Artur Jeż | Alexander Okhotin

Language equations with addition in positional notation

Turku Centre *for* Computer Science

TUCS Technical Report
No 824, August 2007

Language equations with addition in positional notation

Artur Jeż
Institute of Computer Science, University of Wrocław
ul. Joliot-Curie 15, 50–383 Wrocław, Poland
aje@ii.uni.wroc.pl
Alexander Okhotin
Academy of Finland, *and*
Department of Mathematics, University of Turku, *and*
Turku Centre for Computer Science
Turku FIN–20014, Finland
alexander.okhotin@utu.fi

TUCS Technical Report
No 824, August 2007

Abstract

Language equations with an operation of adding numbers written in a positional notation are considered. It is shown that this operation together with union and intersection invests equations with a sufficient expressive power to simulate every trellis automaton, as well as to specify some languages not accepted by any trellis automaton. The results have applications to conjunctive grammars over a unary alphabet and to language equations of a general form.

Keywords: language equations, conjunctive grammars, unary languages

TUCS Laboratory
Discrete Mathematics for Information Technology

1 Introduction

Resolved systems of language equations of the general form

$$\begin{cases} X_1 &= \varphi_1(X_1, \ldots, X_n) \\ &\vdots \\ X_n &= \varphi_n(X_1, \ldots, X_n) \end{cases} \qquad (*)$$

with different operations allowed in their right-hand sides are the oldest and likely the most studied type of language equations.

As established by Ginsburg and Rice [3], if the allowed operations in (*) are concatenation and union, then least solutions of such systems provide semantics to the context-free grammars. If in addition the intersection operation may be used, the resulting systems define the class of *conjunctive grammars* introduced by Okhotin [8, 9]; these grammars have attractive practical properties, such as efficient parsing algorithms, and are surveyed in a recent article [15]. The case of systems (*) using negation but neither union nor intersection was first considered by Leiss [7] and recently studied by Okhotin and Yakimova [14]; by their expressive power, these systems are incomparable with the context-free grammars.

An important case of such equations is the case of a unary alphabet. In the case of union and concatenation, it is well-known that the unary context-free languages are regular. In contrast, equations with concatenation and complementation can specify the nonregular language $\{a^n \,|\, \text{the octal notation of } n \text{ starts with } \mathtt{1}, \mathtt{2} \text{ or } \mathtt{3}\}$, see Leiss [7]. As for equations with union, intersection and complementation, their expressive power over a unary alphabet (or, equivalently, the expressive power of unary conjunctive grammars) was one of the long-standing open problems in the area [8, 15], and it was conjectured that only regular languages can be obtained.

This conjecture has recently been disproved by Jeż [6] by constructing a conjunctive grammar for the language $\{a^{4^n} \mid n \in \mathbb{N}\}$. This grammar, written in the form of a system of language equations, is as follows:

$$\begin{cases} X_1 &= (X_2X_2 \cap X_1X_3) \cup a \\ X_2 &= (X_{12}X_2 \cap X_1X_1) \cup aa \\ X_3 &= (X_{12}X_{12} \cap X_1X_2) \cup aaa \\ X_{12} &= (X_3X_3 \cap X_1X_2) \end{cases} \qquad (1)$$

Its least solution is $X_i = \{a^\ell \mid \text{base-4 notation of } \ell \text{ is } i\mathtt{0}\ldots\mathtt{0}\}$, for $i = 1, 2, 3, 12$.

The system (1) effectively encodes manipulations with the positional notation of numbers. In order to continue the study of unary language equations, it would be convenient to deal explicitly with numbers written in positional notation. The subject of the present paper are equations on languages over alphabets $\Sigma_k = \{\mathtt{0}, \mathtt{1}, \ldots, k-1\}$, in which every word is interpreted

as k-ary notation of a number. Instead of the concatenation operation we shall use addition of numbers written in k-ary notation. Our goal is to establish the expressive power of such equations and apply these results to unary language equations.

2 Languages of numbers written in positional notation

Fix a number $k \geqslant 2$ and consider the alphabet $\Sigma_k = \{\mathtt{0}, \mathtt{1}, \mathtt{2}, \ldots, k-1\}$ of k-ary digits. Words over this alphabet represent non-negative integers written in k-ary notation. Let the empty word $\varepsilon \in \Sigma_k^*$ denote the number $\mathtt{0}$. No representation of a number shall begin with $\mathtt{0}$, that is, the set of valid representations of numbers is $\Sigma_k^* \setminus \mathtt{0}\Sigma_k^*$. We shall consider formal languages over this alphabet, such as the following language of binary notations of all powers of two: $\mathtt{10}^* \subseteq \Sigma_2^*$.

Define a word operation $\boxplus_k : \Sigma_k^* \times \Sigma_k^* \to \Sigma_k^*$, which represents addition of numbers in k-ary notation:

$$u \boxplus_k v = \{\text{the } k\text{-ary notation of } i+j \mid u \text{ is the } k\text{-ary notation of } i, \; v \text{ is the } k\text{-ary notation of } j\}$$

The notation $\boxplus$ will be used when the alphabet is clear from the context. We shall use this operation in the language-theoretic rather than arithmetical context. Then it can be said that $u \boxplus v$ combines the corresponding symbols of u and v and thus computes a certain word of length $\max(|u|, |v|)$ or $\max(|u|, |v|) + 1$. This, in particular, can be used to modify individual symbols of a word:

Example 2.1 (Modifying a digit)**.** *Let $uiv \in \Sigma_k^* \setminus \mathtt{0}\Sigma_k^*$, let $i \neq k-1$. Then $uiv \boxplus \mathtt{10}^{|v|} = u(i+1)v$, that is, one symbol has been modified.*

Note that such a modification is irreversible: there is no $w \in \Sigma_k^*$, such that $u(i+1)v \boxplus_k w = uiv$. Define its inverse, $w \boxminus_k u$, as such a word v that $u \boxplus v = w$.

Let us extend the operation of k-ary addition to languages in the standard way as $K \boxplus L = \{u \boxplus v \mid u \in K, v \in L\}$. Then, for instance, for $k = 10$ it can be said that $\mathtt{9}^+ \boxplus \mathtt{2} = \mathtt{10}^*\mathtt{1}$.

By definition, this operation on languages is *monotone* with respect to the partial ordering of languages by inclusion, that is, whenever $K \subseteq K'$ and $L \subseteq L'$, it holds that $K \boxplus K' \subseteq L \boxplus L'$. It is also *continuous*, in the sense that for every two increasing sequences of languages $\{K_n\}_{n=1}^{\infty}$ and $\{L_n\}_{n=1}^{\infty}$, the sequence $\{K_n \boxplus L_n\}_{n=1}^{\infty}$ has the least upper bound $\bigsqcup_{n>0} K_n \boxplus L_n = \bigsqcup_{n>0} K_n \boxplus \bigsqcup_{n>0} L_n$. These properties are essential for considering language equations with this operator. By the basic results on fixed points, every resolved system

of equations $X_i = \varphi_i(X_1, \ldots, X_n)$ $(i = 1, \ldots, n)$ using only monotone and continuous operations (in particular $\boxplus$, $\cup$ and $\cap$) has a least solution given by $\bigsqcup_{n>0} \varphi^n(\varnothing, \ldots, \varnothing)$, where φ is a vector notation for $(\varphi_1, \ldots, \varphi_n)$.

Our primary motivation for studying these equations is their correspondence to language equations over an alphabet $\{a\}$. Define the bijection $f_k : \Sigma_k^* \setminus 0\Sigma_k^* \to a^*$ as

$$f_k(w) = a^n, \quad \text{where } w \text{ read as } k\text{-ary notation represents } n.$$

This mapping is an isomorphism, since $f_k(u \boxplus_k v) = f_k(u) \cdot f_k(v)$ and $f_k^{-1}(a^m \cdot a^n) = f_k^{-1}(a^m) \boxplus f_k^{-1}(a^n)$. Extend it to languages in a usual way as $f_k(L) = \{f_k(w) \,|\, w \in L\}$, obtaining an isomorphism between unary languages and subsets of $\Sigma_k^* \setminus 0\Sigma_k^*$. This isomorphism extends to systems of language equations over Σ_k using Boolean operations and $\boxplus$, and language equations over $\{a\}$ using Boolean operations and concatenation: $\boxplus$ is replaced with $\cdot$, constants are mapped by f_k, Boolean operations are preserved, and then the solutions of equations correspond as follows:

Proposition 2.1. *Let $\varphi_i(X_1, \ldots, X_n) = \psi_i(X_1, \ldots, X_m)$ be a system of language equations over the alphabet $\{a\}$ and let $\widetilde{\varphi}_i(Y_1, \ldots, Y_n) = \widetilde{\psi}_i(Y_1, \ldots, Y_m)$ be the corresponding language equations over Σ_k. Then a vector of languages $(f_k(L_1), \ldots, f_k(L_n))$ is a solution of the former system if and only if the vector of languages $(L_1, \ldots, L_n)$ is a solution of the latter system. In particular, least solutions are mapped to least solutions and greatest solutions are mapped to greatest solutions.*

The proof is by a straightforward structural induction.

This bijection can be applied to convert between different bases of positional notation. Let $k, \ell \geqslant 2$. For any system of equations over Σ_k the corresponding system of equations over Σ_ℓ is defined by mapping the given base-k system into a unary system and then mapping it back to a base-ℓ system. The transformation preserves the structure of the equations: Boolean operations remain, $\boxplus_k$ is replaced with $\boxplus_\ell$, constants are mapped by the composition of functions $f_\ell^{-1} \circ f^k : \Sigma_k^* \setminus 0\Sigma_k^* \to \Sigma_\ell^* \setminus 0\Sigma_\ell^*$, and the following result easily follows:

Proposition 2.2. *Let $\varphi_i(X_1, \ldots, X_n) = \psi_i(X_1, \ldots, X_m)$ be a system of language equations over the alphabet Σ_k and let $\widetilde{\varphi}_i(Y_1, \ldots, Y_n) = \widetilde{\psi}_i(Y_1, \ldots, Y_m)$ be the corresponding language equations over Σ_ℓ. Then a vector of languages $(L_1, \ldots, L_n)$ is a solution of the former system if and only if the vector of languages $\big(f_\ell^{-1}(f_k(L_1)), \ldots, f_\ell^{-1}(f_k(L_n))\big)$ is a solution of the latter system. In particular, least solutions are mapped to least solutions and greatest solutions are mapped to greatest solutions.*

3 Known representations

Denote by $\mathcal{L}^k_{\cup,\cap,\boxplus}$ the family of languages that occur in least solutions of systems of equations $Y_i = \psi_i(Y_1, \ldots, Y_n)$ over Σ_k, with union, intersection and $\boxplus_k$. Clearly, for every ultimately periodic set of numbers X, $f^{-1}(X)$ is in $\mathcal{L}^k_{\cup,\cap,\boxplus}$. By Proposition 2.1, the system (1) with a nonperiodic solution translates to the following:

Example 3.1. *The following system of language equations over* $\Sigma_4 = \{\mathtt{0},\mathtt{1},\mathtt{2},\mathtt{3}\}$

$$\begin{cases} X_1 &= (X_2 \boxplus X_2 \cap X_1 \boxplus X_3) \cup \{\mathtt{1}\} \\ X_2 &= (X_{12} \boxplus X_2 \cap X_1 \boxplus X_1) \cup \{\mathtt{2}\} \\ X_3 &= (X_{12} \boxplus X_{12} \cap X_1 \boxplus X_2) \cup \{\mathtt{3}\} \\ X_{12} &= X_3 \boxplus X_3 \cap X_1 \boxplus X_2 \end{cases}$$

has the least solution $(\mathtt{10}^*, \mathtt{20}^*, \mathtt{30}^*, \mathtt{120}^*)$.

Consider the equation for X_1 under this substitution: $X_2 \boxplus X_2 = \mathtt{20}^* \boxplus \mathtt{20}^* = \mathtt{10}^+ \cup \mathtt{20}^*\mathtt{20}^*$ and $X_1 \boxplus X_3 = \mathtt{10}^* \boxplus \mathtt{30}^* = \mathtt{10}^+ \cup \mathtt{10}^*\mathtt{30}^* \cup \mathtt{30}^*\mathtt{10}^*$, and clearly their intersection is $\mathtt{10}^+$.

The construction of Example 3.1 generalizes to $w\mathtt{0}^*$, for any $w \in \Sigma_k\Sigma_k \setminus 0\Sigma_k$ [6, Thm. 3]. These results have the following important generalization:

Theorem 3.1 (Jeż [6, Thm. 4])**.** *Every regular language* $L \subseteq \Sigma_k^* \setminus \mathtt{0}\Sigma_k^*$ *is in* $\mathcal{L}^k_{\cup,\cap,\boxplus}$.

We include this system for completeness; for the proof the reader is referred to the cited paper. Let $M = (\Sigma_k, Q, q_0, \delta, F)$ be an NFA recognizing L^R. We use variables

$$\{A_{i,j,q}, A_{i,j} : 1 \leqslant i < k,\ 0 \leqslant j < k,\ q \in Q\} \cup \{S\},$$

with the goal that their least solution is

$$L(A_{i,j}) = ij\mathtt{0}^*, \quad L(A_{i,j,q}) = ijL_M(q), \quad L(S) = L.$$

As mentioned above, $A_{i,j}$ can be defined by this type of language equations, and so we focus only on equations for $A_{i,j,q}$:

$$\begin{aligned} A_{i,j,q} &= \Big(\bigcap_{n=0}^{3} A_{i,n} \boxplus A_{j-n,x,q'}\Big) \cup \{k \cdot i + j :\ \text{if } q = q_0\} \\ &\quad \text{for } j > 3, \text{ every } i, \text{ and every } x, q' \text{ such that } q \in \delta(q', x), \\ A_{i,j,q} &= \Big(\bigcap_{n=1}^{4} A_{i-1,j+n} \boxplus A_{k-n,x,q'}\Big) \cup \{k \cdot i + j :\ \text{if } q = q_0\} \\ &\quad \text{for } j < 4 \text{ and } i \neq 1 \text{ and every } x, q' \text{ such that } q \in \delta(q', x), \\ A_{1,j,q} &= \Big(\bigcap_{n=1}^{4} A_{k-n,0} \boxplus A_{j+n,x,q'}\Big) \cup \{k + j :\ \text{if } q = q_0\} \end{aligned}$$

$$\text{for } j < 4, \text{ every } x, q' \text{ such that } q \in \delta(q', x),$$
$$S = (L \cap \Sigma_k) \cup \bigcup_{i,j,q:\ \delta(q,ji) \cap F \neq \varnothing} A_{i,j,q}.$$

Theorem 3.1 implies that regular constants in such systems of language equations can be effectively expressed via singleton constants. We shall use regular constants below, assuming that they are expressed according to Theorem 3.1.

4 Representing linear conjunctive languages

Let us improve the above result by representing a larger class of formal languages. *Linear conjunctive languages* are defined by linear conjunctive grammars [8], or, in other words, by systems of language equations with $\cup$, $\cap$ and linear concatenation. This family of languages can be equivalently defined by one of the simplest types of cellular automata [11]. These are *trellis automata*, also known as one-way real-time cellular automata, which were studied by Culik, Gruska and Salomaa [2], Ibarra and Kim [5], and others. Our argument will proceed by simulating the computation of such automata.

Let us define and explain trellis automata following Culik et al. [2]. A trellis automaton (TA), defined as a quintuple $(\Sigma, Q, I, \delta, F)$, processes an input string of length $n \geqslant 1$ using a uniform array of $n(n+1)/2$ nodes presented in Figure 1. Each node computes a value from a fixed finite set Q. The nodes in the bottom row obtain their values directly from the input symbols using a function $I : \Sigma \to Q$. The rest of the nodes compute the function $\delta : Q \times Q \to Q$ on the values in their predecessors. The string is accepted if and only if the value computed by the top node belongs to the set of accepting states $F \subseteq Q$.

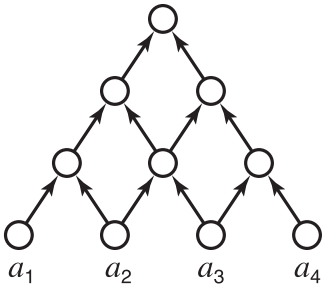

Figure 1: Computation of a trellis automaton.

Definition 4.1. *A trellis automaton is a quintuple $M = (\Sigma, Q, I, \delta, F)$, where:*

- Σ *is the input alphabet,*

- *Q is a finite non-empty set of states,*
- *$I : \Sigma \to Q$ is a function that sets the initial states,*
- *$\delta : Q \times Q \to Q$ is the transition function, and*
- *$F \subset Q$ is the set of final states.*

Extend δ to a function $\delta : Q^+ \to Q$ by $\delta(q) = q$ and

$$\delta(q_1, \ldots, q_n) = \delta(\delta(q_1, \ldots, q_{n-1}), \delta(q_2, \ldots, q_n)),$$

while I is extended to a homomorphism $I : \Sigma^* \to Q^*$. Let $L_M(q) = \{w \mid \delta(I(w)) = q\}$ and define $L(M) = \bigcup_{q \in F} L_M(q)$.

Theorem 4.1 ([11])**.** *A language $L \subseteq \Sigma^+$ is generated by a linear conjunctive grammar if and only if L is recognized by a trellis automaton.*

Linear conjunctive languages are known to be closed under all Boolean operations, concatenation with regular languages and quotient with singletons, but under neither concatenation nor star [2, 11]. In addition, it is known that linear conjunctive languages over a one-letter alphabet generate only regular languages. From this, the following simple result used in the following can be inferred:

Lemma 4.1. *Let L be a linear conjunctive language over an alphabet Σ, let $u, v \in \Sigma^*$ and $a \in \Sigma^*$. Then the language $K = L \cap ua^*v$ is regular.*

Proof. The language $\widetilde{K} = \{u\}^{-1} \cdot K \cdot \{v\}^{-1}$ is linear conjunctive by the closure of this family under quotient with singletons. Since $\widetilde{K}$ is a unary linear conjunctive language, it is regular. Then $K = u\widetilde{K}v$ is regular as well. □

Another simple property of trellis automata relevant to our equations with $\boxplus$ is given in the following lemma:

Lemma 4.2. *Let $\ell = k^n$ for some natural $n > 0$. Then for every languages L, L' such that $f_k(L) = f_\ell(L')$, L is linear conjunctive if and only if L' is linear conjunctive. Given a trellis automaton for either of the languages, a trellis automaton for the other language can be effectively constructed.*

The proof is by a straightforward grouping of digits, and it is omitted.

Let us now prove that the language recognized by any trellis automaton is in $\mathcal{L}^k_{\cup,\cap,\boxplus}$. The basis of the argument is the following simulation of the computation of a trellis automaton by a system of language equations.

Lemma 4.3. *For every $k \geqslant 4$ and for every trellis automaton M over Σ_k, such that $L(M) \cap \mathtt{0}^* = \varnothing$, there exists and can be effectively constructed a resolved system of language equations over the alphabet Σ_k using operations $\cup$, $\cap$ and $\boxplus$ and regular constants, such that the least solution of this system contains a component $((\mathtt{1} \cdot L(M)) \boxplus \mathtt{1}) \cdot \mathtt{10}^*$.*

Proof. In this proof we abuse the notation of $\boxplus$ and $\boxminus$ by allowing their arguments and the result to have leading zeroes. We shall do this only for the second argument equal to $\mathtt{1}$. Under these conditions we define the result to have the same length as the first argument, e.g., $\mathtt{0100} \boxminus \mathtt{1}$ is deemed to be $\mathtt{0099}$. We shall never use this notation in a context where these requirements cannot be fulfilled, that is, for $(k-1)^+ \boxplus \mathtt{1}$ and for $\mathtt{0}^* \boxminus \mathtt{1}$. This abused notation is used only in the text of the proof, while language equations strictly adhere to the definition.

For a given trellis automaton $M = (\Sigma_k, Q, I, \delta, F)$ we define language equations with the set of variables X_q for $q \in Q$, and with an additional variable Y. We will prove that their least solution is $X_q = L_q$, $Y = L$, where

$$L_q = \quad 1((L_M(q) \setminus 0^*) \boxminus 1)10^* = \{1w10^\ell \mid \ell \geqslant 0, w \notin (k-1)^*, w \boxplus 1 \in L_M(q)\}$$

and

$$L = \quad 1((L(M) \setminus 0^*) \boxminus 1)10^* = \{1w10^\ell \mid \ell \geqslant 0, w \notin (k-1)^*, w \boxplus 1 \in L(M)\}.$$

Let us define expressions λ_i and ρ_j, for $i, j \in \Sigma_k$, which depend upon the variables X_q, and which we use as building blocks for constructing equations for X_q. Let us also define constants R_q, which are regular by Lemma 4.1.

$$\begin{aligned}
R_q &= 1\Big(\big((0^*(\Sigma_k \setminus 0) \cup (\Sigma_k \setminus 0)0^*) \cap L_M(q)\big) \boxminus 1\Big)10^* \\
\lambda_i(X) &= 1i\Sigma_k^* \cap \bigcup_{i'} \Big((X \cap 1i'\Sigma^*) \boxplus 10^* \cap 2i'\Sigma_k^* \Big) \boxplus (k+i-2)0^*, \\
&\quad \text{for } i = 0, 1 \\
\lambda_i(X) &= 1i\Sigma_k^* \cap \bigcup_{i'} \Big((X \cap 1i'\Sigma^*) \boxplus 10^* \cap 2i'\Sigma_k^* \Big) \boxplus 1(i-2)0^*, \text{for } i \geqslant 2 \\
\rho_j(X) &= \bigcup_{j'} \Big(\Big((X \cap 1\Sigma_k^* j'10^*) \boxplus 10^* \cap 1\Sigma_k^* j'20^* \Big) \boxplus (k+j-2)10^* \Big) \cap \\
&\quad \cap 1\Sigma_k^* j10^*, \quad \text{for } j = 0, 1 \qquad (2) \\
\rho_j(X) &= \bigcup_{j'} \Big(\Big((X \cap 1\Sigma_k^* j'10^*) \boxplus 10^* \cap 1\Sigma_k^* j'20^* \Big) \boxplus 1(j-2)10^* \Big) \cap \\
&\quad \cap 1\Sigma_k^* j10^*, \quad \text{for } 2 \leqslant j \leqslant k-2 \qquad (3) \\
\rho_{k-1}(X) &= \bigcup_{j'} \Big(\Big((X \cap 1\Sigma_k^* j'10^*) \boxplus 10^* \cap 1\Sigma_k^* j'20^* \Big) \boxplus (k-3)10^* \Big) \cap \\
&\quad \cap 1\Sigma_k^*(k-1)10^* \qquad (4)
\end{aligned}$$

Using this notation, the system of language equations is constructed as follows:

$$\begin{cases} X_q = R_q \cup \bigcup\limits_{\substack{q,q' : \delta(q',q'') = q \\ i,j \in \Sigma_k}} \lambda_i(X_{q''}) \cap \rho_j(X_{q'}) \quad (\text{for all } q \in Q) \\ Y = \bigcup_{q \in F} X_q \end{cases}$$

The construction works as follows: the sets R_q represent the starting part of X_q that we use to compose longer words. A word $w \in \Sigma^{\geqslant 2}$ belongs to $L_M(q)$ iff there are states q', q'' such that $\delta(q', q'') = q$ and $\Sigma_k^{-1} w \in L_M(q'')$ and $w\Sigma_k^{-1} \in L_M(q')$. And so a word $1(w \boxminus 1)10^*$ should belong to X_q if and only if there are two witnesses belonging to $X_{q''}$ and $X_{q'}$ (with some additional constraints). This is specified in ρ and λ, respectively. These expressions represent adding digits at some specific positions, so that selected digits in the original word could be modified in the resulting word, while the rest of the digits remain the same. The main technical difficulty is to force the addition of digits at proper positions. This is achieved by adding the digits in two phases, and by checking the form of intermediate and final results using intersection with regular constants.

Main Claim. *The least solution of the system is* $(\ldots, L_q, \ldots, L)$.

Claim 1. *For every word* $u \in 1\Sigma_k^+ 10^* \setminus 1(k-1)^*10^*$,

$$\lambda_i(\{u\}) = \begin{cases} \{1iw10^m\}, & \text{if } u = 1w10^m, \\ \varnothing, & \text{otherwise.} \end{cases}$$

Proof. Consider the expression λ_i for any i. Let $u = 1i'w'10^m$. In the subexpression corresponding to i', we add $u' = 10^\ell$ for any $\ell \geqslant 0$ and require that the result has $2i'$ as its first two digits. If the leading 1s in u and u' are in the same position, that is, if $|i'w'10^m| = \ell$, then the sum is $2i'w'10^m \in 2i'\Sigma_k^*$. If the 1 in u' is to the left of the leading 1 from u then $u \boxplus u'$ begins with 1. If the leading 1 of u' lands to the right of the leading 1 of u, then either the result does not begin with 2 (if there is no carrying into the first position), or the second digit is not i' (if there is such a carrying). These wrong combinations are filtered out by intersection with $2i'\Sigma_k^*$. Altogether we obtain $(\{1i'w'10^m\} \boxplus 10^*) \cap 2i'\Sigma_k^* = \{2i'w'10^m\}$.

The second addition in λ_i follows the same principle. Our analysis splits depending on the value of i. Consider first $i \in \{0, 1\}$. We start with $u'' = 2i'w'10^m$, add $u''' = (k+i-2)0^\ell$ to it and require that the result begins with $1i$. Since u'' has 2 as the leading symbol, we must modify it to obtain a result of this form. If the positions of 2 and $(k+i-2)$ are the same, that is, $|i'w'10^m| = \ell$, then the result $u'' \boxplus u''' = 1ii'w'10^m$ is as intended. If we add $k+i-2$ left of 2 in u'', then the leading digit is $k+i-2 \in \{k-2, k-1\}$, which is not 1, since $k \geqslant 4$. If we add to the right, then the leading digit is 2 or 3. Therefore, $(\{2i'w'10^m\} \boxplus (k+i-2)0^*) \cap 1i'\Sigma_k^* = \{1ii'w'10^m\}$.

Consider now $i \geqslant 2$. If $i-2$ is at the same position as the leading 2 in u'', then we obtain $1iw10^m$, as intended. If we add $i-2$ to the left of the leading 2 in u'', then the second digit from the left in the result is $i-2$. If the leading 1 is right of 2, then the leading digit in $u \boxplus u' \boxplus u''$ is not 1, (since $k \geqslant 4$). If the leading 1 hits the leading 2, then if we get 1 as a leading symbol, the second symbol is $0 \neq i$. Thus all unintended results are

filtered by intersection with $1i'\Sigma_k^*$, and, as in the previous case, the result is $\{1ii'w'10^m\}$. □

Claim 2. *For every word* $u \in 1\Sigma_k^+10^* \setminus 1(k-1)^*10^*$,

$$\rho_j(\{u\}) = \begin{cases} \{1wj10^m\}, & \text{if } u = 1w10^{m+1} \text{ and } j = k-1, \\ \{1wj10^m\}, & \text{if } u = 1(w \boxminus 1)10^{m+1} \text{ and } j \neq k-1, \\ \varnothing, & \text{otherwise.} \end{cases}$$

Proof. Consider the definition of ρ_j for any j and let $u = 1w'j'10^m$, where $w' \in \Sigma_k^*$ and $j' \in \Sigma_k$. As in Lemma 1, we add any $u' = 10^\ell$ and use intersection with $1\Sigma_k^*j'20^*$ to require that $u \boxplus u'$ has $j'2$ as last non-zero digits. This can be achieved only for $m = \ell$, and so $u \boxplus u' = 1iw'j'20^m$ for $u' = 10^m$. The analysis splits depending on value of j.

Consider first $j \leqslant 1$ and (2). We add $u'' = (k+j-2)10^t$ and require that $u \boxplus u' \boxplus u''$ has $j1$ as last non-zero digits. If $t = m-1$, then we obtain $1(w \boxplus 1)j10^{m-1}$ as intended. Suppose $t \neq m-1$ and that we obtain a word with $j1$ as last non-zero digits. The ending 1 comes from u'', since $k \geqslant 4$. Hence $m > 0$. To obtain j as the second from the last non-zero digit we have to sum up 2 and $k+j-2$, otherwise it would be $k+j-2$ (again we use $k \geqslant 4$). And so we obtain $1(w'j' \boxplus 1)j10^{m-1}$.

Consider now $2 \leqslant j \leqslant k-2$ and (3). We add $u'' = 1(j-2)10^t$ and require that $u \boxplus u' \boxplus u''$ has $j1$ as last non-zero digits. Again for $t = m$ we obtain $1(w \boxplus 1)j10^{m-1}$, as desired. Suppose $m-1 \neq t$ and we obtain word with $j1$ as last non-zero digits. Then the ending 1 must clearly come from u''. hence $m > 0$. To get j as the second digit from the last non-zero digit we have to sum up 2 and $j-2$, otherwise it would be $j-2$. And so we obtain $1(w'j' \boxplus 1)j10^{m-1}$.

Note that this means that for $j \neq k-1$ (despite of the value of j') ρ_j transforms $1w10^m$ into $1(w \boxplus 1)j0^{m-1}$, or equivalently, $1wj10^m$ is obtained from $1(w \boxminus 1)10^{m-1}$.

Consider the case $j = k-1$. We add $u'' = (k-3)10^t$ and require that $u \boxplus u' \boxplus u''$ has $(k-1)1$ as last non-zero digits. If $t = m-1$ then we obtain $1wj10^{m-1}$, as desired. Suppose $t \neq m-1$ and we obtain word with $j1$ as last non-zero digits. Then the ending 1 must come from u''. In particular, $m > 0$. To obtain $k-1$ as the second digit from the last non-zero digit we have to sum up 2 and $k-3$, otherwise it would be $k-3$. And so we obtain $1(w'j')(k-1)10^{m-1}$.

Note, that this means that ρ_{k-1} transforms $1w10^m$ into $1w(k-1)10^{m-1}$. □

Claim 3. $\lambda_i(L_q) = 1(i(L_M(q) \setminus 0^*) \boxminus 1)10^*$.

Proof. Since λ_j is a superposition of $\cup$, $\cap$ and $\boxplus$, $\lambda_i(L_q) = \bigcup_{w \in L_q} \lambda_i(\{w\})$. Then, substituting elements of L_q into Claim 1, we obtain that $\lambda_i(L_q)$ con-

tains all words $1iw10^m$, such that $w \in (L_M(q) \setminus 0^*) \boxminus 1$, which gives the requested expression. □

Claim 4. $\rho_j(L_q) = 1((L_M(q) \setminus 0^*)(j+1 \mod k) \boxminus 1)10^*$.

Proof. As in the previous proof, we use the property that $\rho_j(K) = \bigcup_{w \in K} \rho_j(\{w\})$ for any K. For $j = k-1$, by the definition of L_q and by Claim 2, $\rho_{k-1}(L_q) = \{1w(k-1)10^m \mid 1w10^{m+1} \in L_q\} = 1((L_M(q) \setminus 0^*) \boxminus 1)(k-1)10^* = 1((L_M(q) \setminus 0^*)0 \boxminus 1)10^*$.

For $j \neq k-1$, $\rho_j(L_q) = \{1wj10^m \mid 1(w \boxminus 1)10^{m+1} \in L_q\} = 1(L_M(q) \setminus 0^*)j10^* = 1(((L_M(q) \setminus 0^*)(j+1)) \boxminus 1)10^*$, by definition of L_q and Claim 2. □

Claim 5. *For the least solution $(\ldots, S_q, \ldots)$ of the system and for every $q \in Q$, $L_q \subseteq S_q$.*

Proof. Let $1w10^n \in L_q$, that is, $w \boxplus 1 \in L_M(q)$, where $w \in \Sigma_k^+ \setminus k-1^+$. Using induction on the length of w, let us show that $1w10^n \in S_q$.

If $w \boxplus 1 \in L_M(q)$ has at most one non-zero digit, which is the first one or the last one, then $1w10^n \in R_q \subseteq S_q$ by the equation for X_q.

Otherwise, let $w \boxplus 1 = iuj$, where $i, j \in \Sigma_k$, $u \in \Sigma_k^*$. Since $iuj \in L_M(q)$, there exist states $q', q'' \in Q$, such that $iu \in L_M(q')$, $uj \in L_M(q'')$ and $\delta(q', q'') = q$. Since $iu, uj \notin 0^*$, we can define $w'' = uj \boxminus 1$ and $w' = iu \boxminus 1$. Then, according to the definition of $(\ldots, L_q, \ldots)$, $1w'10^{n+1} \in L_{q'}$ and $1w''10^n \in L_{q''}$. By the induction assumption, $1w'10^{n+1} \in S_{q'}$ and $1w''10^n \in S_{q''}$. We will prove that

$$\lambda_i\big(\{1w''10^n\}\big) \cap \rho_{j-1 \bmod k}\big(\{1w'10^{n+1}\}\big) = \{1w10^n\}.$$

First consider $\lambda_i(1w''10^n)$. By Claim 1, $\lambda_i(1w''10^n) = \{1iw''10^n\}$. To see that $1iw''10^n = 1w10^n$, consider that $w \notin (k-1)^*$, and hence the first symbol of w and of $w \boxplus 1$ are the same. Then $w = iuj \boxminus 1 = i(uj \boxminus 1) = iw''$, which proves that $\lambda_i(1w''10^n) = \{1w10^n\}$.

Consider now $\rho_{j-1}(1w'10^{n+1})$ in the case $j \neq 0$. By Claim 2, it equals $\{1(w' \boxplus 1)(j-1)10^n\}$. Now note that $(w' \boxplus 1)(j-1) = (iu)(j-1) = iuj \boxminus 1 = w$, and hence $\rho_{j-1}(1w'10^{n+1}) = \{1w10^n\}$.

In the case $j = 0$, $\rho_{k-1}(1w'10^{n+1}) = \{1w'(k-1)10^n\}$ by Claim 2. To see that $w'(k-1) = w$, consider that $w = iu0 \boxminus 1 = (iu \boxminus 1)(k-1) = w'(k-1)$. Thus $\rho_{j-1 \bmod k}(\{1w'10^{n+1}\}) = \{1w10^n\}$ for each j.

The claim follows by the equation for X_q. □

Claim 6. *For every $q \in Q$, $L_q \supseteq \varphi_q(\ldots, L_{\tilde{q}}, \ldots)$.*

Proof. Consider any word $1w10^n$ obtained by intersection of $\lambda_i(L_{q''})$ and $\rho_j(L_{q'})$ for some q', q'' such that $\delta(q', q'') = q$. Then $w \notin (k-1)^*$. By Claim 4, $(w \boxplus 1)\Sigma_k^{-1} \in L_M(q')$ and by Claim 3, $\Sigma_k^{-1}(w \boxplus 1) \in L_M(q'')$. Hence, $w \boxplus 1 \in L_M(q)$, and this yields the claim. □

The proof of the main claim proceeds as follows: By Claim 5,

$$(\ldots, \varnothing, \ldots) \sqsubseteq (\ldots, L_q, \ldots) \sqsubseteq (\ldots, S_q, \ldots)$$

Since φ is monotone,

$$\bigsqcup_{n \geqslant 0} \varphi^n(\ldots, \varnothing, \ldots) \sqsubseteq \bigsqcup_{n \geqslant 0} \varphi^n(\ldots, L_q, \ldots) \sqsubseteq \bigsqcup_{n \geqslant 0} \varphi^n(\ldots, S_q, \ldots)$$

Since $(\ldots, S_q, \ldots)$ is a least solution,

$$(\ldots, S_q, \ldots) = \varphi(\ldots, S_q, \ldots) = \bigsqcup_{n \geqslant 0} \varphi^n(\ldots, \varnothing, \ldots).$$

Also, by Claim 6, $\varphi(\ldots, L_q, \ldots) \sqsubseteq (\ldots, L_q, \ldots)$, and hence

$$(\ldots, S_q, \ldots) \sqsubseteq (\ldots, L_q, \ldots) \sqsubseteq (\ldots, S_q, \ldots)$$

This concludes the proof of Lemma 4.3. □

Lemma 4.4. *For every $k \geqslant 4$ and for every trellis automaton M over Σ_k there exists and can be effectively constructed a resolved system of language equations over the alphabet Σ_k using the operations $\cup$, $\cap$ and $\boxplus$ and regular constants, such that its least solution contains a component $\mathtt{1} \cdot L(M)$.*

Proof. For every $j \in \Sigma_k$, consider the language $L(M) \cdot \{j\}^{-1}$. By the closure properties of trellis automata, this language is generated by a trellis automaton M_j. Then, by Lemma 4.3, there exists a system of language equations, such that one of its variables, Y_j, represents the language $(L(M) \cdot \{j\}^{-1}) \boxminus 1$.

Let us combine these equations for all j into a single system, and add a new equation

$$Z = \bigcup_{j=0}^{k-1} (Y_j \cap 1\Sigma^*1) \boxplus (1j \boxminus 1).$$

This equation uses the same technique as in Lemma 4.3. The value of Z is $L(M)$. □

Theorem 4.2. *For every $k \geqslant 2$ and for every trellis automaton M over Σ_k, such that $L(M) \cap \mathtt{0}\Sigma_k^* = \varnothing$, there exists and can be effectively constructed a resolved system of language equations over the alphabet Σ_k using the operations $\cup$, $\cap$ and $\boxplus$ and singleton constants, such that its least solution contains a component $L(M)$.*

Proof. For every $i \in \Sigma_k \setminus \{\mathtt{0}\}$, the language $\{i\}^{-1} \cdot L(M)$ is generated by a certain trellis automaton. By Lemma 4.4, there is a system of language equations, such that one of its variables, Z_i, represents the language $1 \cdot (\{i\}^{-1} \cdot L(M))$.

Combine these systems and add a new variable T with the following equation:

$$T = (L(M) \cap \Sigma_k) \cup Z_1 \cup \bigcup_{\substack{i \in \Sigma_k \setminus \{0,1\} \\ i' \in \Sigma_k}} \Big((Z_i \cap 1i'\Sigma_k^*) \boxplus (i-1)0^* \cap ii'\Sigma_k^* \Big)$$

Consider any Z_i for $i \geqslant 2$. Substituting the value of Z_i into the expression, one first obtains

$$Z_i \cap 1i'\Sigma_k^* = 1(i^{-1} \cdot L(M)) \cap 1i'\Sigma_k^* = \{1i'w \mid ii'w \in L(M)\}.$$

The next operations in the expression are the addition of $(i-1)0^*$ and the intersection with $ii'\Sigma^*$. Let us establish the following fact:

Claim. *For all $i \in \Sigma_k \setminus \{0,1\}$ $i' \in \Sigma_k$ and $w \in \Sigma_k^*$,*

$$\{1i'w\} \boxplus (i-1)0^* \cap ii'\Sigma_k^* = \{ii'w\}.$$

Consider $1i'w \boxplus (i-1)0^\ell$. If $\ell = |i'w|$, then the sum equals $ii'w \in ii'\Sigma_k^*$. If $\ell > |i'w|$, the result is in $(i-1)0^*1i'w$, and hence its intersection with $ii'\Sigma_k^*$ equals $\varnothing$.

Suppose $\ell = |w|$, then $(i-1)0^\ell \boxplus 1i'w = i''i'''w$, and the second digit i''' equals $(i-1)+i'$ modulo k. Since $i' < i'+i-1 < i'+k$, $i''' \neq i'$, and therefore $i''i'''w$ is not in $ii'\Sigma_k^*$ because of a mismatched second digit.

If $\ell < |w|$, there are two subcases. If the addition of $(i-1)0^\ell$ to $i'w$ results in a carry, then $1i'w \boxplus (i-1)0^\ell = 2(i'+1 \mod k)w$, If there is no carry, then $1i'w \boxplus (i-1)0^\ell$ is in $1\Sigma_k^*$ and again cannot be in $ii'\Sigma_k^*$. This concludes the case study necessary to establish the claim, from which there follows

$$\big(\{1i'w \mid ii'w \in L(M)\} \boxplus (i-1)0^*\big) \cap ii'\Sigma^* = L(M) \cap ii'\Sigma^* \tag{5}$$

Summing this up over i', we obtain $L(M) \cap i\Sigma^+$, and summing up the latter over i and adding Z_1, we obtain $L(M) \cap (\Sigma^{\geqslant 2} \setminus 0\Sigma_k^*)$. One-letter words are given separately. Hence, the T-component of the least solution of the system is $L(M)$.

The transition from regular to singleton constants is by Theorem 3.1.

In the cases $k = 2, 3$, consider the language $L' = f_{k^2}^{-1}(f_k(L))$, which is a translation of L from k-ary to k^2-ary notation. By Lemma 4.2, this language is generated by a trellis automaton, and hence can be represented by a system of language equations over Σ_{k^2} with union, intersection and $\boxplus_{k^2}$. Then, by Lemma 2.2, this system can be converted to a system over Σ_k with union, intersection and $\boxplus_k$, representing $f_k^{-1}(f_{k^2}(L')) = L$. This completes the proof for this remaining case. $\square$

5 Separation from linear conjunctive languages

We have shown that every linear conjunctive language over a k-ary alphabet is in $\mathcal{L}^k_{\cup,\cap,\boxplus}$. We shall now establish that $\mathcal{L}^k_{\cup,\cap,\boxplus}$ is a proper superset of the linear conjunctive languages. This is done by specifying the language $\{1^{n+1}0^{2^{2^n}+2^n+1} \mid n \geqslant 0\}$, which is not linear conjunctive, as follows from Buchholz and Kutrib [1, Thms. 4.1, 5.5].

Proposition 5.1. *The language* $L = \{1^n 0^{2^n} 1 0^{2^{2^n}} \mid n \geqslant 0\}$ *is linear conjunctive.*

To see this, consider the well-known fact that the language $L_1 = \{1^n 0^{2^n} \mid n \geqslant 0\}$ is recognized by a trellis automaton [5], that is, it is linear conjunctive. The language $L_2 = \{0^m 1 0^{2^m} \mid m \geqslant 0\}$ is linear conjunctive by the same construction. Then $L = L_1 1 0^* \cap 1^* L_2$ is a linear conjunctive language by their closure properties.

Lemma 5.1. *Consider* $\Sigma_4 = \{\mathtt{0}, \mathtt{1}, \mathtt{2}, \mathtt{3}\}$. *The language* $L' = \{\mathtt{1}^{n+1}\mathtt{0}^{2^{2^n}+2^n} \mid n \geqslant 0\} \subseteq \Sigma_4^*$ *is in* $\mathcal{L}^4_{\cup,\cap,\boxplus}$.

Proof. Consider the above language L over Σ_4. It is in $\mathcal{L}^4_{\cup,\cap,\boxplus}$ by Theorem 4.2. Since $L' = (L \boxplus_4 \mathtt{3}^+\mathtt{0}^*) \cap \mathtt{1}^+\mathtt{0}^*$, the language L' is in $\mathcal{L}^4_{\cup,\cap,\boxplus}$ as well. □

This is sufficient to separate $\mathcal{L}^k_{\cup,\cap,\boxplus}$ from linear conjunctive languages. Let us now consider the complexity of languages in $\mathcal{L}^k_{\cup,\cap,\boxplus}$.

Lemma 5.2. *The family* $\mathcal{L}^k_{\cup,\cap,\boxplus}$ *contains an NP-complete language.*

Proof. Consider the alphabet Σ_7. It is known that the following language of Boolean circuits evaluating to true on given values is linear conjunctive [10]:

$$L = \{\alpha\sigma_1 \ldots \sigma_n \mid \alpha \in \{\mathtt{4}, \mathtt{5}\}^*,\ \sigma_i \in \{1, 2\}^*,\ \alpha \text{ is a description of a circuit with inputs } x_1, \ldots, x_n, \text{ which computes } \mathit{true} \text{ on values } x_i = \mathit{true} \text{ iff } \sigma_i = 2\}$$

The exact form of description α is irrelevant here; what is important that L is in $\mathcal{L}^7_{\cup,\cap,\boxplus}$. Next, consider the language $K = (L \boxplus \{1, 2\}^*) \cap \{4, 5\}^* 3^*$, which is in $\mathcal{L}^7_{\cup,\cap,\boxplus}$ as well. It is easy to see that the language K equals

$$\{\alpha 3^n \mid \alpha \in \{4, 5\}^*,\ \alpha \text{ is a description of a circuit with inputs } x_1, \ldots, x_n, \text{ which evaluates to } \mathit{true} \text{ on some input values}\},$$

and its NP-completeness is obvious. □

Theorem 5.1. *The family of languages* $\mathcal{L}^k_{\cup,\cap,\boxplus}$ *properly includes the family of linear conjunctive languages. It is contained in EXPTIME and contains an NP-complete language.*

The EXPTIME upper bound follows from the P upper bound for conjunctive grammars [8, 9] by the means of Proposition 2.1. The rest is given in Lemmata 5.1 and 5.2.

6 Implications on unary conjunctive grammars

Conjunctive grammars were the starting point of our study of language equations with $\boxplus$, and now we shall use our results to establish some unexpected properties of these grammars.

Definition 6.1. *A conjunctive grammar [8] is a quadruple $G = (\Sigma, N, P, S)$, in which Σ and N are disjoint finite nonempty sets of terminal and nonterminal symbols respectively; P is a finite set of grammar rules, each of the form*

$$A \to \alpha_1 \& \ldots \& \alpha_n \quad (\textit{where } A \in N,\ n \geqslant 1 \textit{ and } \alpha_1, \ldots, \alpha_n \in (\Sigma \cup N)^*)$$

while $S \in N$ is a nonterminal designated as the start symbol.

The semantics of conjunctive grammars is defined by the least solution of the following system of language equations [9]:

$$A = \bigcup_{A \to \alpha_1 \& \ldots \& \alpha_m \in P} \bigcap_{i=1}^{m} \alpha_i \quad (\text{for all } A \in N) \tag{6}$$

The component corresponding to each A is then denoted by $L_G(A)$, and $L(G)$ is defined as $L_G(S)$. An equivalent definition can be given using term rewriting [8] generalizing Chomsky's work rewriting.

Owing to the correspondence between equations over Σ_k with the $\boxplus$ operation and equations of the form (6) over an alphabet $\{a\}$, we can infer the following consequence of Theorem 4.2:

Theorem 6.1. *For every $k \geqslant 2$ and for every linear conjunctive language L and for every trellis automaton M over $\Sigma_k = \{\mathtt{0}, \mathtt{1}, \ldots, k-1\}$, such that no strings in $L(M)$ start with $\mathtt{0}$, there exists and can be effectively constructed a conjunctive grammar generating $f_k(L(M))$.*

Consider the language of computation histories of a Turing machine, VALC(T), which is known to be linear conjunctive [5, 11]. For a TM T over an input alphabet Ω, this language is defined over an alphabet $\Omega \cup \Gamma \cup \{\natural\}$ as

$$\text{VALC}(T) = \{w \natural C_T(w) | w \in \Omega^* \text{ and } C_T(w) \in \Gamma^* \text{ is an accepting computation}\},$$

where $C_T(w)$ encodes the accepting computation of T on any $w \in L(T)$. The details of encoding are irrelevant here. Let us further assume that VALC(T) is defined over an alphabet $\Sigma_k = \{0, \ldots, k-1\} = \Omega \cup \Gamma \cup \{\natural\}$, with $0 \in \Gamma$, so that no string in VALC(T) has a leading zero.

Combining the known linear conjunctive grammar for VALC(T) [11] with our Theorem 4.2, we obtain that for every Turing machine T one can construct a conjunctive grammar for the language $f_k(\text{VALC}(T)) \subseteq a^*$. This leads to unexpected undecidability results for unary conjunctive grammars:

Theorem 6.2. *For every unary conjunctive language $L_0 \subseteq a^*$ the problem of whether a given conjunctive grammar over $\{a\}$ generates the language L_0 is co-RE-complete.*

Proof. The containment of the problem in co-RE is evident, since the equivalence problem of two given recursive languages is in co-RE. It is the co-RE-hardness that has to be established.

Let $G_0 = (\Sigma, N_0, P_0, S_0)$ be a fixed conjunctive grammar generating L_0. Suppose there is an algorithm to check whether $L(G) = L_0$ for any given conjunctive grammar G over $\{a\}$. Let us show that this algorithm could be used to solve the emptiness problem for Turing machines. Depending on the form of L_0, let us consider two cases.

Case I: L_0 contains no subset of the form $a^\ell(a^p)^*$, where $\ell \geqslant 0$ and $p \geqslant 1$. Given a Turing machine T, construct a conjunctive grammar $G_T = (\{a\}, N_T, P_T, S_T)$ for $f_k(\mathrm{VALC}(T))$. On the basis of G_T and G_0, construct a new conjunctive grammar $G = (\{a\}, N_T \cup N_0 \cup \{S, A\}, P_T \cup P_0 \cup P, S)$, where P contains the following four new rules: $S \to S_0$, $S \to S_T A$, $A \to aA$ and $A \to \varepsilon$.

Now, if $L(T) = \varnothing$, then $L(G_T) = \varnothing$, the rule $S \to S_T A$ in G generates nothing, and therefore $L(G) = L(G_0) = L_0$. If $L(T) \neq \varnothing$, then there exists $w\natural C_T(w) \in \mathrm{VALC}(T)$, and hence there exists $a^n \in L(G_T)$. Then the rule $S \to S_T A$ in G can be used to generate all strings in $a^n a^*$, and therefore $L(G)$ contains the subset $a^\ell(a^p)^*$ for $\ell = n$ and $p = 1$. As L_0 contains no such subset by assumption, $L(G) \neq L_0$. This proves the undecidability of this case.

Case II: L_0 contains a subset $a^\ell(a^p)^*$, where $\ell \geqslant 0$ and $p \geqslant 1$. Assume that p is larger than the least cardinality of the alphabet used for $\mathrm{INVALC}(T)$ (if p is too small, any of its multiples can be taken). Define $\mathrm{INVALC}(T)$ over a p-letter alphabet and consider the language $f_p(\mathrm{INVALC}(T) \cdot \mathtt{0})$, which is generated by some conjunctive grammar $G'_T = (\{a\}, N'_T, P'_T, S'_T)$. Using G_0 and G'_T, construct a new grammar $G = (\{a\}, N'_T \cup N_0 \cup \{S, B, C\}, P_T \cup P_0 \cup P, S)$, where the new rules in P are as follows: $S \to S_0 \& B$, $S \to a^\ell S'_T$, $B \to a^i$ (for all $0 \leqslant i < \ell$), $B \to a^{\ell+i}C$ (for all $1 \leqslant i < p$), $C \to a^p C$ and $C \to \varepsilon$.

Note that $L_G(B) = a^* \setminus a^\ell(a^p)^*$. If $L(T) = \varnothing$, then $\mathrm{INVALC}(T) = \Sigma_p^* \setminus \mathtt{0}\Sigma_p^*$, and hence $f_p(\mathrm{INVALC}(T) \cdot \mathtt{0}) = (a^p)^*$. Then the rule $S \to a^\ell S'_T$ generates the language $a^\ell(a^p)^* \subseteq L_0$, while the rule $S \to S_0 \& B$ generates $L_0 \setminus a^\ell(a^p)^*$. Therefore, $L(G) = L_0$.

Otherwise, if $L(T) \neq \varnothing$, then there exists $w \notin \mathrm{INVALC}(T)$, which implies $f_p(w\mathtt{0}) \notin L(G'_T)$ Let $f_p(w\mathtt{0}) = a^{ip}$, for $i \geqslant 0$. Then the string $a^{ip+\ell} \in L_0$ is not generated by the rule $S \to a^\ell S'T$, and it is also not generated by $S \to S_0 \& B$, because it is not in $L_G(B)$. Therefore, $a^{ip+\ell} \notin L(G)$ and $L(G) \neq L_0$. □

If L_0 is not generated by a conjunctive grammar, then the answer to the question of equality of $L(G)$ and L_0 is always negative, which makes the

problem trivial. Hence, the following characterization is obtained:

Corollary 6.1. *For different constant languages $L_0 \subseteq a^*$, the problem of testing whether a given conjunctive grammar over $\{a\}$ generates L_0 is either co-RE-complete or trivial.*

Theorem 6.3. *For conjunctive grammars over a unary alphabet there exist no algorithm to decide whether a given grammar generates a finite language (a regular language).*

Proof. Given a Turing machine T, construct another TM T' that recognizes $\{\varepsilon\}$ if $L(T) \neq \varnothing$, and $\varnothing$ otherwise. Construct a conjunctive grammar G for $f_k(\mathrm{VALC}(T')) \cdot \{a^{k^n} \mid n \geqslant 0\}$. Then $L(G)$ is either nonregular (if $L(T) \neq \varnothing$) or empty. □

Another consequence is related to the growth rate of unary conjunctive languages, Every infinite unary language $L = \{a^{\ell_1}, a^{\ell_2}, \ldots, a^{\ell_n}, \ldots\}$ where $0 \leqslant \ell_1 < \ell_2 < \ldots < \ell_n < \ldots$, can be regarded as an increasing integer sequence, and it is natural to consider the growth rate of such sequences. Obviously, the growth of every regular language is bounded by a linear function. The example of a conjunctive grammar for the language $\{a^{4^n} \mid n \geqslant 0\}$ [6], see Example 3.1, shows that the growth of unary conjunctive languages can be at least exponential, which raises the question of whether there exists any upper bound for the growth of conjunctive languages in the unary case.

The following theorem gives the strongest possible answer to this question:

Theorem 6.4. *For every recursively enumerable set of natural numbers X there exists a conjunctive grammar G over an alphabet $\{a\}$, such that the growth function of $L(G)$ is greater than that of X at any point.*

Proof. Let T be a Turing machine, which recognizes the set $X = \{i_1, i_2, \ldots, i_j, \ldots\}$, where $0 \leqslant i_1 < i_2 < \ldots < i_j < \ldots$, and the numbers are given to it in a binary notation. Consider the language $\mathrm{VALC}(T)$, which contains strings $w_n = f_2^{-1}(a^n) \# C_T(n)$. By the above arguments, there exists a conjunctive grammar G over an alphabet $\{a\}$ that generates $L = f_k(\mathrm{VALC}(T))$ for some $k \geqslant 2$.

Let $g(n)$ be the growth function of L. It is sufficient to show that $g(j) \geqslant i_j$ for each $j \geqslant 1$. To see this, consider the values $g(1), g(2), \ldots, g(j)$. At least one of them describes a computation $w_{j'}$ for $j' \geqslant j$. Since g is an increasing function, we obtain $g(j) \geqslant f_k(w_{j'}) > j' \geqslant j$. □

Note that this quick-growing language is bound to be computationally very easy, as the upper bound of parsing complexity for conjunctive grammars is $DTIME(n^3) \cap DSPACE(n)$ [8, 15].

The next example gives a unary conjunctive language of a polynomial growth.

Proposition 6.1. *There exists a conjunctive grammar G over an alphabet $\{a\}$, such that the growth function of $L(G)$ satisfies $g(n) = \Theta(n^2)$.*

Proof. Consider the set of numbers $X = \{(2^m + 3i) \cdot 2^m \mid m \geqslant 0,\ 2^m \leqslant 2^m + 3i < 2^{m+2}\}$. Let $g(n)$ denote the n-th largest element of X; this is the growth function of the corresponding unary language $L = \{a^n \mid n \in X\}$. The set of binary notations of the numbers in X is

$$f_2^{-1}(L) = \left\{ 1w0^m \;\middle|\; |w| = m-1 \text{ or } |w| = m, \text{ and } f_2(w) \text{ divides by } 3 \right\}.$$

This is clearly a linear context-free language, hence L is conjunctive by Theorem 6.1. Let us prove that $n^2 \leqslant g(n) \leqslant 4n^2$.

Let us prove that for any number $n = 2^m + j$, where $0 \leqslant j < 2^m$, it holds that $g(2^m + j) = (2^m + 3j)2^m$. We first show that $g(2^m) = 2^{2m}$, that is, $(2^m + 3j)2^m$ for $j = 0$. Consider interval $[2^{2m}, 2^{2m+2})$. We know that $2^m \leqslant (2^m + 3j) < 2^{m+2}$, which gives $j \geqslant 0$ and $3j \leqslant 2^{m+2} - 2^m = 3 \cdot 2^m$, hence $0 \leqslant j < 2^m$ and therefore we have exactly 2^m values from this interval in X. Hence in interval $[0, 2^{2m})$ there were $1 + 2 + 4 + \ldots + 2^{m-1} = 2^m - 1$ values from X, as $[0, 2^{2m}) = [0, 4) \cup [4, 16) \cup \ldots \cup [2^{2m-2}, 2^{2m})$. Therefore, $g(2^m + j) = (2^m + 3j)2^m$ by the definition of X and by the fact that j can take up to 2^m values.

Then, to see that $g(n) \geqslant n^2$, consider $g(2^m + j)$ for $0 \leqslant j < 2^m$. We have

$$g(2^m + j) = (2^m + 3j)2^m = 2^{2m} + 3j \cdot 2^m \geqslant 2^{2m} + 2j \cdot 2^m + j^2 = (2^m + j)^2,$$

where the inequality is due to $j \cdot 2^m \geqslant j^2$. On the other hand,

$$g(2^m + j) \leqslant g(2^{m+1}) = 2^{2m+2} \leqslant 4(2^m + j)^2,$$

which proves the upper bound $g(n) \leqslant 4n^2$. □

This construction can be generalized to obtain the following result:

Theorem 6.5. *For every rational number $p/q \geqslant 1$ there exists a conjunctive grammar G over an alphabet $\{a\}$, such that the growth function of $L(G)$ is $g(n) = \Theta(n^{p/q})$.*

Sketch of a proof. The proof follows the same steps as in the case of $p/q = 2$ treated above. Following is the set of numbers to be represented:

$$X = \left\{ 2^{pk} + \lfloor C \cdot i \rfloor \cdot 2^{(p-q)k} \;\middle|\; i, k \geqslant 0, 2^{pk} \leqslant 2^{pk} + C \cdot i \cdot 2^{(p-q)k} < 2^{p(k+1)} \right\},$$

where $C = (2^p - 1)/(2^q - 1)$. □

7 Implications on unary language equations

Language equations with all Boolean operations and concatenation over a multiple-letter alphabet are known to be computationally complete [12]: their unique solutions represent exactly the recursive languages, least solutions represent the recursively enumerable (RE) languages, while greatest solutions represent the co-RE languages. The same results hold for *unresolved* equations of the form

$$\varphi_i(X_1, \ldots, X_n) = \psi_i(X_1, \ldots, X_n)$$

with union, intersection and concatenation [13]. On the other hand, no analogous results are known for a unary alphabet, and as the methods used to establish the existing results essentially use structure in strings [12], one would conjecture that the unary case must be much simpler.

However, using our new methods, the following entirely unexpected result can be established:

Theorem 7.1. *The family of languages representable by unique (least, greatest) solutions of system of language equations of the form* $\varphi_i(X_1, \ldots, X_n) = \psi_i(X_1, \ldots, X_n)$ *over the alphabet* $\{a\}$*, with union, intersection and concatenation, is exactly the family of recursive (recursively enumerable, co-recursively enumerable, respectively) languages.*

Let us first establish a weaker statement on least solutions, which contains the main idea of the proof of Theorem 7.1 in a technically simple form:

Proposition 7.1. *There exists a system of language equations of the form* $\varphi_i(X_1, \ldots, X_n) = \psi_i(X_1, \ldots, X_n)$ *over the alphabet* $\{a\}$*, with union, intersection and concatenation, such that the first component of its least solution is r.e.-complete.*

Proof. Consider a universal Turing machine T, that is, $L(T)$ is r.e.-complete. Assume that the input alphabet of T is $\{\mathtt{1}, \mathtt{2}\}$. For each $w \in L(T)$, let the computation of T on w, $C_T(w)$, be defined over the alphabet $\{3, 4\}$. Define the following language over the alphabet $\Sigma_7 = \{\mathtt{0}, \mathtt{1}, \mathtt{2}, \mathtt{3}, \mathtt{4}, \mathtt{5}, \mathtt{6}\}$:

$$\begin{aligned}\mathrm{VALC}(T) = \{C_T(w)w \mid\ & C_T(w) \in \{\mathtt{3}, \mathtt{4}\}^*,\ w \in \{\mathtt{1}, \mathtt{2}\}^*,\ w \in L(T) \text{ and} \\ & C_T(w) \text{ encodes the accepting computation of } T \text{ on } w\}\end{aligned}$$

Under a proper encoding C_T, this variant of $\mathrm{VALC}(T)$ is linear conjunctive, hence $f_7(\mathrm{VALC}(T)) \subseteq a^*$ is conjunctive by Theorem 4.2, that is, it can be represented by a unique solution of a resolved system of language equations with union, intersection and $\boxplus$.

Consider a variable Y, which assumes a value of a language over a, and construct the following two inequalities

$$\begin{aligned} Y &\subseteq f_7(\{\mathtt{1}, \mathtt{2}\}^*) \\ f_7(\mathrm{VALC}(T)) &\subseteq f_7(\{\mathtt{3}, \mathtt{4}\}^*\mathtt{0}^*) \cdot Y \end{aligned}$$

The language $f_7(L(T))$ is a solution. On the other hand, according to the second inequality, for every string $w \in L(T)$, the number $f_7(C_T(w)w)$ is in $f_7(\{\mathtt{3},\mathtt{4}\}^*\mathtt{0}^*) \cdot Y$, which implies $f_7(w) \in Y$. This makes $Y = f_7(L(T))$ the least solution.

The full system is obtained by expressing the constant languages $f_7(\mathrm{VALC}(T))$, $f_7(\{\mathtt{1},\mathtt{2}\}^*)$ and $f_7(\{\mathtt{3},\mathtt{4}\}^*0^*)$ using conjunctive grammars. $\square$

Let us now proceed with the proof of the more general Theorem 7.1. The correspoding upper bounds for unique, least and greatest solutions are known for all alphabets [12]. It is thus sufficient to represent every recursive (recursively enumerable, co-recursively enumerable) subset of $\Sigma_6^* \setminus \mathtt{0}\Sigma_6^*$ for any single value of $k \geqslant 0$ by a unique (least, greatest, respectively) solution of a system over Σ_k with $\boxplus$, which will extend to unary languages according to Proposition 2.1.

First, redefine the language of computations of a Turing machine as follows:

Definition 7.1. *Let T be a Turing machine recognizing a language $L(T) \subseteq \Sigma_6^* \setminus \mathtt{0}\Sigma_6^*$. For each $i \in \{1,2,3,4,5\}$, the computation of T on on a string $iw \in L(T)$ of length at least 2 starting from the digit i is encoded as a string $C_T^i(iw) \in \{\mathtt{30},\mathtt{300}\}^*\mathtt{300}$. The language* $\mathrm{VALC}_i(T)$ *representing all such valid accepting computations is*

$$\mathrm{VALC}_i(T) = \{C_T^i(iw)\mathtt{1}w \mid iw \in i\Sigma_6^+,\ iw \in L(T)\},$$

The computations of T on ε and on one-symbol strings, provided that they are accepting, are encoded as strings $C_T^0(w) \in \mathtt{1}\Sigma_6^*\mathtt{0}$, and the corresponding language of computations is

$$\mathrm{VALC}_0(T) = \{C_T^0(w) \boxplus w \mid w \in \{\varepsilon,\mathtt{1},\mathtt{2},\mathtt{3},\mathtt{4},\mathtt{5}\} \text{ and } w \in L(T)\}$$

Denote the complements of these languages by

$$\begin{aligned}\mathrm{INVALC}_i(T) &= (\Sigma_k^* \setminus \mathtt{0}\Sigma_k^*) \setminus \mathrm{INVALC}_i(T) \quad \text{and}\\ \mathrm{INVALC}_0(T) &= (\Sigma_k^* \setminus \mathtt{0}\Sigma_k^*) \setminus \mathrm{INVALC}_0(T).\end{aligned}$$

Lemma 7.1. *For every $L \subseteq \mathtt{1}\Sigma_6^+$, for every $x \in \{\mathtt{30},\mathtt{300}\}^*\mathtt{300}$ and for every $u \in \Sigma_6^*$, if $x\mathtt{1}u \in \{\mathtt{30},\mathtt{300}\}^*\mathtt{3000}^* \boxplus L$, then $\mathtt{1}u \in L$.*

Proof. Let $x\mathtt{1}u = y\mathtt{0}^\ell \boxplus \mathtt{1}v$, where $y \in \{\mathtt{30},\mathtt{300}\}^*\mathtt{300}$ and $\mathtt{1}v \in L$. Depending on the length of $|\mathtt{1}v|$, consider the following cases:

1. $|\mathtt{1}v| < \ell$. Then $y\mathtt{0}^\ell \boxplus \mathtt{1}v = y\mathtt{0}^{\ell-|\mathtt{1}v|}\mathtt{1}v$, which is a string containing at least three consecutive zeroes to the left of the leftmost instance of $\mathtt{1}$. Since the string $x\mathtt{1}u$ does not have this property, it follows that $y\mathtt{0}^\ell \boxplus \mathtt{1}v \neq x\mathtt{1}u$, which makes this case impossible.

2. $|\mathtt{1}v| = \ell$. Then $y\mathtt{0}^\ell \boxplus \mathtt{1}v = y\mathtt{1}v$, and thus $y\mathtt{1}v = x\mathtt{1}u$. The leftmost instance of $\mathtt{1}$ in $y\mathtt{1}v$ and $x\mathtt{1}u$ is at the first position of $\mathtt{1}v$ and $\mathtt{1}u$, respectively. Therefore, $y = x$ and $\mathtt{1}v = \mathtt{1}u$.

3. $\ell \leqslant |\mathtt{1}v| \leqslant |y| + \ell$. Let $y = y_1 i y_2$, where $|iy_2| + \ell = |\mathtt{1}v|$. The digit i is either $\mathtt{0}$ or $\mathtt{3}$.

   - If $i = \mathtt{0}$, then y_1 ends with $\mathtt{3}$ or $\mathtt{30}$. The sum $y_1 i y_2 \mathtt{0}^\ell \boxplus \mathtt{1}v$ is thus of the form $y_1 i' z$, where $i' \in \{\mathtt{1}, \mathtt{2}\}$, and the prefix $y_1 i'$ is in $\{\mathtt{30}, \mathtt{300}\}^*\{\mathtt{31}, \mathtt{32}, \mathtt{301}, \mathtt{302}\}$. On the other hand, in $x\mathtt{1}u$, the leftmost occurence of digits outside of $\{\mathtt{3}, \mathtt{0}\}$ must be of the form $\mathtt{3001}$.
   - If $i = \mathtt{3}$, then the sum $y_1 i y_2 \mathtt{0}^\ell \boxplus \mathtt{1}v$ is of the form $y_1 i' z$, where $i' \in \{\mathtt{4}, \mathtt{5}\}$ and $|z| = |v|$. Then the leftmost digit of $y_1 i' z$ not in $\{\mathtt{3}, \mathtt{0}\}$ is not $\mathtt{1}$, while for $x\mathtt{1}u$ it is $\mathtt{1}$.

   In both cases it follows that $y_1 i y_2 \mathtt{0}^\ell \boxplus \mathtt{1}v$ and $x\mathtt{1}u$ must be different, and the case is impossible.

4. $|\mathtt{1}v| \geqslant |y| + \ell$. Then the leading digit of $y\mathtt{0}^\ell \boxplus \mathtt{1}v$ is $\mathtt{1}$ or $\mathtt{2}$, hence again $y\mathtt{0}^\ell \boxplus \mathtt{1}v \neq x\mathtt{1}u$, which rules out this case.

It has thus been established that $y = x$ and $\mathtt{1}v = \mathtt{1}u$ in the only possible case, which yields the claim. □

The first case to be established is the case of least solutions and recursively enumerable languages.

Lemma 7.2. *For every recursively enumerable language $L_0 \subseteq \Sigma_6^* \setminus \mathtt{0}\Sigma_6^*$ there exists a system of language equations of the form*

$$\varphi_j(Y, X_1, \ldots, X_n) = \psi_j(Y, X_1, \ldots, X_n)$$

with union, intersection and $\boxplus$, which has the set of solutions

$$\left\{ (L, f_1(L), \ldots, f_n(L)) \;\middle|\; L_0 \subseteq L \right\}$$

where $f_1, \ldots, f_n : 2^{\Sigma_6^* \setminus \mathtt{0}\Sigma_6^*} \to 2^{\Sigma_6^* \setminus \mathtt{0}\Sigma_6^*}$ are some monotone language functions defined with respect to L_0. In particular, there is a least solution with $Y = L_0$.

Proof. Consider any Turing machine T over the input alphabet $\Sigma_6 = \{\mathtt{0}, \mathtt{1}, \mathtt{2}, \mathtt{3}, \mathtt{4}, \mathtt{5}\}$. A system in variables $(Y, Y_1, \ldots, Y_5, Y_0, X_7, \ldots, X_n)$ will be constructed, where the number n will be determined below, and the set of solutions of this system will be defined precisely by the following conditions,

which ensure that the statement of the lemma is fulfilled:

$$L(T) \cap \{\varepsilon, \mathtt{1}, \mathtt{2}, \mathtt{3}, \mathtt{4}, \mathtt{5}\} \subseteq Y_0 \subseteq \{\varepsilon, \mathtt{1}, \mathtt{2}, \mathtt{3}, \mathtt{4}, \mathtt{5}\}, \tag{7a}$$

$$\{\mathtt{1}w \mid w \in \Sigma_6^+,\ iw \in L(T)\} \subseteq Y_i \subseteq \mathtt{1}\Sigma_6^+ \quad (1 \leqslant i \leqslant 5), \tag{7b}$$

$$Y = Y_0 \cup \bigcup_{i=1}^{5} \{iw \mid \mathtt{1}w \in Y_i\} \tag{7c}$$

$$X_j = K_j \quad (7 \leqslant j \leqslant n) \tag{7d}$$

The languages $K_7, \ldots, K_n$ will be determined below.

For each $i = 1, 2, 3, 4, 5$, consider the above definition of $\mathrm{VALC}_i(T)$ and define a variable Y_i with the equations

$$Y_i \subseteq \mathtt{1}\Sigma_6^+ \tag{8a}$$

$$\mathrm{VALC}_i(T) \subseteq \{\mathtt{30}, \mathtt{300}\}^*\mathtt{3000}^* \boxplus Y_i, \tag{8b}$$

It is claimed that this system is equivalent to (7b).

Suppose (7b) holds for Y_i. Then (8a) is obviously true. To check (8b), consider any $C_T^i(iw)\mathtt{1}w \in \mathrm{VALC}_i(T)$. Since this string represents the computation of T on iw, this implies $iw \in L(T)$, and hence $\mathtt{1}w \in Y_i$ by (7b). Then $C_T^i(iw)\mathtt{1}w \in \{\mathtt{30}, \mathtt{300}\}^*\mathtt{3000}^{|\mathtt{1}w|} \boxplus \mathtt{1}w \subseteq \{\mathtt{30}, \mathtt{300}\}^*\mathtt{3000} \boxplus Y_i$, which proves the inclusion (8b).

Conversely, assuming (8), it has to be proved that for every $iw \in L(T)$, where $w \in \Sigma_6^+$, the string $\mathtt{1}w$ must be in Y_i. Since $iw \in L(T)$, there exists an accepting computation of T: $C_T^i(iw)\mathtt{1}w \in \mathrm{VALC}_i(T)$. Hence, $C_T^i(iw)\mathtt{1}w \in \{\mathtt{30}, \mathtt{300}\}^*\mathtt{3000}^* \boxplus Y_i$ due to the inclusion (8b), and therefore $\mathtt{1}w \in Y_i$ by Lemma 7.1.

Define one more variable Y_0 with the equations

$$Y_0 \subseteq \{\varepsilon, \mathtt{1}, \mathtt{2}, \mathtt{3}, \mathtt{4}, \mathtt{5}\} \tag{9a}$$

$$\mathrm{VALC}_0(T) \subseteq \mathtt{1}\Sigma_6^*\mathtt{0} \boxplus Y_0, \tag{9b}$$

The claim is that (9) holds if and only if (7a).

Assume (7a) and consider any string $C_T^0(w) \boxplus w \in \mathrm{VALC}_0(T)$, where $w \in \{\varepsilon, \mathtt{1}, \mathtt{2}, \mathtt{3}, \mathtt{4}, \mathtt{5}\}$ by definition. Then w is accepted by T, and, by (7a), $w \in Y_0$. Since $C_T^0(w) \in \mathtt{1}\Sigma_6^*\mathtt{0}$, addition of w affects only the last digit, and $C_T^0(w) \boxplus w \in \mathtt{1}\Sigma_6^*\mathtt{0} \boxplus w \subseteq \mathtt{1}\Sigma_6^*\mathtt{0} \boxplus Y_0$, which proves (9b).

The converse claim is that (9) implies that every $w \in L(T) \cap \{\varepsilon, \mathtt{1}, \mathtt{2}, \mathtt{3}, \mathtt{4}, \mathtt{5}\}$ must be in Y_0, The corresponding $C_T^0(w) \boxplus w \in \mathrm{VALC}_0(T)$ is in $\mathtt{1}\Sigma_6^*\mathtt{0} \boxplus w$ by (9b). The string $C_T^0(w) \boxplus w$ ends with $\mathtt{0}$ if $w = \varepsilon$, and if w is a single digit, the string ends with this digit. The language $\mathtt{1}\Sigma_6^*\mathtt{0} \boxplus Y_0$ contains a string of such a form only if $w \in Y_0$.

Next, combine the above six systems together and add a new variable Y with the following equation:

$$Y = Y_0 \cup Y_1 \cup \bigcup_{\substack{i \in \{\mathtt{2},\mathtt{3},\mathtt{4},\mathtt{5}\} \\ i' \in \Sigma_k}} \Big((Y_i \cap \mathtt{1}i'\Sigma_k^*) \boxplus (i-1)\mathtt{0}^* \cap ii'\Sigma_k^* \Big)$$

This equation has been borrowed from the proof of Theorem 4.2, where it was established that it is equivalent to $Y = Y_0 \cup \{iw \mid 1w \in Y_i\}$, that is, to (7c).

The final step of the construction is to express regular and linear conjunctive languages used in the above systems through singleton constants, which can be done according to Theorem 4.2. The variables needed to specify these languages are denoted $(X_7, \ldots, X_n)$, and the equations for these variables have a unique solution $X_j = K_j$ for all j.

This completes the description of the set of solutions of the system. It is easy to see that there is a least solution in this set, with $Y = L(T)$, $Y_0 = L(T) \cap \{\varepsilon, \mathtt{1}, \mathtt{2}, \mathtt{3}, \mathtt{4}, \mathtt{5}\}$, $Y_i = \{\mathtt{1}w | w \in \Sigma_6^+, iw \in L(T)\}$ and $X_j = K_j$. $\square$

The representation of co-recursively enumerable languages by greatest solutions is dual to the case of least solutions and can be established by an analogous argument, which will now be presented. Directly inferring this result from the previous lemma would likely be harder than giving a new proof.

Lemma 7.3. *For every co-recursively enumerable language $L_0 \subseteq \Sigma_6^* \setminus \mathtt{0}\Sigma_6^*$ there exists a system of language equations of the form*

$$\varphi_j(Z, X_1, \ldots, X_n) = \psi_j(Z, X_1, \ldots, X_n)$$

with union, intersection and $\boxplus$, which has the set of solutions

$$\left\{ (L, f_1(L), \ldots, f_n(L)) \;\middle|\; L \subseteq L_0 \right\}$$

where $f_1, \ldots, f_n : 2^{\Sigma_6^* \setminus \mathtt{0}\Sigma_6^*} \to 2^{\Sigma_6^* \setminus \mathtt{0}\Sigma_6^*}$ are some monotone language functions defined with respect to L_0. In particular, there is a greatest solution with $Z = L_0$.

Proof. Let T be a Turing machine over $\Sigma_6 = \{\mathtt{0}, \mathtt{1}, \mathtt{2}, \mathtt{3}, \mathtt{4}, \mathtt{5}\}$. The system in variables $(Z, Z_1, \ldots, Z_5, X_0, X_7, \ldots, X_n)$ to be constructed will have a set of solutions satisfying the following properties:

$$Z_0 \subseteq \overline{L(T)} \cap \{\varepsilon, \mathtt{1}, \mathtt{2}, \mathtt{3}, \mathtt{4}, \mathtt{5}\} \tag{10a}$$

$$Z_i \subseteq \{\mathtt{1}w \mid w \in \Sigma_6^+,\ iw \notin L(T)\} \quad (1 \leqslant i \leqslant 5), \tag{10b}$$

$$Z = Z_0 \cup \bigcup_{i=1}^{5} \{iw \mid \mathtt{1}w \in Z_i\} \tag{10c}$$

$$X_j = K_j \quad (7 \leqslant j \leqslant n) \tag{10d}$$

The number n and the vector of languages $(K_7, \ldots, K_n)$ will be determined below. This set of solution satisfies the statement of the lemma.

The equations defining the value of each Z_i $(1 \leqslant i \leqslant 5)$ are as follows:

$$Z_i \subseteq \mathtt{1}\Sigma_6^+ \tag{11a}$$

$$\{\mathtt{30}, \mathtt{300}\}^*\mathtt{3000}^* \boxplus Z_i \subseteq \mathrm{INVALC}_i(T), \tag{11b}$$

It is claimed that (11) holds if and only if (10b).

If Z_i satisfies (10b), then (11a) follows immediately, and in order to prove (11b), one has to consider any string not in $\mathrm{INVALC}_i(T)$ and show that it is not in $\{\mathtt{30},\mathtt{300}\}^*\mathtt{3000}^* \boxplus Z_i$. By definition, a string is not in $\mathrm{INVALC}_i(T)$ if it is in $\mathrm{VALC}_i(T)$, so take any $C_T^i(iw)\mathtt{1}w \in \mathrm{VALC}_i(T)$, where $C_T^i(iw) \in \{\mathtt{30},\mathtt{300}\}^*\mathtt{300}$ and $iw \in L(T)$. Suppose $C_T^i(iw)\mathtt{1}w \in \{\mathtt{30},\mathtt{300}\}^*\mathtt{3000}^* \boxplus Z_i$. Then, by Lemma 7.1, $\mathtt{1}w \in Z_i$, hence $iw \notin L(T)$ by (10b), which yields a contradiction.

The converse is established as follows. Assuming (11), consider any $iw \in L(T)$, where $w \in \Sigma_6^+$. It is sufficient to prove that $\mathtt{1}w \notin Z_i$. Suppose $\mathtt{1}w \in Z_i$, then $C_T^i(iw)w \in \{\mathtt{30},\mathtt{300}\}^*\mathtt{3000}^* \boxplus Z_i \subseteq \mathrm{INVALC}_i(T)$ by (11b). However, $C_T^i(iw)w$ is in $\mathrm{VALC}_i(T)$ and thus cannot be in $\mathrm{INVALC}_i(T)$. The contradiction obtained proves this case.

Define the following equations for the variable Z_0:

$$Z_0 \subseteq \{\varepsilon,\mathtt{1},\mathtt{2},\mathtt{3},\mathtt{4},\mathtt{5}\} \tag{12a}$$

$$\mathtt{1}\Sigma_6^*\mathtt{0} \boxplus Z_0 \subseteq \mathrm{INVALC}_0(T) \tag{12b}$$

Again, the claim is that these equations are equivalent to (10a).

Let Z_0 be a subset of $\{\varepsilon,\mathtt{1},\mathtt{2},\mathtt{3},\mathtt{4},\mathtt{5}\} \setminus L(T)$, as stated in (10a). This immediately implies (12a). Consider any string not in $\mathrm{INVALC}_0(T)$; proving that it is not in $\mathtt{1}\Sigma_6^*\mathtt{0}\boxplus Z_0$ will establish (12b). A string not in $\mathrm{INVALC}_0(T)$ is in $\mathrm{VALC}_0(T)$, so let $C_T^0(w)\boxplus w \in \mathrm{VALC}_0(T)$ for any $w \in \{\varepsilon,\mathtt{1},\mathtt{2},\mathtt{3},\mathtt{4},\mathtt{5}\}$, and suppose $C_T^0(w) \boxplus w \in \mathtt{1}\Sigma_6^*\mathtt{0} \boxplus Z_0$. If $w = \varepsilon$, then the last digit of $C_T^0(w) \boxplus w$ is $\mathtt{0}$, and hence $\varepsilon \in Z_0$. If $w = i \in \{\mathtt{1},\mathtt{2},\mathtt{3},\mathtt{4},\mathtt{5}\}$, the last digit of $C_T^0(w) \boxplus w$ is i and therefore $i \in Z_0$. In either case it follows that $w \in Z_0$, hence, by (10a), $w \notin L(T)$, which contradicts the accepting computation $C_T^0(w)$.

Conversely, assume (12) and suppose there exists $w \in \{\varepsilon,\mathtt{1},\mathtt{2},\mathtt{3},\mathtt{4},\mathtt{5}\}$, which is at the same time in $L(T)$ and in Z_0. Then there exists an accepting computation $C_T^0(w) \boxplus w \in \mathrm{VALC}(0)T$, that is, $C_T^0(w) \boxplus w \notin \mathrm{INVALC}(0)T$. However, $C_T^0(w) \boxplus w \in \mathtt{1}\Sigma_6^*\mathtt{0} \boxplus Z_0$, because $C_T^0(w) \in \mathtt{1}\Sigma_6^*\mathtt{0}$ and $w \in Z_0$ by assumption, which contradicts (12b). The contradiction obtained proves that no such w exists, which establishes (10a).

The equation for Z is the same as in Theorem 4.2 and in Lemma 7.2:

$$Z = Z_0 \cup Z_1 \cup \bigcup_{\substack{i\in\{\mathtt{2},\mathtt{3},\mathtt{4},\mathtt{5}\}\\ i'\in\Sigma_k}} \Big((Z_i \cap \mathtt{1}i'\Sigma_k^*) \boxplus (i-1)\mathtt{0}^* \cap ii'\Sigma_k^* \Big)$$

As in the previous cases, it is equivalent to (10c).

Linear conjunctive constants are expressed as in Theorem 4.2, using extra variables $(X_7,\ldots,X_n)$.

The set of solutions has been described, and, clearly, the greatest of them is $Z_0 = \overline{L(T)} \cap \{\varepsilon,\mathtt{1},\mathtt{2},\mathtt{3},\mathtt{4},\mathtt{5}\}$, $Z_i = \{\mathtt{1}w \mid w \in \Sigma_6^+,\ iw \notin L(T)\}$, $Z = \overline{L(T)}$, where the latter can be a complement of any given recursively enumerable set. □

Finally, the case of recursive languages and unique solutions can be established by combining the constructions of Lemmata 7.2 and 7.3.

Lemma 7.4. *For every recursive language $L \subseteq \Sigma_6^* \setminus 0\Sigma_6^*$ there exists a system of language equations of the form $\varphi_i(Y, Z, X_1, \ldots, X_n) = \psi_i(Y, Z, X_1, \ldots, X_n)$ with union, intersection and $\boxplus$, such that its unique solution is $Y = Z = L$, $X_i = K_i$, where $(K_1, \ldots, K_n)$ is some vector of languages.*

Proof. As a recursive language, L is both recursively enumerable and co-recursively enumerable, hence both Lemmata 7.2 and 7.3 apply. Consider both systems of language equations given by these lemmata, let Y be the variable from Lemma 7.2 let Z be the variable from Lemma 7.3, and let $X_1, \ldots, X_n$ be the rest of the variables in these systems combined. The set of solutions of the systems obtained is

$$\left\{ (Y, Z, f_1(Y, Z), \ldots, f_n(Y, Z)) \;\middle|\; Z \subseteq L \subseteq Y \right\}$$

Add one more equation to the system:

$$Y = Z$$

This condition collapses the bounds $Z \subseteq L \subseteq Y$ to $Z = L = Y$, and the resulting system has the unique solution

$$\big\{(L, L, f_1(L, L), \ldots, f_n(L, L))\big\},$$

which completes the proof. □

This construction yields further results on unary language equations over a unary alphabet by the same technique as in the multiple-letter case [13]. For instance, the solution existence problem is co-r.e.-complete, while the solution uniqueness problem is Π_2-complete. These results will be established in detail in a paper in progress.

8 Conclusions

We have considered the expressive power of language equations over sets of positional notations of numbers. If the allowed operations are union, intersection and addition in positional notation, the family of languages represented by solutions of such equations was shown to be a proper superset of linear conjunctive languages. It is contained in EXPTIME and includes some NP-complete languages. Our motivation for the study of these equations came from the study of conjunctive grammars over a unary alphabet, and our results have strong implications on these grammars.

If the complementation is further allowed, then the resulting equations turn out to be computationally universal. This result has immediate strong implications on language equations over a unary alphabet, on which we elaborate in an upcoming paper.

Acknowledgements

Research of the first author supported by MNiSW grant number N206 024 31/3826, 2006–2008, part of the research was done during the visit to Turku supported by the European Science Foundation short visit grant as part of the project "Automata: from Mathematics to Applications", reference number 1763. Research of the second author supported by the Academy of Finland under grant 118540.

References

[1] T. Buchholz, M. Kutrib, "On time computability of functions in one-way cellular automata", *Acta Informatica*, 35:4 (1998), 329–352.

[2] K. Culik II, J. Gruska, A. Salomaa, "Systolic trellis automata", I and II, *International Journal of Computer Mathematics*, 15 (1984), 195–212, and 16 (1984), 3–22.

[3] S. Ginsburg, H. G. Rice, "Two families of languages related to ALGOL", *Journal of the ACM*, 9 (1962), 350–371.

[4] J. Hartmanis, "Context-free languages and Turing machine computations", *Proceedings of Symposia in Applied Mathematics*, Vol. 19, AMS, 1967, 42–51.

[5] O. H. Ibarra, S. M. Kim, "Characterizations and computational complexity of systolic trellis automata", *Theoretical Computer Science*, 29 (1984), 123–153.

[6] A. Jeż, "Conjunctive grammars can generate non-regular unary languages", *Developments in Language Theory* (DLT 2007, Turku, Finland, July 3–6, 2007), LNCS 4588, 242–253.

[7] E. L. Leiss, "Unrestricted complementation in language equations over a one-letter alphabet", *Theoretical Computer Science*, 132 (1994), 71–93.

[8] A. Okhotin, "Conjunctive grammars", *Journal of Automata, Languages and Combinatorics*, 6:4 (2001), 519–535.

[9] A. Okhotin, "Conjunctive grammars and systems of language equations", *Programming and Computer Software*, 28 (2002), 243–249.

[10] A. Okhotin, "The hardest linear conjunctive language", *Information Processing Letters*, 86:5 (2003), 247–253.

[11] A. Okhotin, "On the equivalence of linear conjunctive grammars to trellis automata", *Informatique Théorique et Applications*, 38:1 (2004), 69–88.

[12] A. Okhotin, "Decision problems for language equations with Boolean operations", *Automata, Languages and Programming* (ICALP 2003, Eindhoven, The Netherlands, June 30–July 4, 2003), LNCS 2719, 239–251.

[13] A. Okhotin, "Unresolved systems of language equations: expressive power and decision problems", *Theoretical Computer Science*, 349:3 (2005), 283–308.

[14] A. Okhotin, O. Yakimova, "On language equations with complementation", *Developments in Language Theory* (DLT 2006, Santa Barbara, USA, June 26–29, 2006), LNCS 4036, 420–432.

[15] A. Okhotin, "Nine open problems for conjunctive and Boolean grammars", *Bulletin of the EATCS*, 91 (2007), 96–119.

Turku
Centre *for*
Computer
Science

Lemminkäisenkatu 14 A, 20520 Turku, Finland | www.tucs.fi

University of Turku
- Department of Information Technology
- Department of Mathematical Sciences

Åbo Akademi University
- Department of Computer Science
- Institute for Advanced Management Systems Research

Turku School of Economics and Business Administration
- Institute of Information Systems Sciences

ISBN 978–952–12–1913–9
ISSN 1239-1891