ELSEVIER
Theoretical Computer Science 191 (1998) 145–156
Theoretical Computer Science

Makanin's algorithm is not primitive recursive[1]

Antoni Kościelski, Leszek Pacholski

Institute of Computer Science, Wrocław University, Przesmyckiego 20, 51-151 Wrocław, Poland

Received March 1996
Communicated by A. Razborov

Abstract

We prove that, contrary to the common belief, the algorithm deciding satisfiability of equations in free groups, given by Makanin, is not primitive recursive.

0. Introduction

The problem whether the set of all equations that are satisfiable in some free group – or, equivalently, in all groups – is recursive (usually called the satisfiability problem for group equations), and the analogous problem for semi-groups (usually called the satisfiability problem for semigroup equations) were first formulated by Markov in early 1960s (see [2]). Special cases of the second problem were solved affirmatively by Markov (see [2]), Khmelevskiĭ [4], Plotkin, [13] and Lentin [8]. But the full solution turned out to be extremely difficult and eluded researchers for many years.

The breakthrough came in a series of papers by Makanin [9–11]. The first of these gave a positive solution to the satisfiability problem for semigroup equations. The second, which appeared a few years later, together with corrections published in the third paper, established the analogous, much more difficult result for groups.

Makanin's decision procedure for equational satisfiability in semi-groups has received a lot of attention in the literature, mainly from computer scientists. Several improvements of Makanin's algorithm have been given (see [12, 1, 3, 15, 6]).

As regards Makanin's result for groups, not much further research seems to have been done. We can only mention the 1984 paper by Razborov [14], which presents an algorithm for generating all solutions to a given group equation.

E-mail: {kosciels, pacholsk}@tcs.uni.wroc.pl.

[1] A preliminary version of the main result of this paper has been presented at the 31st IEEE Symposium on Foundations of Computer Science (see [7]). This research was partially supported by KBN grants 2 P301 034 07 and 8 S503 022 07.

PII S0304-3975(96)00321-0

In this paper we prove that, contrary to the common belief, the Makanin's decision procedure for equational satisfiability in groups presented in [10, 11] is not primitive recursive.[2] In fact, one can prove (see [5]) that the functions defined in [11] which give an upper bound on the number of iterations of elementary steps in Makanin's algorithm are not primitive recursive. However, it is impossible to give a concise and comprehensive proof of this statement without copying a large part of Makanin's paper, since definitions of the actual functions used in [10, 11] are quite complicated and are mixed together with the proof of correctness of algorithm and with the algebraic arguments. Moreover, the fact that the bounds are not primitive recursive does not imply that the algorithm itself is not primitive recursive.

To circumvent the difficulty we introduce the notion of an abstract Makanin algorithm. It captures the algorithmic properties used to prove the decidability of the solvability problem for equations in free groups, and abstracts from the algebraic context which obscures the overall structure of the algorithm. Abstract Makanin algorithm can be considered a skeleton, on which with complicated data structures (generalised equations) and algebraic arguments, the actual algorithm is built. The main algorithmic features and parameters used by Makanin can be still recognised in the abstract notion.[3]

In Section 1 we establish the halting property for abstract Makanin algorithms. The abstract Makanin algorithm is parametrised by two "complexity parameters" that in the case of the actual Makanins algorithm are very large. In Section 2 we give an example of an abstract Makanin algorithm which, even for small "complexity parameters" is not primitive recursive. Since the algorithmic structure of the abstract algorithm is inherent in the algorithm presented in [10] this proves that the last one is not primitive recursive. Moreover, the example we give has been obtained by a simplification of the actual Makanins algorithm, and it mimics the actual behaviour of Makanin's algorithm working on "generalised equations".

By Ac we shall denote the Ackermann function i.e. the function defined by recursion as follows:

$$Ac(0,n) = n+1,$$

$$Ac(m+1,0) = Ac(m,1),$$

$$Ac(m+1,n+1) = Ac(m,Ac(m+1,n)).$$

It is well known that any function growing as fast or faster than the Ackermann function is not primitive recursive.

By $\mathbb{N}$ we denote the set of non-negative integers, $\mathbb{N}^+$ is the set of positive integers.

[2] The belief was justified by the fact that the bounds given in [10] were primitive recursive. In this paper, when talking about Makanin's algorithm for groups, we refer to the algorithm of [10] *together with the corrections made in* [11].

[3] See the comments after Definition 1.2.

1. Abstract Makanin algorithms

In this section we define the notion of an abstract Makanin algorithm and prove that it always terminates. We first give an auxiliary definition.

Definition 1.1. If $f: X \to X$, then we define a function $f^I: \mathbb{N} \times X \to X$ by putting $f^I(0;x) = x$ and $f^I(n+1;x) = f(f^I(n;x))$, for $n \in \mathbb{N}$.

Now we are ready to give the definition of the main notion of this section.

Definition 1.2. Let $\perp$ be an element which does not belong to $\mathbb{N}$. An abstract Makanin algorithm is a tuple $\langle E, r, p, s, e, f \rangle$ such that

(i) E is a set and $r: E \to E$,

(ii) $p: E \to \mathbb{N} \cup \{\perp\}$ is a function such that

(ii.1) $\{p(r^I(i;x)): i \in \mathbb{N}\}$ is finite, for each $x \in E$,

(ii.2) $r(x) = x$ if and only if $p(x) = \perp$,

(iii) $s: E \to \mathbb{N}$, $f: \mathbb{N} \to \mathbb{N}$, $f(n+1) > f(n)$, for $n \in \mathbb{N}$, and $s(r(x)) \leqslant f(s(x))$, for $x \in E$,

(iv) $e: \mathbb{N} \to \mathbb{N}$ is a non-decreasing function such that, if $x \in E$, $r(x) \neq x$, and $k \in \mathbb{N}^+$, then there exists $j < k$ such that $card(\{i < k: p(r^I(i;x)) = p(r^I(j;x))\}) \leqslant e(s(x))$.

Original Makanin's algorithm consists of a series of reductions of "generalised equations" that are executed until a "simple" equation is obtained. In the abstract notion defined above the set of generalised equations and the reduction procedure are represented, respectively, by the set E and the reduction function r. The set of simple equations is represented by those elements of E that cannot be further reduced, i.e. $r(x) = x$. The original Makanin's algorithm is non-deterministic. Our notion, describes a deterministic process, which corresponds to the behaviour of the original algorithm on one of the branches of the computation tree.

There are two parameters describing a generalised equation which are present in the notion of an abstract Makanin algorithm. These are the size (s), and the parameter functions (p), which, in the case of actual generalised equations, correspond respectively, to the length, and the "index" of a generalised equation – or the consecutive number of the "main dependence" of a generalised equation being considered, in the numbering of all dependencies. The size of an input x and the cardinality of the set $P(x)$, of parameters which appear in the process of reduction, determine the execution of an abstract Makanin algorithm for an input x. The functions e and f determine the behaviour of an abstract Makanin algorithm for all inputs. The function f bounds the growth of the size of a generalised equation in the process of reduction, and e describes the fact, that in every finite sequence of equations, which is obtained during the execution of an abstract Makanin algorithm, there is an equation, whose parameter does not appear too many times. Actual bounds for the functions corresponding to f and e in [11] are respectively $max\{2x, 1\}$ and F_2 (see Lemma 10.4 of [10] with the

corrections made in [11]), where F_2 is defined in [11] by a complicated recursion and grows faster than the Ackermann function, so is not primitive recursive.

We shall need the following easy fact.

Proposition 1.3. *If $\langle E, r, p, s, e, f\rangle$ is an abstract Makanin algorithm, and $n \leqslant m$, then*

$$s(r^I(n;x)) \leqslant f^I(m; s(x)) \tag{1}$$

for all $x \in E$.

Proof. The proof proceeds by induction. Clearly, by (iii) we have $f(m) \geqslant m$, for $m \in \mathbb{N}$. This easily gives the result for $n = 0$. Moreover, if $m \geqslant n+1$, and (1) holds for n, then

$$\begin{aligned} s(r^I(n+1;x)) &= s(r^I(n; r(x))) \leqslant f^I(m-1; s(r(x))) \leqslant f^I(m-1; f(s(x))) \\ &= f^I(m; s(x)). \qquad \square \end{aligned}$$

Below we introduce some notions that are not necessary to prove or state the results of this section, but fix intuitions and can help to understand the meaning of theorems, we are going to prove.

Definition 1.4. Let $\mathscr{A} = \langle E, r, p, s, e, f\rangle$ be an abstract Makanin algorithm. Then

(i) The execution of $\mathscr{A}$ on an input x is the sequence $\langle r^I(n;x)\colon n \in \mathbb{N}\rangle$. We put $P(x) = \{p(r^I(n;x))\colon n \in \mathbb{N}\} \setminus \{\perp\}$.

(ii) We put $H = \{x \in E\colon r(x) = x\}$, and we say that $\mathscr{A}$ halts for an input x after n steps, if $r^I(n;x) \in H$.

(iii) The functions e and f are called the complexity parameters of $\mathscr{A}$.

(iv) A function q is the time complexity of $\mathscr{A}$, if for each input $x \in E$, $\mathscr{A}$ halts after $\leqslant q(card(P(x)), s(x))$ steps, and q is minimal with this property, i.e., if for each $y \in E \setminus H$, there exists x such that $s(x) = s(y)$, $card(P(x)) = card(P(y))$, and $\mathscr{A}$ on the input x halts after exactly $q_x = q(card(P(x)), s(x))$ steps (i.e. $r^I(q_x; x) \in H$ and $r^I(q_x - 1; x) \notin H$).

Definition 1.5. Given functions $e, f : \mathbb{N} \to \mathbb{N}$ we define $g : \mathbb{N}^2 \to \mathbb{N}$ and $h : \mathbb{N}^3 \to \mathbb{N}$ as follows:

(i) $g(0,k) = 1$,
(ii) $h(0,t,k) = g(t,k)$,
(iii) $h(j+1,t,k) = h(j,t,k) + g(t, f^I(h(j,t,k); k))$,
(iv) $g(t+1,k) = h(e(k),t,k)$.

Before proceeding further we state some easy, but useful properties of functions f and g.

Lemma 1.6. *Functions g and h defined above have the following properties:*

(i) *they assume positive values,*

(ii) *h is an increasing function of the first variable,*
(iii) *if* $e(v) > 0$, *then* $g(i+1, v) > g(i, v)$,
(iv) *if e and f are non-decreasing, then g and h are non-decreasing functions of the last variable,*
(v) *if e and f are increasing and* $f(0) > 0$, *then, for all* $n, m \in \mathbb{N}$, (a) *and* (b) *below hold.*
  (a) $Ac(m, n) < g(m+1, n+1)$,
  (b) $Ac(m+1, n) < h(n+1, m+1, n+1)$,
(vi) *g and h are not primitive recursive.*

Proof. Parts (i)–(iii) are obvious, (iv) follows by a routine induction, and (vi) is an immediate consequence of (v).

The proof of (v) uses parts (i)–(iv) and the fact that if $f(0) > 0$, for an increasing function f then

$$f^I(x; y) \geqslant x + y.$$

We proceed by induction as follows.

To get (a) for $m = 0$, we notice that

$$g(1, n+1) = h(e(n+1), 0, n+1) \geqslant e(n+1) + 1 \geqslant n + 2 > Ac(0, n).$$

If (a) holds for a fixed m and every n, then (b) holds for m and $n = 0$. In fact, we have $h(0, m+1, 1) \geqslant 1$, therefore $f^I(h(0, m+1, 1); 1) \geqslant h(0, m+1, 1) + 1 \geqslant 2$, and finally

$$h(1, m+1, 1) > g(m+1, f^I(h(0, m+1, 1); 1) \geqslant g(m+1, 2) > Ac(m, 1) = Ac(m+1, 0).$$

To prove, that (b) for m and n implies (b) for m and $n+1$ notice that

$$\begin{aligned} h(n+2, m+1, n+2) &> g(m+1, f^I(h(n+1, m+1, n+2); n+2)) \\ &\geqslant g(m+1, h(n+1, m+1, n+2) + n + 2)) \\ &\geqslant g(m+1, h(n+1, m+1, n+1) + 1) \\ &\geqslant g(m+1, Ac(m+1, n) + 1) \\ &> Ac(m, Ac(m+1, n)) = Ac(m+1, n+1). \end{aligned}$$

Finally if (b) holds for a fixed m and arbitrary n then

$$g(m+2, n+1) = h(e(n+1), m+1, n+1) \geqslant h(n+1, m+1, n+1) > Ac(m+1, n),$$

and (a), for $m+1$ and every n, follows. □

Theorem 1.7. *If* $\mathscr{A} = \langle E, r, p, s, e, f \rangle$ *is an abstract Makanin algorithm then, for each input* $x \in E$, $\mathscr{A}$ *halts after* $g(card(P(x)), s(x)) - 1$ *steps. In other words, for every* $x \in E$, $r^I(g(card(P(x)), s(x)) - 1; x) \in H$.

Proof. Let $k>0$, and put $P_k(x)=\{p(r^I(i;x)): i<k\}$. We claim that

$$\text{if } r^I(k-1;x)\notin H \text{ and } card(P_k(x))\leqslant t, \quad \text{then } k<g(t,s(x)). \tag{2}$$

We shall prove (2) by induction on t. It is obvious that (2) holds for $t=0$. Now, as the main induction hypothesis, assume that (2) holds for t.

Having t fixed, we shall prove by induction on j that

$$\text{if } r^I(k-1;x)\notin H, \quad \text{and for some } a\in P(x), \quad card(\{a\}\cup P_k(x))\leqslant t+1$$
$$\text{and} \quad card(\{i<k: p(r^I(i;x))=a\})=j, \quad \text{then } k<h(j,t,s(x)). \tag{3}$$

If $j=0$, then $P_k(x)$ has at most t elements, so by the main induction hypothesis, we have $k<g(t,s(x))=h(0,t,s(x))$.

Now, assume that $r^I(k-1;x)\notin H$, that (3) holds for t and j, and that

$$card(\{i<k: p(r^I(i;x))=a\})=j+1.$$

Let l be the greatest integer such that $l<k$, and $p(r^I(l;x))=a$. We claim that

$$l<h(j,t,s(x)) \tag{4}$$

and

$$k-l-1<g(t,s(r^I(l+1;x)))\leqslant g(t,f^I(h(j,t,s(x));s(x))). \tag{5}$$

Clearly (4) holds if $l=0$. Moreover, $card(\{i<l: p(r^I(i;x))=a\})=j$, so for $l>0$, (4) follows from the hypothesis that (3) holds for t and j.

Of course, (5) holds if $k-l-1=0$. Moreover, $card(\{p(r^I(i;r^I(l+1;x)): i<k-l-1\})\leqslant t$, so for $k-l-1>0$, we can apply (2) for t to get the first inequality of (5). Now, Proposition 1.3, and Lemma 1.6(iv) imply that $s(r^I(l+1;x))\leqslant f^I(l+1;s(x))$, which finishes the proof of (5).

Using (4) and (5) we finally get

$$k=l+1+(k-1-l)<h(j,t,s(x))+g(t,f^I(h(j,t,s(x));s(x)))=h(j+1,t,s(x))$$

so (3) holds for all $j\in\mathbb{N}$, and t fixed above.

Now, to finish the proof of (2) notice that by Definition 1.2(iv), there is an $a\in P_k(x)$ such that $card\{i<k: p(r^I(i;x)=a\}\leqslant e(s(x))$, hence $k<h(e(s(x)),t,s(x))=g(t+1, s(x))$.

To conclude the proof of the theorem, let $t=card(P(x))$ and suppose that $r^I(g(t,s(x))-1;x)\notin H$. Then an application of (2) to $k=g(t,s(x))$ gives $g(t,s(x))<g(t,s(x))$, so, we have arrived at a contradiction. Therefore $r^I(g(card(P(x)),s(x))-1;x)\in H$. □

We have proved the halting property of abstract Makanin algorithm. However, by Lemma 1.6(vi) function g, which bounds the number of steps of an abstract Makanin algorithm, is not primitive recursive.

2. An example

Now, we are going to construct an example of an abstract Makanin algorithm, for which the number of steps cannot be bounded by a primitive recursive function, even if the complexity parameters are given by slowly growing, increasing functions. First, we shall roughly describe our construction in a non-formal way. The objects upon which our algorithm operates will be pairs $\langle \vec{a}; v\rangle$, where $\vec{a}$ is a finite sequence of fixed length (vector) of non-negative integers and v is a (integer) parameter which controls the reduction process of $\vec{a}$. Initially $\vec{a}=\langle a_0,\dots,a_m\rangle$, and v has some initial value. At each step of the algorithm the control value v is changed to $f(v)$, for some fixed function f, and the first non-zero coordinate of $\vec{a}$ is decremented by 1 (so, the algorithm stops if $\vec{a}=\langle 0,\dots,0\rangle$). Moreover, if the coordinate, which is decremented, is not the first one, then all smaller coordinates are given new, initial positive value $e(v)$, which is computed from the control parameter v using a fixed function e.

Example 2.1. Let $f(v)=v+1$, $e(v)=v$. Let $\langle \vec{a}; v\rangle \to \langle \vec{b}; w\rangle$ denote that $\langle \vec{b}; w\rangle$ is obtained from $\langle \vec{a}; v\rangle$ in one step. Finally, let $\xrightarrow{*}$ be the transitive closure of $\to$. Then we have

$$\begin{aligned}&\langle 0,0,0,1;1\rangle \to \langle 1,1,1,0;2\rangle \to \langle 0,1,1,0;3\rangle \to \langle 3,0,1,0;4\rangle \to \langle 2,0,1,0;5\rangle\\ &\xrightarrow{*} \langle 0,0,1,0;7\rangle \to \langle 7,7,0,0;8\rangle \xrightarrow{*} \langle 0,7,0,0;15\rangle \to \langle 15,6,0,0;16\rangle\\ &\xrightarrow{*} \langle 0,6,0,0;31\rangle \to \langle 31,5,0,0;32\rangle \xrightarrow{*} \langle 0,5,0,0;63\rangle \to \langle 63,4,0,0;64\rangle,\end{aligned}$$

and so on.

Now we shall give a formal definition of a function *next*, which will be used to construct an abstract Makanin algorithm.

Definition 2.2. Given functions $e, f : \mathbb{N} \to \mathbb{N}$ we define a function

$$next: \bigcup_{m\in\mathbb{N}} (\mathbb{N}^{m+1}\times\mathbb{N}) \to \bigcup_{m\in\mathbb{N}} (\mathbb{N}^{m+1}\times\mathbb{N})$$

as follows:

$$next(\langle a_0,\dots,a_m; v\rangle) = \begin{cases} \langle a_0,\dots,a_m; v\rangle & \text{if } \langle a_0,\dots,a_m\rangle = \langle 0,\dots,0\rangle,\\ \langle b_0,\dots,b_m; f(v)\rangle & \text{otherwise,}\end{cases}$$

where

$$b_i = \begin{cases} e(v) & \text{if for each } i'\leqslant i,\quad a_{i'}=0,\\ a_i - 1 & \text{if } a_i>0 \text{ and for each } i'<i,\quad a_{i'}=0,\\ a_i & \text{otherwise.}\end{cases}$$

If $\langle a_0,\dots,a_m;v\rangle \in \mathbb{N}^{m+1}\times\mathbb{N}$, then we define $(\langle a_0,\dots,a_m;v\rangle)_i = a_i$, for $i\leqslant m$, and $(\langle a_0,\dots,a_m;v\rangle)_{m+1} = v$.

Let $i\leqslant m+1$. An element $\langle a_0,\dots,a_m;v\rangle \in \mathbb{N}^{m+1}\times\mathbb{N}$ is called a zero of degree i, if $a_n = 0$, for all $n < i$, and is called a zero of degree exactly i if in addition $a_i \neq 0$. A zero of degree exactly i, for $i < m+1$ is called proper.

Lemma 2.3. *Assume that $i\leqslant m$, $\langle\vec{a};v\rangle \in \mathbb{N}^{m+2}$ is a proper zero of degree i. Then for each $j\in\mathbb{N}$*

(i) *$next^I(g(i,v);\langle\vec{a};v\rangle)$ is a zero of degree i,*

(ii) *if $0 < l < g(i,v)$, then $next^I(l;\langle\vec{a};v\rangle)$ is not a zero of degree i,*

(iii) *$next^I(h(j,i,v);\langle\vec{a};v\rangle)$ is a zero of degree i,*

(iv) *if $0 < j\leqslant(next(\langle\vec{a};v\rangle))_i$ and $h(j-1,i,v) < l < h(j,i,v)$, then $next^I(l;\langle\vec{a};v\rangle)$ is not a zero of degree i.*

(v) *if $j\leqslant(next(\langle\vec{a};v\rangle))_i$, then $(next^I(h(j,i,v);\langle\vec{a};v\rangle))_i = (next(\langle\vec{a};v\rangle))_i - j$.*

Proof. The proof is divided into two parts.

Part 1: In the first part we shall prove that if (i) and (ii) hold for a fixed i and every proper zero $\langle\vec{a};v\rangle$ of degree i, then (iii)–(v) hold for i, every proper zero $\langle\vec{a};v\rangle$ of degree i, and every $j\in\mathbb{N}$.

We proceed by induction on j. Let $\langle\vec{a};v\rangle$ be a proper zero of degree i.

Let $j = 0$. Clearly (iv) holds. Moreover, it is easy to see that (i) implies (iii). To prove (v) notice that if x is not a zero of degree i, then by the definition of *next*

$$(x)_i = (next(x))_i. \tag{6}$$

Therefore, (v) follows by (ii) and the definition of h.

Now, assume that (iii)–(v) hold for j. Notice that

$$next^I(h(j+1,i,v);\langle\vec{a};v\rangle) = next^I(g(i,f^I(h(j,i,v);v));next^I(h(j,i,v);\langle\vec{a};v\rangle)).$$

If $next^I(h(j,i,v);\langle\vec{a};v\rangle)$ is a proper zero, then by the definition of *next*

$$next^I(h(j,i,v);\langle\vec{a};v\rangle) = \langle\vec{a}';f^I(h(j,i,v);v)\rangle$$

for some $\vec{a}'$ and (iii) follows from (i). The case when $next^I(h(j,i,v);\langle\vec{a};v\rangle)$ is a non-proper zero, is obvious.

In a similar way we can prove (iv) using (ii). We use the fact, which easily follows from the induction hypothesis on (v), that $next^I(h(j,i,v);\langle\vec{a};v\rangle)$ is a proper zero of degree i, for $j < (next(\langle\vec{a};v\rangle))_i$.

Now, to prove (v), let $j+1\leqslant(next(\langle\vec{a};v\rangle)_i$. By induction hypothesis that (iii) and (v) hold for j, it follows that $next^I(h(j,i,v);\langle\vec{a};v\rangle)$ is a zero of degree exactly i. Let $l = f^I(h(j,i,v);v)$. Then for some $\vec{a}'$

$$next^I(h(j,i,v);\langle\vec{a};v\rangle) = \langle\vec{a}';f^I(h(j,i,v);v)\rangle = \langle\vec{a}';l\rangle.$$

Now, we have

$$\begin{aligned}(next^I(h(j+1,i,v);\langle\vec{a};v\rangle))_i &= (next^I(g(i,l);\langle\vec{a}';l\rangle))_i\\ &= (next^I(1;\langle\vec{a}';l\rangle))_i\\ &= (\langle\vec{a}';l\rangle)_i - 1\\ &= (next(\langle\vec{a};v\rangle)_i - j) - 1\\ &= (next(\langle\vec{a};v\rangle))_i - (j+1),\end{aligned} \tag{7}$$

where the second equality follows from (6) and (ii), the third from the property that $\langle\vec{a}';l\rangle = next^I(h(j,i,v);\langle\vec{a};v\rangle)$ is a zero of degree exactly i and the fourth from (v) for j. Clearly (7) proves (v) for $j+1$.

Part 2: Now, using Part 1, by induction on i, we shall prove that (i) and (ii) hold for each proper zero $\langle\vec{a};v\rangle$ of degree i. Let $i=0$. Clearly each $\langle\vec{a};v\rangle \in \mathbb{N}^{m+2}$ is a zero of degree 0, so (i) holds, and (ii) follows from the definition of g.

Now, assume, as an induction hypothesis, that (i) and (ii) hold for i and

$$\langle\vec{a};v\rangle \text{ is a proper zero of degree } i+1. \tag{8}$$

By Part 1(iii), $next^I(g(i+1,v);\langle\vec{a};v\rangle) = next^I(h(e(v),i,v);\langle\vec{a};v\rangle)$ is a zero of degree i. Moreover, since $(next(\langle\vec{a};v\rangle)_i = e(v)$, by (v) we get

$$(next^I(g(i+1,v);\langle\vec{a};v\rangle))_i = (next^I(h(e(v),i,v);\langle\vec{a};v\rangle))_i = 0, \tag{9}$$

so $next^I(g(i+1,v);\langle\vec{a};v\rangle)$ is a zero of degree $i+1$, and (i) follows.

To prove (ii) notice that if

$$0 < l < g(i+1,v) = h(e(v),i,v),$$

then either $0 < l < g(i,v) = h(0,i,v)$, or $l = h(j,i,v)$ for some $j < e(v)$, or there exists j, $0 < j \leqslant e(v)$ such that $h(j-1,i,v) < l < h(j,i,v)$. In the second case, by (iii) and (v), $next^I(l;\langle\vec{a};v\rangle)$ is a zero of degree exactly i, so it is not a zero of degree $i+1$. In the first and the third case, $next^I(l;\langle\vec{a};v\rangle)$ is not a zero of degree i, respectively by the induction hypothesis (ii) and by (iv), and hence it is not a zero of degree $i+1$. □

Proposition 2.4. *If* $l \geqslant 0$, *then*

$$(next^I(l;\langle\underbrace{0,\ldots,0}_{m \text{ times}},1;v\rangle)_m = 0.$$

Proof. Obvious. □

Definition 2.5. Let $e, f : \mathbb{N} \to \mathbb{N}$. Then we define $\mathscr{A} = \langle E, r, p, s, e, f\rangle$ as follows:

$$E = \{next^I(k; \langle \underbrace{0, \ldots, 0}_{m \text{ times}}, 1; v\rangle)\colon k, m, v \in \mathbb{N}^+\},$$

$$r(\langle \vec{a}, v\rangle) = next(\langle \vec{a}, v\rangle) \text{ for all } (\langle \vec{a}, v\rangle) \in E,$$

$$p(\langle a_0, \ldots, a_m; v\rangle) = \begin{cases} \perp & \text{if for every } j \leqslant m, \quad a_j = 0, \\ i & \text{if } a_i > 0, \text{ and for every } j < i, \quad a_j = 0, \end{cases}$$

$$f^{-1}(v) = \min\{i\colon f(i) \geqslant v\},$$

$$s(\langle \vec{a}, v\rangle) = f^{-1}(v).$$

Proposition 2.6. *Let* $P^+(x) = \{p(r^I(x; i)\colon i \in \mathbb{N}^+\} \setminus \{\perp\}$. *If* $\langle \vec{a}; v\rangle$ *is a zero of degree exactly* n *and* $e(v) > 0$, *then*

$$P^+(\langle \vec{a}; v\rangle) \supseteq \{0, \ldots, n-1\}.$$

Proof. Assume that $\langle \vec{a}; v\rangle$ is a zero of degree exactly n, and let $i < n$. By Lemma 2.3(i) and (ii) it follows that $r^I(g(i, v); \langle \vec{a}; v\rangle)$ is a zero of degree exactly i. Hence, $p(r^I(g(i, v); \langle \vec{a}; v\rangle)) = i$. □

Theorem 2.7. *If* $\mathscr{A} = \langle E, r, p, s, e, f\rangle$, *where* e *is non-decreasing and* f *is increasing, then* $\mathscr{A}$ *is an abstract Makanin algorithm.*

Proof. It is clear that (i) and (ii) of Definition 1.2 hold. Moreover, it is easy to prove that $s(r(\langle \vec{a}; v\rangle)) \leqslant f(s(\langle \vec{a}; v\rangle))$, so (iii) holds too.

To prove (iv) let $\langle \vec{a}; v\rangle \in \mathbb{N}^{m+2} \cap (E \setminus H)$, and let $k \in \mathbb{N}$. We can assume that $r^I(k-1;\ \langle \vec{a}; v\rangle) \notin H$.

Let

$$i = \max\{p(r^I(j; \langle \vec{a}; v\rangle))\colon j < k \text{ and } r^I(j; \langle \vec{a}; v\rangle) \notin H\}.$$

By Proposition 2.4, $p(x) < m$, for each $x \in \mathbb{N}^{m+2} \cap E$, so $i < m$.

By the definition of E,

$$\langle \vec{a}; v\rangle = next^I(n; \langle \underbrace{0, \ldots, 0}_{m \text{ times}}, 1; v'\rangle)$$

for some $n, v' \in \mathbb{N}^+$. Let

$$n_0 = \max\{j < n\colon next^I(j; \langle \underbrace{0, \ldots, 0}_{m \text{ times}}, 1; v'\rangle) \text{ is a zero of degree } i+1\},$$

and let

$$\langle \vec{a}_0; v_0\rangle = next^I(n_0; \langle \underbrace{0, \ldots, 0}_{m \text{ times}}, 1; v'\rangle).$$

Clearly, $\langle \vec{a}_0; v_0 \rangle$ is a proper zero of degree $i+1$.

We claim that

$$k \leqslant g(i+1, v_0) + n_0 - n. \tag{10}$$

Otherwise, by the definition of i

$$p(r^I(g(i+1, v_0) + n_0 - n; \langle \vec{a}; v \rangle)) \leqslant i.$$

But $r^I(g(i+1, v_0) + n_0 - n; \langle \vec{a}; v \rangle) = next^I(g(i+1, v_0); \langle \vec{a}_0; v_0 \rangle)$ and by Lemma 2.3 (i), $next^I(g(i+1, v_0); \langle \vec{a}_0; v_0 \rangle)$ is a zero of degree $i+1$. So

$$p(r^I(g(i+1, v_0) + n_0 - n; \langle \vec{a}; v \rangle)) > i,$$

and we get a contradiction.

Now, we shall prove that

$$\begin{aligned} &\text{if } p(r^I(j; \langle \vec{a}; v \rangle)) = i \quad \text{for some } j < k, \\ &\text{then } h(l, i, v_0) = j + n - n_0 \quad \text{for some } l < e(v_0). \end{aligned} \tag{11}$$

Let $j < k$. Then by (10), $j + n - n_0 < g(i+1, v_0) = h(e(v_0), i, v_0)$. Consequently, either $j + n - n_0 < h(0, i, v_0)$ or, $h(l, i, v_0) \leqslant j + n - n_0 < h(l+1, i, v_0)$, for some $l < e(v_0)$.

The first case $0 < j + n - n_0 < g(i, v_0)$, so by Lemma 2.3(ii), $next^I(j + n - n_0; \langle \vec{a}_0; v_0 \rangle)$ equals $r^I(j; \langle \vec{a}; v \rangle)$, and therefore is not a zero of degree i, contrary to the assumption. In the second case Lemma 2.3(ii) implies that $j + n - n_0 = h(l, i, v)$, for some $l < e(v_0)$.

Since h is increasing with respect to the first argument, by (11), $p(r^I(j; \langle \vec{a}; v \rangle)) = i$ holds for at most $e(v_0)$ many integers $j < k$.

To conclude the proof of (iv) it suffices to notice that

$$v_0 \leqslant f^I(n - n_0 - 1; v_0) = f^{-1}(f^I(n - n_0; v_0)) = f^{-1}(v) = s(\langle \vec{a}; v \rangle)$$

which implies that $e(v_0) \leqslant e(s(\langle \vec{a}; v \rangle))$. □

Theorem 2.8. *Let $\mathscr{A} = \langle E, r, p, s, e, f \rangle$ be an abstract Makanin algorithm and let g be defined from e, f by Definition 1.5. If $e(1) > 0$, then the time complexity of $\mathscr{A}$ is $g(\cdot, \cdot) - 1$.*

More formally, for each $m, v \in \mathbb{N}^+$, there exists $x \in E$, for which $card(P(x)) = m$, $s(x) = v$, and $r^I(g(m, v) - 2; x) \notin H$.

Proof. Of course we have

$$H = \{\langle \underbrace{0, \ldots, 0}_{m+1 \text{ times}}; v \rangle \in E\colon\ m, v \in \mathbb{N}^+\}.$$

Let $m, v \in \mathbb{N}^+$ and let $x = next(\langle \underbrace{0, \ldots, 0}_{m \text{ times}}, 1; v \rangle)$. By Proposition 2.6, we have

$$P(x) \supseteq P^+(\langle \underbrace{0, \ldots, 0}_{m \text{ times}}, 1; v \rangle) \supseteq \{0, \ldots, m-1\}.$$

On the other hand, by the definition of p, and Proposition 2.4 we get $p(r^I(l;x)) < m$, for any l such that $r^I(l;x) \notin H$, so $card(P(x)) = m$.

By Lemma 2.3(ii),

$$r^I(g(m,s(x)) - 2; x) = next^I(g(m,v) - 1; \langle \underbrace{0,\ldots,0}_{m \text{ times}}, 1; v\rangle) \notin H. \qquad \square$$

To conclude, let us formulate the main result of this section. Notice, that we can obtain abstract algorithms which are not primitive recursive, even if we require that complexity parameters grow very slowly. As mentioned earlier, the complexity parameters in the original Makanin's algorithm (especially F_2) grow very fast.

Corollary 2.9. *If $e, f : \mathbb{N} \to \mathbb{N}$ are increasing and $f(0) > 0$, then there exists an abstract Makanin algorithm with complexity parameters e and f, whose complexity is not primitive recursive.*

Proof. Immediate from Theorem 1.6(v) and Theorem 2.8. $\square$

References

[1] H. Abdulrab and J.P. Pecuchet, Solving word equations, *J. Symbolic Comput.* **8** (1989) 499–521.

[2] S.I. Adyan and G.S. Makanin, Investigation on algorithmic questions of algebra, *Trudy Matem. Inst. Steklova* **168** (1984) 197–217 (in Russian). English translation in *Proc. Steklov Inst. Math.* **3** (1986) 207–226.

[3] J. Jaffar, Minimal and complete word unification, *J. Assoc. Comput. Mach.* **37** (1990) 47–85.

[4] Yu.I. Khmelevskiĭ, Solution of word equations in three unknowns, *Dokl. Akad. Nauk SSSR* **177** (1967) 1023–1025 (in Russian).

[5] A. Kościelski, An analysis of Makanin's algorithm deciding solvability of equations in free groups, in: K. Schultz, ed., *Word Equations and Related Topics*, Lecture Notes in Computer Science, Vol. 572 (Springer, Berlin, 1990) 12–60.

[6] A. Kościelski and L. Pacholski, Complexity of Makanin's algorithm, *J. Assoc. Comput. Mach.* **43** (1996) 670–684.

[7] A. Kościelski and L. Pacholski, Complexity of unification in free groups and free semigroups, in: *Proc. 31st Ann. IEEE Symp. on Foundations of Computer Science*, Los Alamitos (1990) 824–829.

[8] A. Lentin, Equations in free monoids, in Nivat, ed., *Automata, Languages and Programming* (1972) 67–85.

[9] G.S. Makanin, The problem of solvability of equations in a free semigroup, *Mat. Sbornik* **103** (1977) 147–236 (in Russian). English translation in *Math. USSR Sbornik* **32** (1977) 129–198.

[10] G.S. Makanin, Equations in a free group, *Izvestiya AN SSSR* **46** (1982) 1199–1273 (in Russian). English translation in *Math. USSR Izvestiya* **21** (1983) 483–546.

[11] G.S. Makanin, Decidability of the universal and positive theories of a free group, *Izvestiya AN SSSR* **48** (1984) 735–749 (in Russian). English translation in *Math. USSR Izvestiya* **25** (1985) 75–88.

[12] J.P. Pecuchet, Solutions principales et rang d'un système d'équations avec constantes dans le monoïde libre, *Discrete Math.* **48** (1984) 253–274.

[13] G. Plotkin, Building in equational theories, *Machine Intelligence* **7** (1972) 73–90.

[14] A.A. Razborov, On systems of equations in a free group, *Izvestiya AN SSSR* **48** (1984) 779–832 (in Russian). English translation in *Math. USSR Izvestiya* **25** (1985) 115–162.

[15] K.U. Schulz, Word unification and transformation of generalized equations, *J. Automated Reasoning* **11** (1993) 149–184.