



# Boundedness of Cost Register Automata over the Integer Min-Plus Semiring

Andrei Draghici  

Department of Computer Science, University of Oxford, UK

Radosław Piórkowski  

Department of Computer Science, University of Oxford, UK

Andrew Ryzhikov  

Department of Computer Science, University of Oxford, UK

---

## Abstract

Cost register automata (CRAs) are deterministic automata with registers taking values from a fixed semiring. A CRA computes a function from words to values from this semiring. CRAs are tightly related to well-studied weighted automata. Given a CRA, the boundedness problem asks if there exists a natural number  $N$  such that for every word, the value of the CRA on this word does not exceed  $N$ . This problem is known to be undecidable for the class of linear CRAs over the integer min-plus semiring  $(\mathbb{Z} \cup \{+\infty\}, \min, +)$ , but very little is known about its subclasses. In this paper, we study boundedness of copyless linear CRAs with resets over the integer min-plus semiring. We show that it is decidable for such CRAs with at most two registers. More specifically, we show that it is, respectively, NL-complete and in coNP if the numbers in the input are presented in unary and binary. We also provide complexity results for two classes with an arbitrary number of registers. Namely, we show that for CRAs that use the minimum operation only in the output function, boundedness is PSPACE-complete if transferring values to other registers is allowed, and is coNP-complete otherwise. Finally, for each  $f_i$  in the hierarchy of fast-growing functions, we provide a stateless CRA with  $i$  registers whose output exceeds  $N$  only on runs longer than  $f_i(N)$ . Our construction yields a non-elementary lower bound already for four registers.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Quantitative automata

**Keywords and phrases** cost register automata, boundedness, decidability

**Digital Object Identifier** 10.4230/LIPIcs.CSL.2025.20

**Funding** All authors are supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant agreement No. 852769, ARiAT).

**Acknowledgements** We thank the anonymous reviewers for their helpful comments.



## 1 Introduction

A cost register automaton (CRA), introduced by Alur et al. [3], is a deterministic finite automaton over finite words equipped with a finite set of registers storing values from a fixed semiring. When a CRA reads a word, the values of its registers are updated in a write-only way using the semiring operations, and at the end it outputs a value from the semiring. Thus, a CRA defines a function from finite words to elements of a semiring. CRAs can hence be used to model quantitative behaviour of systems, and are tightly related to well-studied weighted automata (WAs) introduced by Schützenberger in [36], see also surveys [18, 19]. In this context, a fundamental question is, given a CRA or WA, to decide if a certain property of the function computed by it (or even just a property of the image of this function) holds.

In this paper, we study the boundedness problem, which, given a CRA, asks if there exists a natural number  $N$  such that the output of the CRA on every input is smaller than  $N$ . As discussed in more detail below, very little is known about decidability of this problem



© Andrei Draghici, Radosław Piórkowski, and Andrew Ryzhikov;  
licensed under Creative Commons License CC-BY 4.0

33rd EACSL Annual Conference on Computer Science Logic (CSL 2025).

Editors: Jörg Endrullis and Sylvain Schmitz; Article No. 20; pp. 20:1–20:23



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

compared to other natural problems for WAs and CRAs. Our work thus addresses and partially closes this gap, by studying it for restricted classes of CRAs. Below we provide an overview of such classes, and put them in the context of known classes of WAs.

We concentrate on CRAs over the integer min-plus semiring  $(\mathbb{Z} \cup \{+\infty\}, \min, +)$ . All CRAs are thus assumed to be over this semiring unless stated otherwise. Such CRAs can be seen as a variant of counter automata, specifically, as an extension of integer vector addition systems with states [21, 8]. Reachability properties of the latter are characterised by Presburger arithmetic, the first order theory of integer numbers with addition and order [21], which has good algorithmic features. For CRAs over the integer min-plus semiring, the situation changes significantly due to the presence of minimum operations in the updates. This makes the computed functions highly nonlinear: for example, they can compute iterated minimums. The results for integer vector addition systems thus cannot be directly used for CRAs. On the positive side, for subclasses of CRAs with decidable properties, this opens a possibility of finding new decidable extensions of Presburger arithmetic, by finding logical characterisation of the functions computed by CRAs from such subclasses.

More generally, many fundamental properties of CRAs and WAs can be defined by simple formulas in first-order logic, and can thus be considered as model checking CRAs against a fixed formula. For example, boundedness can be expressed by the formula

$$\exists N \in \mathbb{Z} \cup \{+\infty\}. N < +\infty \wedge \forall v \in \mathbb{Z} \cup \{+\infty\}. \mathcal{I}(v) \rightarrow (v < N),$$

where  $\mathcal{I}(v)$  is the predicate that holds true for  $v \in \mathbb{Z} \cup \{+\infty\}$  if and only if the CRA outputs the value  $v$  on some word. Understanding the decidability landscape of natural properties such as boundedness can thus be seen as a first step towards much more general model checking algorithms for subclasses of CRAs and WAs.

### Relations between CRAs and WAs

In general, CRAs are strictly more expressive than WAs [3]. However, WAs are equally expressive to linear CRAs, which are CRAs where the updates of the registers are restricted to affine transformations. Transforming a linear CRA into an equivalent WA and vice versa can be done in polynomial time [3]. Hence, linear CRAs can be seen as a deterministic model for inherently nondeterministic WAs. WAs, and thus CRAs, find their applications in the areas of language and speech processing [34], verification [9], image processing [10], and the analysis of on-line algorithms [5] and probabilistic systems [39]. Functions computable by WAs are exactly those that can be defined in weighted monadic second order logic, a natural extension of monadic second order logic [16, 17]. Functions computed by a subclass of CRAs were also characterised in [31] by maximal partition logic, a logic with regular quantifiers that allow to partition words into segments and then aggregate the values computed for them.

### Ambiguity hierarchy of WAs

To make decision problems tractable for WAs, it is usually required to restrict their expressiveness. The most well-studied way of doing that is by bounding the ambiguity, that is, the number of accepting runs labelled by a word. A WA is called finitely (respectively, linearly, polynomially or exponentially) ambiguous if there exists a constant (respectively, a linear, polynomial or exponential) function  $f(n)$  such that for every word  $w$  the number of accepting runs labelled by  $w$  is bounded by  $f(|w|)$ . If  $f(n) = 1$ , a WA is called unambiguous. Most classical decision problems, such as universality, inclusion and equivalence, are undecidable already for linearly ambiguous WAs over the integer min-plus semiring [28, 1, 12]. For finitely ambiguous WAs over this semiring, universality, inclusion and equivalence become decidable [40, 23].

For the boundedness problem, the situation is very different. A seminal paper [1] establishes undecidability of several classical decision problems for linearly ambiguous WAs over the integer min-plus semiring in a uniform fashion. However, it only proves boundedness to be undecidable for general (that is, exponentially ambiguous) WAs, and provides no classes with decidable boundedness. This indicates that boundedness is somehow different to other mentioned decision problems.

We remark that boundedness is PSPACE-complete for WAs over the *natural* min-plus semiring  $(\mathbb{N} \cup \{+\infty\}, \min, +)$  [1, 22, 29, 37], which requires completely different techniques than the integer case. We also remark that it is decidable for copyless linear CRAs over the semiring of positive rational numbers with usual addition and multiplication, and is undecidable for general WAs over the same semiring [11]. Transferring any such results to the min-plus semiring is unlikely, since these semirings has very different properties.

### Restricted classes of CRAs

One useful feature of CRAs is that, by adding syntactic restrictions on them, it is possible to introduce subclasses whose expressiveness is incomparable to known classes of WAs. This allows to obtain a finer decidability landscape compared to the case where only the formalism of WAs is used.

One notable example of that is the class of so called copyless linear CRAs. Informally, a CRA is called copyless if for every transition, the value of each of its registers can only be used once in the updates. Copyless linear CRAs are strictly less expressive than linearly ambiguous WAs, and their expressivity is incomparable to unambiguous WAs [2]. A further restriction of copyless linear CRAs to the case where the minimum operation is only allowed in the output function makes them equally expressive to finitely sequential WAs, which are unions of WAs whose underlying NFAs are deterministic [4, 13].

Restricting the number of registers in a class of CRAs also usually provides a subclass whose expressivity is incomparable to that of known classes of WAs. For example, there exists a copyless (but not linear) CRA with only 3 registers that computes a function not computed by any polynomially ambiguous WA [32]. For copyless linear CRAs restricted to only three registers universality is undecidable [15]. Moreover, there exist copyless linear CRAs with only two registers that are not equivalent to any unambiguous WA [2], see Example 6 on page 6 for one such CRA. All these results indicate that CRAs with few registers are quite expressive. Results on finding a CRA with the minimum number of registers computing a given function are presented in [13, 14, 25, 26].

More generally, for many problems the case of automata with two counters or registers often turns out to be already difficult enough. For example, most decision problems are undecidable for two-counter automata [33]. For vector addition systems with states, decidability of the reachability problem in the two-counter case was established in a seminal paper [24] several years before the proof that it is decidable for arbitrary number of counters was found [30], and it took over 20 more years to establish the precise computational complexity of the two-counter case [7].

### Our contributions

The main technical contribution of the paper, Theorem 17, states that boundedness is decidable for copyless linear CRAs with resets with at most two registers. Namely, we show that it is NL-complete if the numbers in the updates are presented in unary, and is in coNP if they are presented in binary. Functions computed by CRAs, even when they have only two

registers, are highly nonlinear and complex, mainly due to the presence of nested minimum operations. We illustrate this by providing in Section 6 a series of CRAs with very long shortest runs outputting a given value. To show that the two-register case is decidable, we identify several possible shapes of small witnesses of unboundedness (Section 4.2). The main challenge is then showing that if a CRA is unbounded, it contains one of these witnesses, which we do by carefully analysing the growth of the output value for a run providing a large enough value, and showing how to rearrange the cycles of this run to obtain a witness (Section 4.3). This requires in particular some geometrical arguments on the cones generated by the weight vectors of the cycles.

Our second contribution is establishing the complexity of boundedness for copyless linear CRAs with resets where the number of registers is arbitrary, but the minimum operation only occurs in the output function. As mentioned above, such CRAs compute the same class of functions as finitely sequential WAs. We show that boundedness is PSPACE-complete if registers are allowed to transfer values to other registers (Theorem 33), and is coNP-complete otherwise (Theorem 32). For upper bounds on the complexity, our techniques again rely on the combinatorial and geometrical analysis of cycles.

## 2 Main definitions

Symbols  $\mathbb{N}$  and  $\mathbb{Z}$  stand for natural and integer numbers respectively. Let  $\mathbb{N}_+ := \mathbb{N} \setminus \{0\}$ . We assume that the reader is familiar with the basic concepts in the area of formal languages and automata, see e.g. [38]. The Kleene star is denoted as  $(\cdot)^*$ . In regular expressions, we write  $\cup$  for the sum and  $\varepsilon$  for the empty string. The language of a nondeterministic finite automaton (NFA)  $\mathcal{A}$  is denoted as  $\mathcal{L}(\mathcal{A})$ . When speaking of underlying digraphs of NFAs, we use standard terms like simple cycle, reachability and backwards-reachability. We emphasise that by a simple cycle we mean a cycle that does not visit the same vertex more than once, except for its first and last vertex. General cycles do not have this restriction.

In this paper, we focus on CRAs over the integer min-plus semiring  $(\mathbb{Z} \cup \{+\infty\}, \min, +)$ . Hence, in what follows, we fix  $\mathbb{K} := (\mathbb{Z} \cup \{+\infty\}, \min, +)$ . For the proofs we present, the main focus lies on the operations on registers performed by CRAs, so we look at them in depth in Section 2.1. Then, in Section 2.2, we define CRAs and the boundedness problem.

### 2.1 Expressions, valuations and substitutions

Fix a finite set of variables  $X$ . By  $\text{Expr}(X)$  we denote the set of *expressions* constructed using operations  $\min\{\cdot, \cdot\}$  and  $+$ , variables from  $X$  and constants from  $\mathbb{K}$ . Expressions can be seen as polynomials over  $\mathbb{K}$ . Due to associativity, we allow arbitrary arity of the semiring operations in the expression notation. For an expression  $e \in \text{Expr}(\emptyset) \subseteq \text{Expr}(X)$  without variables, we denote by  $\text{eval}(e) \in \mathbb{K}$  its value.

A *substitution over  $X$*  is a function  $\nu: X \rightarrow \text{Expr}(X)$ . We denote the set of all substitutions over  $X$  by  $\text{Sub}(X)$ . When defining substitutions, we often treat them as sets of “*argument*  $\leftarrow$  *value*” pairs. When  $X$  is clear from the context, we implicitly extend partial substitutions with the identity mapping for the omitted arguments. For example, if  $X' = \{x_1, x_2\} \subsetneq X$ , then  $\nu = \{x_1 \leftarrow e_1, x_2 \leftarrow e_2\}$  denotes a substitution satisfying  $\nu(x_i) = e_i$  for  $x_i \in X'$  and  $\nu(x) = x$  for  $x \in X \setminus X'$ . A *valuation over  $X$*  is a substitution  $\mu: X \rightarrow \mathbb{K}$ . We denote by  $\text{Val}(X) \subset \text{Sub}(X)$  the set of all valuations over  $X$ . We write  $\mathbf{0} \in \text{Val}(X)$  for the valuation with  $\mathbf{0}(x) = 0$  for every  $x \in X$ . We assume that there is a fixed order on the set of registers, and thus in particular consider valuations equivalently as vectors.

► **Example 1** (An expression, a substitution and a valuation). Fix  $X := \{x, y, z\}$ .

$$\begin{aligned} e &:= \min\{10, x + 5, y + z\} && \in \text{Expr}(X) && \text{(an expression)} \\ \nu &:= \{x \leftarrow 2 + \min\{x, y\}, y \leftarrow 3\} && \in \text{Sub}(X) && \text{(a substitution)} \\ \mu &:= \{x \leftarrow 2, y \leftarrow 5, z \leftarrow 10\} && \in \text{Val}(X) && \text{(a valuation)} \end{aligned}$$

Substitutions can be *applied* to expressions: given  $e \in \text{Expr}(X)$  and  $\nu \in \text{Sub}(X)$ , by  $e[\nu]$  we denote the result of simultaneously replacing each occurrence of  $x$  with  $\nu(x)$  for every  $x \in X$ . Substitutions can be composed: for  $\nu, \nu' \in \text{Sub}(X)$ , we define the *composition*  $(\nu; \nu') \in \text{Sub}(X)$  as  $(\nu; \nu')(x) = \nu'(x)[\nu]$ . Applying a valuation  $\mu$  to an expression  $e$  yields an expression without variables – thus, with a defined value from  $\mathbb{K}$ . We hence define  $\text{eval}_\mu(e) := \text{eval}(e[\mu])$ . For a valuation  $\mu$  and a substitution  $\nu$ , we let  $\text{eval}_\mu(\nu) := \text{eval}(\mu; \nu)$ .

► **Example 2** (Application and composition of substitutions). Continuing Example 1, we have

$$\begin{aligned} e[\nu] &= \min\{10, (2 + \min\{x, y\}) + 5, (3) + (z)\} && \in \text{Expr}(X) && (e \text{ with } \nu \text{ applied to it}) \\ e[\mu] &= \min\{10, (2) + 5, (5) + (10)\} && \in \text{Expr}(\emptyset) && (e \text{ with } \mu \text{ applied to it}) \\ \text{eval}_\mu(e) &= \text{eval}(e[\mu]) = 7 && \in \mathbb{K} && \text{(the value of } e[\mu]) \end{aligned}$$

For  $e, e' \in \text{Expr}(X)$ , we write  $e \equiv e'$  if  $\text{eval}_\mu(e) = \text{eval}_\mu(e')$  for every  $\mu \in \text{Val}(X)$ . Additionally, for  $\nu, \nu' \in \text{Sub}(X)$ , we write  $\nu \equiv \nu'$  whenever  $\nu(x) \equiv \nu'(x)$  for every  $x \in X$ . For an expression  $e$ , by  $\text{maxc}(e)$  we denote the maximal absolute value of constants different to  $+\infty$  appearing in  $e$ , and 0 if there are none. We extend  $\text{maxc}$  naturally to substitutions.

### Copyless linear substitution with resets

In this paper, our main focus is on a special family of substitutions which are *copyless* and *linear with resets*. An expression  $e$  is in a *canonical linear form* if

$$e = \min\{x_1 + c_1, x_2 + c_2, \dots, x_k + c_k\}$$

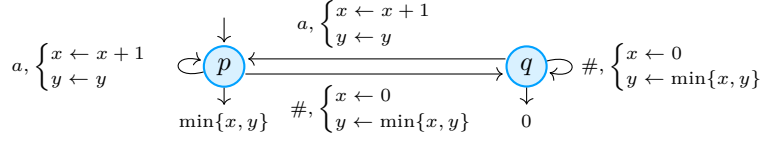
for some pairwise-different  $x_1, \dots, x_k \in X$ ,  $c_1, \dots, c_k \in \mathbb{K}$ , and  $k \in \mathbb{N}$ . Any expression  $e'$  such that  $e' \equiv e$  for an expression  $e$  in a canonical linear form is called *linear*. A substitution  $\nu$  is *linear with resets* if for every  $x \in X$ ,  $\nu(x)$  is either linear or 0 (a *reset*). A linear substitution with resets is *in a canonical form* if all its linear expressions are in a canonical linear form. We remark that the only reason why we use linear substitutions with resets instead of affine substitutions (that is, substitutions whose expressions are sums of linear and constant expressions) is to simplify the presentation of our techniques. Clearly, by adding more registers one can transform an affine CRA into a linear CRA with resets.

A substitution  $\nu$  is *copyless* if for all pairs  $x, x' \in X$  such that  $x \neq x'$  the expressions  $\nu(x)$  and  $\nu(x')$  feature disjoint sets of variables. By  $\text{Expr}_{\text{lin}}(X)$  and  $\text{Sub}_{\text{clr}}(X)$  we denote the sets of linear expressions and copyless linear substitutions with resets, respectively.

► **Example 3** (Copyless linear substitutions with resets). Consider the substitutions  $\nu$  and  $\nu'$ :

$$\nu := \begin{cases} x \leftarrow y + 5 \\ y \leftarrow \min\{x, z - 2\} \\ z \leftarrow 5 - 5 \end{cases} \equiv \begin{cases} x \leftarrow \min\{y + 5\} \\ y \leftarrow \min\{x + 0, z + (-2)\} \\ z \leftarrow 0 \end{cases}, \quad \nu' := \begin{cases} x \leftarrow \min\{x\} \\ y \leftarrow \min\{x, y\} \end{cases}$$

We have that  $\nu \in \text{Sub}_{\text{clr}}(\{x, y, z\})$  because it is equivalent to a linear substitution with resets in a canonical form, and variables  $x, y, z$  occur at most once in its expressions. In contrast,  $\nu'$  is linear, but not copyless, because  $x$  occurs in both  $\nu'(x)$  and  $\nu'(y)$ .



■ **Figure 1** An example of a CRA over the semiring  $\mathbb{K}$ .

## 2.2 Cost register automata

► **Definition 4** (CRA). A copyless linear CRA with resets over  $\mathbb{K}$  is a tuple

$$\mathcal{C} = (X, Q, \Sigma, \delta, q_{\text{ini}}, \text{out})$$

consisting of a finite set  $X$  of registers, a finite alphabet  $\Sigma$ , a finite set  $Q$  of control states, an initial state  $q_{\text{ini}} \in Q$ , and two functions:

- a transition rule function  $\delta: Q \times \Sigma \rightarrow Q \times \text{Sub}_{\text{chr}}(X)$ ,
- an output function  $\text{out}: Q \rightarrow \text{Expr}_{\text{lin}}(X)$ .

We also make the following assumptions that clearly preserve the property of being bounded or unbounded. We assume that all substitutions and expressions in  $\mathcal{C}$  are in a canonical form. Furthermore, we assume that  $+\infty$  never occurs as a constant in a substitution, since every such transition can be replaced with a transition leading to a new state, a self-loop incrementing the value of the corresponding register in this new state, and then a transition back.

In this paper, we consider only copyless linear CRAs with resets, hence for the rest of the paper we simply call them CRAs.

► **Definition 5** (Semantics of a CRA). A CRA  $\mathcal{C} = (X, Q, \Sigma, \delta, q_{\text{ini}}, \text{out})$  naturally induces a (possibly infinite) deterministic labelled transition system  $[[\mathcal{C}]] := (\text{Conf}_{\mathcal{C}}, \Sigma, \Delta_{\mathcal{C}}, c_{\text{ini}})$ , where  $\text{Conf}_{\mathcal{C}} := Q \times \text{Val}(X)$  is the set of configurations,  $c_{\text{ini}} := (q_{\text{ini}}, \mathbf{0})$  is the initial configuration and  $\Delta_{\mathcal{C}}: \text{Conf}_{\mathcal{C}} \times \Sigma \rightarrow \text{Conf}_{\mathcal{C}}$  is a transition function defined as  $\Delta_{\mathcal{C}}((q, \mu), \sigma) := (q', \text{eval}_{\mu}(\nu))$ , where  $(q', \nu) = \delta(q, \sigma)$ . We extend  $\Delta_{\mathcal{C}}$  to words as usually: for a configuration  $t$ , a letter  $\sigma \in \Sigma$  and a word  $w \in \Sigma^*$ ,  $\Delta_{\mathcal{C}}(t, \sigma w) := \Delta_{\mathcal{C}}(\Delta_{\mathcal{C}}(t, \sigma), w)$  and  $\Delta_{\mathcal{C}}(t, \varepsilon) := t$ . We further define the function  $\mathcal{C}: \Sigma^* \rightarrow \mathbb{K}$  calculated by  $\mathcal{C}$  as  $\mathcal{C}(w) = \text{eval}_{\mu}(\text{out}(q))$ , where  $(q, \mu) = \Delta_{\mathcal{C}}(c_{\text{ini}}, w)$ . Note that since we are only interested in the boundedness problem, the assumption that the initial value of each register is zero does not restrict the expressiveness of CRAs.

► **Example 6.** In Figure 1, we give an example of a CRA over the semiring  $\mathbb{K}$  and alphabet  $\Sigma = \{a, \#\}$ . It has two states  $p$  and  $q$ , where  $p$  is the initial state, and two registers  $x$  and  $y$ . The output function for state  $p$  (words ending with  $a$  and the empty word) is  $\min\{x, y\}$  and for state  $q$  (words ending with  $\#$ ) is 0. Both registers are initialised with value 0. If the input word ends with  $\#$  or is empty, this CRA outputs 0. Call a sequence of consecutive  $a$ 's a *block*. A block is *maximal* if it is not contained in a longer block. If the input word ends with  $a$ , the CRA outputs the length of the shortest maximal block. Register  $x$  stores the number of  $a$ 's read in the current block, and  $y$  stores the minimum length of maximal blocks read so far. This CRA is a copyless linear CRA with resets.

We say that a CRA  $\mathcal{C}$  is *bounded* if there is  $N \in \mathbb{N}$  such that  $\mathcal{C}(w) < N$  for all  $w \in \Sigma^*$ . We are now ready to state the main decision problem we are interested in.

► **Problem 7** (CRA boundedness).

Input CRA  $\mathcal{C}$ .

Question *Is  $\mathcal{C}$  bounded?*

### 3 Regular substitution languages

#### 3.1 Substitution languages associated to CRAs

In the boundedness problem, the input alphabet of a CRA is redundant, since the formulation is existentially quantified for the word and the underlying finite automaton is deterministic. For this reason, in this paper we focus on sequences of substitutions that a CRA can perform.

A *language of substitutions* is an arbitrary subset of  $\text{Sub}_{\text{clr}}(X)^*$ . Every word  $w \in \Sigma^*$  read by a CRA  $\mathcal{C}$  induces a sequence of substitutions that  $\mathcal{C}$  performs when reading  $w$ . Fix a CRA  $\mathcal{C} = (X, \Sigma, Q, q_{\text{ini}}, \delta, \text{out})$ . We define the language of substitutions  $\mathcal{L}_x(\mathcal{C})$  induced by it. Observe that the set of substitutions that occur in the transitions and output expressions of  $\mathcal{C}$  is finite. We denote it by  $\Gamma_{\mathcal{C},x}$ . The regular language  $\mathcal{L}_x(\mathcal{C})$  is then formally defined as the language of the following automaton  $\text{NFA}_x(\mathcal{C})$ . To simplify the presentation, we assume that the last substitution in the sequence corresponding to a word saves the output into a designated output register  $x \in X$ .

► **Definition 8** ( $\text{NFA}_x(\mathcal{C})$ ). *Given a CRA  $\mathcal{C} = (X, Q, \Sigma, \delta, q_{\text{ini}}, \text{out})$ , define a nondeterministic finite automaton  $\text{NFA}_x(\mathcal{C}) := (Q', \Gamma_{\mathcal{C},x}, \delta', q_{\text{ini}}, \{q_{\text{fin}}\})$  where  $Q' := Q \cup \{q_{\text{fin}}\}$  and the transition relation  $\delta' \subseteq Q' \times \Gamma_{x,\mathcal{A}} \times Q'$  contains transitions:*

$$\begin{aligned} (q, \nu, q') \in \delta' & \quad \text{for every } q, q' \in Q, \text{ and } \nu, \sigma \text{ such that } \delta(q, \sigma) = (q', \nu), \\ (q, \{x \leftarrow \text{out}(q)\}, q_{\text{fin}}) \in \delta' & \quad \text{for every } q \in Q. \end{aligned}$$

Fix an NFA  $\mathcal{A} = (Q, S, \delta, q_{\text{ini}}, Q_{\text{fin}})$  and a set of registers  $X = \{x_1, \dots, x_d\}$  for the rest of this subsection. An alternating sequence  $\pi = q_0 \xrightarrow{\nu_1} q_1 \xrightarrow{\nu_2} q_2 \rightarrow \dots \rightarrow q_{n-1} \xrightarrow{\nu_n} q_n$  of states from  $Q$  and letters from  $S$  is called a *run in  $\mathcal{A}$  labelled by the word  $w = \nu_1 \nu_2 \dots \nu_n$* . We write  $q_0 \xrightarrow{\pi, w} q_n$  to denote the fact that  $\pi$  is a run labelled by  $w$  that begins in  $q_0$  and ends in  $q_n$ . For such a run and  $\mu \in \text{Val}(X)$ , we define  $\text{eval}_\mu(\pi) := \text{eval}_\mu(w)$ . We identify words with compositions of the corresponding sequences of substitutions. The set of runs of  $\mathcal{A}$  is denoted by  $\text{Runs}(\mathcal{A})$ . A run  $\pi$  is *accepting* if  $q_{\text{ini}} \xrightarrow{\pi} q_{\text{fin}}$ . An NFA  $\mathcal{A}$  *accepts*  $w \in S^*$  whenever there exists an accepting run of  $\mathcal{A}$  labelled by  $w$ . The *language of  $\mathcal{A}$* , denoted  $\mathcal{L}(\mathcal{A})$ , is the set of words accepted by  $\mathcal{A}$ .

To be able to refer to segments of runs, we combine the notation for single transitions  $p \xrightarrow{\nu} q$  and runs  $q \xrightarrow{\pi, w} r$ . For example, we may consider a run  $\pi = p \xrightarrow{\nu} q \xrightarrow{\pi', w} r$  labelled by a word  $\nu \cdot w$ . When the labelling is not important, we write  $q \xrightarrow{\pi} r$ .

There is an obvious correspondence between runs in  $\llbracket \mathcal{C} \rrbracket$  and in  $\text{NFA}_x(\mathcal{C})$ . In particular, for  $k \in \mathbb{K}$ , there exists  $w \in \Sigma^*$  such that  $\mathcal{C}(w) = k$  if, and only if, there exists  $u \in \mathcal{L}(\mathcal{A})$  such that  $\text{eval}_0(u)(x) = k$ . This allows us to restate the boundedness problems in terms of languages of substitutions. We say that a regular language  $L \subseteq \text{Sub}_{\text{clr}}(X)^*$  *has bounded output* in  $x \in X$  if there exists  $N \in \mathbb{N}$  such that for every  $w \in L$  we have  $\text{eval}_0(s)(x) < N$ .

► **Problem 9** (Boundedness of regular  $d$ -register substitution languages).

Input *NFA  $\mathcal{A}$  over a finite set  $S \subset \text{Sub}_{\text{clr}}(X)$ ,  $|X| = d$ , output register  $x \in X$ .*

Question *Does  $\mathcal{L}(\mathcal{A})$  have bounded output in  $x$ ?*

Note that boundedness for CRAs (Problem 7) easily reduces to this problem in deterministic logarithmic space. Indeed, it suffices to compute  $\text{NFA}(\mathcal{A})$  for a given CRA with Definition 8. This reformulation allows us to use a rich framework of regular languages, which streamlines the proofs presented in later sections.

► **Definition 10** (Elementary substitutions). *We say that a substitution  $\nu \in \text{Sub}_{\text{clr}}(X)$  is elementary if it has one of the following forms for some  $x, y \in X$ ,  $c \in \mathbb{K}$ :*

$$\begin{array}{lll} \{x \leftarrow x + c\} & \text{(additive sub.)} & \{x \leftarrow y, y \leftarrow x\} & \text{(transposition)} \\ \{x \leftarrow \min\{x, y\}, y \leftarrow 0\} & \text{(minimum sub.)} & \{x \leftarrow 0\} & \text{(reset sub.)} \end{array}$$

Let  $\text{Sub}_{\text{elem}}(X) \subseteq \text{Sub}_{\text{clr}}(X)$  be the set of elementary substitutions, and let  $T_X \cup R_X \cup A_X \cup M_X$  be its partition into sets of transpositions, and reset, additive, and minimum substitutions.

► **Lemma 11.** *For every  $\nu \in \text{Sub}_{\text{clr}}(X)$  in a canonical form, there exists a word of substitutions  $u \in \text{Sub}_{\text{elem}}(X)^*$  of length  $O(d^2)$  such that  $u \equiv \nu$ .*

Note that the statement of the above lemma is not true in the general setting of  $\text{Sub}(X)$ , as its proof relies on copylessness. Note also that Lemma 11 implies the existence of a homomorphism to-elem:  $\text{Sub}_{\text{clr}}(X)^* \rightarrow \text{Sub}_{\text{elem}}(X)^*$  such that  $\nu \equiv \text{to-elem}(\nu)$  for every  $\nu \in \text{Sub}_{\text{clr}}(X)$ . For a finite set  $S \subset \text{Sub}(X)$ , we define  $\text{maxc}(S) := \max\{\text{maxc}(\nu) \mid \nu \in S\}$ . The following two claims are not difficult to prove.

▷ **Claim 12** (Maximal constant grows linearly w.r.t. length). Fix a finite  $S \subset \text{Sub}_{\text{elem}}(X)$ . For every  $w \in S^*$ , we have  $\text{maxc}(w) \leq |w| \cdot \text{maxc}(S)$ .

▷ **Claim 13** (Elementary substitutions assumption). We may assume w.l.o.g. that the alphabet  $S$  of  $\mathcal{A}$  consists only of elementary substitutions, i.e.,  $S \subset \text{Sub}_{\text{elem}}(X)$ .

### 3.2 The structure of witnesses with additive and reset substitutions only

In this subsection, we show how to simplify and decompose runs that do not contain any minimum substitutions or transpositions. We then use these results in the complexity upper bounds and to analyse runs with a more complex structure.

For the rest of this subsection, fix a set of registers  $X = \{x_1, \dots, x_d\}$ , an output register  $x \in X$ , and an NFA  $\mathcal{A}$  over a finite alphabet  $S \subset A_X \cup R_X \subset \text{Sub}_{\text{elem}}(X)$  of elementary additive and reset substitutions. Similarly to Claim 13, this covers a more general case where the alphabet of  $\mathcal{A}$  consists of substitutions adding integer values to some registers and resetting other registers.

Let  $w \in A_X^*$  be such that  $w \equiv \{x_1 \leftarrow x_1 + c_1, \dots, x_d \leftarrow x_d + c_d\}$  for some  $c_1, \dots, c_d \in \mathbb{K}$ . We define  $\text{eff}(w) := (c_1, \dots, c_d) \in \mathbb{K}^X$ , and we call it the *effect* of  $w$ . The *integer conic hull*  $\text{Cone}_{\mathbb{N}}(V)$  of a set of vectors  $V$  is the set of linear combinations of vectors from  $V$  with nonnegative integer coefficients. The following theorem is a direct consequence of a classical result by Carathéodory, see, e.g., [35].

► **Theorem 14** (Only  $d$  vectors are sufficient to represent a positive point). *Let  $V \subseteq \mathbb{Z}^d$ . If all components of a vector  $\vec{b}$  are strictly positive and  $\vec{b} \in \text{Cone}_{\mathbb{N}}(V)$ , then there exists  $V' \subseteq V$  with  $|V'| \leq d$  and a constant  $\lambda > 0$  such that  $\lambda \vec{b} \in \text{Cone}_{\mathbb{N}}(V')$ .*

We now state two important lemmas describing the shape of runs labelled by words over  $A_X$  and  $A_X \cup R_X$ , respectively.

► **Lemma 15** (Decomposition lemma for additive runs). *For every run  $q \xrightarrow{\pi, w} q'$  of  $\mathcal{A}$  such that  $w \in A_X^*$ , there exist  $n \in \mathbb{N}$ , words  $w_1, \dots, w_n, w', z_1, \dots, z_{n+1} \in A_X^*$ , and integers  $a_1, \dots, a_n \in \mathbb{N}$  such that*

- *for every word  $z \in z_1 w_1^* z_2 \dots z_n w_n^* z_{n+1}$ , there exists a run  $q \xrightarrow{z} q'$ ,*
- *$\text{eff}(w) = \text{eff}(w') + a_1 \text{eff}(w_1) + \dots + a_n \text{eff}(w_n)$ , and*
- *$|w_1|, \dots, |w_n|, |w'| \leq |Q|$  and  $|z_1 \dots z_{n+1}| \leq |Q|^2$ .*



**Proof idea.** We iterate a process that eliminates all the simple cycles from  $\pi$ . These simple cycles are labelled by some words  $w_1, \dots, w_d$  and the process returns a cycle-free run  $\pi'$  that is labelled by a word  $w'$ . We can describe the additive effect of the run  $\pi$  as the effect of  $w'$  plus the effect of all the simple cycles that we eliminated. Finally, we argue there exists a short run from  $q$  to  $q'$  that contains a vertex from each of these simple cycles. ◀

**Proof.** Consider the following iterative process on  $\pi$ . Initialise  $\vec{b} = \mathbf{0} \in \mathbb{Q}^X$ .

- Identify the first simple cycle  $r \xrightarrow{\pi_i, w_i} r$  in  $\pi$  and replace it by  $r$ . By simple cycle we mean a run where only the first and last states are equal.
- Update the value of the vector  $\vec{b}$  to  $\vec{b} + \text{eff}(w_i)$ .

Let  $q \xrightarrow{\pi', w'} q'$  be the resulting run. This process guarantees that  $\pi'$  is cycle-free and that  $\text{eff}(w) = \text{eff}(w') + \vec{b} = \text{eff}(w') + a_1 \text{eff}(w_1) + \dots + a_n \text{eff}(w_n)$ , where  $a_i$  is the number of times we have eliminated a simple cycle with effect  $w_i$ , for all  $1 \leq i \leq n$ .

Let  $Q' \subseteq Q$  be the set of states visited by  $\pi$ . The shortest run that visits all states of  $Q'$  has length at most  $|Q|^2$ . Since every word  $w_i$  corresponds to a simple cycle eliminated by the process, it follows that there exists words  $z_1, \dots, z_{n+1}$  such that for every word  $z \in z_1 w_1^* z_2 \dots z_n w_n^* z_{n+1}$ , there exists a run  $q \xrightarrow{z} q'$  and  $|z_1 \dots z_{n+1}| \leq |Q|^2$  even if  $n$  can be as large as  $2^{|Q|}$ . ◀

► **Lemma 16** (Pumping lemma). *If  $S \subseteq A_X \cup R_X$ , then there exists a word  $w \in \mathcal{L}(\mathcal{A})$  with  $\text{eval}_{\mathbf{0}}(w) > 2d|Q|C$ , where  $C := \max_C(S)$  if and only if there exist words  $\alpha_1, \dots, \alpha_{d+1}, \beta_1, \dots, \beta_d \in A_X^*$  such that*

- $\alpha_1 \beta_1^+ \alpha_2 \dots \alpha_d \beta_d^+ \alpha_{d+1} \eta_X \subseteq \mathcal{L}(\mathcal{A})$ ,
- $|\beta_1|, \dots, |\beta_d| \leq |Q|$
- $|\alpha_1 \dots \alpha_{d+1}| < (d+1)|Q|^2$ , and
- for each  $N \in \mathbb{N}$ , there are  $a_1, \dots, a_d \in \mathbb{N}$  with  $\text{eval}_{\mathbf{0}}(\alpha_1 \beta_1^{a_1} \alpha_2 \dots \alpha_d \beta_d^{a_d} \alpha_{d+1} \eta_X)(x_1) > N$ .

**Proof idea.** Given a run in  $\mathcal{A}$  of sufficiently large value, we can split it into different segments such that in each segment we know for every register if it is going to be reset in the future or not. Thus, for every register, we can determine if a segment of the run is relevant for determining its final value. Subsequently, we identify all cycles of this run and argue that since all the registers hold large values at the end of the run, the total !! effect of the relevant cycles (the ones after the last reset of the register) must be a large positive number for each register. This allows us to use Lemma 15 to conclude that we do not have too many such cycles and that we can pump them up in order to achieve unbounded register values. ◀

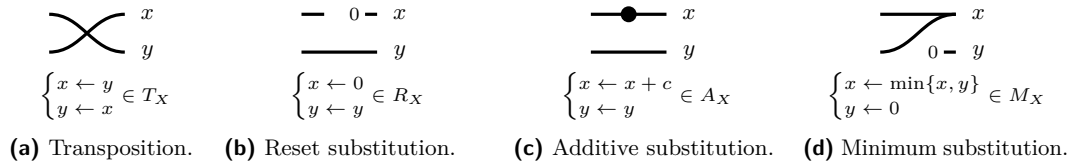
**Proof.** The right to left implication of this lemma is immediate, so we focus on the left to right implication. Assume there exists a run  $\pi = q_{\text{ini}} \xrightarrow{w} q_{\text{fin}}$ , for some  $q_{\text{fin}} \in Q_{\text{fin}}$  such that  $\text{eval}_{\mathbf{0}}(w)(x_1) > 2d|Q|C$ .

We can assume that the value of each register is reset at least once along  $\pi$ , since we can add additional reset substitutions to the beginning of  $w$  without changing the value of  $\text{eval}_{\mathbf{0}}(w)(x_1)$ . For each  $i$ ,  $1 \leq i \leq d$ , the  $\nu_i$  be the last reset substitution for register  $x_i$  in  $w$ . Without loss of generality, we can assume that in  $w$  the registers are reset for the last time in the increasing order of their indices. Then  $\pi$  can be represented as

$$\pi = q_{\text{ini}} \xrightarrow{\pi_1, w_1} q_1 \xrightarrow{\nu_1} q'_1 \xrightarrow{\pi_2, w_2} q_2 \xrightarrow{\nu_2} \dots \xrightarrow{\pi_d, w_d} q_d \xrightarrow{\nu_d} q'_d \xrightarrow{\pi_{d+1}, w_{d+1}} q_{d+1} \xrightarrow{\eta_X} q_{\text{fin}}.$$

Observe that for each  $i$ ,  $1 \leq i \leq d-1$ , we can replace in  $w_i$  all reset substitutions of registers  $x_{i+1}, \dots, x_d$  with the identity substitution without changing the value of  $\text{eval}_{\mathbf{0}}(w)(x_1)$ .





■ **Figure 2** Pictorial representation of elementary substitutions in  $\text{Sub}_{\text{elem}}(\{x, y\})$ . Register  $x$  is always drawn above  $y$ . A black dot stands for adding a constant – our graphical notation disregards the particular values of constants in the substitutions. A branching depicts the minimum operation. For the operations on  $y$ , the diagrams are symmetric. The effect of gluing several drawings together horizontally naturally corresponds to composition of depicted substitutions.

For a word  $w = \nu_1 \nu_2 \cdots \nu_n \in \text{Sub}_{\text{elem}}(X)^*$ , we define the *output derivation tree*  $\text{out-tree}(w)$  as the derivation tree of the expression  $w(x)$ . This tree can have three kinds of leaves: constants 0 originating from reset or minimum substitutions, arbitrary constants from  $\mathbb{K}$  coming from additive substitutions, and variables occurring in  $\nu_1(x)$  or  $\nu_1(y)$  of the first substitution  $\nu_1$ . We say that a path from a leaf to the root in  $\text{out-tree}(w)$  is *leading* if its starting leaf comes from a substitution  $\nu_i$  with the smallest  $i$  among all leaves that have a path to the root. A path is called  *$x$ -aligned* if all its vertices originate from expressions  $(\nu_i(x))_{1 \leq i \leq n}$ . A word  $w$  is called  *$x$ -aligned* if the leading path of  $\text{out-tree}(w)$  is  $x$ -aligned.

► **Example 20** (Depiction of  $\text{out-tree}(w)$ ). For  $w \in \text{Sub}_{\text{elem}}(X)^*$ ,  $\text{out-tree}(w)$  corresponds to the tree rooted at the rightmost position corresponding to  $x$  in the pictorial representation. Consider

$$w := \begin{array}{cccccccc} \left\{ \begin{array}{l} x \leftarrow x \\ y \leftarrow y + 3 \end{array} \right\} & \left\{ \begin{array}{l} x \leftarrow x \\ y \leftarrow 0 \end{array} \right\} & \left\{ \begin{array}{l} x \leftarrow x + 2 \\ y \leftarrow y \end{array} \right\} & \left\{ \begin{array}{l} x \leftarrow x \\ y \leftarrow y + 2 \end{array} \right\} & \left\{ \begin{array}{l} x \leftarrow \min\{x, y\} \\ y \leftarrow 0 \end{array} \right\} & \left\{ \begin{array}{l} x \leftarrow y \\ y \leftarrow x \end{array} \right\} & \left\{ \begin{array}{l} x \leftarrow x + 3 \\ y \leftarrow y \end{array} \right\} \\ \left\{ \begin{array}{l} x \leftarrow x + 3 \\ y \leftarrow y \end{array} \right\} & \left\{ \begin{array}{l} x \leftarrow \min\{x, y\} \\ y \leftarrow 0 \end{array} \right\} & \left\{ \begin{array}{l} x \leftarrow x \\ y \leftarrow y + 5 \end{array} \right\} & \left\{ \begin{array}{l} x \leftarrow x \\ y \leftarrow 0 \end{array} \right\} & \left\{ \begin{array}{l} x \leftarrow \min\{x, y\} \\ y \leftarrow 0 \end{array} \right\} & & \end{array}$$

The depiction of  $w$ , in line with Definition 19, is as follows:



The  $\text{out-tree}(w)$  corresponds to the subgraph drawn in black. The leading path  $\pi$  of  $\text{out-tree}(w)$  is marked with a blue outline. Expressions in  $w$  corresponding to vertices of  $\pi$  are typeset on blue background. As not all of them come from  $x \leftarrow e$  mappings, path  $\pi$  is not  $x$ -aligned. An  $x$ -aligned word  $w'$  such that  $w(x) \equiv w'(x)$  has the following shape:



► **Lemma 21** (Leading branch  $x$ -aligned assumption). *We can assume that each  $w \in \mathcal{L}(\mathcal{A})$  is  $x$ -aligned.*

Therefore, in the remainder of this section we assume that each  $w \in \mathcal{L}(\mathcal{A})$  is  $x$ -aligned. This means that only parts (b), (c) and (d) from Figure 2 (and not their symmetric  $y$ -counterparts) can be segments of runs.

## 4.2 Unboundedness witnesses

In this subsection, we define the shape of witnesses that prove unboundedness of CRAs. Next subsection shows that if a CRA is unbounded, it must contain such a witness. For the rest of this section, fix two substitutions

$$\eta = \{x \leftarrow \min\{x, y\}, y \leftarrow 0\} \text{ and } \rho = \{x \leftarrow x, y \leftarrow 0\},$$

which will be referred to throughout the whole section. Let  $N := |Q|$  and  $C := \max c(S)$ . We introduce two new notations for special types of runs of  $\mathcal{A}$ :

$$\begin{array}{c} \boxed{\phantom{r_1}} \\ \hline \alpha_1, w_1 \\ \hline r_1 \longrightarrow s_1 \end{array} \quad \text{and} \quad \begin{array}{c} \boxed{\phantom{r_2}} \\ \hline \alpha_2, w_2 \\ \hline r_2 \longrightarrow s_2 \end{array}$$

used to signify that  $w_1 \in A_X^*$  (i.e., has additive substitutions only), and  $w_2 \in (A_X \cup \{\rho\})^*$ .

► **Definition 22** (Unboundedness witness). *A run  $\pi$  in  $\mathcal{A}$  is called a trivial unboundedness witness if it has the form*

$$\pi = q_{\text{ini}} \xrightarrow{\alpha_1} r \xrightarrow{\theta} r \xrightarrow{\alpha_2} q_{\text{fin}}$$

such that  $|\pi| \leq 3N$ ,  $\text{eff}(\theta)(x) > 0$  and  $q_{\text{fin}} \in Q_{\text{fin}}$ .

A run  $\pi$  in  $\mathcal{A}$  is called a nontrivial unboundedness witness if it has the form

$$\pi = q_{\text{ini}} \xrightarrow{\pi_a} q_a \xrightarrow{\pi_b} q_b \xrightarrow{\pi_c} q_c$$

such that  $|\pi_a| \leq N$ ,  $|\pi_b|, |\pi_c| \leq 3N^2$ , run  $\pi_b$  is pumpable, and  $\pi_c$  is sustainable, where pumpable and sustainable runs are defined below.

A trivial or nontrivial unboundedness witness is called just an unboundedness witness.

Intuitively,  $\pi_a$  from this definition is used to reach a gadget enabling pumping,  $\pi_b$  witnesses that we can pump up the value of  $x$  to an arbitrarily large number and then end up in  $q_b$ , and  $\pi_c$  certifies that we can maintain a large value of  $x$  to be output in  $q_c \in Q_{\text{fin}}$ .

► **Definition 23** (Pumpable run). *A run  $\pi_b$  is called pumpable if it has one of the four forms:*

■ **Type A.1** (a cycle with a positive effect on  $x$ , then a reset of  $y$ )

$$\pi_b = q_a \xrightarrow{\theta} q_a \xrightarrow{\alpha} s \xrightarrow{\rho} q_b \quad \left( \begin{array}{c} \boxed{\phantom{q_a}} \\ \hline \theta \\ \hline \boxed{\phantom{q_a}} \end{array} \right)$$

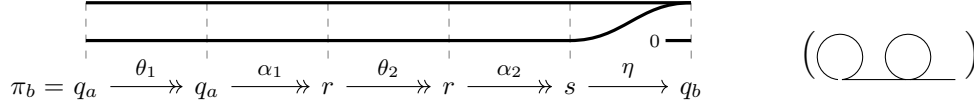
such that  $\theta$  is a cycle with  $\text{eff}(\theta)(x) > 0$  and  $\alpha$  is a run with no  $\eta$  substitutions.

■ **Type A.2** (a cycle with a positive effect on both  $x$  and  $y$ , then a minimum substitution)

$$\pi_b = q_a \xrightarrow{\theta} q_a \xrightarrow{\alpha} s \xrightarrow{\eta} q_b \quad \left( \begin{array}{c} \boxed{\phantom{q_a}} \\ \hline \theta \\ \hline \boxed{\phantom{q_a}} \end{array} \right)$$

such that  $\theta$  is a cycle with no  $\eta$  and no  $\rho$  substitutions and with  $\text{eff}(\theta) \in \mathbb{N}_+^2$ , and  $\alpha$  is a run with no  $\eta$  and no  $\rho$  substitutions.

- **Type A.3** (two cycles combining for a positive effect on both  $x$  and  $y$ , then a minimum)

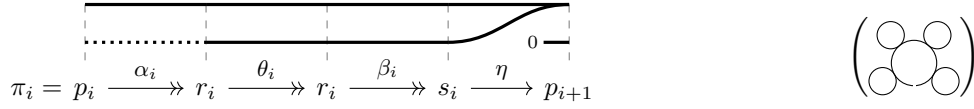


such that  $\theta_1, \theta_2$  are cycles with no  $\eta$  and no  $\rho$  substitutions and with  $a_1 \text{eff}(\theta_1) + a_2 \text{eff}(\theta_2) \in \mathbb{N}_+^2$ , for some  $a_1, a_2 \in \mathbb{N}$ . Furthermore,  $\alpha_1, \alpha_2$  are runs with no  $\eta$  and no  $\rho$  substitutions.

- **Type B** (a cycle with a positive effect on  $x$  together with cycles supporting its value)

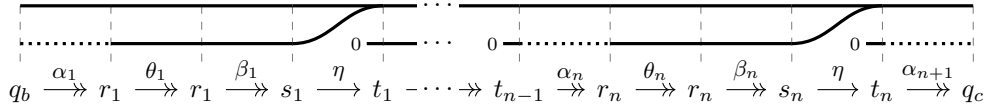
$$\pi = p_1 \xrightarrow{\pi_1} p_2 \xrightarrow{\pi_2} p_3 \rightarrow \dots \rightarrow p_{n-1} \xrightarrow{\pi_{n-1}} p_n \xrightarrow{\pi_n} p_1$$

for some  $n \in \mathbb{N}$ , where  $p_1 = q_a = q_b$ , and for every  $i$ ,  $1 \leq i \leq n$ :



such that  $\alpha_i$  is a run with no  $\eta$  substitutions,  $\theta_i$  is a cycle with no  $\eta$  and no  $\rho$  substitutions with  $\text{eff}(\theta_i) \in \mathbb{N} \times \mathbb{N}_+$  and  $\beta_i$  is a run with no  $\eta$  substitutions. Furthermore, we require  $\text{eff}(\alpha_1 \theta_1 \beta_1 \alpha_2 \theta_2 \beta_2 \dots \alpha_n \theta_n \beta_n)(x) > 0$ .

- **Definition 24** (Sustainable run). A run  $\pi_c$  is called sustainable if it is labelled by  $w_c \in (A_X \cup \{\rho, \eta\})^*$  and has the following form:



for  $n \in \mathbb{N}$ , runs  $\alpha_i, \beta_i$  of the form as depicted in the picture, and cycles  $\theta_i$  such that  $\text{eff}(\theta_i) \in \mathbb{Z} \times \mathbb{N}_+$  for every  $i$ ,  $1 \leq i \leq n$ .

- **Proposition 25.** Given  $\mathcal{A}$ , deciding if there exists a run in it which is an unboundedness witness is in NL if the numbers in the substitutions are presented in unary, and in NP if they are in binary.

**Proof.** Intuitively, using nondeterminism, we can guess a nontrivial witness  $\pi = \pi_a \pi_b \pi_c$  as in Definition 22 and verify that it has all the required properties. The case of a trivial witness is handled in a similar way and is thus omitted.

Indeed, starting in  $q_{\text{ini}}$ , we guess one transition at a time until we verify the existence of a witness or exceed the bound  $N + 4N^2$  on its length. During the traversal, we guess the positions of states  $q_a, q_b, q_c$  at appropriate distances from  $q_{\text{ini}}$ . This splits the search into three phases corresponding to  $\pi_a, \pi_b$  and  $\pi_c$ . We need to verify that  $\pi_b$  is of one of four types of pumpable runs (cf. Definition 23). At any point in time, if the definition of a pumpable run requires it, we can guess that the current state  $r$  marks the start of the occurrence of a simple cycle  $\theta$ . In this case, we store  $r$ , follow only labels from  $A_X \cup \{\rho\}$  and compute the effect of the run until  $r$  occurs again. This can easily be done in NL or NP depending on the representation of the numbers in the substitutions. ◀

In order to complete the proof of Proposition 18, we need to show that the existence of an unboundedness witness is equivalent to the fact that  $\mathcal{A}$  is not bounded. We show it by two implications stated below. We start with proving Lemma 26, and Lemma 29 is proved in the next subsection.

► **Lemma 26.** *If there exists an unboundedness witness then  $\mathcal{L}(\mathcal{A})$  is not bounded.*

**Proof.** The proof is straightforward in case of a trivial unboundedness witness. Fix a nontrivial unboundedness witness  $\pi$  as in Definition 22:

$$\pi = q_{\text{ini}} \xrightarrow{\pi_a} q_a \xrightarrow{\pi_b} q_b \xrightarrow{\pi_c} q_c.$$

Fix an arbitrary number  $N \in \mathbb{N}$ . We construct a run  $\pi'$  such that  $\text{eval}_{\mathbf{0}}(\pi')(x) \geq N$ . We first prove that

▷ **Claim 27.** For every  $M \in \mathbb{N}$  there is a run  $\pi'_b$  of  $\mathcal{A}$  from  $q_a$  to  $q_b$  such that  $\text{eval}_{\mathbf{0}}(\pi_a \pi'_b) = (M', 0)$  for some  $M' > M$ .

First, by Claim 12 we have that  $-CN \leq \text{eval}_{\mathbf{0}}(\pi_a)(x) \leq CN$ . The claim is immediate if  $\pi_b$  contains a certificate of type A.1, A.2, or A.3. Indeed, we simply repeat the cycles that occur there a sufficient number of times. If  $\pi_b$  contains a certificate of type B, then  $\pi_b = p_1 \xrightarrow{\pi_1} p_2 \xrightarrow{\pi_2} p_3 \rightarrow \dots \rightarrow p_{n-1} \xrightarrow{\pi_{n-1}} p_n \xrightarrow{\pi_n} p_1$  is a cycle such that the overall effect on  $x$  is positive. Since every sub-path  $\pi_1, \dots, \pi_n$  contains a cycle with a positive effect on  $y$  and a non-negative effect on  $x$ , there is a path  $\pi''_b$  that repeats each such cycle  $M + CN$  times. Note that  $\pi''_b$  is now a cycle with positive effect on  $x$  for any value of  $x$  smaller than  $M + CN$ . Thus, we can take  $\pi'_b$  to be the cycle  $\pi''_b$  taken  $M + CN$  times.

This finishes the proof of the claim. Now, it suffices to show the following:

▷ **Claim 28.** For every  $M \in \mathbb{N}$  there exists a run  $\pi'_c$  such that  $\text{eval}_{\mathbf{0}}(\pi_a \pi'_b \pi'_c)(x) \geq M$ .

We prove this claim by induction on the number of occurrences of  $\eta$  in  $\pi_c$ . Assume that  $\pi_c$  does not contain any occurrences of  $\eta$ . By Claim 27, there exists a path  $\pi'_b$  such that  $\text{eval}_{\mathbf{0}}(\pi_a \pi'_b)(x) \geq M + CN$ . Thus,  $\text{eval}_{\mathbf{0}}(\pi_a \pi'_b \pi_c)(x) \geq M$  by Claim 12, since the length of  $\pi_c$  is bounded.

Assume now that there is at least one occurrence of  $\eta$  in  $\pi_c$ . Then  $\pi_c$  can be represented as follows:

$$\pi_c = q_b \xrightarrow{\rho_1, \alpha_1} q_1 \xrightarrow{\rho_2, \alpha_2} q_2 \xrightarrow{\rho_3, \alpha_3} \dots \xrightarrow{\rho_r, \alpha_r} q_{\text{fin}},$$

where the cutting points are states reached after reading  $\eta$ . Assume that there is a run  $q_{\text{ini}} \xrightarrow{\pi_{i-1}, w_{i-1}} q_{i-1}$  such that  $\text{eval}_{\mathbf{0}}(w_{i-1})(x) \geq M'$ , for every  $M' \in \mathbb{N}$ . Then we can use the cycle with positive effect on  $y$  inside  $\rho_i$  in order to conclude that there exists a run  $q_{\text{ini}} \xrightarrow{\pi_i, w_i} q_i$  such that  $\text{eval}_{\mathbf{0}}(w_i) \geq M'$ , for every  $M' \in \mathbb{N}$ . This concludes the proof. ◀

► **Lemma 29.** *If  $\mathcal{L}(\mathcal{A})$  is not bounded, there exists an unboundedness witness.*

### 4.3 Unboundedness implies the existence of a witness

**Proof of Lemma 29.** Assume that  $\mathcal{L}(\mathcal{A})$  is not bounded. Let  $M := 15C^2N^3$  and  $W := 12CN^2$ . Let  $\pi$  be the shortest accepting run of  $\mathcal{A}$  such that  $\text{eval}_{\mathbf{0}}(\pi)(x) > M + W$ , and let  $w$  be the word labelling  $\pi$ . Since  $\pi$  is  $x$ -aligned, it can be split into two parts

$$\pi = q_{\text{ini}} \xrightarrow{\pi_{\text{pre}}} q_0 \xrightarrow{\pi_{\text{suf}}} q_{\text{fin}}$$

for some  $q_0 \in Q$  and  $q_{\text{fin}} \in Q_{\text{fin}}$ , such that  $\pi_{\text{suf}}$  is the shortest suffix of  $\pi$  satisfying  $\text{out-tree}(\pi) = \text{out-tree}(\pi_{\text{suf}})$ . Note that  $\text{eval}_{\mathbf{0}}(\pi_{\text{pref}})(x) = 0$ , and  $\pi_{\text{suf}}$  features no substitution  $\nu$  that resets  $x$  (i.e., for which  $\nu(x) = 0$ ). Recall that we defined

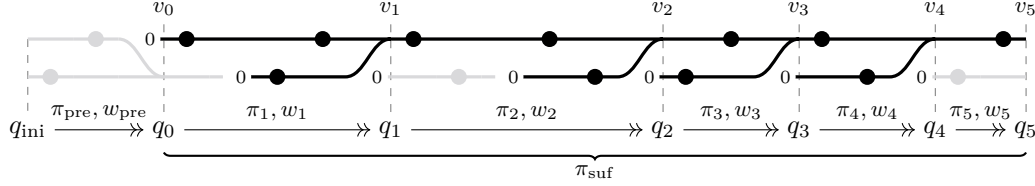
$$\eta = \{x \leftarrow \min\{x, y\}, y \leftarrow 0\} \text{ and } \rho = \{x \leftarrow x, y \leftarrow 0\}.$$

Since  $\pi$  is  $x$ -aligned, we have that  $w_{\text{suf}} \in (A_X \cup \{\eta, \rho\})^*$ , where  $w_{\text{suf}}$  is the word labelling  $\pi_{\text{suf}}$ . Let  $n \in \mathbb{N}$  be the number of substitutions  $\eta$  in  $w$ . Split the run  $\pi_{\text{suf}}$  into segments (some possibly empty)

$$q_0 \xrightarrow{\pi_1, w_1} q_1 \xrightarrow{\pi_2, w_2} q_2 \dashrightarrow \cdots \dashrightarrow q_n \xrightarrow{\pi_{n+1}, w_{n+1}} q_{n+1}$$

such that  $q_1, \dots, q_n$  are all the states reached directly after reading  $\eta$  each time. Define  $v_i := \text{eval}_0(\pi_{\text{pref}}\pi_1 \cdots \pi_i)(x)$  for  $0 \leq i \leq n+1$ . Observe that  $v_0 = 0$  and  $v_{n+1} \geq M + W$ .

► **Example 30.** Consider a run  $\pi$  partitioned into segments as defined above:



Let us overlook the slight inaccuracy that a run reaching a large value would have many more additive transitions (cf. Claim 12). The out-tree( $w$ ) is drawn in black, other (irrelevant) lines are drawn in light grey. Run  $\pi_{\text{suf}}$  is the shortest one that contains all black lines. There are  $n = 4$  occurrences of  $\eta$  in  $w_{\text{suf}}$ , thus  $\pi_{\text{suf}}$  is split into 5 parts, and runs  $\pi_1, \dots, \pi_4$  end with a transition labelled by  $\eta$ .

We first show that unboundedness guarantees the existence of a sustainable part  $\pi_c$  of a witness. Define  $m := \max\{i \mid v_i < M\}$ .

▷ **Claim 31.** For every  $i > m$  and every transition  $s \xrightarrow{\nu} t$  in  $\mathcal{A}$  such that  $t$  is a state visited by  $\pi_i$  and  $\nu(y) = 0$ , there exists a sustainable run  $\pi_c$  from  $t$  to  $q_{\text{fin}}$  of length at most  $3N^2$ .

We prove the claim by downward induction on  $i$ , the index of the segment  $\pi_i$  incident with the state  $t$  of  $\nu$ . Base case ( $i = n+1$ ) is trivial, as  $w_{n+1} \in A_X^*$ . Assume our claim holds for  $i+1$ . Fix a transition  $s \xrightarrow{\nu} t$  such that  $t$  is a state visited by  $\pi_i$ , and  $\nu(y) = 0$ . Similarly to the case of a trivial boundedness witness, a steep increase on  $y$  implies existence of a run  $\pi'_c = t \xrightarrow{\alpha} r \xrightarrow{\theta, w_\theta} s \xrightarrow{\beta, w_\beta} q_i$  such that  $w_\theta, w_\beta \in A_X^*$ ,  $\alpha, \beta$  are simple paths and  $\theta$  a simple cycle, and that  $\text{eff}(\theta)(y) > 0$ . Finally, by applying the inductive hypothesis to  $s \xrightarrow{\eta} q_i$ , we get a pumpable  $\pi''_c$  from  $q_i$  to  $q_{\text{fin}}$ ; we have thus constructed a sustainable run  $\pi'_c \pi''_c$ , as required.

It remains to show how to find  $\pi_b$ , the pumpable part of the run. We consider two cases.

**Case 1:**  $v_{i+1} - v_i > 4CN$  for some  $i \in \mathbb{N} \cap [m, n]$ . (aim: pumpable run of type A)  
Fix such  $i \in \mathbb{N}$ . Let  $\mathcal{A}' = (S, Q \cup \{q'_{i+1}\}, q_i, \{q'_{i+1}\}, \delta')$ , where  $\delta'$  is constructed from  $\delta$  by removing transitions with minimum substitutions, and adding  $q \xrightarrow{\eta} q'_{i+1}$  whenever  $q \xrightarrow{\eta} q_{i+1}$  for some  $q \in Q$ . Note that  $w_i \in \mathcal{L}(\mathcal{A}')$  and that  $\text{eval}_0(w_i)(x) > 4CN$ , thus automaton  $\mathcal{A}'$  satisfies the premises of Lemma 16. Using this lemma, we obtain a short pumpable run  $\pi'_b$  of type A of  $\mathcal{A}'$  – cases A.1, A.2 and A.3 were designed to match all possible loop arrangements. It naturally induces  $\pi_b$  in  $\mathcal{A}$  from  $q_i$  to  $q_{i+1}$  of the same properties. Since  $q_i$  is reachable from  $q_{\text{ini}}$ , there exists a short run  $\pi_a$  between them. Finally, by Claim 31, we obtain a sustainable part  $\pi_c$  of the witness, which completes the analysis of this case.

**Case 2:**  $v_{i+1} - v_i \leq 4CN$  for all  $i \in \mathbb{N} \cap [m, n]$ . (aim: witness exists or contradiction)  
This case proves to be more difficult, as it involves a more complicated type B pumpable run. The proof is by contradiction. Here, since  $v_{n+1} - v_m > (M + W) - M = 12CN^2$

## 20:16 Boundedness of Cost Register Automata over the Integer Min-Plus Semiring

and each  $v_{i+1}$  provides an increase of at most  $4CN$  compared to the previous value  $v_i$ , any maximal increasing subsequence of  $(v_i)_{m \leq i \leq n}$  must have at least  $\frac{12CN^2}{4CN} = 3N$  elements. Therefore, there exist  $k, \ell \in \mathbb{N}$  such that

$$m \leq k < \ell \leq n, \quad v_k < v_\ell, \quad M < v_i < M + W \quad \text{for } k \leq i \leq \ell, \quad \text{and} \quad q_k = q_\ell$$

thus  $\pi_{(k\ell)} := \pi_{k+1} \cdots \pi_\ell$  is a cycle. For each  $i$ ,  $k \leq i \leq \ell$ , define  $c_i \in \mathbb{K}$  to be the effect on  $x$  of the run  $\pi_i$  without its last transition labelled with  $\eta$ . We have  $v_{i+1} \leq v_i + c_i$ , and therefore  $\sum_{i=k}^{\ell-1} c_i > v_\ell - v_k > 0$ . Assume that  $\mathcal{A}$  has no pumpable run of type B from  $q_k$  to  $q_\ell$  (otherwise we obtain a witness easily). Therefore, there exists  $i \in \mathbb{N} \cap [k, \ell]$  such that  $\pi_i$  decomposes into

$$\pi_i := q_{i-1} \xrightarrow{\alpha, u} r_1 \xrightarrow{\beta, v} r_2 \xrightarrow{\eta} q_i,$$

where  $\beta$  is the run of maximal length labelled by some  $v \in A_X^*$  without  $\rho$ , run  $\alpha$  is labelled by  $u \in (A_X \cup \{\rho\})^*$  (possibly empty), and no simple cycle  $\theta$  with  $\text{eff}(\theta) \in \mathbb{N} \times \mathbb{N}_+$  is reachable from  $r_1$  and backwards-reachable from  $r_2$ . Note that  $\text{eval}_{\mathbf{0}}(\pi_{\text{pre}} \pi_1 \pi_2 \cdots \pi_{i-1} \alpha)(y) = 0$ .

Assume that  $\alpha$  contains a simple cycle  $\theta$ . If  $\text{eff}(\theta)(x) \leq 0$ , this cycle can be removed from  $\pi$  without decreasing the output value, which contradicts the assumption that  $\pi$  is the shortest. If otherwise  $\text{eff}(\theta)(x) > 0$ , we obtain a pumpable run of type A.1 featuring  $\theta$  and a simple path to  $r_1$ . Again, a sustainable run to the final state  $q_{\text{fin}}$  is guaranteed by Claim 31, and thus in this case the proof is finished.

Hence, we can assume that  $\alpha$  does not have simple cycles. Due to Claim 12, we have that  $\text{eff}(\alpha)(x) \in [-CN, CN]$ . By Lemma 15,  $\text{eff}(\beta)$  decomposes into  $\text{eff}(\beta) = \text{eff}(\beta') + (A, B)$  for a run  $\beta'$  from  $r_1$  to  $r_2$  of length  $\leq N$ , and  $(A, B) \in \text{Cone}_{\mathbb{N}}(\Theta)$ , where  $\Theta$  is the set of simple cycles that are reachable from  $r_1$  and backwards-reachable from  $r_2$  by transitions not involving  $\rho$ . By assumption,  $\text{eff}(\theta) \notin \mathbb{N} \times \mathbb{N}_+$  for any  $\theta \in \Theta$ . Thus, for each  $\theta \in \Theta$ , either  $\text{eff}(\theta)(x) < 0$  or  $\text{eff}(\theta)(y) \leq 0$ . Hence, for  $\theta$  with  $\text{eff}(\theta)(y) > 0$ , we have  $\text{eff}(\theta)(x) < 0$ . Take  $\theta_{\circlearrowleft}$  such that  $\text{eff}(\theta_{\circlearrowleft}) = (a, b)$ ,  $b > 0$  (and hence  $a < 0$ ) and  $\frac{b}{-a}$  is the largest possible among cycles satisfying these conditions. Every simple cycle has length at most  $N$ , therefore its effect belongs to  $[-CN, CN]^2$ . Thus,  $\frac{CN}{1} \geq \frac{b}{-a}$ . Let  $\ell(t) := \frac{b}{a}t$ . If there exists  $\theta'$  such that  $\text{eff}(\theta')$  lies above line  $\ell$ , then we have identified two cycles that span a cone having a nonempty intersection with the positive quadrant; this yields a pumpable run of type A.3, and, by Claim 31, we get a sustainable run starting at  $q_i$ .

Otherwise,  $\text{Cone}_{\mathbb{N}}(\Theta)$  lies below  $\ell$ . Since  $\pi_i$  ends with  $\eta$  and has effect at least  $M$ ,  $\text{eff}(\beta)(y) > M$ , therefore  $B > 0$ . This in turn implies  $A < 0$ , because  $(A, B)$  is below  $\ell$ . Hence  $B < \ell A = \frac{b}{a}A \leq -CNA$ . We know that  $\text{eval}_{(v_{i-1}, 0)}(\pi_i) \geq M$ , therefore

$$\begin{cases} \text{eval}_{(v_{i-1}, 0)}(\alpha\beta)(x) \geq M \\ \text{eval}_{(v_{i-1}, 0)}(\alpha\beta)(y) \geq M \end{cases} \quad \text{and thus} \quad \begin{cases} v_{i-1} + \text{eff}(\alpha)(x) + \text{eff}(\beta')(x) + A \geq M \\ 0 + \text{eff}(\beta')(y) + B \geq M \end{cases}$$

Since  $v_i < M + W$ , and effects of simple runs  $\alpha$  and  $\beta'$  are bounded by Claim 12, we get

$$\begin{cases} \mathcal{M} + W + 2CN + A \geq \mathcal{M} \\ CN + B \geq M \end{cases} \quad \text{and} \quad \begin{cases} 12C^2N^3 + 2C^2N^2 \geq -CNA \\ B \geq 15C^2N^3 - CN \end{cases}$$

Since  $B < -CNA$ , we have  $15C^2N^3 - CN < 12C^2N^3 + 2C^2N^2$  and finally  $3C^2N^3 < 2C^2N^2 + CN$  which yields a contradiction that concludes the proof.  $\blacktriangleleft$



## 5 Output-minimum CRAs

In this section, we consider CRAs in which minimum substitutions can only appear in the output function. We call such CRAs *output-minimum* for brevity.

The main results of this section are as follows.

► **Theorem 32.** *Boundedness of output-minimum CRAs with no transpositions is coNP-complete, even if the numbers in the substitutions are presented in unary.*

► **Theorem 33.** *Boundedness of output-minimum CRAs is PSPACE-complete, even if the numbers in the substitutions are presented in unary.*

For the rest of the section, fix the set of registers  $X := \{x_1, \dots, x_d\}$ . Let  $\eta_{X'} := \{x_1 \leftarrow \min\{x \mid x \in X'\}\}$  for  $X' \subseteq X$ . Once again, we use the formalism of regular languages of substitutions presented in Section 3. Recall that  $\text{Sub}_{\text{elem}}(X) = T_X \cup R_X \cup A_X \cup M_X$ . Since we are considering output-minimum CRAs, similarly to Claim 13, we can assume that the alphabet contains only elementary substitutions from  $T_X \cup R_X \cup A_X$ , with the only exception of a minimum transition which comes at the end of the word. Thus, in Section 5.1 and Section 5.2 we consider the language boundedness problem for NFAs  $\mathcal{A}$  such that for some  $X' \subseteq X$ , we have, respectively,  $\mathcal{L}(\mathcal{A}) \subseteq (A_X \cup R_X)^* \eta_{X'}$  and  $\mathcal{L}(\mathcal{A}) \subseteq (A_X \cup R_X \cup T_X)^* \eta_{X'}$ . As shown in Section 3, this is enough to prove Theorems 32 and 33.

### 5.1 Output-minimum CRAs with no transpositions

► **Proposition 34.** *Boundedness for regular subsets of  $(A_X \cup R_X)^* \eta_{X'}$  is coNP-complete, even if the numbers in the substitutions are presented in unary.*

**Proof.** As a certificate of unboundedness, we consider substitutions  $\alpha_1, \dots, \alpha_{d+1}, \beta_1, \dots, \beta_d$  respecting the conditions of Lemma 16, together with a run  $\pi$  of  $\mathcal{A}$  witnessing that  $\alpha_1 \beta_1^+ \alpha_2 \dots \alpha_d \beta_d^+ \alpha_{d+1} \eta_{X'} \subseteq \mathcal{L}(\mathcal{A})$ . Checking the second and third conditions of Lemma 16 is trivial and by having  $\pi$  in the certificate, it is also easy to check the first condition in linear time.

Checking the last conditions requires a bit more work. As argued in the proof of Lemma 16, all substitutions in  $\beta_1, \dots, \beta_d$  can be modified so that they become additive substitutions without changing the value of  $\text{eval}_{\mathbf{0}}(\alpha_1 \beta_1^{\alpha_1} \alpha_2 \dots \alpha_d \beta_d^{\alpha_d} \alpha_{d+1} \eta_{X'})(x_1)$ . Now, let the vectors  $\vec{v}_1, \dots, \vec{v}_d$  be the effects of the modified substitutions  $\beta_1, \dots, \beta_d$ . By Theorem 14, we only need to solve the following linear program

$$\exists a_1, \dots, a_d \in \mathbb{Q}_{\geq 0} \text{ s.t. } a_1 \vec{v}_1 + a_2 \vec{v}_2 + \dots + a_d \vec{v}_d > \mathbf{0},$$

which can be done in polynomial time.

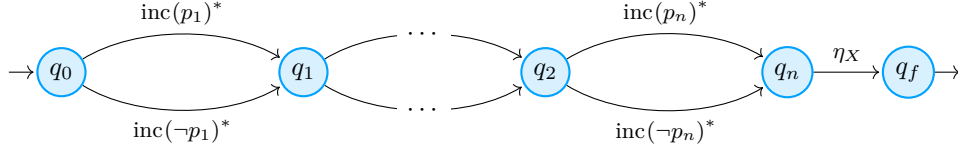
To prove coNP-hardness, we reduce the satisfiability problem, which is NP-complete [20], to the complement of the boundedness problem.

► **Problem 35 (Satisfiability).**

**Input** A set  $C = \{c_1, \dots, c_m\}$  of clauses over boolean variables  $p_1, \dots, p_n$ .

**Question** Does there exist an assignment of Boolean values to the variables satisfying all the clauses?

As registers, we take the set of all clauses:  $X = C = \{c_1, \dots, c_m\}$ . Let  $C_p := \{c \in C \mid p \models c\}$  and  $C_{\neg p} := \{c \in C \mid \neg p \models c\}$  be the sets of clauses satisfied by  $p = \top$  and  $p = \perp$ , respectively. Also, for a literal  $x$ , let  $\text{inc}(x) := \{c \leftarrow c + 1 \mid c \in C_x\}$ . Consider the generalised NFA  $\mathcal{A}$  in Figure 3.



■ **Figure 3** Generalised NFA  $\mathcal{A}$  for satisfiability. Transitions are labelled by regular expressions.

It is readily seen that for any  $N \in \mathbb{N}$ , there exists a word

$$w \in \left( \text{inc}(C_{p_1})^N \cup \text{inc}(C_{-p_1})^N \right) \cdots \left( \text{inc}(C_{p_n})^N \cup \text{inc}(C_{-p_n})^N \right) \eta_X$$

such that  $\text{eval}_0(w)(x_1) \geq N$  if and only if the variable assignment induced by  $w$  for variables  $p_1, \dots, p_n$  satisfies all the clauses  $c_1, \dots, c_m$ . ◀

## 5.2 Output-minimum CRAs with transpositions

► **Proposition 36.** *Boundedness for regular subsets of  $(A_X \cup R_X \cup T_X)^* \eta_X$  is in PSPACE.*

**Proof idea.** We prove containment in PSPACE by operating on an exponentially larger NFA  $\mathcal{P}_A$ , called permutation NFA. This NFA encodes all possible register permutations inside its state space, and hence its alphabet contains only additive and reset substitutions. It can be checked in NPSpace whether this larger NFA admits a certificate of the type presented in Lemma 16. By Savitch's theorem we get that the problem is in PSPACE [38]. ◀

**Proof.** Fix a finite alphabet  $S \subset A_X \cup R_X \cup T_X$  and let  $C = \maxc(S)$ . Fix NFA  $\mathcal{A} = (Q, S \cup \{\eta_{X'}\}, \delta, q_{\text{ini}}, Q_{\text{fin}})$  such that  $\mathcal{L}(\mathcal{A}) \subseteq S^* \eta_{X'}$ . Let  $G_X$  be the set of all permutations of the set  $X$  and let  $\mathcal{P}_A = (Q', S' \cup \{\eta_{X'}\}, \delta', q'_{\text{ini}}, Q'_{\text{fin}})$ , where  $Q' = Q \times G_X$ ,  $S' = S \setminus T_X$ , and  $Q'_{\text{fin}} = Q_{\text{fin}} \times G_X$ . We also take  $q'_{\text{ini}} = (q_{\text{ini}}, \tau_0)$ , where  $\tau_0$  is the identity permutation. Let  $\text{id} \in S$  be the additive substitution that adds 0 to every register. For  $s \in T_X$  and  $\tau \in X$  we define  $(\tau \circ s)(x) = s(\tau(x))$ . Finally,  $\delta'$  contains transitions

$$\begin{aligned} ((q, \tau), \text{id}, (q', \tau \circ s)) &\in \delta' && \text{for every } (q, s, q') \in \delta \text{ such that } s \in A_X, \\ ((q, \tau), s, (q', \tau)) &\in \delta' && \text{for every } (q, s, q') \in \delta \text{ such that } s \notin T_X. \end{aligned}$$

Clearly,  $\mathcal{A}$  is unbounded if and only if  $\mathcal{P}_A$  is unbounded if and only if there exist words  $\alpha_1, \dots, \alpha_{d+1}, \beta_1, \dots, \beta_d$  that adhere to the conditions of Lemma 16 and run  $\pi$  of  $\mathcal{P}_A$  witnessing that  $\alpha_1 \beta_1^+ \alpha_2 \dots \alpha_d \beta_d^+ \alpha_{d+1} \eta_{X'} \subseteq \mathcal{L}(\mathcal{P}_A)$ . Since  $|Q'| = |Q| \cdot d!$ , we can store one state with polynomial space. Hence, an NPSpace algorithm can non-deterministically search for the run  $\pi$  without constructing  $\mathcal{P}_A$  explicitly. Verifying the first three conditions of Lemma 16 can be done on the fly in linear space. Also, a non-deterministic algorithm can guess and verify that  $\vec{v}_1 \dots, \vec{v}_d$  are the effects of cycles  $\beta_1, \dots, \beta_d$  on the fly. Finally, it is easy to verify that a positive linear combination of them has positive effect on all registers. Thus, we can conclude the argument by recalling that NPSpace = PSPACE by Savitch's theorem [38]. ◀

► **Proposition 37.** *Boundedness for regular subsets of  $(A_X \cup R_X \cup T_X)^* \eta_X$  is PSPACE-hard, even if the numbers in the substitutions are presented in unary.*

**Proof idea.** We reduce the DFA intersection problem, which is PSPACE-complete [27]. For every state of each DFA, we create a separate register in the constructed CRA  $\mathcal{C}$ . We simulate reading a letter by all the DFAs by moving a large value to the registers corresponding to

the new active states of the DFAs, and keeping the values of all remaining registers zero. These large values come from a self-loop transition in the initial state of  $\mathcal{C}$ , and  $\mathcal{C}$  cannot return to the initial state afterwards. All DFAs accept the same word if and only if the large values can be simultaneously brought to the registers corresponding to the final states of the DFAs. The output of  $\mathcal{C}$  is thus set to be the minimum of these registers. ◀

**Proof.** We reduce from the following PSPACE-complete problem [27]:

► **Problem 38** (DFA intersection).

**Input**  $n \in \mathbb{N}$ , alphabet  $\Sigma$ ,  $n$  DFAs  $\mathcal{A}_i = (Q_i, \Sigma, \delta_i, q_{\text{ini}}^{(i)}, Q_{\text{fin}}^{(i)})$ ,  $1 \leq i \leq n$ .

**Question** Does there exist a word accepted by all the DFAs?

We can assume that each DFA has only one final state, which can be ensured as follows: add a new letter  $\#$  to  $\Sigma$  and two new states  $q_i^+$ ,  $q_i^-$  to each  $\mathcal{A}_i$ . For each  $i$ , make this new letter  $\#$  send all states from  $Q_{\text{fin}}^{(i)}$  to  $q_i^+$ , and all other states of  $\mathcal{A}_i$  to  $q_i^-$ . Make the new letter induce a self-loop for both  $q_i^+$ ,  $q_i^-$  and make  $q_i^+$  to be the only final state in each DFA.

We construct an NFA  $\mathcal{A}$  such that the language  $\mathcal{L}(\mathcal{A})$  is bounded if and only if  $\bigcap_{1 \leq i \leq n} \mathcal{L}(\mathcal{A}_i)$  is empty. Let  $X = \{q_i^{(j)} \mid 1 \leq j \leq n, q_i \in Q_j\}$ . The idea is that the registers correspond to the states of the DFAs, and register  $r_i^{(j)}$  has a positive value after reading  $w \in \Sigma$  if and only if the  $i$ 'th state in  $\mathcal{A}_j$  is active after reading  $w$ . Next, we describe  $\mathcal{L}(\mathcal{A})$  in terms of a regular language.

Let  $\nu_{\text{inc}} = \{q_{\text{ini}}^{(i)} \leftarrow q_{\text{ini}}^{(i)} + 1 \mid 1 \leq i \leq n\}$  be a substitution that increments the registers representing initial states for each  $\mathcal{A}_i$ . Also, for every  $\sigma \in \Sigma$ , let

$$T_{i,\sigma} = \{\{q' \leftarrow q\} \cup \{q \leftarrow 0 \mid q \neq q'\} \mid (q, \sigma, q') \in \delta_i, q \neq q'\} \text{ and}$$

$$T_\sigma = T_{1,\sigma} \cdot T_{2,\sigma} \cdots T_{n,\sigma},$$

$$T = \bigcup_{\sigma \in \Sigma} T_\sigma.$$

There is a substitution in  $T_{i,\sigma}$  that simulates every transition inside  $\mathcal{A}_i$  for letter  $\sigma \in \Sigma$ . The idea is that after we guess the next letter  $\sigma \in \Sigma$ , for each  $\mathcal{A}_i$  we need to simulate the transition that is executed when reading this letter. If we pick the correct transition, we move our positive value from register  $q_i$  to  $q'_i$ , and resetting all other registers does not change their values. However, if we pick a wrong transition, we reset our positive value and we can never recover. Then, a substitution from  $T_\sigma$  simulates executing a transition labelled by letter  $\sigma$  in all  $\mathcal{A}_i$  and a substitution from  $T$  simulates choosing a letter  $\sigma \in \Sigma$  and executing a transition labelled by  $\sigma$  in all  $\mathcal{A}_i$ . Finally let  $\nu_{\text{out}} = \{x \leftarrow \min\{q_{\text{fin}}^{(i)} \mid 1 \leq i \leq n\}\}$ . We argue that  $\mathcal{L}(\mathcal{A}) = \nu_{\text{inc}}^* \cdot T^* \cdot \nu_{\text{out}}$  is unbounded if and only if  $\bigcap_{1 \leq i \leq n} L(\mathcal{A}_i)$  is non-empty.

Consider a word  $w = \sigma_1 \dots \sigma_m \in \Sigma^*$  and an integer  $N \in \mathbb{N}$ . For every  $1 \leq i \leq n$  it follows inductively that there exists a word  $w'_j \in \text{inc}^{N+1} \cdot T_{\sigma_1} \cdot T_{\sigma_2} \cdots T_{\sigma_j}$  such that  $\text{eval}_0(w'_j)(q^{(i)}) = N + 1$ , for  $q \in Q_i$  if and only if  $q_{\text{ini}}^{(i)} \xrightarrow{\sigma_1 \dots \sigma_j} q^{(i)}$ . Thus, there exists a word  $w'_m \in \nu_{\text{inc}}^* \cdot T^* \cdot \nu_{\text{out}}$  such that  $\text{eval}_0(w'_m) = N + 1$  if and only if  $\bigcap_{1 \leq i \leq n} L(\mathcal{A}_i)$  is non-empty. ◀

## 6 Stateless CRAs

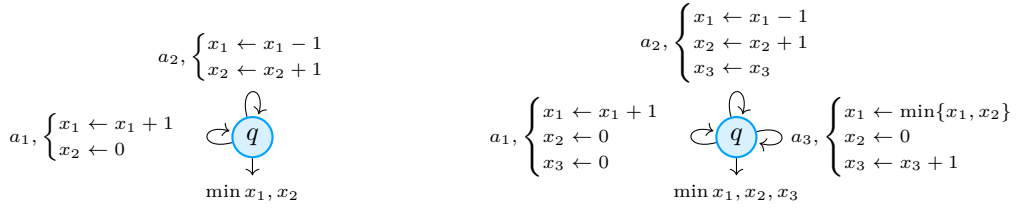
In this section, for every  $d \geq 2$  we present a fairly restricted family of unbounded CRAs with  $d$  registers such that the length of a shortest run outputting a value  $N \in \mathbb{N}$  is lower bounded by  $F_{d-1}(N)$ , the  $(d-1)$ st function in the hierarchy of fast-growing functions. These functions

## 20:20 Boundedness of Cost Register Automata over the Integer Min-Plus Semiring

are defined as follows. Let  $F_1(n) = 2n$  and for every  $k \geq 2$  let  $F_k(n) = F_{k-1} \circ \dots \circ F_{k-1}(1)$ , where  $\circ$  denotes the composition of functions, and this composition is taken  $n$  times. For example,  $F_2(n) = 2^n$  and  $F_3(n) = 2^{2^{\dots^2}} = \text{Tower}(n)$ , where exponentiation is taken  $n$  times. We construct these CRAs inductively in the following theorem.

► **Theorem 39.** *For every  $d \geq 2$ , there exists a stateless CRA  $\mathcal{C}$  with  $d$  registers and  $d$  transitions such that for every  $N \in \mathbb{N}_+$ , any run of  $\mathcal{C}_d$  that outputs a value of at least  $N$  must have length at least  $F_{d-1}(N)$ .*

**Proof.** For  $d = 2$  and  $d = 3$  consider the two CRAs in Figure 4.



■ **Figure 4** Unbounded CRAs  $\mathcal{C}_2$  (left) and  $\mathcal{C}_3$  (right) with 2 and 3 registers.

Let us prove that  $\mathcal{C}_2$  and  $\mathcal{C}_3$  satisfies the statement of the lemma. Let  $N \in \mathbb{N}_+$  be an arbitrary positive integer. Since we are dealing with stateless CRAs, we denote a configuration  $(q, \{x_1 \leftarrow k_1, x_2 \leftarrow k_2, \dots, x_n \leftarrow k_n\})$  by a vector  $(k_1, k_2, \dots, k_n)$ . Clearly, for both  $\mathcal{C}_2$  and  $\mathcal{C}_3$ , the transitions labelled by  $a_1$  are the only ones that can increase the value of register  $x_1$  and since these transitions reset the values of all other counters, any run outputting  $N$  must start by taking this transition  $m$  many times,  $m > 0$ , reaching in  $\mathcal{C}_2$  and  $\mathcal{C}_3$  the configurations  $(m, 0)$  and  $(m, 0, 0)$ , respectively. The value  $m$  can be seen as the initial budget that is necessary for increasing the values of other registers. Clearly, the shortest run that outputs  $N$  in  $\mathcal{C}_2$  reaches the configuration  $(2N, 0)$ , then takes  $N$  times the transition labelled by  $a_2$  and outputs. Since it must start by getting to the configuration  $(2N, 0)$ , its length is at most  $F_1(N) = 2N$ .

Let now  $\pi_3$  be a shortest run in  $\mathcal{C}_3$  that outputs  $N$ . Clearly,  $\pi_3$  needs to increase the value of register  $x_3$ . The transition labelled by  $a_3$  is the only one that increases  $x_3$ , however, it contains a minimum update for  $x_1$ . Since  $\pi_3$  is a shortest path outputting  $N$ , before reading  $a_3$  it reaches a configuration in which the values of registers  $x_2, x_3$  are equal, otherwise some transitions can be removed from it without changing the output value.

Thus,  $\pi_3$  initialises the budget by reading  $a_1^m$ , and then, before reading  $a_3$ , it reads a word in  $a_2^*$  which applies the function  $F_1^{-1}(\cdot) = \frac{\cdot}{2}$  to the value of register  $x_1$ . We argue that in order to output  $N$ ,  $m = F_2^{-1}(N) = 2^N$ . Indeed,  $\pi_3$  must reach a value  $m$  in register  $x_1$  and then apply  $N$  many times the function  $F_1^{-1}(\cdot)$  to register  $x_1$ , so  $m = F_2^{-1}(N)$ . Thus, the length of  $\pi_3$  must be longer than  $F_2(N)$ . Furthermore, we see that  $\pi_3$  has the following shape  $(0, 0, 0) \rightarrow (F_2(N), 0, 0) \rightarrow (N, N, N)$ .

Assume now that there exists  $\mathcal{C}_{d-1}$  with the property from the statement of the lemma. We modify it by adding a new letter  $a_d$  to the alphabet  $\Sigma$ , and extend the substitutions of the transitions as follows:

- add  $x_d \leftarrow 0$  to  $a_1$ ,
- add  $x_d \leftarrow x_d$  to  $a_i$  for  $1 < i < d - 1$ , and
- let the substitution of  $a_d$  be  $\{x_1 \leftarrow \min\{x_1, \dots, x_{d-1}\}, x_2 \leftarrow 0, \dots, x_{d-1} \leftarrow 0, x_d \leftarrow x_d + 1\}$ .

We know that there exists a shortest path  $\pi_{d-1}$  with the following shape  $(0, \dots, 0) \rightarrow (F_{d-2}(N), 0, \dots, 0) \rightarrow (N, N, \dots, N, 0)$ . So, in order to increase the register  $x_d$  by one, we need to have enough budget on register  $x_1$  to be able to apply the function  $F_{d-2}^{-1}$  to its value. Since we need to increase the value of  $x_d$  by one  $N$  times, it follows that we need to repeat this process  $N$  times so that  $\pi_d$  has shape  $(0, \dots, 0) \rightarrow (F_{d-1}(N), 0, \dots, 0) \rightarrow (N, N, \dots, N, N)$ . Thus its length must be at least  $F_{d-1}(N)$ . ◀

## 7 Conclusions and open problems

The most obvious open problem left by this work is the decidability of boundedness for copyless linear CRAs with resets with more than two registers. We conjecture that it is decidable for arbitrary number of registers. Our techniques and the shapes of the witnesses for the two-register case might be useful for proving that.

Another interesting open problem is the precise complexity of the two-register case where numbers in the substitutions are presented in binary. We have proved that this problem is NL-hard and in coNP, but no better bound is known even if minimum substitutions are only allowed in the output. In the latter case, it follows from our results that the witness of unboundedness consists of at most two cycles and some paths connecting them. This relates to the following natural problem whose complexity we were not able to find in the literature. Let  $G = (V, E)$  be a digraph, and  $\omega : E \rightarrow \mathbb{Z}^2$  be a (bi-criteria) weighting function on its edges. Given  $G$  and  $\omega$ , find a cycle in  $G$  such that the sum of weights of its edges is component-wise positive. This is of course a generalisation of the problem of finding a cycle of negative weight in a digraph, which can be solved in polynomial time by e.g. Bellman-Ford-Moore algorithm [6].

---

### References

- 1 Shaull Almagor, Udi Boker, and Orna Kupferman. What's decidable about weighted automata? *Information and Computation*, 282:104651, 2022. doi:10.1016/j.ic.2020.104651.
- 2 Shaull Almagor, Michaël Cadilhac, Filip Mazowiecki, and Guillermo A. Pérez. Weak cost register automata are still powerful. *International Journal of Foundations of Computer Science*, 31(6):689–709, 2020. doi:10.1142/S0129054120410026.
- 3 Rajeev Alur, Loris D'Antoni, Jyotirmoy Deshmukh, Mukund Raghothaman, and Yifei Yuan. Regular functions and cost register automata. In *28th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2013)*, pages 13–22, 2013. doi:10.1109/LICS.2013.65.
- 4 Rajeev Alur and Mukund Raghothaman. Decision problems for additive regular functions. In Fedor V. Fomin, Rusins Freivalds, Marta Z. Kwiatkowska, and David Peleg, editors, *Automata, Languages, and Programming - 40th International Colloquium, ICALP 2013, Riga, Latvia, July 8-12, 2013, Proceedings, Part II*, volume 7966 of *Lecture Notes in Computer Science*, pages 37–48. Springer, 2013. doi:10.1007/978-3-642-39212-2\_7.
- 5 Benjamin Aminof, Orna Kupferman, and Robby Lampert. Reasoning about online algorithms with weighted automata. *ACM Transactions on Algorithms*, 6(2):28:1–28:36, 2010. doi:10.1145/1721837.1721844.
- 6 Jørgen Bang-Jensen and Gregory Z Gutin. *Digraphs: theory, algorithms and applications*. Springer Science & Business Media, 2008.
- 7 Michael Blondin, Alain Finkel, Stefan Göller, Christoph Haase, and Pierre McKenzie. Reachability in two-dimensional vector addition systems with states is pspace-complete. In *30th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2015, Kyoto, Japan, July 6-10, 2015*, pages 32–43. IEEE Computer Society, 2015. doi:10.1109/LICS.2015.14.

- 8 Michael Blondin, Christoph Haase, Filip Mazowiecki, and Mikhail A. Raskin. Affine extensions of integer vector addition systems with states. *Log. Methods Comput. Sci.*, 17(3), 2021. doi:10.46298/LMCS-17(3:1)2021.
- 9 Krishnendu Chatterjee, Laurent Doyen, and Thomas A. Henzinger. Quantitative languages. *ACM Transactions on Computational Logic*, 11(4):23:1–23:38, 2010. doi:10.1145/1805950.1805953.
- 10 Karel Culík and Jarkko Kari. Digital images and formal languages. In Grzegorz Rozenberg and Arto Salomaa, editors, *Handbook of Formal Languages, Volume 3: Beyond Words*, pages 599–616. Springer, 1997. doi:10.1007/978-3-642-59126-6\_10.
- 11 Wojciech Czerwiński, Engel Lefauchaux, Filip Mazowiecki, David Purser, and Markus A. Whiteland. The boundedness and zero isolation problems for weighted automata over non-negative rationals. In Christel Baier and Dana Fisman, editors, *LICS '22: 37th Annual ACM/IEEE Symposium on Logic in Computer Science, Haifa, Israel, August 2 - 5, 2022*, pages 15:1–15:13. ACM, 2022. doi:10.1145/3531130.3533336.
- 12 Laure Daviaud. Containment and equivalence of weighted automata: Probabilistic and max-plus cases. In Alberto Leporati, Carlos Martín-Vide, Dana Shapira, and Claudio Zandron, editors, *Language and Automata Theory and Applications - 14th International Conference, LATA 2020, Milan, Italy, March 4-6, 2020, Proceedings*, volume 12038 of *Lecture Notes in Computer Science*, pages 17–32. Springer, 2020. doi:10.1007/978-3-030-40608-0\_2.
- 13 Laure Daviaud. Register complexity and determinisation of max-plus automata. *ACM SIGLOG News*, 7(2):4–14, 2020. doi:10.1145/3397619.3397621.
- 14 Laure Daviaud, Pierre-Alain Reynier, and Jean-Marc Talbot. A generalised twinning property for minimisation of cost register automata. In *31st Annual ACM/IEEE Symposium on Logic in Computer Science, (LICS 2016)*, pages 857–866, 2016. doi:10.1145/2933575.2934549.
- 15 Laure Daviaud and Andrew Ryzhikov. Universality and forall-exactness of cost register automata with few registers. In Jérôme Leroux, Sylvain Lombardy, and David Peleg, editors, *48th International Symposium on Mathematical Foundations of Computer Science, MFCS 2023, August 28 to September 1, 2023, Bordeaux, France*, volume 272 of *LIPICs*, pages 40:1–40:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICs.MFCS.2023.40.
- 16 Manfred Droste and Paul Gastin. Weighted automata and weighted logics. *Theoretical Computer Science*, 380(1-2):69–86, 2007. doi:10.1016/J.TCS.2007.02.055.
- 17 Manfred Droste and Paul Gastin. Weighted automata and weighted logics. In *Handbook of weighted automata*, pages 175–211. Springer, 2009.
- 18 Manfred Droste, Werner Kuich, and Heiko Vogler. *Handbook of Weighted Automata*. Springer Berlin, Heidelberg, 1st edition, 2009. doi:10.1007/978-3-642-01492-5.
- 19 Manfred Droste and Dietrich Kuske. Weighted automata. In Jean-Éric Pin, editor, *Handbook of Automata Theory*, pages 113–150. European Mathematical Society Publishing House, Zürich, Switzerland, 2021. doi:10.4171/Automata-1/4.
- 20 Michael R Garey and David S Johnson. *Computers and intractability*, volume 174. freeman San Francisco, 1979.
- 21 Christoph Haase and Simon Halfon. Integer vector addition systems with states. In Joël Ouaknine, Igor Potapov, and James Worrell, editors, *Reachability Problems - 8th International Workshop, RP 2014, Oxford, UK, September 22-24, 2014. Proceedings*, volume 8762 of *Lecture Notes in Computer Science*, pages 112–124. Springer, 2014. doi:10.1007/978-3-319-11439-2\_9.
- 22 Kosaburo Hashiguchi. New upper bounds to the limitedness of distance automata. *Theor. Comput. Sci.*, 233(1-2):19–32, 2000. doi:10.1016/S0304-3975(97)00260-0.
- 23 Kosaburo Hashiguchi, Kenichi Ishiguro, and Shuji Jimbo. Decisability of the equivalence problem for finitely ambiguous automata. *International Journal of Algebra and Computation*, 12(03):445–461, 2002. doi:10.1142/S0218196702000845.

- 24 John E. Hopcroft and Jean-Jacques Pansiot. On the reachability problem for 5-dimensional vector addition systems. *Theoretical Computer Science*, 8:135–159, 1979. doi:10.1016/0304-3975(79)90041-0.
- 25 Daniel Kirsten and Sylvain Lombardy. Deciding Unambiguity and Sequentiality of Polynomially Ambiguous Min-Plus Automata. In *26th International Symposium on Theoretical Aspects of Computer Science (STACS 2009)*, volume 3 of *LIPICs*, pages 589–600, 2009. doi:10.4230/LIPICs.STACS.2009.1850.
- 26 Ines Klimann, Sylvain Lombardy, Jean Mairesse, and Christophe Prieur. Deciding unambiguity and sequentiality from a finitely ambiguous max-plus automaton. *Theoretical Computer Science*, 327(3):349–373, 2004. doi:10.1016/j.tcs.2004.02.049.
- 27 Dexter Kozen. Lower bounds for natural proof systems. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, pages 254–266. IEEE Computer Society, 1977. doi:10.1109/SFCS.1977.16.
- 28 Daniel Krob. The equality problem for rational series with multiplicities in the tropical semiring is undecidable. *International Journal of Algebra and Computation*, 4(3):405–426, 1994. doi:10.1142/S0218196794000063.
- 29 Hing Leung and Viktor Podolskiy. The limitedness problem on distance automata: Hashiguchi’s method revisited. *Theor. Comput. Sci.*, 310(1-3):147–158, 2004. doi:10.1016/S0304-3975(03)00377-3.
- 30 Ernst W. Mayr. An algorithm for the general petri net reachability problem. In *Proceedings of the 13th Annual ACM Symposium on Theory of Computing, May 11-13, 1981, Milwaukee, Wisconsin, USA*, pages 238–246. ACM, 1981. doi:10.1145/800076.802477.
- 31 Filip Mazowiecki and Cristian Riveros. Maximal partition logic: Towards a logical characterization of copyless cost register automata. In Stephan Kreutzer, editor, *24th EACSL Annual Conference on Computer Science Logic, CSL 2015, September 7-10, 2015, Berlin, Germany*, volume 41 of *LIPICs*, pages 144–159. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2015. doi:10.4230/LIPICs.CSL.2015.144.
- 32 Filip Mazowiecki and Cristian Riveros. Copyless cost-register automata: Structure, expressiveness, and closure properties. *Journal of Computer and System Sciences*, 100:1–29, 2019. doi:10.1016/j.jcss.2018.07.002.
- 33 Marvin L. Minsky. *Computation: Finite and Infinite Machines*. Prentice-Hall, USA, 1967.
- 34 Mehryar Mohri. Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2):269–311, 1997. URL: <https://aclanthology.org/J97-2003>.
- 35 Alexander Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, Inc., USA, 1986.
- 36 Marcel-Paul Schützenberger. On the definition of a family of automata. *Information and Control*, 4(2):245–270, 1961. doi:10.1016/S0019-9958(61)80020-X.
- 37 Imre Simon. On semigroups of matrices over the tropical semiring. *RAIRO Theor. Informatics Appl.*, 28(3-4):277–294, 1994. doi:10.1051/ITA/1994283-402771.
- 38 Michael Sipser. *Introduction to the Theory of Computation*. Course Technology, Boston, MA, third edition, 2013.
- 39 Moshe Y. Vardi. Automatic verification of probabilistic concurrent finite-state programs. In *26th Annual Symposium on Foundations of Computer Science (FOCS 1985)*, pages 327–338, 1985. doi:10.1109/SFCS.1985.12.
- 40 Andreas Weber. Finite-valued distance automata. *Theoretical Computer Science*, 134(1):225–251, 1994. doi:10.1016/0304-3975(94)90287-9.