} \]
  are programs.

The set of axioms of the algorithmic theory of natural numbers consists of three formulas:

Ax1) \[ (\forall x) \neg s(x) = 0 \]
Ax2) \[ (\forall x)(\forall y) s(x) = s(y) \Rightarrow x = y \]
Ax3) \[ (\forall x)(y := 0) \bigcup \text{if } \neg x = y \text{ then } y := s(y) \text{ if} \] \( x = y \)

5. Prize

A box of champagne is offered for he/she who will present the first proof of the conjecture, or will present a counterargument.

6. Some hints

6.1. **Find an equivalent program and prove it halts.** It seems to me that the best way to a successful answer is to find a program \( K \) such that \( K \) is equivalent to the program \( P \) and such that a proof of halting property is given.

Below I quote a few examples. Remark that there is infinitely many programs that are equivalent to the program \( P \).

First program \( P_1 \) is equivalent to the program \( P \). We do not know whether it stops for every natural number \( n \). (The compound sign \( <> \) is to be read as "not equal").
while n<>1 do
begin
  r1 := n; r2 := 0; r3 := 0; r4 := 0; r5 := 0;
  while r1 <> r2 do
    r2 := r2 + 1;
    if r3 = 0 then r3 := 1; r4 := r4 + 1 else r3 := 0 fi;
    r5 := r5 + 1 + 1 + 1
  done
  if r3 = 0 then n := r4 else n := r5 + 1 fi
end
done

The program $P_1$ does not use multiplication. Addition is not necessary either. One can write $s(x)$ instead of $x+1$. It is easy to prove that the internal instruction while always halts. The program $P_1$ has two while instructions nested one in another.

We can transform our program to have only one while instruction.
begin q := 1;
while ((q = 1 and n<>1) or q<>1) do
  if q = 1 and n<>1 then
    r1 := n; r2 := 0; r3 := 0; r4 := 0; r5 := 0; q := 0;
  else
    if r1 <> r2 then
      r2 := r2 + 1; r5 := r5 + 1 + 1 + 1;
      if r3 = 0 then r3 := 1; r4 := r4 + 1 else r3 := 0 fi;
      else
        q := 1; if r3 = 0 then n := r4 else n := r5 + 1 fi
      fi
    done
  fi
end

Still I do not know how to prove the halting property of the program $P_2$. The third program $D$ builds a tree and visits it in breadth first search for the number $n$. 
begin
var q: queue, n,x,y,z: integer;
q :=∅; x := 1; q := insert(1,q);
while x <> n do
    q := insert (2*x, q);
y := (x-1) div 3; z := (x-1) mod 3;
    if z =0 and odd(y)
        D:
            q := insert(y, q);
        fi;
    x:= first(q); q := deletefirst(q);
end

Now we have to prove that the halting properties of programs \( P \) and \( D \) are equivalent (\( P \) true \( \iff \) \( D \) true) and to prove that program \( D \) halts (\( D \) true). The first task is easy.

One can combine programs \( P \) and \( D \) in such a way that the new program \( DP \) executes alternatively one iteration of program \( P \), then, one iteration of program \( D \).

found := false; r3:=∅; q :=∅; // q is a queue of integers
setG :=∅; setD = ∅; // setD and setG are sets(e.g. priority queues)
while ~ found
    do
        if r3
            then
                if even(n) then n:= n div 2 else n:= 3*n+1 fi;
                insert(n) to setG;
                if n ∈ setD then found := true fi
            else
                x:= first(q); q := delete(q);
                q := put(2*x, q);
                insert(2*x) to setD;
                if n ∈ setG then found := true fi;
                y := (x-1) div 3; z:= (x-1) mod 3;
                if z=0 and even(x)
                    then
                        q := put(y, q);
                        insert(y) to setD;
                        found := y ∈ setG;
                    fi;
                fi;
            r3 := 1-r3;
        done

6.2. Prove the halting property of program \( P \) from axioms9. An alternative way of searching is to prove the implication (\( \forall x \)Ntrue \( \implies \) (\( \forall n \)Ptrue. \( N \) is the program from the algorithmic axiom Ax3'.

Let me recall a few of useful inference rules
\[ \alpha \Rightarrow \beta \]

while \( \beta \) do \( K \) done true \( \Rightarrow \) while \( \alpha \) do \( K \) done true

and

while \( \alpha \) do \( K \) done true
while \( \alpha \) do \( L; K; M \) done true

where the programs \( L \) and \( M \) always halt and do not change value of any variable occurring in \( \alpha \) or in \( K \).

It will suffice to prove the following formula.

\((\forall n > 0) \bigcup \text{if even}(n) \text{ then } n := n \text{ div } 2 \text{ else } n := 3 \ast n + 1 \text{ fi}(n = 1)\).

6.3. Other logical frameworks. a) Perhaps it may be easier for you to discuss formulas of \textit{weak second order logic} with the quantifiers over finite subsets or sequences of the set of standard natural numbers.

Our hypothesis reads: for every natural number \( n > 0 \), there exist a finite sequence \( s = s_0, \ldots, s_k \) of natural numbers such that: \( n = s_0 \), \( s_k = 1 \) and for every \( j = 1, \ldots, k \), \( s_{j+1} = s_j \text{ div } 2 \) if \( s_j \) is an even number, \( s_{j+1} = 3 \ast s_j + 1 \) if \( s_j \) is an odd number.

b) Another possibility is to consider \textit{infinite disjunctions}. Program \( P \) terminates iff

\[ n = 1 \wedge (n \neq 1 \wedge (\text{even}(n) \wedge n \text{ div } 2 = 1)) \wedge (n \neq 1 \wedge (\text{even}(n) \wedge n \text{ div } 2 = 2)) \wedge (n \neq 1 \wedge (\text{even}(n) \wedge n \text{ div } 2 = 4)) \wedge (n \neq 1 \wedge ((\text{even}(n) \wedge n \text{ div } 2 = 8)) \wedge (\neg \text{even}(n) \wedge 3 \ast n + 1 = 16)) \wedge \ldots \]

One may observe that the scheme of this infinite disjunction is as follow

\[ \bigvee_{i=0}^{\infty} \text{if even}(n) \text{ then } n := n \text{ div } 2 \text{ else } n := 3 \ast n + 1 \text{ fi} \quad (n = 1) \]

To verify it you need to know that the following equivalence is an algorithmic tautology

if \( \gamma \) then \( K \) else \( M \) fi \( \alpha \Leftrightarrow ((\gamma \wedge K \alpha) \vee (\neg \gamma \wedge M \alpha)) \).

REFERENCES


contains an excellent survey and bibliography till 1985, an electronic version of the paper with the annotated bibliography till 1996 is accessible as http://www.cecm.sfu.ca/organics/papers/lagarias/

[2] A link to a conference devoted exclusively to Collatz problem
http://www.math.gmu.edu/~chamberl/conf.html


an important page with records of numerical research and more ...


INSTITUTE OF INFORMATICS, UNIVERSITY OF WARSAW
E-mail address: salwicki@mimuw.edu.pl
URL: http://www.mimuw.edu.pl/~salwicki