Weak Cost Register Automata are Still Powerful

Shaull Almagor[1], Michaël Cadilhac[1], Filip Mazowiecki[2], and Guillermo Pérez[3]

[1] University of Oxford
[2] LABRI, Université de Bordeaux
[3] Université Libre de Bruxelles

Abstract We consider one of the weakest variants of cost register automata over a tropical semiring, namely copyless cost register automata over $\mathbb{N}$ with updates using min and increments. We show that this model can simulate, in some sense, counter machines with zero-tests. We deduce that a number of problems pertaining to that model are undecidable, in particular equivalence, disproving a conjecture of Alur et al. from 2012. To emphasize how weak these machines are, we also show that they can be simulated by a restricted form of linearly-ambiguous weighted automata.

1 Introduction

Cost register automata (CRA) [2] encompass a wealth of computation models for functions from words to values (herein, integers). In their full generality, a CRA is simply a DFA equipped with registers that are updated upon taking transitions. The updates are expressions built using a prescribed set of operations (e.g., $+, \times, \min, \dots$), constants, and the registers themselves.

In this work, we will focus on CRA computing integer values, where the updates may only use "$+c$", for any constant c, and min. For instance:
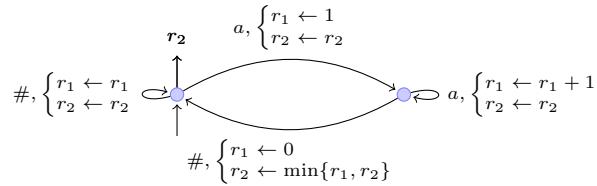
With r_1 initialized to 0 and r_2 to ∞, this CRA computes the length of the minimal nonempty block of a's between two #'s. This model has the same expressive power as weighted automata (WA) over the structure $(\mathbb{Z}, \min, +)$, but the use of registers can simplify the design of functions.

The example above enjoys an extra property that can be used to restrain the model (since a lot of interesting problems are undecidable on WA [1]). Indeed, *no register is used twice in any update function*; this property is called *copylessness*. This syntactic restriction, introduced by Alur et al. [2] and studied by Mazowiecki and Riveros [5], provably weakens the model. It was the hope of Alur et al. that this would provide a model for which equivalence is decidable.

Semilinearity and decidability of equivalence. Recall that a set $R \subseteq \mathbb{Z}^k$ is semilinear if it is expressible in first-order logic with addition: $\text{FO}[<, +]$. This latter logic being decidable [7], semilinearity is a useful tool to show decidability results. For instance, let $f, g \colon A^* \to \mathbb{Z}$ be expressible in some model for which the images of functions are effectively semilinear. Suppose further that the function $h \colon w \mapsto \min\{2 \times f(w), 2 \times g(w) + 1\}$ is also in that model. Since the image $h(A^*)$ is effectively semilinear, one can check whether it is always even: this would show that $f(w) \leq g(w)$ for all w. A first natural question is thus, is copyless CRA (CCRA) such a model?

Iterating min *breaks semilinearity.* Deterministic automata equipped with copyless registers with only "+c" updates are quite well-behaved [3, Section 6]; in particular, the set

$$R = \{\bar{r} \mid \bar{r} \text{ are the values of the registers at the end of an accepting run}\}$$

is semilinear. Naturally, $\min\{x, y\}$ is expressible in $\text{FO}[<, +]$, hence $\text{FO}[<, +] = \text{FO}[<, +, \min]$ (even, and this is not immediate, when the extra value ∞ is added [4]). This entails that if we were to give to these automata the ability to do a *constant* number of min, we would still have that R is semilinear. In this paper, we show that if the number of min is unbounded along runs, then the set is not semilinear (see the proof of Theorem 3 for a simple construction), and that it is undecidable to check whether R is semilinear.

Contributions. Beyond considerations on semilinearity, we show that CCRA over $\mathbb{N}$ can simulate counter machines with zero-tests (Theorem 1). Intuitively, the only words mapped by the CCRA to an even value are the correct executions of the counter machine. This construction is then used to show that equivalence

is undecidable for CCRA over $\mathbb{N}$ and that upper-boundedness is undecidable for WA. To better gauge the expressiveness of CCRA, we show that they are a weak form of *linearly-ambiguous* WA, that is, WA for which a word w has no more than $k \times |w|$ accepting runs, for some constant k (see drawing on page 5). Since the problems we tackle are decidable for *finitely-ambiguous* WA, CCRA are arguably the simplest generalization of deterministic WA for which equivalence is undecidable.

2 Preliminaries

We assume familiarity with automata theory, for which we settle some notations. We write $\mathbb{N}$ for $\{0, 1, 2, \ldots\}$, $\mathbb{Z}$ for the integers, and define $\mathbb{N}_\infty = \mathbb{N} \cup \{\infty\}$ and $\mathbb{Z}_\infty = \mathbb{Z} \cup \{\infty\}$. Naturally, $\min\{\ldots, \infty, \ldots\}$ stays the same when removing the ∞ value, and we set $\min \emptyset = \infty$. For any $k \geq 1$, we write $[k]$ for $\{1, 2, \ldots, k\}$. We write ε for the empty word.

Automata. An automaton (NFA) is a tuple (Q, A, δ, q_0, F), where Q is the set of states, A the alphabet, $\delta \subseteq Q \times (A \cup \{\varepsilon\}) \times Q$ the transition function, q_0 the initial state, and $F \subseteq Q$ the set of final states. We rely on the usual vocabulary pertaining to automata: a *run* is a word in δ^* starting in q_0, and such that each transition is consistent with the next; it is accepting if the last reached state is in F; a word $w \in A^*$ is accepted if there is an accepting run labeled by w.

If δ is a function from $Q \times A$ to Q, the automaton is *deterministic* (DFA). If there is a $k \in \mathbb{N}$ such that each accepted word w is the label of at most $k \times |w|$ accepting runs, the automaton is *linearly-ambiguous.*

Tropicalites. The only semirings (i.e., algebraic structures) that we will use are $(\mathbb{Z}_\infty, \min, +)$ and $(\mathbb{N}_\infty, \min, +)$. As with rings, matrix multiplication is well-defined in semirings; e.g., over 2×2 matrices, if $(a_{ij}) = (b_{ij}) \cdot (c_{ij})$, then:

$$a_{2,1} = \min\{b_{2,1} + c_{1,1}, b_{2,2} + c_{2,1}\} \ .$$

Weighted automata. Weighted automata will only be used in Section 3 and Theorem 4. A weighted automaton $\mathcal{W}$ over $\mathbb{K}$ ($\mathbb{K}$-WA) is a tuple $(\mathcal{A}, \lambda, \mu, \nu)$ where $\mathcal{A} = (Q, A, \delta, q_0, F)$ is an NFA, and $\lambda \in \mathbb{K}, \mu \colon \delta \to \mathbb{K}$, and $\nu \colon F \to \mathbb{K}$. Given a run $t_1 \cdot t_2 \cdots t_n \in \delta^*$ ending in a state $q \in F$ in cA, its *weight* is $\lambda + \mu(t_1) + \mu(t_2) + \mu(t_n) + \nu(q)$. The weight $\mathcal{W}(w)$ of a word $w \in A^*$ is the minimum weight for all accepting runs over w in the NFA. The $\mathbb{K}$-WA is *deterministic* (resp. *linearly-ambiguous*) if $\mathcal{A}$ is. We use $\mathbb{K}$-DetWA and $\mathbb{K}$-LinWA for these restrictions.

Registers and counters. A central goal of this work is to present a simulation of some *counter* machine with zero-tests by a *register* machine without zero-test but with more complicated update functions. To avoid confusion, we will stick to that vocabulary, and use c_i for counters and r_i for registers.

Cost register automata. In this work, we only consider cost register automata over $\mathbb{K} \in \{\mathbb{Z}_\infty, \mathbb{N}_\infty\}$ where the registers are updated using expressions that use min and "$+c$" for $c \in \mathbb{K}$. A precise, formal definition of the model will only be needed for Proposition 2; to present the main constructions, we will simply rely on the following more intuitive definition.

A $\mathbb{K}$-CRA $\mathcal{C}$ of dimension k is a DFA equipped with k registers $r_1, r_2, \ldots, r_k$ taking values in $\mathbb{K}$. The initial values of the registers are specified by a vector in $\mathbb{K}^k$, and each transition further induces a transformation of the form:

$$(\forall i \in [k]) \quad r_i \leftarrow \min\{r_1 + m_{1,i}, \ r_2 + m_{2,i}, \ \ldots, \ r_k + m_{k,i}, \ m_{k+1,i}\}$$

where each $m_{i,j}$ is in $\mathbb{K}$ (hence it can be ∞, making the subexpression irrelevant). Each final state is paired with an output function of the shape:

$$\min\{r_1 + m_1, \ r_2 + m_2, \ \ldots, \ r_k + m_k, \ m_{k+1}\}$$

where again the m_i's are in $\mathbb{K}$.

Given a word $w \in A^*$, the value of $\mathcal{C}$ on w, written $\mathcal{C}(w)$, is defined if w reaches a final state in the underlying DFA, and it is computed in the obvious way: the registers are initialized, then updated along the (single) run in the DFA, and the output is determined by the output function at the final state.

The $\mathbb{K}$-CRA is said to be *copyless* ($\mathbb{K}$-CCRA) if all the update functions verify, using the notations above, that for all $i \in [k]$, $|\{j \mid m_{i,j} \neq \infty\}| \leq 1$; in words, for each i, at most one of the subexpressions "$r_i + m_{i,j}$" will evaluate to a non-∞ value: the value of r_i impacts at most one register.

Vector addition systems with states and zero-tests. The main construction of this paper focuses on simulating counters with zero-tests. The precise formalism for our counter machines is a variant of vector addition systems with states (VASS) over $\mathbb{Z}^k$, equipped with transitions that can only be fired if a designated counter is zero. For any k, we define the *update alphabet* C_k as:

$$C_k = \bigcup_{i \in [k]} \{\mathbf{inc}_i, \mathbf{dec}_i, \mathbf{chk}_i\} \ ,$$

the intended meaning being that $\mathbf{inc}_i$ will increment the i-th counter, $\mathbf{dec}_i$ will decrement it, and $\mathbf{chk}_i$ will check that it is zero.

A $\mathbb{Z}$-VASS$^\mathbf{z}$ $\mathcal{V}$ of dimension k is a DFA (Q, C_k, δ, q_0, F). Consider a *configuration* $K = (q, \bar{c}) \in Q \times \mathbb{Z}^k$; writing $(\overline{e_i}) \in \mathbb{Z}^k$ for the standard basis:

- If $\delta(q, \mathbf{inc}_i) = q'$, then K can reach the configuration $(q', \bar{c} + \overline{e_i})$;
- If $\delta(q, \mathbf{dec}_i) = q'$, then K can reach the configuration $(q', \bar{c} - \overline{e_i})$;
- If $\delta(q, \mathbf{chk}_i) = q'$, then K can reach the configuration $(q', \bar{c})$ iff $c_i = 0$.

We say that the $\mathbb{Z}$-VASS$^\mathbf{z}$ reaches a state q if $(q_0, \bar{0})$ reaches, by a sequence of configurations, $(q, \bar{c})$ for some $\bar{c}$. We write $L_{\mathcal{V},q} \subseteq (C_k)^*$ for the *reachability language* of q, that is, the language of updates along the runs reaching q.

**Proposition 1.** *The following problem is undecidable:*
> ***Given:*** *A $\mathbb{Z}$-VASS$^{\mathbf{z}}$ $\mathcal{V}$ and a state q*
> ***Question:*** *Is $L_{\mathcal{V},q}$ empty?*

The problem stays undecidable even if $|L_{\mathcal{V},q}| \leq 1$ is guaranteed.

Proof. We see classical Minsky machines as an extension of $\mathbb{Z}$-VASS$^{\mathbf{z}}$ to streamline the reduction. Define $C'_k = C_k \cup \bigcup_{i \in [k]} \{\mathbf{chk}_i\}$. A k-counter machine is an automaton over C'_k, with the $\mathbb{Z}$-VASS$^{\mathbf{z}}$ semantics, augmented with the property that a transition labeled $\overline{\mathbf{chk}}_i$ can only be taken if the i-th counter is *non*zero.

Minsky [6] showed that the emptiness of reachability languages is undecidable for these machines—in particular, even if it is assumed that there is at most one run reaching the given state. To show the same for $\mathbb{Z}$-VASS$^{\mathbf{z}}$, we need only remove the transitions labeled $\overline{\mathbf{chk}}_i$, while preserving the reachability languages. To do so, it suffices to replace them with the following gadget, where j is a new counter and some states are omited:
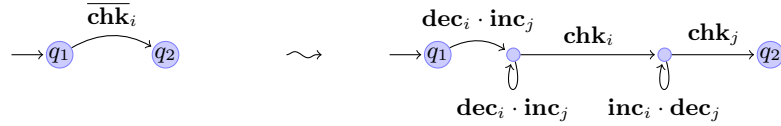

It is easily checked that upon reaching state q_2, the i-th counter is restored to its value in q_1, the j-th is 0, and the state can only be reached if the i-th counter were strictly positive. □

3 CCRA and weighted automata

With the plethora of models computing functions from word to values in modern literature, it is imperative to justify studying the seemingly artificial CCRA. In this section, we provide a normal form that will demonstrate that these machines are but deterministic weighted automata with a small dose of nondeterminism. In particular, all the problems we show to be undecidable in Section 5 turn out to be decidable for deterministic (or even finitely-ambiguous) weighted automata; this gives credence to the assertion that $\mathbb{N}_\infty$-CCRA is one of the weakest models for which equivalence, for instance, is undecidable.

In the following proposition, it is shown that any $\mathbb{K}$-CCRA can be expressed as a a DFA making nondeterministic jumps into a DetWA; graphically, every $\mathbb{K}$-CCRA is equivalent to:
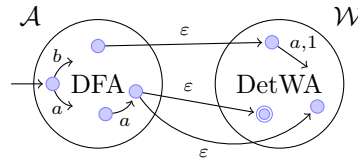
Proposition 2. *Let $f\colon A^* \to \mathbb{K}$ be a $\mathbb{K}$-CCRA. There are a DFA $\mathcal{A}$ with state set Q and initial state q_0, a $\mathbb{K}$-DetWA $\mathcal{W}$ with state set Q', and a function $\eta\colon Q \to \mathcal{P}(Q')$ such that:*

$$(\forall w \in A^*) \qquad f(w) = \min\{\mathcal{W}^q(v) \mid w = uv \wedge q \in \eta(q_0.u)\} \ ,$$

where $\mathcal{W}^q$ is $\mathcal{W}$ with the initial state set to q, and $q_0.u$ is the state reached by reading u in $\mathcal{A}$.

Proof. We first sketch the proof idea. Consider a nondeterministic variant of a given $\mathbb{K}$-CCRA $\mathcal{C}$ where updates of the form $r_1 \leftarrow \min\{r_2, r_3\}$ become nondeterministic jumps between the updates $r_1 \leftarrow r_2$ and $r_1 \leftarrow r_3$. The final value of this variant is set to be the minimum output of any run. Then this variant has the same output value as the original CRA, by distributivity of the min operation. We implement that strategy using a DFA $\mathcal{A}$ which, on resets ($r_1 \leftarrow 0$), starts a new run within a DetWA that follows the increments ($r_1 \leftarrow r_1 + m$) and movements ($r_i \leftarrow r_1$) of the register.

We now formalize the definition of CRA. A $\mathbb{K}$-CCRA $\mathcal{C}$ of dimension k is a tuple $(\mathcal{A}', \overline{\lambda}, \mu, \nu)$ where $\mathcal{A}' = (Q, A, \delta, q_0, F)$ is a DFA, $\overline{\lambda} \in \mathbb{K}^{1 \times k}$ is the initial value of the k registers, $\nu\colon F \to \mathbb{K}^{(k+1) \times 1}$ gives the output function for each final state, and $\mu\colon Q \times A \to \mathbb{K}^{(k+1) \times (k+1)}$ provides the update functions. To compare with the definition on page 4, using the notation therein, $\mu(q, a)$ is:

$$\begin{pmatrix} m_{1,1} & m_{1,2} & \cdots & m_{1,k} & \infty \\ m_{2,1} & m_{2,2} & \cdots & m_{2,k} & \infty \\ \vdots & \vdots & \ddots & \vdots & \\ m_{k,1} & m_{k,2} & \cdots & m_{k,k} & \infty \\ m_{k+1,1} & m_{k+1,2} & \cdots & m_{k+1,k} & 0 \end{pmatrix}$$

It can be readily checked that $(\overline{r}', 0) = (\overline{r}, 0) \cdot \mu(q, a)$ indeed verifies, for all $i \in [k]$, recalling that the multiplication is made in the semiring $(\mathbb{K}, \min, +)$, that:

$$r_i' = \min\{r_1 + m_{1,i}, \ r_2 + m_{2,i}, \ \ldots, \ r_k + m_{k,i}, \ m_{k+1,i}\} \ .$$

Note that the $(k+1)$-th component is a virtual register that will be maintained to 0. Given a run $(q_0, w_0) \cdot (q_1, w_1) \cdots (q_n, w_n) \in (Q \times A)^*$ in $\mathcal{A}'$ ending in a state $q_{n+1} \in F$, the output value is then defined as:

$$\mathcal{C}(w_0 w_1 \cdots w_n) = \overline{\lambda} \cdot \mu(q_0, w_0) \cdot \mu(q_1, w_1) \cdots \mu(q_n, w_n) \cdot \nu(q_{n+1}) \ .$$

Now that the precise definition of $\mathbb{K}$-CRA is settled, we present the construction. We will assume that $\overline{\lambda} \in \{0, \infty\}^k$ and that the updates are in one of two possible forms:

- $r_i \leftarrow \min\{r_1 + m_1, r_2 + m_2 \ldots, r_k + m_k\}$, that is, no constant term appears;
- $r_i \leftarrow 0$.

In symbols, this means that if $\mu(q, a) = (m_{i,j})$, then for any $i \in [k]$, either $m_{k+1,i}$ is ∞ or all $m_{j,i}$, for $j \in [k]$, are ∞. Any $\mathbb{K}$-CCRA can be put under that form using standard techniques.

The automaton $\mathcal{A}$ is the underlying automaton of $\mathcal{C}$, augmented with the information of *which registers* were reset by the previous transition. More precisely, $\mathcal{A} = (Q \times \mathcal{P}([k]), A, \delta_{\mathcal{A}}, q'_0, \emptyset)$ where $q'_0 = (q_0, \{i \mid \lambda_i = 0\})$; note that the final states are irrelevant. The transition function $\delta_{\mathcal{A}}$ is defined by:

$$\delta_{\mathcal{A}}((q, \cdot), a) = (\delta(q, a), E) \quad \text{where } E = \{i \mid \mu(q, a)_{k+1,i} = 0\} .$$

The $\mathbb{K}$-DetWA $\mathcal{W}$ consists of k copies of $\mathcal{C}$, one for each register. Formally, $\mathcal{W} = (\mathcal{B}, \overline{0}, \mu_{\mathcal{W}}, \nu_{\mathcal{W}})$ with $\mathcal{B} = (Q \times [k], A, \delta_{\mathcal{B}}, (q_0, 1), F \times [k])$; here, the initial valuation is irrelevant. We now define the transition function $\delta_{\mathcal{B}}$ and the weight function $\mu_{\mathcal{W}}$. Let (q, x) be a state of $\mathcal{B}$ and $a \in A$. By copylessness, there is at most one y such that $\mu(q, a)_{x,y}$ is not ∞. If one such y exists, then:

$$\delta_{\mathcal{B}}((q, x), a) = (\delta(q, a), y)$$
$$\mu_{\mathcal{W}}((q, x), a) = \mu(q, a)_{x,y} .$$

The output function of $\mathcal{W}$ is then, for any $q \in Q, i \in [k]$, $\nu_{\mathcal{W}}(q, i) = \nu(q)_i$.

Finally, $\eta: Q \times \mathcal{P}([k]) \to \mathcal{P}(Q \times [k])$ is defined as $\eta(q, E) = \{(q, i) \mid i \in E\}$.

Consider a word $w \in A^*$, and a factorization $w = uv$. The word u reaches a state q in $\mathcal{C}$, and a state (q, E) in $\mathcal{A}$. The last transition taken in $\mathcal{C}$ reading u updated all the registers $r_i, i \in E$, with the value 0. For each of these i's, there will be a run over v in $\mathcal{W}$, starting at (q, i), which follows the updates applied to r_i. This process thus simulates the nondeterministic variant of $\mathcal{C}$ described above, showing the Proposition. $\qquad\square$

Corollary 1. $\mathbb{K}$-CCRA $\subseteq$ $\mathbb{K}$-LinWA.

Proof. With the notations of Proposition 2, let us see $\mathcal{A}, \eta$, and $\mathcal{W}$ as a single $\mathbb{K}$-WA, where the weights in the $\mathcal{A}$ part are set to 0. For any word w, each run on w consists of a run over a prefix u within $\mathcal{A}$, and a run over the leftover suffix v within $\mathcal{W}$ starting in some state $q \in \eta(q_0.u)$. Thus there are at most $|w| \times |Q'|$ runs, hence the WA is linearly ambiguous. $\qquad\square$

As an application of this specific form, it is not hard to show that some specific functions are not expressible using a $\mathbb{Z}_\infty$-CCRA. Let minblock (resp. lastblock) be the function from $\{a, \#\}^*$ to $\mathbb{N}$ which, given $w = \#a^{n_1}\#a^{n_2}\#\cdots\#a^{n_k}\#$ returns $\min\{n_i\}_{i \in [k]}$ (resp. n_k):

Proposition 3. *The following functions are not expressible by a $\mathbb{Z}_\infty$-CCRA:*

- $c^i \cdot w \mapsto i + \text{minblock}(w)$, *with* $w \in \{a, \#\}^*$;
- $u \cdot \$ \cdot v \mapsto \text{lastblock}(u) + \text{lastblock}(v)$, *with* $u, v \in \{a, \#\}^*$.

Proof (sketch). In both cases, one has to reason about when the nondeterministic jump, given by η in Proposition 2, is made in the minimal run, bearing in mind that neither minblock nor lastblock are computable by a DetWA.

For the first example, the jump has to be made at the beginning of the minimal block of a's, after reading a #; thus the number of c's cannot be taken into account. For the second example, if the jump is made just before the last block of a's in v, then the value of the last block in u is disregarded. If it is made just before the last block in u, then the DetWA part has to compute lastblock on v, which is not possible. □

Note that the first function of Proposition 3 is expressible by a LinWA, and the second by an unambiguous WA (i.e., at most one run per accepted word). Moreover, since minblock is not expressible by an unambiguous WA, the classes of functions expressed by CCRA and unambiguous WA are incomparable.

4 Simulation of $\mathbb{Z}$-VASS$^{\mathbf{z}}$ using CCRA

Let $\mathcal{V}$ be a $\mathbb{Z}$-VASS$^{\mathbf{z}}$ and q a state of $\mathcal{V}$. Recall that C_k is the update alphabet of symbols $\mathbf{inc}_i, \mathbf{dec}_i$, and $\mathbf{chk}_i$, and that $L_{\mathcal{V},q} \subseteq (C_k)^*$ is the reachability language of q. In this section, we devise a simulation of $\mathcal{V}$ using CCRA in the following sense: Given a word $w \in (C_k)^*$, the $\mathbb{Z}_\infty$-CCRA (resp. $\mathbb{N}_\infty$-CCRA) will output 0 (resp. an even value) iff $w \in L_{\mathcal{V},q}$.

We will first tackle the simulation using $\mathbb{Z}_\infty$-CCRA before presenting a construction for $\mathbb{N}_\infty$-CCRA. The $\mathbb{Z}$ case is quite simpler, and reminiscent of the methodology of [1], but it provides some intuition for the construction for $\mathbb{N}$.

In both cases, we present how the counter increments ($\mathbf{inc}$), decrements ($\mathbf{dec}$), and zero-tests ($\mathbf{chk}$) are implemented for a single counter before showing how multiple counters can be handled. The automaton structure of the source $\mathbb{Z}$-VASS$^{\mathbf{z}}$, with accepting state q, can then be followed by the CRA while simulating the counters.

4.1 Simulation by $\mathbb{Z}_\infty$-CCRA

Simulation of a single counter Since we are working with a single counter, we drop the indices of the letters in C_1. A single counter c will be simulated by 3 registers: r^+ and r^-, carrying the values of c and $-c$, respectively, and $r^{\mathbf{z}}$ which shall be 0 if each time the letter $\mathbf{chk}$ was read, c was 0. If at any time $\mathbf{chk}$ was read while c was nonzero, then $r^{\mathbf{z}}$ will be strictly smaller than 0. This is implemented as follows:

$$\mathbf{inc}: \begin{cases} r^+ \leftarrow r^+ + 1 \\ r^- \leftarrow r^- - 1 \\ r^{\mathbf{z}} \leftarrow r^{\mathbf{z}} \end{cases} \quad \mathbf{dec}: \begin{cases} r^+ \leftarrow r^+ - 1 \\ r^- \leftarrow r^- + 1 \\ r^{\mathbf{z}} \leftarrow r^{\mathbf{z}} \end{cases} \quad \mathbf{chk}: \begin{cases} r^+ \leftarrow 0 \\ r^- \leftarrow 0 \\ r^{\mathbf{z}} \leftarrow \min\{r^{\mathbf{z}}, r^+, r^-\} \end{cases}$$

Observation 1. *If $r^{\mathbf{z}}$ becomes strictly smaller than 0, it will stay so after reading any word in $(C_1)^*$.*

Observation 2. *Assume $r^+ = r^- = r^{\mathbf{z}} = 0$. After reading i letters $\mathbf{inc}$ and j letters $\mathbf{dec}$, in any order, then reading a final $\mathbf{chk}$, the new values of the registers verify:*

1. If $i = j$, then $r^+ = r^- = r^{\mathbf{z}} = 0$;
2. Otherwise $r^{\mathbf{z}} < 0$.

This simulates the original counter in the following sense:

Proposition 4. *Let $\mathcal{V}$ be a $\mathbb{Z}$-VASS$^{\mathbf{z}}$ of dimension 1 and q a state of $\mathcal{V}$. There is a $\mathbb{Z}_\infty$-CCRA $\mathcal{C}$ such that:*

$$(\forall w \in (C_1)^*) \quad w \in L_{\mathcal{V},q} \Leftrightarrow \mathcal{C}(w) = 0 \ .$$

Proof. Let $\mathcal{V} = (Q, C_1, \delta, q_0, F)$ and $q \in Q$. The $\mathbb{Z}_\infty$-CCRA $\mathcal{C}$ with 3 registers is defined as having $(Q, C_1, \delta, q_0, \{q\})$ as the automaton structure, and the updates are dictated by the letter being read, as defined above. On state q, $\mathcal{C}$ outputs $r^{\mathbf{z}}$. □

Simulation of multiple counters It is quite straightforward to combine multiple $r^{\mathbf{z}}$ registers into one. Indeed, if k counters are simulated using registers r_i^+, r_i^-, and $r_i^{\mathbf{z}}$, $i \in [k]$, then at the end of the simulation, one can set:

$$flag \leftarrow \min\{r_1^{\mathbf{z}}, r_2^{\mathbf{z}}, \ldots, r_n^{\mathbf{z}}\} \ ,$$

so that *flag* would be 0 iff the execution saw no illegal zero-tests; *flag* is negative otherwise.

Proposition 5. *Let $\mathcal{V}$ be a $\mathbb{Z}$-VASS$^{\mathbf{z}}$ of dimension k and q a state of $\mathcal{V}$. There is a $\mathbb{Z}_\infty$-CCRA $\mathcal{C}$ such that:*

$$(\forall w \in (C_k)^*) \qquad w \in L_{\mathcal{V},q} \Leftrightarrow \mathcal{C}(w) = 0 \ .$$

Remark 1. Here, we were mostly interested in having a specific output if the simulated execution was correct. If we wanted, by contrast, to output one of the counters on correct executions, we would need one more idea, that we present here since it is similar to the techniques of the next section.

Suppose that we wish to output the register r iff *flag* is 0; recall that *flag* may only be 0 or negative. We will do so by repeatedly reading a new letter, and having r be the only possible *even* output value, provided *flag* is 0—no even value is produced if *flag* is negative.

We may assume that, by construction, *flag* is even and r is a multiple of 4; we further assume that we have a register $r^{\frac{1}{2}}$ that contains *half* of r's value. We add the letter z to our alphabet, to be read at the end of the simulation; reading z increases $r^{\frac{1}{2}}$ by 2 and *flag* by 4. The output value is then set as:

$$\min\{r + 1, flag + 1, r^{\frac{1}{2}}\} \ .$$

Hence the only time an even output is produced is when $r^{\frac{1}{2}}$ is minimal. This only happens when it is equal to r and *flag* started at 0; for instance, in the following graphics, $r = 4, r^{\frac{1}{2}} = 2$, and the left-hand side depicts the case *flag* = 0, while the right-hand side depicts *flag* = −2. It can be seen that reading zero or two z's in the left-hand picture would make the output be some odd value, while reading one correctly returns r. In the right-hand picture, reading zero, one, or two z's would make the output be some odd value, as claimed.
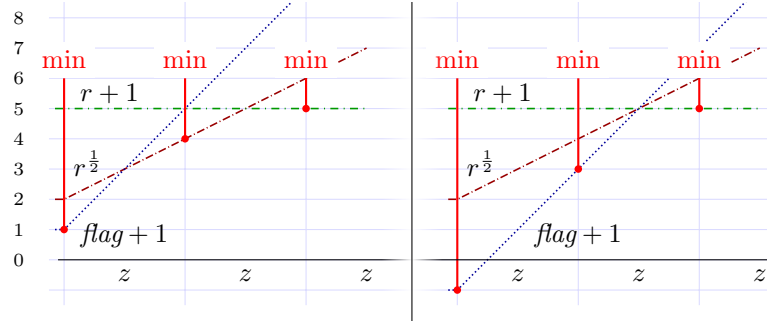
4.2 Simulation by $\mathbb{N}_\infty$-CCRA

Translating the strategy above to the $\mathbb{N}$ setting turns out to be a nontrivial matter. Indeed, one might expect that it would be enough to increase the updates so that no negative number appears therein. This would contribute a linear blowup to the values, but does not seem to change the overall behavior. However, the resets made while reading $\mathbf{chk}_i$ would have to be equal to that blowup, and this would require copying.

The simulation will thus follow two phases. First, one that corresponds to the strategy for $\mathbb{Z}$ with the updates tweaked to be positive; second, after reading a $\mathbf{chk}_i$, a *climb-back* phase that puts the registers back in a manageable state (called "ready" later on). For this latter phase, the $\mathbb{N}_\infty$-CCRA will read a word in $\mathbf{cb}_i^* \cdot \mathbf{chkcb}_i$—the letter $\mathbf{cb}$ standing for *climb-back*. Further, combining the acceptance conditions of multiple counters will also require some new letters; the alphabet of the automaton is thus:

$$C_k' = C_k \cup \bigcup_{i \in [k]} \{\mathbf{cb}_i, \mathbf{chkcb}_i, z_i\} \ .$$

Simulation of a single counter Again, since we are working with a single counter, we drop the indices of the letters in C_1'. A single counter in the $\mathbb{Z}$-VASS$^{\mathbf{z}}$ will be simulated by 7 different registers, each with a simple intended meaning:

- r^+ and r^- should respectively count the number of increments and decrements of the counter;
- r^{u} increases each time the counter is either incremented or decremented; it counts the number of *updates* to the counter. The register $r^{\frac{\mathrm{u}}{2}}$ should be half of r^{u};
- $r^{\mathbf{z}}$ will be a witness that the $\mathbf{chk}$ letter has always been read when the simulated counter was zero, and that the climb-back phases were done correctly;
- Finally, we will need two internal registers r^{cb} and r^{2cb}, used solely in the climb-back phase.

To simplify the discussion, we give names to some states of the registers:

- They are *ready* if $r^+ = r^- = r^{\mathbf{z}} = r^{\frac{\mathrm{u}}{2}} = \frac{1}{2} \times r^{\mathrm{u}}$;

– They are *to-climb* if $r^+ = r^- = 0$ and $r^{\mathbf{z}} = r^{\frac{\mathbf{u}}{2}} = \frac{1}{2} \times r^{\mathbf{u}}$;
– They are *dead* if $r^{\mathbf{z}} < r^{\frac{\mathbf{u}}{2}}$.

In the first two states, we also assume that the internal registers r^{cb} and $r^{2\mathrm{cb}}$ are zero.

Goal of the construction. We will show that if the registers are ready and we read an equal number of **inc**'s and **dec**'s followed by a **chk**, then the registers become to-climb. There is then a precise number i such that reading $\mathbf{cb}^i \cdot \mathbf{chkcb}$ will put the registers back in ready mode. Crucially, if the numbers of **inc**'s and **dec**'s are not equal, or an incorrect number of **cb**'s is read, then the registers become dead.

The updates are as follows, where the registers not shown are simply preserved. As we saw in Remark 1, we will require that the values of the registers be divisible by some values, hence rather than incrementing with 1, we increment by a value $e \in \mathbb{N}$ (for *Einheit*, unit) to be determined later. Note that these are indeed copyless updates.

$$\mathbf{inc} : \begin{cases} r^+ \leftarrow r^+ + e \\ r^{\mathbf{u}} \leftarrow r^{\mathbf{u}} + e \\ r^{\frac{\mathbf{u}}{2}} \leftarrow r^{\frac{\mathbf{u}}{2}} + \frac{e}{2} \\ r^{\mathbf{z}} \leftarrow r^{\mathbf{z}} + \frac{e}{2} \end{cases} \quad \mathbf{dec} : \begin{cases} r^- \leftarrow r^- + e \\ r^{\mathbf{u}} \leftarrow r^{\mathbf{u}} + e \\ r^{\frac{\mathbf{u}}{2}} \leftarrow r^{\frac{\mathbf{u}}{2}} + \frac{e}{2} \\ r^{\mathbf{z}} \leftarrow r^{\mathbf{z}} + \frac{e}{2} \end{cases} \quad \mathbf{chk} : \begin{cases} r^+ \leftarrow 0 \\ r^- \leftarrow 0 \\ r^{\mathbf{z}} \leftarrow \min\{r^{\mathbf{z}}, r^+, r^-\} \end{cases}$$

$$\mathbf{cb} : \begin{cases} r^+ \quad \leftarrow r^+ + e \\ r^- \quad \leftarrow r^- + e \\ r^{\frac{\mathbf{u}}{2}} \quad \leftarrow r^{\frac{\mathbf{u}}{2}} + \frac{e}{2} \\ r^{\mathbf{z}} \quad \leftarrow r^{\mathbf{z}} + \frac{e}{2} \\ r^{\mathrm{cb}} \quad \leftarrow r^{\mathrm{cb}} + e \\ r^{2\mathrm{cb}} \leftarrow r^{\mathrm{cb}} + 2 \times e \end{cases} \quad \mathbf{chkcb} : \begin{cases} r^{\mathrm{cb}} \quad \leftarrow 0 \\ r^{2\mathrm{cb}} \leftarrow 0 \\ r^{\mathbf{u}} \quad \leftarrow r^{2\mathrm{cb}} \\ r^{\mathbf{z}} \quad \leftarrow \min\{r^{\mathbf{z}}, r^{\mathrm{cb}}, r^{\mathbf{u}}\} \end{cases}$$

Observation 3. *If the registers are dead, they will stay so after reading any word in $(C_1')^*$.*

Lemma 1. *Assume the registers are ready. After reading i letters **inc** and j letters **dec**, in any order, then reading a final **chk**, the new values of the registers verify:*

1. *If $i = j$, then they are to-climb;*
2. *Otherwise, they are dead.*

Proof. Suppose $r^+ = r^- = r^{\mathbf{z}} = r^{\frac{\mathbf{u}}{2}} = \frac{1}{2} \times r^{\mathbf{u}}$, and let us name that value s. After reading i letters **inc** and j letters **dec**, the new values are:

$$r^+ = s + e \times i, \quad r^- = s + e \times j, \quad r^{\mathbf{z}} = r^{\frac{\mathbf{u}}{2}} = \frac{1}{2} \times r^{\mathbf{u}} = s + e \times \frac{i+j}{2} \ .$$

Now, if $i = j$ then $r^+ = r^- = r^{\mathbf{z}} = r^{\frac{\mathbf{u}}{2}} = \frac{1}{2} \times r^{\mathbf{u}}$, thus reading **chk** will indeed make the registers to-climb. Otherwise, one of r^+ or r^- is smaller than $r^{\mathbf{z}}$, and reading **chk** will make the registers dead. □

Lemma 2. *Assume the registers are to-climb. After reading* $\mathbf{cb}^i \cdot \mathbf{chkcb}$*, the new values of the registers verify:*

1. *If i is equal to the starting value of $r^{\mathbf{z}}$ multiplied by $\frac{2}{e}$, then they are ready;*
2. *Otherwise, they are dead.*

Proof. Suppose $r^+ = r^- = 0$ and $r^{\mathbf{z}} = r^{\frac{\mathrm{u}}{2}} = \frac{1}{2} \times r^{\mathrm{u}}$; we name that latter value s. After reading i letters $\mathbf{cb}$, the new values are:

$$r^+ = r^- = r^{\mathrm{cb}} = \frac{1}{2} \times r^{2\mathrm{cb}} = e \times i, \quad r^{\mathbf{z}} = r^{\frac{\mathrm{u}}{2}} = s + e \times \frac{i}{2}, \quad r^{\mathrm{u}} = 2 \times s \ .$$

Now if $i = \frac{2 \times s}{e}$, then $r^+ = r^- = r^{\mathrm{cb}} = \frac{1}{2} \times r^{2\mathrm{cb}} = r^{\mathbf{z}} = r^{\frac{\mathrm{u}}{2}} = 2 \times s$. Reading **chkcb** thus makes the registers ready. If i is smaller than $\frac{2 \times s}{e}$ then $r^{\mathrm{cb}} < r^{\mathbf{z}}$; if it is greater, then $r^{\mathrm{u}} < r^{\mathbf{z}}$: in both cases, reading **chkcb** makes the registers dead. $\qquad\square$

Simulation of multiple counters We just saw how to simulate a single counter in the sense that the registers are not dead iff the input word describes a correct run (i.e., one in which **chk** is only read if the counter is 0). Let us now exhibit a method that combines multiple such simulations, and outputs an even value iff none of the simulations is dead. To do so, we will repeatedly read new letters $z_1, z_2, \ldots, z_k$ at the very end of the execution, in a similar fashion as Remark 1.

Let us suppose we have k simulated counters, hence k sets of 7 registers. We will only use $r_i^{\mathbf{z}}$, for each i, but we will have *one* more register in our $\mathbb{N}_\infty$-CCRA, named r^{avg}. The purpose of r^{avg} is to hold the average of all the $r_i^{\frac{\mathrm{u}}{2}}$; this is easily achieved by adding to the above updates:

$$r^{\mathrm{avg}} \leftarrow r^{\mathrm{avg}} + \frac{e}{2 \times k}$$

whenever a $r_i^{\frac{\mathrm{u}}{2}}$ is incremented (always by $\frac{e}{2}$). Now for each i, the new letter z_i will update the registers with:

$$\begin{cases} r_i^{\mathbf{z}} & \leftarrow r_i^{\mathbf{z}} + \frac{e}{2} \\ r^{\mathrm{avg}} \leftarrow r^{\mathrm{avg}} + \frac{e}{2 \times k} \end{cases}$$

The output value of the $\mathbb{N}_\infty$-CCRA is then set to

$$\min\{r^{\mathrm{avg}}, r_1^{\mathbf{z}} + 1, r_2^{\mathbf{z}} + 1, \ldots, r_k^{\mathbf{z}} + 1\} \ . \tag{1}$$

We further assume that e was chosen so that all the registers are even.

If r^{avg} was the average of the $r_i^{\mathbf{z}}$'s before reading the z_i's—and this only happens if none of the register set was dead—it will stay so reading z_i's. Consequently, there is a number of each letter z_i that can be read so that all the $r_i^{\mathbf{z}}$'s are equal, making r^{avg} the output value of the $\mathbb{N}_\infty$-CCRA.

If r^{avg} was greater than the average of the $r_i^{\mathbf{z}}$'s—implying that at least one set of registers was dead—then r^{avg} will never be the output of the CCRA after reading z_i's.

Theorem 1 (Simulation). *Let $\mathcal{V}$ be a $\mathbb{Z}$-VASS$^{\mathbf{z}}$ of dimension k and q a state of $\mathcal{V}$. Write $h\colon (C'_k)^* \to (C_k)^*$ for the function that erases the letters $\mathbf{cb}_i, \mathbf{chkcb}_i$, and z_i. There is a $\mathbb{N}_\infty$-CCRA $\mathcal{C}$ such that for all $w \in (C_k)^*$:*

$$w \in L_{\mathcal{V},q} \Leftrightarrow (\exists! w' \in h^{-1}(w))[\mathcal{C}(w') \text{ is even}] \ .$$

Proof. The only detail left to deal with is the uniqueness of the w'. We can certainly make sure that $\mathcal{C}$ outputs a value iff the input is of the form:

$$(\mathbf{inc}_i + \mathbf{dec}_i + \mathbf{chk}_i \cdot \mathbf{cb}_i^* \cdot \mathbf{chkcb}_i)_i^* \cdot (z_i)_i^* \ ,$$

but even if the first half (without the z_i's) is indeed unique, as per Lemma 2, the z_i's need not be so. To preserve uniqueness, this latter part should be replaced by:

$$\bigcup_{i \in [k]} \prod_{j \in [k] \setminus \{i\}} (z_j)^* \ .$$

This serves two purposes: first, the order on the z_i's is fixed; second, one of the z_j will *not* be used, hence the condition that all the $r_i^{\mathbf{z}}$ be equal will only be verified when they are all valued $r_j^{\mathbf{z}}$. Naturally, only one such j exists, it is simply the index of the maximum $r_i^{\mathbf{z}}$. $\qquad\square$

5 Applications

We draw a number of undecidability results as a consequence of the construction of the previous section.

Theorem 2 (Equivalence). *The following problem is undecidable:*
> ***Given:*** *Two $\mathbb{N}_\infty$-CCRA $\mathcal{C}$ and $\mathcal{C}'$ over A^**
> ***Question:*** *$(\forall w \in A^*)[\mathcal{C}(w) = \mathcal{C}'(w)]$*

Proof. Let $\mathcal{V}$ be a $\mathbb{Z}$-VASS$^{\mathbf{z}}$ and q a state of $\mathcal{V}$, and consider the $\mathbb{N}_\infty$-CCRA $\mathcal{C}$ that simulates $L_{\mathcal{V},q}$. We reduce deciding if that language is empty (which is undecidable by Proposition 1) to the problem at hand. Equation (1), defining the output of $\mathcal{C}$, is such that r^{avg} is the minimum iff the execution was correct. Thus replacing this output function by:

$$\min\{r_1^{\mathbf{z}} + 1, r_2^{\mathbf{z}} + 1, \ldots, r_k^{\mathbf{z}} + 1\}$$

changes the output value of a word iff it was a correct run. Calling $\mathcal{C}'$ this modified version, it holds that $(\forall w \in A^*)[\mathcal{C}(w) = \mathcal{C}'(w)]$ iff $L_{\mathcal{V},q} = \emptyset$. $\qquad\square$

Clearly, it is undecidable whether the image of a $\mathbb{N}_\infty$-CCRA is always odd. Further:

Theorem 3 (Semilinearity). *The following problem is undecidable:*
> ***Given:*** *A $\mathbb{N}_\infty$-CCRA $\mathcal{C}$ over A^**
> ***Question:*** *Is $\mathcal{C}(A^*)$ semilinear, i.e., an eventually periodic set?*

Proof. We provide an independent construction which bears some similarities with the "climb-back" method. It doubles a register r in the following sense: if r is a register with starting value s, then reading $\mathbf{inc}^{s/2} \cdot \mathbf{chk}$ *doubles* the value of r; if any other number of $\mathbf{inc}$'s is read (which happens in particular when s is odd), the new value of r will be some odd number.

Consider a register r with initial value s, and suppose we have an additional register r' holding $2 \times s$. We introduce two new registers, r^{cb} and $r^{2\mathrm{cb}}$ initialized with 0. Upon reading a word $\mathbf{cb}^i \cdot \mathbf{chkcb}$, we apply the updates:

$$
\mathbf{cb}\colon \begin{cases} r & \leftarrow r + 2 \\ r^{\mathrm{cb}} & \leftarrow r^{\mathrm{cb}} + 4 \\ r^{2\mathrm{cb}} & \leftarrow r^{2\mathrm{cb}} + 8 \end{cases}
\qquad
\mathbf{chkcb}\colon \begin{cases} r^{\mathrm{cb}} & \leftarrow 0 \\ r^{2\mathrm{cb}} & \leftarrow 0 \\ r' & \leftarrow r^{2\mathrm{cb}} \\ r & \leftarrow \min\{r, r' + 1, r^{\mathrm{cb}} + 1\} \end{cases}
$$

After reading $\mathbf{cb}^i$, it holds that $r = s + 2 \times i$, $r^{\mathrm{cb}} = 4 \times i$, and $r^{2\mathrm{cb}} = 8 \times i$.

If $i = \frac{s}{2}$, then $r = r^{\mathrm{cb}} = r' = 2 \times s$, hence after reading $\mathbf{chkcb}$, we have indeed $r = \frac{r'}{2} = 2 \times s$, and the extra registers are reset: we are back to our starting hypothesis.

If $i \neq \frac{s}{2}$, then either $r' < r$ (when $i > \frac{s}{2}$) or $r^{\mathrm{cb}} < r$ (when $i < \frac{s}{2}$). In both cases, after reading $\mathbf{chkcb}$, r becomes odd, and will stay so after reading any other word.

As a side note, consider the $\mathbb{N}_\infty$-CCRA with the above updates and r initialized to 2, that reads words in $(\mathbf{cb}^* \cdot \mathbf{chkcb})^*$. Then the only even outputs of this machine are the powers of two, a nonsemilinear set.

This concludes the construction, and we now present the reduction.

Let $\mathcal{V}$ be a $\mathbb{Z}$-VASS$^{\mathbf{z}}$ and q a state of $\mathcal{V}$, and consider the $\mathbb{N}_\infty$-CCRA $\mathcal{C}$ that simulates $L_{\mathcal{V},q}$. We assume that $|L_{\mathcal{V},q}| \leq 1$, and again reduce deciding $L_{\mathcal{V},q} = \emptyset$ to the problem at hand.

First we note that we may assume that $\mathcal{C}$ outputs all the odd numbers, for instance by adding a letter ℓ and, upon reading ℓ^n, outputing $2 \times n + 1$. Also recall that if $L_{\mathcal{V},q}$ is nonempty, then there is a unique w such that $\mathcal{C}(w)$ is even.

We now modify $\mathcal{C}$ into $\mathcal{C}'$ to incorporate the above machinery. We simply store in a new register r the output value of $\mathcal{C}$, and proceed by reading words of the form $\mathbf{cb}^i \cdot \mathbf{chkcb}$ with the updates as above. If $L_{\mathcal{V},q} = \emptyset$, then $\mathcal{C}'((C'_k)^*)$ is all the odd numbers, a semilinear set. Otherwise, there is one (and only one) even value s in the image of $\mathcal{C}$, and it holds that:

$$
\mathcal{C}'((C'_k)^*) = (2\mathbb{N} + 1) \cup \{2^i \times s \mid i \geq 0\} \;,
$$

a nonsemilinear set. □

Theorem 4 (Upperboundedness). *The following problem is undecidable:*
   Given: *A $\mathbb{Z}_\infty$-WA $\mathcal{A}$ over A^**
   Question: $(\exists c \in \mathbb{Z})(\forall w \in A^*)[\mathcal{A}(w) \leq c]$

Proof. Let $\mathcal{V}$ be a $\mathbb{Z}$-VASS$^{\mathbf{z}}$ and q a state of $\mathcal{V}$, and consider the $\mathbb{Z}_\infty$-CCRA $\mathcal{C}$ that simulates $L_{\mathcal{V},q}$. Relying on Proposition 2, let $\mathcal{W}$ be a $\mathbb{Z}_\infty$-WA equivalent to

$\mathcal{C}$. Tweak $\mathcal{W}$ to output the same as $\mathcal{C}$ *plus one*, hence $\mathcal{W}(w)$ is 1 iff $w \in L_{\mathcal{V},q}$. Now let $\mathcal{W}'$ be $\mathcal{W}$ with an added letter $\#$ that jumps from the final states of $\mathcal{W}$ to the its initial states; formally, let $\mathcal{W} = (\mathcal{A}, \lambda, \mu, \nu)$ with $\mathcal{A} = (Q, A, \delta, q_0, F)$, then $\mathcal{W}'$ is $(\mathcal{A}', \lambda, \mu', \nu)$ where $\mathcal{A}' = (Q, A \uplus \{\#\}, \delta \cup \{(q, \#, q_0) \mid q \in F\}, q_0, F)$, and μ' agrees with μ on δ and is extended by $\mu(q, \#, q_0) = \nu(q) + \lambda$.

In essence, $\mathcal{W}'$ is iterating $\mathcal{W}$:

$$\mathcal{W}'(w_1 \# w_2 \# \cdots \# w_k) = \sum_{i \in [k]} \mathcal{W}(w_i) \ .$$

From this, we see that if $\mathcal{W}$ is always negative or zero, $\mathcal{W}'$ is bounded, otherwise, if $\mathcal{W}(w) = 1$, then $\mathcal{W}'((w\#)^c \cdot w) = c + 1$, hence $\mathcal{W}'$ is unbounded. $\qquad\square$

6 Conclusion

Open: Upperboundedness of $\mathbb{Z}_\infty$-CCRA; characterizing linearly ambiguous.

References

1. Shaull Almagor, Udi Boker, and Orna Kupferman. What's decidable about weighted automata? In *Automated Technology for Verification and Analysis, 9th International Symposium, ATVA 2011, Taipei, Taiwan, October 11-14, 2011. Proceedings*, pages 482–491, 2011. `doi:10.1007/978-3-642-24372-1_37`.
2. Rajeev Alur, Loris D'Antoni, Jyotirmoy V. 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, New Orleans, LA, USA, June 25-28, 2013*, pages 13–22, 2013. `doi:10.1109/LICS.2013.65`.
3. Michaël Cadilhac, Alain Finkel, and Pierre McKenzie. Unambiguous constrained automata. *Int. J. Found. Comput. Sci.*, 24(7):1099–1116, 2013. `doi:10.1142/S0129054113400339`.
4. Stéphane Gaubert and Ricardo Katz. Rational semimodules over the max-plus semiring and geometric approach to discrete event systems. *Kybernetika*, 40(2):153–180, 2004.
5. Filip Mazowiecki and Cristian Riveros. Copyless cost-register automata: Structure, expressiveness, and closure properties. In *33rd Symposium on Theoretical Aspects of Computer Science, STACS 2016, February 17-20, 2016, Orléans, France*, pages 53:1–53:13, 2016. `doi:10.4230/LIPIcs.STACS.2016.53`.
6. Marvin L. Minsky. Recursive unsolvability of Post's problem of "tag" and other topics in theory of Turing machines. *Annals of Mathematics*, 74(3):pp. 437–455, 1961.
7. Mojzesz Presburger. Über de vollständigkeit eines gewissen systems der arithmetik ganzer zahlen, in welchen, die addition als einzige operation hervortritt. In *Comptes Rendus du Premier Congrès des Mathématiciens des Pays Slaves*, pages 92–101, Warsaw, 1927.