Theoretical Computer Science 850 (2021) 185–201

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Optimal run problem for weighted register automata

Hiroyuki Seki[a,*], Reo Yoshimura[a], Yoshiaki Takata[b]

[a] *Nagoya University, Japan*
[b] *Kochi University of Technology, Japan*

ARTICLE INFO

Article history:
Received 29 January 2020
Received in revised form 31 October 2020
Accepted 3 November 2020
Available online 6 November 2020

Keywords:
Weighted register automaton
Optimal run
Register type
Complexity

ABSTRACT

Register automata (RA) are a computational model that can handle data values by adding registers to finite automata. Recently, weighted register automata (WRA) were proposed by extending RA so that weights can be specified for transitions. In this paper, we first investigate decidability and complexity of decision problems on the weights of runs in WRA. We then propose an algorithm for the optimal run problem related to the above decision problems. For this purpose, we use a register type as an abstraction of the contents of registers, which is determined by binary relations (such as $=$, $<$, etc.) handled by WRA. Also, we introduce a subclass where both the applicability of transition rules and the weights of transitions are determined only by a register type. We present a method of transforming a given WRA satisfying the assumption to a weighted directed graph such that the optimal run of WRA and the minimum weight path of the graph correspond to each other. Lastly, we discuss the optimal run problem for weighted timed automata as an example.

1. Introduction

There have been many extensions of finite automata that can manipulate data values. Among them, register automata (abbreviated as RA) introduced in [16] have the advantages that important decision problems including membership and emptiness are decidable and the class of languages accepted by RA is closed under standard language operations except complementation. In a k-RA, k registers are associated with each state. An input is a finite sequence of pairs of a symbol from a finite alphabet and a data value from an infinite set. Each transition can compare the contents of the registers and the current input data value and if this test succeeds, the input data value is loaded to the registers specified by the transition and the state is changed. The complexity of decision problems for RA has been analyzed [21,13]. Also, [17] points out that RA is a good formal model for querying structured data such as XML documents. Recently, weighted RA was proposed in [5] by incorporating weights into RA so that various quantities such as time, information flow and costs needed for transitions and/or data manipulations can be formally represented as weights. A k-WRA is a k-RA equipped with weight functions for transitions and data manipulations. The weight function for data manipulations can represent weights depending on data values such as the cost depending on the elapsed time in timed automata. A semiring is assumed to represent weights and to assign a weight to a switch (one step move), a run (accepting sequence of switches), and a data word (accepted sequence of pairs of a symbol and a data value) in a systematic way. Closure properties of the data series, which is a mapping from data words to weights, defined by WRA are discussed and an MSO logical counterpart of WRA is proposed

[*] Corresponding author.
E-mail addresses: seki@i.nagoya-u.ac.jp (H. Seki), yoshimura.reo@sqlab.jp (R. Yoshimura), takata.yoshiaki@kochi-tech.ac.jp (Y. Takata).

https://doi.org/10.1016/j.tcs.2020.11.003

and studied in depth in [5]. However, decidability and complexity of basic problems for WRA were not discussed. Timed automata (abbreviated as TA) are well-known extensions of finite automata that can deal with time by clock variables [3]. TA was extended to weighted TA (WTA) and the optimal-reachability problems have been investigated [4,19]. In [5], TA and WTA are shown to be regarded as subclasses of RA and WRA, respectively.

In this paper, we discuss optimal run problems and related decision problems for WRA, motivated by [3]. First, we clarify the decidability and complexity of the decision problems on weight computation and weight realizability. More concretely, we show that the problem to decide whether there is a run of a given data word whose weight takes a given value in a given WRA is NP-complete, and the problem to compute the weight of a given data word, which is the sum of all runs of the data word, in a given WRA is in PSPACE and #P-hard. We also show that the following two weight realizability problems are both undecidable: the problem to decide whether there is a run in a given WRA whose weight takes a given value and the problem to decide whether there is a data word whose weight in a given WRA equals to a given value. Note that the former two problems and the latter two problems can be regarded as extensions of the membership and emptiness problems for RA, which are known to be NP-complete and PSPACE-complete, respectively.

Next, we utilize register type, which was introduced in [24] as an abstract representation of the contents of registers, by identifying the data values indistinguishable by comparisons allowed in the guards of transitions. We show an equivalence transformation from a given k-WRA to a k-WRA such that the exact register type is annotated to each state by associating register types with states before and after a transition. A WRA obtained by this transition decomposition by register type is called a normal form WRA.

Then, we move to the main topic, the optimal run problem for WRA, which is the problem to compute a run whose weight takes the infimum among all the runs in a given WRA. The idea is simple and similar to the one in [4]: A given WRA is translated into a directed graph where a node stands for a configuration (a pair of a state and the contents of registers) and an edge between two nodes stands for switches between them where the weight of the edge is the infimum of the weights of those switches. In order to determine the weight of each edge, the infimum of the weights must be independent of the contents of registers. However, this does not hold in general, unlike for WTA. To overcome this issue, we introduce two reasonable assumptions: for each transition, the infimum of the weights of switches realized by the transition is uniquely determined independent of the contents of registers (weighted simulation); and the above infimum can be computed when weighted simulation holds (weight computability). These two assumptions are a weighted version of simulation and progress properties proposed in [24]. For a given WRA satisfying the above two properties, we can construct a directed graph as intended, and we can obtain an optimal run by an existing graph algorithm that computes the minimum-weight path in the constructed graph. We also show that if weighted simulation and weight computability hold, the run weight bounding problem, which is a decision-problem counterpart of the optimal run problem, becomes PSPACE-complete.

After that, we introduce restricted WRA, abbreviated as RWRA, which is a subclass of WRA such that the weight of every data manipulation is determined only by (a transition and) the result of comparisons allowed in the guards between the contents of registers and an input data value. When we apply the transition decomposition by register type to a given RWRA, the weight of switch is uniquely determined only by (a state and) the register types. We introduce two necessary conditions of weighted simulation property and weight computability, which are obtained by ignoring the requirements for weights, respectively. These conditions are similar to simulation and progress properties in [24]. We show that if the given normal form RWRA satisfies these two necessary conditions, the RWRA can be regarded as a WRA with weighted simulation and weight computability.

Finally, we discuss the optimal run problem for WTA as an example of the application of the proposed method. We focus on the subclass of WRA obtained from WTA by the translation of [5]. Intuitively, a register type corresponds to a clock region of TA [3]. Moreover, [4] shows that there always exists an optimal (minimum weight) path that visits only boundary regions and limit regions because all clock constraints of TA are linear. If we restrict the register types to those corresponding to boundary regions and limit regions, a WTA is regarded as an RWRA having simulation and progress properties. By the fact mentioned in the previous paragraph, the WTA is regarded as a WRA having weighted simulation and weight computability properties and the proposed method of solving the optimal run problem can be applied to the WTA. In this case, the directed graph constructed by our method corresponds to the subregion graph in [4].

This paper is an extended version of [22] by adding detailed proofs of some complexity results and proposing a concrete subclass of WRA, called RWRA, for which the optimal run problem can be solved under some reasonable assumptions.

Related work Register automata (RA) were proposed by Kaminski and Francez [16] as finite-memory automata where they show that the membership and emptiness problems are decidable, and the class of languages recognized by RA are closed under union, concatenation and Kleene-star. Later, the computational complexity of the above two problems are analyzed in [21,13]. In [12], register context-free grammars (RCFG) as well as pushdown automata over an infinite alphabet were introduced as extensions of RA and the equivalence of the two models were shown. Properties of RCFG such as closure and complexity of decision problems are investigated in depth in [12,23,24].

As extensions of finite automata other than RA, data automata [10], pebble automata (PA) [20] and nominal automata (NA) [9] are known. Libkin and Vrgoč [18] argue that RA is the only model that has efficient data complexity for membership among the above mentioned formalisms. Neven, et al. consider variations of RA and PA, which are either one way or two ways, deterministic, nondeterministic or alternating. They show inclusion and separation relationships among these automata, FO($\sim$, $<$) and EMSO($\sim$, $<$), and give the answer to some open problems including the undecidability of the universality problem for RA [20].

Nominal automata (NA) are defined by a data set with symmetry and finite supports, and properties of NA are investigated including Myhill-Nerode theorem, closure and determinization in [9]. (Usual) RA with equality and RA with total order can be regarded as NA where the data sets have equality symmetry and total order symmetry, respectively. Linear temporal logic with freeze quantifier [14,13] and two variable logic with data equality FO$^2(\sim, <, +1)$ [8] are well-known logical counterparts of automata with data values, whose expressive powers are incomparable.

Time-optimal reachability and the related and generalized problems for weighed timed automata (WTA) have been investigated. The single-source optimal reachability problem for WTA is solved by a branch-and-bound algorithm in [7]. Alur, et al. [4] solved the optimal reachability problem for WTA, which is more general than the single-source one, by introducing limited regions and transforming a WTA to a weighted graph. The decision version of the optimal reachability problem is shown to be PSPACE-complete in [19].

The existing study most related to this paper is Babari, et al.'s [5,6], where RA is extended to weighted RA (WRA), and properties including closure and MSO logical characterizations are studied in depth as mentioned in the beginning of this section. Note that WRA is different from cost register automata [2] where data values and weights are not separated and the basic problems are undecidable even for very restricted subclasses such as copyless cost register automata (CRA) [1]. This paper partially answers to open problems and conjectures raised in [5] about the decidability of the optimal run problem for WRA under reasonable assumptions as well as the complexity of decision problems for WRA which are counterparts of the membership and emptiness problems for models without weights.

2. Definitions

Let $\mathbb{B} = \{0, 1\}$ be the set of truth values, $\mathbb{N} = \{0, 1, \ldots\}$ be the set of natural numbers and $\mathbb{R}_{\geq 0}$ be the set of nonnegative reals. For a natural number $k \in \mathbb{N}$, let $[k] = \{1, \ldots, k\}$. By $|\beta|$, we mean the cardinality of β if β is a set and the length of β if β is a finite sequence. Let Σ be a finite alphabet and D be an infinite set of data values. We call $w \in (\Sigma \times D)^+$ a *data word* (over Σ and D). For a finite collection $\mathcal{R}$ of binary relations over D, $\mathbb{D} = \langle D, \mathcal{R} \rangle$ is called a *data structure*.

Intuitively, an automaton is equipped with a certain number of registers that can store a data value. Formally, an assignment of data values to k registers (abbreviated as k-register assignment or just assignment if k is irrelevant) is a mapping $\theta : [k] \to D$. The collection of k-register assignments is denoted as Θ_k. For a k-register assignment θ, $\theta(i)$ $(i \in [k])$ is the data value assigned to the i-th register by θ. Let F_k denote the set of guard formulas (or simply, guards) defined by $\varphi := tt \mid x_i^R \mid x_i^{R^{-1}} \mid \mathrm{in}^R \mid \varphi \wedge \varphi \mid \neg\varphi$ $(i \in [k],\ R \in \mathcal{R})$. For an assignment θ, a data value $d \in D$ and a guard φ, the satisfaction relation $(\theta, d) \models \varphi$ is defined inductively on the structure of φ as $(\theta, d) \models x_i^R$ iff $(\theta(i), d) \in R$, $(\theta, d) \models x_i^{R^{-1}}$ iff $(d, \theta(i)) \in R$, $(\theta, d) \models \mathrm{in}^R$ iff $(d, d) \in R$ and the meaning of tt, $\wedge$ and $\neg$ are defined in the usual way. Define $ff \equiv \neg tt$ and $\varphi_1 \vee \varphi_2 \equiv \neg(\neg\varphi_1 \wedge \neg\varphi_2)$.

Definition 1 *([16,17]).* A k-register automaton (k-RA) over a finite alphabet Σ and a data structure $\mathbb{D}$ is a tuple $A = (Q, Q_0, T, Q_f)$ where

- Q is a finite set of states,
- $Q_0, Q_f \subseteq Q$ are sets of initial and final states, respectively,
- $T \subseteq Q \times \Sigma \times F_k \times 2^{[k]} \times Q$ is a set of state transitions. □

Let $A = (Q, Q_0, T, Q_f)$ be a k-RA over Σ and $\langle D, \mathcal{R} \rangle$. A state transition (or transition) $t = (q, a, \varphi, \Lambda, q') \in T$ where $q, q' \in Q, a \in \Sigma, \varphi \in F_k, \Lambda \in 2^{[k]}$ is written as $q \to^a_{\varphi,\Lambda} q'$ and we denote by label(t) the second component a of t. The description length of a k-RA $A = (Q, Q_0, T, Q_f)$ is defined as $\|A\| = |Q| + |T| \max\{\|t\| \mid t \in T\}$, where $\|t\| = \log|Q| + k + \|\varphi\|$ for $t = q \to^a_{\varphi,\Lambda} q' \in T$ and $\|\varphi\|$ is the description length of φ, defined in a usual way. In this definition, we assume that we need $O(|Q|)$ bits to specify Q_0 and Q_f and $O(\log|Q| + k + \|\varphi\|)$ bits to specify a single transition $q \to^a_{\varphi,\Lambda} q'$ (i.e. $O(\log|Q|)$ bits for q and q', and k bits for Λ and $\|\varphi\|$ bits for φ). We consider $|\Sigma|$ is constant and do not take into account the description length of $a \in \Sigma$.

For an assignment $\theta \in \Theta_k$, $\Lambda \in 2^{[k]}$ and a data value $d \in D$, the updated assignment $\theta[\Lambda \leftarrow d] \in \Theta_k$ is $\theta[\Lambda \leftarrow d](i) = d$ if $i \in \Lambda$ and $\theta[\Lambda \leftarrow d](i) = \theta(i)$ otherwise. For a state $q \in Q$ and an assignment $\theta \in \Theta_k$, (q, θ) is called an *instantaneous description (ID)*. For two IDs $c = (q, \theta)$ and $c' = (q', \theta')$, if there are $d \in D$, $t = q \to^a_{\varphi,\Lambda} q' \in T$ such that $(\theta, d) \models \varphi$ and $\theta' = \theta[\Lambda \leftarrow d]$, then $c \vdash_{t,d} c'$ is called a *switch* from c to c' by t and d in A. The initial value of any register is $\perp$ $(\perp \in D)$. An initial ID and an accepting ID are $c_0 \in Q_0 \times \perp^k$ and $c_f \in Q_f \times \Theta_k$, respectively. A *run* in A is a finite sequence of switches from an initial ID to an accepting ID $\rho = c_0 \vdash_{t_1,d_1} c_1 \vdash_{t_2,d_2} c_2 \cdots \vdash_{t_n,d_n} c_n$. The label of a run ρ is label$(\rho) =$ (label$(t_1), d_1) \ldots$ (label$(t_n), d_n)$ and ρ is called a run of label(ρ) in A. For $w \in (\Sigma \times D)^+$, $\mathrm{Run}_A(w)$ is the set of all runs of w in A.

We define $L(A) = \{w \mid \mathrm{Run}_A(w) \neq \emptyset\}$, called the *data language* recognized by A. A data language $L \subseteq (\Sigma \times D)^+$ is recognizable if there is an RA A such that $L = L(A)$.

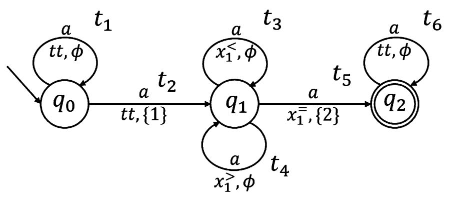


Fig. 1. RA A_1.

Example 1. Let $\Sigma = \{a\}$, $\mathcal{R} = \{<, =, >\}$. An example of 2-RA A_1 is shown in Fig. 1. For an input data word w, A_1 loads any data value, say d_i, in w to the first register nondeterministically by t_2. After that, every time a data value not equal to d_i comes, A_1 stays at q_1 by t_3 or t_4 until the same value d_i comes, at which A_1 moves to q_2 by t_5. In this way, A_1 nondeterministically chooses two positions having an identical data value d_i from the input data word, and the data values between them are not equal to d_i. We have $L(A_1) = \{(a, d_1) \ldots (a, d_n) \in (D \times \Sigma)^+ \mid \exists i, j \in [n],\ i < j,\ d_i = d_j\}$.

We will use notations Σ, $\mathbb{D} = \langle D, \mathcal{R}\rangle$ and $\mathcal{S} = (S, +, \cdot, 0, 1)$ to implicitly denote a finite alphabet, a data structure and a semiring, respectively.

Definition 2 *([5]).* A k-register weighted automaton (k-WRA) over $\Sigma, \mathbb{D}, \mathcal{S}$ is a tuple $\mathcal{A} = (Q, Q_0, T, Q_f, \mathrm{wt})$ where

- (Q, Q_0, T, Q_f) is a k-RA over $\Sigma, \mathbb{D}$, called the *base RA* of $\mathcal{A}$,
- $\mathrm{wt} = (\mathrm{wtt}, \mathrm{wtd})$ where $\mathrm{wtt} : T \to S$ and $\mathrm{wtd} : (T \times [k]) \to ((D \times D) \to S)$. $\square$

Let $\mathcal{A} = (Q, Q_0, T, Q_f, \mathrm{wt})$ be a k-WRA as above. $\mathrm{wtt}(t)$ represents the weight of a transition $t \in T$. $\mathrm{wtd}(t, j)$ is the weight of the j-th register at a transition $t \in T$. More precisely, $\mathrm{wtd}(t, j)(\theta(j), d)$ represents the weight needed for manipulating the j-th register for a switch $(q, \theta) \vdash_{t,d} c'$. The weight of a switch $c \vdash_{t,d} c'$ is defined as

$$\mathrm{wt}((q, \theta) \vdash_{t,d} c') = \prod_{j=1}^{k} \mathrm{wtd}(t, j)(\theta(j), d) \cdot \mathrm{wtt}(t).$$

A run in $\mathcal{A}$ is just a run in the base RA of $\mathcal{A}$. The weight of a run $\rho = c_0 \vdash_{t_1,d_1} c_1 \vdash_{t_2,d_2} c_2 \cdots \vdash_{t_n,d_n} c_n$ in $\mathcal{A}$ is defined as

$$\mathrm{wt}(\rho) = \prod_{i=1}^{n} \mathrm{wt}(c_{i-1} \vdash_{t_i,d_i} c_i).$$

As shown in the above definitions, the function wtd depends on the contents of registers and data values in a given data word. This is the main reason why some problems for WRA are difficult to solve and motivates us to introduce register type, weighted simulation property, weight computability and the subclass RWRA to make the setting more amenable without losing too much generality.

We assume that there are polynomials $p_1(n), p_2(n)$ such that for any $t \in T$, $\mathrm{wtt}(t)$ can be computed in $p_1(\|t\|)$ time and for $t \in T, j \in [k], d_1, d_2 \in D$, $\mathrm{wtd}(t, j)(d_1, d_2)$ can be computed in $p_2(\|t\| + \|d_1\| + \|d_2\|)$ time where $\|d\|$ is the description length of $d \in D$. We define the description length of a k-WRA $\mathcal{A}$ as $\|\mathcal{A}\| = \|A_b\|$ where A_b is the base RA of $\mathcal{A}$.

A data series over Σ, D and $\mathcal{S}$ is a mapping $U : (\Sigma \times D)^+ \to S$. The data series recognized by a WRA $\mathcal{A}$ is the data series $[\![\mathcal{A}]\!]$ defined as $[\![\mathcal{A}]\!](w) = \sum_{\rho \in \mathrm{Run}_{\mathcal{A}}(w)} \mathrm{wt}(\rho)$ for each $w \in (\Sigma \times D)^+$. A data series $U : (\Sigma \times D)^+ \to S$ is recognizable if there is a WRA that recognizes U.

Example 2. Let $\Sigma = \{a\}$, $\mathbb{D} = \langle \mathbb{N}, \{<, =, >\}\rangle$, and the semiring $\mathcal{S}_{\mathrm{trpc}} = (\mathbb{R}_{\geq 0} \cup \{\infty\}, \min, +, \infty, 0)$, known as a tropical semiring, where min acts as the addition and $+$ acts as the multiplication of the semiring. Let $\mathcal{A}_2$ be 2-WRA that has A_1 of Example 1 as its base RA. The weight functions $\mathrm{wt} = (\mathrm{wtt}, \mathrm{wtd})$ are defined as: $\mathrm{wtt}(t_3) = 1$ and $\mathrm{wtt}(t) = 0$ for every transition t other than t_3, and $\mathrm{wtd}(t, j)(d, d') = 0$ for every argument. $\mathcal{A}_2$ nondeterministically chooses two positions having an identical data value d_i and counts the data values greater than d_i between them by t_3. The data series recognized by $\mathcal{A}_2$ is such that for $w \in (\Sigma \times D)^+$, $[\![\mathcal{A}_2]\!](w) = \min\{$the number of d in $d_{i+1}, \ldots, d_{j-1}$ such that $d > d_i \mid w = (a, d_1) \ldots (a, d_n)$, $i, j \in [n]$, $i < j$, $d_i = d_j$ and for $k = i + 1, \ldots, j - 1$, $d_i \neq d_k\}$.

3. Decision problems

In this section, we analyze the computational complexity of the following problems for WRA. The results are summarized in Table 1.

Table 1
Complexity results.

Problem	Complexity
Run weight computation	NP-complete
Data word weight computation	PSPACE-solvable, #P-hard (#P-complete when a weight is a natural number, a transition weight function is bounded and every register manipulation weight is 1)
Run weight realizability	Undecidable
Data word weight realizability	Undecidable

Definition 3 *(The weight computation problems).* Input: a k-WRA $\mathcal{A}$ over Σ, $\mathbb{D}$, $\mathcal{S}$ and a data word $w \in (\Sigma \times D)^+$. For the run weight computation problem, a weight $s \in S$ is also given.
(The run weight computation problem) $\exists \rho \in \mathrm{Run}_{\mathcal{A}}(w).\, \mathrm{wt}(\rho) = s$?
(The data word weight computation problem) Compute $[\![\mathcal{A}]\!](w)$.
The input size of both problems is $\|\mathcal{A}\| + \|w\|$ where $\|w\| = n + \sum_{i\in[n]} \|d_i\|$ for $w = (a_1, d_1)\cdots(a_n, d_n)$.

Definition 4 *(The weight realizability problems).* Input: a k-WRA $\mathcal{A}$ over Σ, $\mathbb{D}$, $\mathcal{S}$ and a weight $s \in S$
(The run weight realizability problem) $\exists w.\exists \rho \in \mathrm{Run}_{\mathcal{A}}(w).\, \mathrm{wt}(\rho) = s$?
(The data word weight realizability problem) $\exists w.\, [\![\mathcal{A}]\!](w) = s$?
The input size of both problems is $\|\mathcal{A}\|$.

Theorem 1. *The run weight computation problem is NP-complete.*

Proof. Assume we are given a k-WRA $\mathcal{A} = (Q, Q_0, T, Q_f, \mathrm{wt})$ over Σ, $\langle D, \mathcal{R}\rangle$, $\mathcal{S} = (S, +, \cdot, 0, 1)$, a data word $w \in (\Sigma \times D)^+$ and $s \in S$.

(NP solvability) Let $w = (a_1, d_1)\cdots(a_n, d_n)$ and $m = \max\{\|d_i\| \mid i \in [n]\}$. By the assumption on complexity of computing weights of WRA, $\mathrm{wt}(c \vdash_{t,d} c')$ can be computed in $O(p_1(\|t\|) + p_2(\|t\| + m)k)$ time. Thus, for any run $\rho \in \mathrm{Run}_{\mathcal{A}}(w)$, the weight $\mathrm{wt}(\rho)$ can be computed in $O((p_1(\|t\|) + p_2(\|t\| + m)k)|w|)$ time. Hence, we can nondeterministically choose a run of w and test whether $\mathrm{wt}(\rho) = s$ in polynomial time.

(NP-hardness) We restrict the problem as:

For every transition $t \in T$, $j \in [k]$ and $d_1, d_2 \in D$, $\mathrm{wtt}(t) = \mathrm{wtd}(t, j)(d_1, d_2) = 1$. Also $s = 1$.

Then, for any switch $c \vdash_{t,d} c'$, we have $\mathrm{wt}(c \vdash_{t,d} c') = 1$. This implies that for every run $\rho \in \mathrm{Run}_{\mathcal{A}}(w)$, we have $\mathrm{wt}(\rho) = 1 = s$. Therefore, the problem restricted in this way asks for an input k-WRA $\mathcal{A}$ and a data word w, whether $\exists \rho \in \mathrm{Run}_{\mathcal{A}}(w)$. The k-WRA in this setting can be regarded as a RA (standard register automata without weight) and the above problem is equivalent to the membership problem that asks whether a given data word w is accepted by $\mathcal{A}$ regarded as a RA. Hence the run weight computation problem is NP-hard because the membership problem for RA is NP-complete [17]. □

To discuss the complexity of the data word computation problem, we use the complexity class #P, the class of function problems that can be solved by counting the number of accepting runs of a polynomial-time non-deterministic Turing machine. An example of #P-complete problem is #2SAT [15]: How many different variable assignments will satisfy a given 2-CNF formula?

Let $\mathcal{N} = (\mathbb{N}, +, \cdot, 0, 1)$ be the semiring of natural numbers.

Lemma 1. *The data word weight computation problem of k-WRA $\mathcal{A} = (Q, Q_0, T, Q_f, (\mathrm{wtt}, \mathrm{wtd}))$ over Σ, $\langle D, \mathcal{R}\rangle$ and $\mathcal{N}$ is #P-hard even if $\mathrm{wtt}(t) = \mathrm{wtd}(t, j)(d, d') = 1$ for every $t \in T$, $j \in [k]$, $d, d' \in D$.*

Proof. We reduce #2SAT problem, which is known to be #P-complete, to the data word weight computation problem. Let $\phi = c_1 \wedge c_2 \wedge \cdots \wedge c_m$ be a given 2-CNF, where each c_i ($i \in [m]$) is a clause consisting of two literals and $z_1, \ldots, z_n$ are Boolean variables appearing in ϕ. We construct n-WRA $\mathcal{A}_\phi = (Q, Q_0, T, Q_f, (\mathrm{wtt}, \mathrm{wtd}))$ over Σ, $\langle D, \mathcal{R}\rangle$, $\mathcal{N}$ and input data word w from ϕ as follows. Let $\Sigma = \{a\}$, D be an infinite set containing $\top$ and $\bot$, and let $\mathcal{R} = \{=, \neq\}$ where $=$ and $\neq$ are (an extension of) the equality on Boolean values (logical equivalence) and its negation, respectively. Note that $\bot$ is the initial value. The values of all weight functions wtt and wtd are defined as $1 \in \mathbb{N}$. Let $Q = \{q_i \mid i \in [n]\} \cup \{q_{k,l} \mid k \in [m], l \in [2]\} \cup \{q'_k \mid k \in [m]\} \cup \{q_r, q_f\}$, $Q_0 = \{q_1\}$ and $Q_f = \{q_f\}$. The input word is $w = (a, \top)\cdots(a, \top)$ of length $|w| = n + 2m$. We construct the following transitions and add them to T: The first group of transitions nondeterministically simulates an assignment of a Boolean value to each z_i ($i \in [n]$). If x_i is updated to be $\top$, it means z_i is assigned tt, and otherwise, it means z_i is assigned ff.

$$q_1 \rightarrow^{a}_{tt,\{1\}} q_2,\; q_1 \rightarrow^{a}_{tt,\emptyset} q_2,\; \ldots,\; q_n \rightarrow^{a}_{tt,\{n\}} q_{1,1},\; q_n \rightarrow^{a}_{tt,\emptyset} q_{1,1}.$$

The second group of transitions deterministically evaluates the truth value of each clause $c_k = y_{k,1} \vee y_{k,2}$ ($k \in [m]$).

$$\begin{array}{lll}
q_{k,1} \rightarrow^{a}_{x_i^=,\emptyset} q'_k, & q_{k,1} \rightarrow^{a}_{x_i^{\neq},\emptyset} q_{k,2} & \text{if } y_{k,1} = z_i,\\
q_{k,1} \rightarrow^{a}_{x_i^{\neq},\emptyset} q'_k, & q_{k,1} \rightarrow^{a}_{x_i^=,\emptyset} q_{k,2}, & \text{if } y_{k,1} = \overline{z_i},\\
q_{k,2} \rightarrow^{a}_{x_i^=,\emptyset} q_{k+1,1}, & q_{k,2} \rightarrow^{a}_{x_i^{\neq},\emptyset} q_{\mathrm{r}}, & \text{if } y_{k,2} = z_i,\\
q_{k,2} \rightarrow^{a}_{x_i^{\neq},\emptyset} q_{k+1,1}, & q_{k,2} \rightarrow^{a}_{x_i^=,\emptyset} q_{\mathrm{r}}, & \text{if } y_{k,2} = \overline{z_i},\\
q'_k \rightarrow^{a}_{tt,\emptyset} q_{k+1,1} & &
\end{array}$$

where $q_{m+1,1}$ is the final state q_f. The state q_{r} is a dead state with no outgoing transition. The states q'_k are used to skip the evaluation of literals when a preceding literal evaluates to $\top$ in the clause.

For a truth-value assignment $\alpha : \{z_1, \ldots, z_n\} \to \{tt, ff\}$, let $\theta_\alpha \in \Theta_n$ be $\theta_\alpha(x_i) = \top$ if $\alpha(z_i) = tt$ and $\theta_\alpha(x_i) = \bot$ otherwise. Assume $\mathcal{A}_\phi$ is fed with the input data word $w = (a, \top) \ldots (a, \top)$ of length $n + 2m$. After conducting the first group of transitions, the assignment of $\mathcal{A}_\phi$ becomes θ_α for some truth-value assignment α. Because the second group of transitions deterministically verifies whether ϕ evaluates to tt without register update, that part of the run is uniquely determined. In other words, there is a one-to-one correspondence between the set of maximal sequences of switches of w in $\mathcal{A}_\phi$ and the set of assignments. Therefore, a maximal sequence of switches of w in $\mathcal{A}_\phi$ is a run ρ of w if and only if ϕ is satisfied by the truth-value assignment α corresponding to the assignment θ_α obtained by ρ.

Consequently, the number of the truth-value assignments satisfying ϕ and the number of the runs of w in $\mathcal{A}_\phi$, namely, the summation of the weights of the all runs of w are equal. $\mathcal{A}_\phi$ can be constructed in polynomial time of the size of ϕ. Thus, the polynomial time reduction from #2SAT to the data word weight computation problem is completed. □

Lemma 2. *The data word weight computation problem for k-WRA is PSPACE-solvable. When the semiring is $\mathcal{N}$, and* wtt *is bounded and* wtd$(t, j)(d_1, d_2) = 1$ *for every $t \in T$ and $j \in [k]$ and $d_1, d_2 \in D$ for a given k-WRA $\mathcal{A} = (Q, Q_0, T, Q_f, (\mathrm{wtt}, \mathrm{wtd}))$, the problem becomes #P-solvable.*

Proof. PSPACE-solvability is easy to show. The weight of a run of an input data word can be calculated in polynomial time by the proof of Theorem 1, and we need additional polynomial space to store the sum of the weights of all runs of the input data word.

Next, we discuss #P-solvability. Assume a k-WRA $\mathcal{A} = (Q, Q_0, T, Q_f, (\mathrm{wtt}, \mathrm{wtd}))$ over Σ, $\langle D, \mathcal{R}\rangle$, $\mathcal{N}$ where wtt is bounded and $\mathrm{wtd}(t, j)(d_1, d_2) = 1$ for every $t \in T$ and $j \in [k]$ and $d_1, d_2 \in D$. If $\mathrm{wtt}(t) = 1$ for every $t \in T$ in addition, then $\mathrm{wt}(\rho) = 1$ for every run ρ in $\mathcal{A}$. From such $\mathcal{A}$, we can construct a polynomial-time nondeterministic Turing machine $\mathcal{M}_\mathcal{A}$ that simulates $\mathcal{A}$ so that the number of accepting runs of an input w in $\mathcal{M}_\mathcal{A}$ equals $[\![\mathcal{A}]\!](w)$. In the general case where $\mathrm{wtt}(t) \neq 1$ for some $t \in T$, from a given k-WRA $\mathcal{A} = (Q, Q_0, T, Q_f, (\mathrm{wtt}, \mathrm{wtd}))$, we construct k-WRA $\mathcal{A}' = (Q', Q'_0, T', Q'_f, (\mathrm{wtt}', \mathrm{wtd}'))$ such that $[\![\mathcal{A}']\!] = [\![\mathcal{A}]\!]$ and $\mathrm{wtd}' = \mathrm{wtd}$ and $\mathrm{wtt}'(t') = 1$ for every $t' \in T'$, as follows. For $M = \max\{\mathrm{wtt}(t) \mid t \in T\}$, let $Q' = Q \times [M]$ and $T' = \{(q, i) \rightarrow^{a}_{\varphi,\Lambda} (q', j) \mid t = q \rightarrow^{a}_{\varphi,\Lambda} q' \in T,\ i \in [M],\ 1 \le j \le \mathrm{wtt}(t)\}$. Note that M is a constant by the assumption. Also let $Q'_0 = \{(q_0, 1) \mid q_0 \in Q_0\}$ and $Q'_f = \{(q_f, i) \mid q_f \in Q_f,\ i \in [M]\}$. $\mathcal{A}'$ satisfies $[\![\mathcal{A}']\!] = [\![\mathcal{A}]\!]$ because for each run $q_0 \vdash_{t_1,d_1} q_1 \vdash_{t_2,d_2} q_2 \cdots \vdash_{t_n,d_n} q_n$ in $\mathcal{A}$, there are exactly $\prod_{j=1}^{n} \mathrm{wtt}(t_j)$ runs $(q_0, 1) \vdash_{t'_1,d_1} (q_1, i_1) \vdash_{t'_2,d_2} (q_2, i_2) \cdots \vdash_{t'_n,d_n} (q_n, i_n)$ in $\mathcal{A}'$ where t'_j is the transition obtained from t_j and $1 \le i_j \le \mathrm{wtt}(t_j)$ for $j \in [n]$. This construction of $\mathcal{A}'$ can be done in polynomial time. Therefore, the data word weight computation problem is in #P under the given condition. □

Theorem 2. *Let $\mathcal{A} = (Q, Q_0, T, Q_f, (\mathrm{wtt}, \mathrm{wtd}))$ be a k-WRA over Σ, $\langle D, \mathcal{R}\rangle$, $\mathcal{N}$. If* $\max\{\mathrm{wtt}(t) \mid t \in T\}$ *is uniformly bounded and* $\mathrm{wtd}(t, j)(d_1, d_2) = 1$ *for every $t \in T$, $j \in [k]$ and $d_1, d_2 \in D$, then the data word weight computation problem is #P-complete.*

Proof. By Lemmas 1 and 2. □

Theorem 3. *The run weight realizability problem for k-WRA is undecidable even if $k = 1$, all the values of weight functions are one and every relation of the data structure is decidable.*

Proof. We prove the theorem by a reduction from the Post correspondence problem (PCP). Let $I = \langle (u_1, \ldots, u_m), (v_1, \ldots, v_m)\rangle$ be a given instance of PCP over Σ where $u_i, v_i \in \Sigma^*$ for $i \in [m]$. From I, we construct a 1-WRA $\mathcal{A}_I = (\{q_0, q, q_f\}, \{q_0\}, T, \{q_f\}, \mathrm{wt})$ over $\{a\}, \langle D, \mathcal{R}\rangle, \mathcal{N}$ where the data structure $\langle D, \mathcal{R}\rangle$, the set T of transitions and the weight functions $\mathrm{wt} = (\mathrm{wtt}, \mathrm{wtd})$ are defined as follows.

- $D = \Sigma^* \times \Sigma^*$ with $\perp = (\varepsilon, \varepsilon) \in D$ as the initial value and $\mathcal{R} = \{R_i \mid i \in [m]\} \cup \{\mathrm{EQ}\}$ where for $x, y, x', y' \in \Sigma^*$, $(x, y)R_i(x', y') \Leftrightarrow (x' = xu_i$ and $y' = yv_i)$ for $i \in [m]$ and $(x, y)\mathrm{EQ}(x', y') \Leftrightarrow (x = y)$.
- $T = \{q_0 \rightarrow^a_{x_1^{R_i}, \{1\}} q,\ q \rightarrow^a_{x_1^{R_i}, \{1\}} q \mid i \in [m]\} \cup \{q \rightarrow^a_{x_1^{\mathrm{EQ}}, \emptyset} q_f\}$.
- $\mathrm{wtt}(t) = \mathrm{wtd}(t, 1)(d_1, d_2) = 1$ for every $t \in T$, $d_1, d_2 \in D$.

It is easy to see that I has a solution of PCP if and only if there is a run ρ of some $w \in (\{a\} \times D)^+$ in $\mathcal{A}_I$ such that $\mathrm{wt}(\rho) = 1$. □

Corollary 1. *The data word weight realizability problem of k-WRA is undecidable even if $k = 1$, all the values of weight functions are one and every relation of the data structure is decidable.* □

The above results imply that the realizability problems are already undecidable for ordinary RA (w/o weights). This motivates us to introduce a subclass of WRA for which the realizability problems and related optimization problems are solvable while the weights make sense, which are given in section 5.2.

4. Transition decomposition by register type

In this section, we will define a *normal form* WRA. First, we introduce a *register type* as a finite abstraction of assignments with respect to the relations in $\mathcal{R}$ of a given data structure $\langle D, \mathcal{R}\rangle$.

Definition 5 *([24]).* A register type (of k registers) for a data structure $\langle D, \mathcal{R}\rangle$ is an arbitrary function $\gamma : ([k] \times [k]) \rightarrow (\mathcal{R} \rightarrow \mathbb{B})$. Let Γ_k denote the collection of all register types of k registers. For an assignment $\theta \in \Theta_k$ and a register type $\gamma \in \Gamma_k$, if $\forall i, j \in [k] \forall R \in \mathcal{R}.(\gamma(i, j)(R) = 1 \Leftrightarrow (\theta(i), \theta(j)) \in R)$ holds, we write $\theta : \gamma$ and we say that the type of θ is γ. □

Let $\mathcal{A} = (Q, Q_0, T, Q_f, \mathrm{wt})$ be an arbitrary k-WRA. From $\mathcal{A}$, we define k-WRA $\mathcal{A}' = (Q', Q_0', T', Q_f', \mathrm{wt}')$ as follows: $Q' = Q \times \Gamma_k$. $Q_0' = Q_0 \times \{\gamma_0\}$ where γ_0 is defined as $\forall R \in \mathcal{R}\big[(\forall i, j \in [k].\gamma_0(i, j)(R) = 1) \Leftrightarrow ((\perp, \perp) \in R)\big]$. $Q_f' = Q_f \times \Gamma_k$. T' is the smallest set of transitions $t' = (p, \gamma) \rightarrow^a_{\varphi', \Lambda} (q, \gamma')$ satisfying the following condition:

$t = p \rightarrow^a_{\varphi, \Lambda} q \in T$, $\gamma, \gamma' \in \Gamma_k$, $\varphi' = \varphi \wedge \prod_{R \in \mathcal{R}} (\prod_{i=1}^k \alpha_i^R \wedge \beta_i^R) \wedge \delta^R$ where $\alpha_i^R \in \{x_i^R, \neg x_i^R\}$, $\beta_i^R \in \{x_i^{R^{-1}}, \neg x_i^{R^{-1}}\}$, $\delta^R \in \{\mathrm{in}^R, \neg \mathrm{in}^R\}$, and $\gamma'(i, j)(R) = 1$ if and only if $i \notin \Lambda$, $j \notin \Lambda$ and $\gamma(i, j)(R) = 1$, or
$i \notin \Lambda$, $j \in \Lambda$ and $\alpha_i^R = x_i^R$, or
$i \in \Lambda$, $j \notin \Lambda$ and $\beta_j^R = x_j^{R^{-1}}$, or
$i \in \Lambda$, $j \in \Lambda$ and $\delta^R = \mathrm{in}^R$.

In the above definition, φ' is the conjunction of φ and

- α_i^R, which determines whether the contents of the i-th register and an input data value d satisfy R,
- β_i^R, which determines whether an input data value d and the contents of the i-th register satisfy R,
- δ^R, which determines whether d is reflexive on R.

This implies that when t' is applied and d is loaded to the registers specified by Λ, the register type γ' of the resultant assignment $\theta[\Lambda \rightarrow d]$ is determined by the register type γ of θ and Λ and φ' and does not depend on d. Therefore, if $t \in T$, $\gamma \in \Gamma_k$ and φ' are given, the transition belonging to T' is uniquely determined. We write that transition as $s_{t,\gamma,\varphi'}$. Finally, define $\mathrm{wt}' = (\mathrm{wtt}', \mathrm{wtd}')$ where for each $t' = s_{t,\gamma,\varphi'} \in T'$, $\mathrm{wtt}'(t') = \mathrm{wtt}(t)$ and for $j \in [k]$, $\mathrm{wtd}'(t', j) = \mathrm{wtd}(t, j)$. This completes the definition of k-WRA $\mathcal{A}'$.

Example 3. Let $k = 2$, $\mathcal{R} = \{R\}$ and consider transition $t = p \rightarrow^a_{x_1^R, \{2\}} q$ and register type γ such that $\gamma(i, j)(R) = 1$ for $(i, j) \in \{(1, 1), (1, 2), (2, 2)\}$ and $\gamma(2, 1)(R) = 0$. For example, we let $\alpha_1^R = x_1^R$, $\beta_1^R = x_1^{R^{-1}}$, $\alpha_2^R = x_2^R$, $\beta_2^R = x_2^{R^{-1}}$, $\delta^R = \mathrm{in}^R$, and let

$$\varphi' = x_1^R \wedge x_1^{R^{-1}} \wedge x_2^R \wedge x_2^{R^{-1}} \wedge \mathrm{in}^R.$$

Also assume $\theta : \gamma$, $(\theta, d) \models \varphi'$ and $\theta' = \theta[\{2\} \leftarrow d]$. These three assumptions mean

$$(\theta(1), \theta(1)), (\theta(1), \theta(2)), (\theta(2), \theta(2)) \in R,$$
$$(\theta(2), \theta(1)) \notin R,$$

$$(\theta(1), d), (d, \theta(1)), (\theta(2), d), (d, \theta(2)), (d, d) \in R, \text{ and}$$
$$\theta'(1) = \theta(1) \text{ and } \theta'(2) = d,$$

respectively. Therefore, we have that

$$(\theta'(1), \theta'(1)), (\theta'(1), \theta'(2)), (\theta'(2), \theta'(1)), (\theta'(2), \theta'(2)) \in R.$$

Hence, we construct the rule:

$$(p, \gamma) \to^{a}_{x_1^R \wedge x_1^{R^{-1}} \wedge x_2^R \wedge x_2^{R^{-1}} \wedge \mathrm{in}^R, \{2\}} (q, \gamma^{(1)})$$

where $\gamma^{(1)}(i, j)(R) = 1$ for $i, j \in [2]$. Since an input data value is loaded to the second register by t, the register type after a switch by t does not depend on $\theta(2)$ (the content of the second register before the switch). Therefore, we can merge constructed transitions with the same guard on x_1 and an input data value by omitting the guard condition on x_2 as follows:

$$(p, \gamma) \to^{a}_{x_1^R \wedge x_1^{R^{-1}} \wedge \mathrm{in}^R, \{2\}} (q, \gamma^{(1)}), \qquad (p, \gamma) \to^{a}_{x_1^R \wedge x_1^{R^{-1}} \wedge \neg\mathrm{in}^R, \{2\}} (q, \gamma^{(2)}),$$
$$(p, \gamma) \to^{a}_{x_1^R \wedge \neg x_1^{R^{-1}} \wedge \mathrm{in}^R, \{2\}} (q, \gamma^{(3)}), \qquad (p, \gamma) \to^{a}_{x_1^R \wedge \neg x_1^{R^{-1}} \wedge \neg\mathrm{in}^R, \{2\}} (q, \gamma^{(4)})$$

where

$$\gamma^{(i)}(1,2)(R) = 1, \quad \gamma^{(i)}(2,1)(R) = 1 \quad (i \in \{1,2\}),$$
$$\gamma^{(i)}(1,2)(R) = 1, \quad \gamma^{(i)}(2,1)(R) = 0 \quad (i \in \{3,4\}),$$
$$\gamma^{(i)}(2,2)(R) = 0 \ (i \in \{2,4\}), \quad \gamma^{(i)}(j,j)(R) = 1 \ (\text{otherwise}).$$

Lemma 3. *Let $\mathcal{A} = (Q, Q_0, T, Q_f, \mathrm{wt})$ be an arbitrary k-WRA and $\mathcal{A}' = (Q', Q'_0, T', Q'_f, \mathrm{wt}')$ be the k-WRA obtained from $\mathcal{A}$ by the transition decomposition by register type. Also let $w = (a_1, d_1) \cdots (a_n, d_n) \in (\Sigma \times D)^+$ be an arbitrary data word. For a run $\rho = c_0 \vdash_{t_1,d_1} c_1 \vdash_{t_2,d_2} \cdots \vdash_{t_n,d_n} c_n \in \mathrm{Run}_{\mathcal{A}}(w)$, there exists a run $\rho' = c'_0 \vdash_{s_{t_1,\gamma_0,\varphi'_1},d_1} c'_1 \vdash_{s_{t_2,\gamma_1,\varphi'_2},d_2} \cdots \vdash_{s_{t_n,\gamma_{n-1},\varphi'_n},d_n} c'_n \in \mathrm{Run}_{\mathcal{A}'}(w)$ such that $\mathrm{wt}(\rho') = \mathrm{wt}(\rho)$. Conversely, for a run $\rho' \in \mathrm{Run}_{\mathcal{A}'}(w)$, there exists a run $\rho \in \mathrm{Run}_{\mathcal{A}}(w)$ such that $\mathrm{wt}(\rho) = \mathrm{wt}(\rho')$.*

Proof. Consider a data word w and a run ρ stated in the lemma and assume $c_i = (q_i, \theta_i), \theta_i : \gamma_i$ for $i \in \{0\} \cup [n]$. By the construction of T', there exists a unique transition $s_{t_i,\gamma_{i-1},\varphi'_i} = (q_{i-1}, \gamma_{i-1}) \to^{a_i}_{\varphi'_i,\Lambda} (q_i, \gamma_i) \in T'$ such that $((q_{i-1}, \gamma_{i-1}), \theta_{i-1}) \vdash_{s_{t_i,\gamma_{i-1},\varphi'_i},d_i} ((q_i, \gamma_i), \theta_i)$ in $\mathcal{A}'$ where φ'_i is determined by whether $(\theta_{i-1}, d_i) \models x_j^R$, $(\theta_{i-1}, d_i) \models x_j^{R^{-1}}$ and $(\theta_{i-1}, d_i) \models \mathrm{in}^R$ hold or not for $j \in [k]$ and $R \in \mathcal{R}$. If we concatenate the above switches, we obtain a run ρ' of w in $\mathcal{A}'$ and $\mathrm{wt}(\rho') = \mathrm{wt}(\rho)$.

Conversely, for $i \in [n]$, let $c'_{i-1} \vdash_{s_{t_i,\gamma_{i-1},\varphi'_i},d_i} c'_i$ be a switch in $\mathcal{A}'$ where $s_{t_i,\gamma_{i-1},\varphi'_i} = (q_{i-1}, \gamma_{i-1}) \to^{a_i}_{\varphi'_i,\Lambda} (q_i, \gamma_i) \in T'$. The transition of $\mathcal{A}$ corresponding to $s_{t_i,\gamma_{i-1},\varphi'_i} \in T'$ is exactly $t_i \in T$. By the construction of T', $(\theta_{i-1}, d_i) \models \varphi'_i$ implies $(\theta_{i-1}, d_i) \models \varphi_i$. Therefore, $c_{i-1} \vdash_{t_i,d_i} c_i$ is a switch in $\mathcal{A}$. The rest of the proof is similar to the former case; we lift the obtained switches to the run. □

A WRA obtained by the above transformation is called a *normal form WRA*.

Corollary 2. *Let $\mathcal{A} = (Q, Q_0, T, Q_f, \mathrm{wt})$ be an arbitrary k-WRA and $\mathcal{A}' = (Q', Q'_0, T', Q'_f, \mathrm{wt}')$ be the normal form k-WRA obtained from $\mathcal{A}$. Then, $[\![\mathcal{A}']\!] = [\![\mathcal{A}]\!]$ holds.*

Proof. By Lemma 3 and the definition of the weight function wt' of $\mathcal{A}'$. □

5. The optimal run problem

5.1. Definition of the problem

We introduce the problem of computing the optimal (infimum) weight of the runs from an initial ID to an accepting ID of a given WRA. We assume the tropical semiring $\mathcal{S}_{\mathrm{trpc}}$ (see Example 2) because by $\mathcal{S}_{\mathrm{trpc}}$ we can represent the minimum weight by the addition of the semiring. Of course, we could use the max-tropical semiring $(\mathbb{R} \cup \{-\infty\}, \max, +, -\infty, 0)$ instead.

Definition 6 *(The optimal run problem).* Input: a k-WRA $\mathcal{A}$ over $\Sigma, \langle D, \mathcal{R}\rangle, \mathcal{S}_{\mathrm{trpc}}$
Output: The infimum of $\{\mathrm{wt}(\rho) \mid \exists w \in (\Sigma \times D)^+.\ \rho \in \mathrm{Run}_{\mathcal{A}}(w)\}$ □

By Lemma 3 and the definition of the problem, the following property holds.

Corollary 3. *Let $\mathcal{A} = (Q, Q_0, T, Q_f, \mathrm{wt})$ be an arbitrary k-WRA and $\mathcal{A}' = (Q', Q'_0, T', Q'_f, \mathrm{wt}')$ be the normal form k-WRA obtained from $\mathcal{A}$. The solutions to the optimal run problem for $\mathcal{A}$ and $\mathcal{A}'$ are the same.* □

Let $\mathcal{A}$ and $\mathcal{A}'$ be as assumed in the above corollary. We will transform $\mathcal{A}'$ to an edge-weighted directed graph $G = \langle V, E\rangle$ such that the solution of the optimal run problem is equal to the weight of the minimum-weight path of G. The difficulty lies in the requirement that we must construct G without knowing an input data word w to $\mathcal{A}'$ or assignments appearing in a run of w in $\mathcal{A}'$. To overcome this problem, we introduce two properties in the next subsection.

5.2. Weighted simulation and weight computability

Let $\mathcal{A} = (Q, Q_0, T, Q_f, \mathrm{wt})$ be an arbitrary k-WRA and $\mathcal{A}' = (Q', Q'_0, T', Q'_f, \mathrm{wt}')$ be the normal form k-WRA obtained from $\mathcal{A}$. We say that k-WRA $\mathcal{A}'$ has *weighted simulation property* if every $t' = (p, \gamma) \rightarrow^{a}_{\varphi',\Lambda} (q, \gamma') \in T'$ satisfies the following condition: for every $\theta_1, \theta_2 \in \Theta_k$ such that $\theta_1 : \gamma$ and $\theta_2 : \gamma$, $\mathrm{wt}_{t'}(\theta_1) = \mathrm{wt}_{t'}(\theta_2)$ holds where $\mathrm{wt}_{t'}(\theta) = \inf\{\mathrm{wt}(((p, \gamma), \theta) \vdash_{t',d} ((q, \gamma'), \theta[\Lambda \leftarrow d])) \mid d \in D\}$. Also, we say that k-WRA $\mathcal{A}'$ has *weight computability* if the above infimum, denoted as $\mathrm{wt}(t')$, can be computed in polynomial time of $\|\mathcal{A}\|$.

Assume a data structure $\mathbb{D} = \langle \mathbb{R}, \{R\}\rangle$ where $(d, d') \in R$ iff $d' - d \geq 1$ and consider a 1-WRA $\mathcal{A}$ over $\Sigma = \{a\}$, $\mathbb{D}$ and $\mathcal{S}_{\mathrm{trpc}}$. Since no d satisfies $d - d \geq 1$, every $\theta \in \Theta_1$ has type γ where $\gamma(1, 1)(R) = 0$ (meaning that $\theta(1) - \theta(1) < 1$). Let $t = p \rightarrow^{a}_{tt,\emptyset} p$ be a transition of $\mathcal{A}$ and consider the transition $t' = (p, \gamma) \rightarrow^{a}_{\varphi',\emptyset} (p, \gamma)$ where $\varphi' = x_1^R \wedge \neg x_1^{R^{-1}} \wedge \neg \mathrm{in}^R$ constructed from t as a transition of the normal form 1-WRA and assume $\mathrm{wtt}(t') = 0$. Note that $(\theta, d) \models \varphi'$ iff $d - \theta(1) \geq 1$ for every $\theta \in \Theta_1$ and $d \in \mathbb{R}$. If $\mathrm{wtd}(t', 1)(d, d') = |d' - d|$, then for every $\theta \in \Theta_1$ such that $\theta : \gamma$, $\mathrm{wt}_{t'}(\theta) = \inf\{d - \theta(1) \mid d - \theta(1) \geq 1\} = 1$ and so t' satisfies the condition required by weighted simulation property. On the other hand, if $\mathrm{wtd}(t', 1)(d, d') = |d|$, then $\mathrm{wt}_{t'}(\theta) = \inf\{|\theta(1)| \mid \exists d \in \mathbb{R}.\ d - \theta(1) \geq 1\} = |\theta(1)|$, and in this case t' does not satisfy the condition required by weighted simulation property.

Weighted simulation is a natural extension of the property of TA and WTA that the infinite set of IDs can be divided into finite sets called clock regions such that any IDs belonging to a same clock region are indistinguishable. The above two properties are undecidable in general even if a binary relation appearing in the guard of a transition is decidable.

For example, we can show that weighted simulation property is undecidable by reducing the Post correspondence problem (PCP) over a finite alphabet Σ that asks for a given instance $I = (u_1, \ldots, u_n; v_1, \ldots, v_n)$ where $u_i, v_i \in \Sigma^*$ for $1 \leq i \leq n$, whether there exist $i_1, \ldots, i_m \in [n]$ $(m \geq 1)$ such that $u_{i_1} \cdots u_{i_m} = v_{i_1} \cdots v_{i_m}$. Assume that we are given an instance I of PCP. Let $\mathbb{D}_I = \langle [n]^*, \{R_I\}\rangle$ be the data structure such that for $d, d' \in [n]^*$,

$$(d, d') \in R_I \text{ iff } d = \varepsilon \text{ and } d' \text{ is a solution of } I.$$

From I, we construct 1-WRA $\mathcal{A} = (Q, Q_0, T, Q_f, (\mathrm{wtt}, \mathrm{wtd}))$ over $\Sigma = \{a\}$, $\mathbb{D}_I$, $\mathcal{S}_{\mathrm{trpc}}$ where $Q = \{q_0, q_1, q_2\}$, $Q_0 = \{q_0\}$, $Q_f = \{q_2\}$, $T = \{t_1 = q_0 \rightarrow^{a}_{tt,\{1\}} q_1,\ t_2 = q_1 \rightarrow^{a}_{x_1^{R_I},\emptyset} q_2\}$ and $\mathrm{wtt}(t_m) = 0$, $\mathrm{wtd}(t_m, 1)(d, d') = 0$ for $m \in [2]$, $d, d' \in [n]^*$. Let $\mathcal{A}' = (Q', Q'_0, T', Q'_f, \mathrm{wt}')$ be the normal form WRA obtained from $\mathcal{A}$. Let $\gamma_0 \in \Gamma_1$ be the register type such that $\gamma_0(1, 1)(R_I) = 0$; i.e., $\theta : \gamma_0$ iff $(\theta(1), \theta(1)) \notin R_I$. Note that every $\theta \in \Theta_1$ satisfies $\theta : \gamma_0$ because R_I is irreflexive. Consider the transition $t'_2 = (q_1, \gamma_0) \rightarrow^{a}_{x_1^{R_I} \wedge \neg x_1^{R_I^{-1}} \wedge \neg \mathrm{in}^{R_I},\emptyset} (q_2, \gamma_0) \in T'$. Then, for $\theta \in \Theta_1$, we obtain

$$\mathrm{wt}_{t'_2}(\theta) = \begin{cases} 0 & \text{if } (\theta(1), d) \in R_I \text{ for some } d \in [n]^*, \\ \infty & \text{otherwise.} \end{cases}$$

(Note that $(\theta, d) \models x_1^{R_I}$ implies $(\theta, d) \models \neg x_1^{R_I^{-1}} \wedge \neg \mathrm{in}^{R_I}$ for any $d \in [n]^*$ because $(\theta(1), d) \in R_I$ implies $d \neq \varepsilon$.) By the definition of R_I, $(\theta(1), d) \in R_I$ if and only if $\theta(1) = \varepsilon$ and d is a solution of I. Hence, if I has a solution d, then $\mathrm{wt}_{t'_2}(\theta_0) = 0$ for $\theta_0 \in \Theta_1$ such that $\theta_0(1) = \varepsilon$ since $(\theta_0(1), d) \in R_I$, and $\mathrm{wt}_{t'_2}(\theta) = \infty$ for every $\theta \in \Theta_1$ other than θ_0. If I does not have a solution, then $t'_1 = (q_0, \gamma_0) \rightarrow^{a}_{\neg x_1^{R_I} \wedge \neg x_1^{R_I^{-1}} \wedge \neg \mathrm{in}^{R_I},\{1\}} (q_1, \gamma_0) \in T'$ satisfies $\mathrm{wt}_{t'_1}(\theta) = 0$ for every $\theta \in \Theta_1$, and for every $t'' \in T'$ other than t'_1, $\mathrm{wt}_{t''}(\theta) = \infty$ for every $\theta \in \Theta_1$. Therefore, $\mathcal{A}'$ does not have weighted simulation property if and only if I have a solution, and hence the reduction completes.

Weighted simulation says that if two assignments θ_1, θ_2 have a same register type γ, the infimum of the weights of switches from (p, θ_1) to (q, θ'_1) by t' is the same as that from (p, θ_2) to (q, θ'_2) by t'. This property, together with weight computability, enables us to compute the infimum of the weights from (p, γ) to (q, γ') without knowing an assignment or an input data value.

5.3. Transformation to a directed graph

We will present a transformation from a given k-WRA to an edge-weighted directed graph when weighted simulation and weight computability hold. Let $\mathcal{A}$ be a k-WRA over Σ, $\langle D, \mathcal{R}\rangle$ and $\mathcal{S}_{\mathrm{trpc}}$ that satisfies weighted simulation and weight computability and $\mathcal{A}' = (Q', Q_0', T', Q_f', \mathrm{wt}')$ be the normal form k-WRA obtained from $\mathcal{A}$.

Construct the edge-weighted directed graph $G = \langle V, E\rangle$ where V and E are the sets of nodes and edges respectively, where $V = Q'$ and $E \subseteq V \times V \times T' \times \mathbb{R}_{\geq 0}$ is defined as follows: For each transition $s_{t,\gamma,\varphi'} = (p,\gamma) \rightarrow^{a}_{\varphi',\Lambda} (q,\gamma') \in T'$ of $\mathcal{A}'$, compute $\mathrm{wt}(s_{t,\gamma,\varphi'})$, which is possible by weighted simulation and weight computability. If $\mathrm{wt}(s_{t,\gamma,\varphi'}) < \infty$, add $((p,\gamma),(q,\gamma'), s_{t,\gamma,\varphi'}, \mathrm{wt}(s_{t,\gamma,\varphi'}))$ to E. Let $e(s_{t,\gamma,\varphi'})$ denote the edge created from $s_{t,\gamma,\varphi'}$. Also let $e(\rho)$ denote the sequence $e(s_1)e(s_2)\ldots e(s_n)$ for a run $\rho = ((q_0,\gamma_0),\perp^k) \vdash_{s_1,d_1} ((q_1,\gamma_1),\theta_1) \vdash_{s_2,d_2} ((q_2,\gamma_2),\theta_2)\cdots \vdash_{s_n,d_n} ((q_n,\gamma_n),\theta_n)$ of $\mathcal{A}'$.

For a path π in an edge-weighted directed graph, the weight of π is the sum of the weights of the edges in π, denoted by $\mathrm{wt}(\pi)$.

Lemma 4. *Let $\mathcal{A}$ and $\mathcal{A}'$ be the WRA above, and $\mathcal{A}'$ have weighted simulation property and weight computability. Let $G = \langle V, E\rangle$ be the directed graph obtained from $\mathcal{A}'$ by the above construction. Then there is a path π in G starting with an initial state and ending with a final state of $\mathcal{A}'$ if and only if there is a run ρ in $\mathcal{A}'$ such that $\pi = e(\rho)$. Moreover, for such a path π in G, $\mathrm{wt}(\pi) = \inf\{\mathrm{wt}(\rho) \mid \rho$ is a run in $\mathcal{A}'$ such that $\pi = e(\rho)\}$.*

Proof. Let $\pi = e_1e_2\ldots e_n$ be a path in G starting with an initial state and ending with a final state of $\mathcal{A}'$ where $e_i \in E$ ($i \in [n]$). Let $s_i \in T'$ be the third component of e_i for $i \in [n]$; i.e., $\pi = e(s_1)e(s_2)\ldots e(s_n)$. By the definition of weighted simulation property, for every transition $s = (p,\gamma) \rightarrow^{a}_{\varphi',\Lambda} (q,\gamma') \in T'$, $\mathrm{wt}(s) < \infty$ iff for every $\theta \in \Theta_k$ of type γ, there is a switch $((p,\gamma),\theta) \vdash_{s,d} ((q,\gamma'),\theta[\Lambda \leftarrow d])$ for some $d \in D$. This implies that there is a run ρ in $\mathcal{A}'$ that is a sequence of switches by $s_1, s_2, \ldots, s_n$, and thus $\pi = e(\rho)$. The converse direction can be proved by retracing the above discussion in reverse order.

Let $\pi = e_1e_2\ldots e_n$ be the above-mentioned path in G, and let $s_i = (q_{i-1},\gamma_{i-1}) \rightarrow^{a_i}_{\varphi_i',\Lambda_i} (q_i,\gamma_i) \in T'$ be the third component of e_i for $i \in [n]$. By the construction of G, $\mathrm{wt}(e_i) = \inf\{\mathrm{wt}(((q_{i-1},\gamma_{i-1}),\theta) \vdash_{s_i,d} ((q_i,\gamma_i),\theta[\Lambda_i \leftarrow d])) \mid d \in D\}$ for any $\theta \in \Theta_k$ such that $\theta : \gamma_{i-1}$. Therefore $\mathrm{wt}(\pi) = \mathrm{wt}(e_1) + \cdots + \mathrm{wt}(e_n) = \inf\{\mathrm{wt}(\rho) \mid \rho$ is a run in $\mathcal{A}'$ such that $\pi = e(\rho)\}$. $\square$

By Lemmas 3 and 4, the optimal run problem for a given k-WRA $\mathcal{A}$ can be solved by solving the minimum weight path problem for the directed graph G obtained from $\mathcal{A}$ via the normal form $\mathcal{A}'$ if $\mathcal{A}'$ satisfies weighted simulation and weight computability. Furthermore, we can find the original transition $t \in T$ of $\mathcal{A}$ from a given transition $s_{t,\gamma,\varphi'} \in T'$ as described in the proof of Lemma 3. In this way, we can easily reconstruct the run in $\mathcal{A}$ that provides the infimum weight from a minimum path found in G.

The description length $\|\mathcal{A}'\|$ of k-WRA $\mathcal{A}' = (Q', Q_0', T', Q_f', \mathrm{wt}')$ can be represented by the following relationship between the sizes of the corresponding components of $\mathcal{A}'$ and $\mathcal{A}$: $|\Gamma_k| = 2^{k^2|\mathcal{R}|}$, $|Q'| = |Q| \times |\Gamma_k|$, $|Q_0'| = |Q_0|$, $|Q_f'| = |Q_f| \times |\Gamma_k|$, $|T'| = (|Q| \times |\Gamma_k|) \times |\Sigma| \times 2^{2k|\mathcal{R}|+|\mathcal{R}|} \times 2^k \times (|Q| \times 1)$.

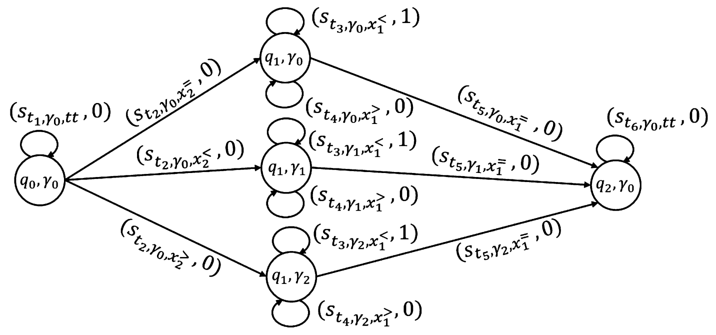
Theorem 4. *When the normal form k-WRA $\mathcal{A}'$ constructed from k-WRA $\mathcal{A} = (Q, Q_0, T, Q_f, \mathrm{wt})$ has weighted simulation property and weight computability, the time complexity of the optimal run problem for k-WRA $\mathcal{A}$ is $O(2^{k^2|\mathcal{R}|}|Q|(4^{k|\mathcal{R}|}2^{|\mathcal{R}|+k}|\Sigma||Q|+k^2|\mathcal{R}|))$.*

Proof. The above complexity is derived from the time complexity $O(|E| + |V|\log|V|)$ of Dijkstra algorithm by $|V| = |Q'|$, $|E| = |T'|$. $\square$

Example 4. Consider the WRA $\mathcal{A}_2$ of Example 2 again. Let $\mathcal{A}_2'$ be the normal form WRA obtained from $\mathcal{A}_2$. $\mathcal{A}_2'$ satisfies weighted simulation and weight computability. We show the directed graph G_1 for $\mathcal{A}_2'$ (see Fig. 2). For a label (t', w) of an edge, t' represents the applied transition and w represents the infimum of the weights of switches corresponding to the edge. The register types $\gamma_0, \gamma_1, \gamma_2$ in the node labels are as follows where γ_0 is the initial register type:

$$\begin{aligned}
&\gamma_0(1,2)(<) = 0, \quad \gamma_0(2,1)(<) = 0, \quad \gamma_0(1,2)(=) = \gamma_0(2,1)(=) = 1,\\
&\gamma_1(1,2)(<) = 0, \quad \gamma_1(2,1)(<) = 1, \quad \gamma_1(1,2)(=) = \gamma_1(2,1)(=) = 0,\\
&\gamma_2(1,2)(<) = 1, \quad \gamma_2(2,1)(<) = 0, \quad \gamma_2(1,2)(=) = \gamma_2(2,1)(=) = 0,\\
&\gamma_m(j,j)(<) = 0, \quad \gamma_m(j,j)(=) = 1, \quad \text{for } m \in \{0\} \cup [2],\ j \in [2],\\
&\gamma_m(i,j)(>) = \gamma_m(j,i)(<) \quad \text{for } m \in \{0\} \cup [2],\ i,j \in [2].
\end{aligned}$$

The edge with $(s_{t_1,\gamma_0,tt}, 0)$ represents the three edges generated from t_1 in $\mathcal{A}_2$. The optimal paths of G_1 are the simple paths from (q_0,γ_0) to (q_2,γ_0), and the weight infimum is 0.



$(s_{t_3,\gamma_0,x_1^<},1)$
$(s_{t_2,\gamma_0,x_2^=},0)$
q_1,γ_0
$(s_{t_1,\gamma_0,tt},0)$
$(s_{t_4,\gamma_0,x_1^>},0)$
$(s_{t_5,\gamma_0,x_1^=},0)$
$(s_{t_6,\gamma_0,tt},0)$
$(s_{t_3,\gamma_1,x_1^<},1)$
$(s_{t_2,\gamma_0,x_2^<},0)$
$(s_{t_5,\gamma_1,x_1^=},0)$
q_0,γ_0
q_1,γ_1
q_2,γ_0
$(s_{t_4,\gamma_1,x_1^>},0)$
$(s_{t_3,\gamma_2,x_1^<},1)$
$(s_{t_2,\gamma_0,x_2^>},0)$
$(s_{t_5,\gamma_2,x_1^=},0)$
q_1,γ_2
$(s_{t_4,\gamma_2,x_1^>},0)$

Fig. 2. The directed graph G_1 for $\mathcal{A}'_2$ of Example 2.

Now, consider the corresponding decision problem of the optimal run problem shown below. We will show this problem is PSPACE-complete.

Definition 7 *(The run weight bounding problem).* Input: a k-WRA $\mathcal{A}$ over Σ, $\langle D, \mathcal{R}\rangle$, $\mathcal{S}_{\mathrm{trpc}}$ and a weight $s \in \mathbb{R}_{\geq 0}$
Output: $\exists w. \exists \rho \in \mathrm{Run}_{\mathcal{A}}(w). \mathrm{wt}(\rho) < s$?
(In other words, $\inf\{\mathrm{wt}(\rho) \mid \exists w. \rho \in \mathrm{Run}_{\mathcal{A}}(w)\} < s$?)

Theorem 5. *The run weight bounding problem for k-WRA over Σ, $\langle D, \mathcal{R}\rangle$, $\mathcal{S}_{\mathrm{trpc}}$ is PSPACE-complete if weighted simulation and weight computability hold.*

Proof. (PSPACE-solvability) Let $\mathcal{A}$ be a given k-WRA and $\mathcal{A}'$ be the normal form k-WRA obtained from $\mathcal{A}$. Construct the edge-weighted directed graph G from $\mathcal{A}'$ as mentioned above. By Lemmas 3 and 4, the run weight bounding problem for $\mathcal{A}$ can be rephrased as "Is there a path π in G between an initial and final states of $\mathcal{A}'$ such that $\mathrm{wt}(\pi) < s$?" Without loss of generality, we can assume that π contains no loop and thus $|\pi| \leq |Q'|$, since we assume $\mathcal{S}_{\mathrm{trpc}}$. Assume that computing $\mathrm{wt}(t')$ for each $t' \in T'$ takes at most $O(\|\mathcal{A}\|^c)$ space where c is a constant. Since each state $(p, \gamma) \in Q'$ can be represented in $O(\log|Q| + k^2|\mathcal{R}|)$ bits, an $O(\log|Q| + k^2|\mathcal{R}| + \|\mathcal{A}\|^c)$ space-bounded nondeterministic Turing machine can examine every path in G between an initial and final states of $\mathcal{A}'$ whose length is at most $|Q'|$. Therefore this problem can be solved in PSPACE.

(PSPACE-hardness) As in the proof of NP-hardness in Theorem 1, we assume the value of every weight function is 0. Then, for every data word w and every run $\rho \in \mathrm{Run}_{\mathcal{A}}(w)$, $\mathrm{wt}(\rho) = 0$. When we choose any real number $s > 0$ for the input semiring value, the run weight bounding problem is expressed as: for a given k-WRA $\mathcal{A}$, $\exists w. \exists \rho \in \mathrm{Run}_{\mathcal{A}}(w)$?, which is equivalent to the emptiness problem for k-RA. Because the emptiness problem for RA is PSPACE-complete [17], the run weight bounding problem is PSPACE-hard. □

6. Restricted WRA

In the previous section, we discussed the optimal run problem for WRA that have weighted simulation property and weight computability. We introduced register type as an abstraction of register assignments, and we gave a method of constructing an edge-weighted graph from a given WRA. Each edge in the constructed graph is labeled with the infimum of the weights of the corresponding switches. By weighted simulation property, the infimum of the weights of the switches of WRA is determined by register type regardless of input data words. In this section, we will define a subclass of WRA whose weight is *a priori* determined by the register type. The advantage of this subclass over general WRA is that we can slightly relax weighted simulation property and weight computability since the weights are determined by the register type, not depending on data values in an input word. (See the discussion after the definition of k-WRA in Section 2.)

In this section, we assume that a collection of binary relations $\mathcal{R}$ is closed under the inverse, namely, $\mathcal{R}^{-1} = \{R^{-1} \mid R \in \mathcal{R}\} \subseteq \mathcal{R}$ for simplicity. We fix a data structure as $\mathbb{D} = \langle D, \mathcal{R}\rangle$.

Definition 8. A restricted k-register weighted automaton (k-RWRA) over $\Sigma, \mathbb{D}, \mathcal{S}$ is a tuple $\mathcal{A} = (Q, Q_0, T, Q_f, \mathrm{wt})$ where

- (Q, Q_0, T, Q_f) is a k-RA over $\Sigma, \mathbb{D}$.
- $\mathrm{wt} = (\mathrm{wtt}, \mathrm{wtd})$ where $\mathrm{wtt} : T \to S$ and $\mathrm{wtd} : (T \times [k]) \to (\mathbb{B}^{2|\mathcal{R}|} \to S)$.

The definition of RWRA is the same as that of WRA except for the weight function. In an RWRA, the weight function wtd is determined depending on which transition and register are concerned and whether each binary relation in $\mathcal{R}$ between the contents of the register and the input data value holds or not.

We introduce auxiliary functions to formally define the weight of the switch of RWRA. The function $\phi : D \times D \times \mathcal{R} \to \mathbb{B}$ is defined as

$$\phi(d_1, d_2, R) = \begin{cases} 1 & \text{if } (d_1, d_2) \in R \\ 0 & \text{otherwise} \end{cases}$$

for $d_1, d_2 \in D$ and $R \in \mathcal{R}$. Note that d_1 stands for the data value stored in a register and d_2 stands for an input data value. Using ϕ, the function $\Phi_{\mathcal{R}} : D \times D \to \mathbb{B}^{2|\mathcal{R}|}$ is defined as

$$\Phi_{\mathcal{R}}(d_1, d_2) = (\phi(d_1, d_2, R_1), ..., \phi(d_1, d_2, R_{|\mathcal{R}|}), \phi(d_2, d_2, R_1), ..., \phi(d_2, d_2, R_{|\mathcal{R}|}))$$

for $d_1, d_2 \in D$ and $\{R_1, ..., R_{|\mathcal{R}|}\} = \mathcal{R}$.

By $\mathrm{wtt}(t)$, we denote the weight of a transition $t \in T$, and by $\mathrm{wtd}(t, j)$ we denote the weight function of the j-th register for a transition $t \in T$ as in the case of WRA. The weight of a switch $(q, \theta) \vdash_{t,d} c'$ is defined as

$$\mathrm{wt}((q, \theta) \vdash_{t,d} c') = \prod_{j=1}^{k} \mathrm{wtd}(t, j)(\Phi_{\mathcal{R}}(\theta(j), d)) \cdot \mathrm{wtt}(t).$$

We introduce two properties of a WRA called *simulation property* and *progress property*, which are obtained from weighted simulation property and weight computability, respectively, by ignoring the requirements for weights in the following sense. While weighted simulation property requires that for $t' = (p, \gamma) \to^{a}_{\varphi', \Lambda} (q, \gamma')$, $\theta_1 : \gamma$, $\theta_2 : \gamma$, $\mathrm{wt}_{t'}(\theta_1) = \mathrm{wt}_{t'}(\theta_2)$, simulation property only requires that if a switch by t' exists from $((p, \gamma), \theta_1)$ for some $d \in D$, then a switch by t' also exists from $((p, \gamma), \theta_2)$ for some $d' \in D$. While weight computability requires that $\mathrm{wt}(t')$ can be computed in polynomial time, progress property requires that it is decidable in polynomial time whether there is at least one switch by t'.

Let $\mathcal{A} = (Q, Q_0, T, Q_f, \mathrm{wt})$ be an arbitrary k-WRA and $\mathcal{A}' = (Q', Q'_0, T', Q'_f, \mathrm{wt}')$ be the normal form k-WRA obtained from $\mathcal{A}$. We say that $\mathcal{A}'$ has *simulation property* if every $t' = (p, \gamma) \to^{a}_{\varphi', \Lambda} (q, \gamma') \in T'$ satisfies the following condition: for every $\theta, \theta' \in \Theta_k$ of type γ and $d \in D$ such that $(\theta, d) \models \varphi'$, there exists a data value $d' \in D$ that satisfies $(\theta', d') \models \varphi'$. We say that $\mathcal{A}'$ has *progress property* if for a given $t' = (p, \gamma) \to^{a}_{\varphi', \Lambda} (q, \gamma')$ it is decidable in polynomial time of $\|\mathcal{A}\|$ whether there exist $\theta \in \Theta_k$ and $d \in D$ such that $\theta : \gamma$ and $(\theta, d) \models \varphi'$.

Simulation property implies that if a transition t' can be applied to an ID $((p, \gamma), \theta)$ with a data value d, then for any other ID $((p, \gamma), \theta')$, there exists a data value d' such that t' can be applied to $((p, \gamma), \theta')$ with d'.

Simulation property is a necessary but not sufficient condition of weighted simulation property by the following reason. Assume that the infimum of the weights of the switches from an ID $c = ((p, \gamma), \theta)$ is the same as the infimum of those from another ID $c' = ((p, \gamma), \theta')$. If there is a switch from c, then the infimum is not ∞ and by weighted simulation property, there is a switch from c' also. Hence, simulation property holds. The inverse does not always hold because simulation property does not require anything about weights. Progress property is also a necessary but not sufficient condition of weight computability. However, the inverse direction holds if $\mathcal{A}$ (and $\mathcal{A}'$) is an RWRA.

Lemma 5. *Let $\mathcal{A}' = (Q', Q'_0, T', Q'_f, (\mathrm{wtt}', \mathrm{wtd}'))$ be the normal form k-RWRA of a k-RWRA $\mathcal{A} = (Q, Q_0, T, Q_f, (\mathrm{wtt}, \mathrm{wtd}))$ and assume that $\mathcal{A}'$ has simulation property and progress property. Assume further that* wtt *and* wtd *can be computed in polynomial time of the size of the arguments of these functions. Then, $\mathcal{A}'$ has weighted simulation property and weight computability when regarded as a normal form WRA.*

Proof. Simulation property guarantees that for a transition $t' \in T'$, assignments θ, θ' with same register type γ and a data value d, if there is a switch $((p, \gamma), \theta) \vdash_{t',d} c$, then there is a switch $((p, \gamma), \theta') \vdash_{t',d'} c'$ for some data value d'. Since $\mathcal{A}'$ is in normal form, $\Phi_{\mathcal{R}}(\theta(j), d) = \Phi_{\mathcal{R}}(\theta'(j), d')$ for every $j \in [k]$ by the guard of t'. Hence, $\mathrm{wt}(((p, \gamma), \theta) \vdash_{t',d} c) = \mathrm{wt}(((p, \gamma), \theta') \vdash_{t',d'} c')$ by the definition of the weight of a switch of RWRA. Therefore, weighted simulation holds.

From the above discussion, $\mathrm{wt}(t') = \mathrm{wt}(((p, \gamma), \theta) \vdash_{t',d} c)$ if there exist some $\theta \in \Theta_k$ of type γ and some $d \in D$ such that a switch $((p, \gamma), \theta) \vdash_{t',d} c$ exists; otherwise, $\mathrm{wt}(t') = \infty$. We can decide whether $\mathrm{wt}(t') = \infty$ or not in polynomial time of $\|\mathcal{A}\|$ by progress property of $\mathcal{A}'$.

When $\mathrm{wt}(t') \neq \infty$, $\mathrm{wt}(t') = \prod_{j=1}^{k} \mathrm{wtd}'(t', j)(\Phi_{\mathcal{R}}(\theta(j), d)) \cdot \mathrm{wtt}'(t')$ for the above-mentioned θ and d. Assume that $t' = s_{t,\gamma,\varphi'}$ for some $t \in T$. From the above discussion again, $\Phi_{\mathcal{R}}(\theta(j), d)$ is determined by the guard expression φ' of t' regardless of θ and d; more precisely, $\Phi_{\mathcal{R}}(\theta(j), d)$ equals a tuple of bits that represents the selection of α^R_j and δ^R in φ' for each $R \in \mathcal{R}$. This tuple of bits can be extracted from φ' in linear time of $\|\varphi'\|$, which equals $\|\varphi\| + (2k+1)|\mathcal{R}|$ for the guard expression φ of t. By the definition of normal form, $\mathrm{wtt}'(t') = \mathrm{wtt}(t)$ and $\mathrm{wtd}'(t', j) = \mathrm{wtd}(t, j)$, and thus we can calculate $\mathrm{wt}(t')$ in polynomial time of $\|\mathcal{A}\|$, i.e., weight computability holds. □

By Lemma 5, our method of solving the optimal run problem for WRA in the previous section can be applied to an RWRA with simulation and progress properties.

Theorem 6. *Assume that we are given a normal form k-RWRA $\mathcal{A}_r = (Q_r, Q_{0r}, T_r, Q_{fr}, (\mathrm{wtt}_r, \mathrm{wtd}_r))$ over Σ, $\mathbb{D}$ and $\mathcal{S}_{\mathrm{trpc}}$ that has simulation property and progress property. From $\mathcal{A}_r$, we can construct a normal form k-WRA $\mathcal{A} = (Q_r, Q_{0r}, T_r, Q_{fr}, (\mathrm{wtt}, \mathrm{wtd}))$ over Σ, $\mathbb{D}$ and $\mathcal{S}_{\mathrm{trpc}}$ such that $\mathcal{A}$ has weighted simulation property and weight computability, and the optimal weights of $\mathcal{A}_r$ and $\mathcal{A}$ are the same.*

Proof. We construct $\mathcal{A}$ as follows: The base RA of $\mathcal{A}$ is the same as that of $\mathcal{A}_r$. The weight function wtt of $\mathcal{A}$ is also the same as wtt_r of $\mathcal{A}_r$. The weight function $\mathrm{wtd}(t_r, j)$ of $\mathcal{A}$ can be defined as follows. For a given $t_r \in T_r$ and $j \in [k]$,

$$\mathrm{wtd}(t_r, j)(\theta(j), d) = \mathrm{wtd}_r(t_r, j)(\Phi_{\mathcal{R}}(\theta(j), d))$$

for $\theta \in \Theta_k$ and $d \in D$. $\mathcal{A}$ has weighted simulation property and weight computability by Lemma 5. By the definition of $\mathcal{A}$, the optimal weights of $\mathcal{A}_r$ and $\mathcal{A}$ are the same. □

Note that the inverse of Theorem 6 does not hold. Assume that $\mathcal{A}$ is a normal form WRA that has weighted simulation property. For any transition $t = (p, \gamma) \rightarrow^{a}_{\varphi', \Lambda} (q, \gamma')$ in $\mathcal{A}$ and assignments θ, θ' of type γ, the infimum of the weights of switches from $((p, \gamma), \theta)$ is the same as the infimum of those from $((p, \gamma), \theta')$ by weighted simulation property. However, there may exist switches $((p, \gamma), \theta) \vdash_{t,d} c$ and $((p, \gamma), \theta') \vdash_{t,d'} c'$ such that at least one of them does not contribute to the infimum and

$$\mathrm{wt}(((p, \gamma), \theta) \vdash_{t,d} c) \neq \mathrm{wt}(((p, \gamma), \theta') \vdash_{t,d'} c').$$

In such a case, we cannot construct a normal form RWRA equivalent to $\mathcal{A}$ because the weight function wtd_r of a normal form RWRA must satisfy that for any t, $j \in [k]$,

$$\mathrm{wtd}_r(t, j)(\Phi_{\mathcal{R}}(\theta(j), d)) = \mathrm{wtd}_r(t, j)(\Phi_{\mathcal{R}}(\theta'(j), d'))$$

for the above mentioned θ, θ', d, d'. Hence the above inequation never holds.

In the next section, we show the optimal run problem for weighted timed automata (WTA) as an example of the application of our method. The optimal run problem for WTA can be regarded as the optimal run problem for RWRA because a clock region of WTA can be represented by a register type and the weight of a transition is determined by the region when we consider the optimal run.

7. Weighted timed automata

Weighted timed automata (WTA) are an extension of timed automata (TA) by introducing the weight to TA. We first take a glance at the original definition of weighted timed automaton of [4]. A k-clock weighted timed automaton (abbreviated as k-WTA) over a finite alphabet Σ is $\mathcal{T} = (Q, Q_0, T, Q_f, (\mathrm{wtt}, \mathrm{wtd}))$ where

- Q is a finite set of states,
- $Q_0, Q_f \subseteq Q$ are sets of initial and finial states, respectively,
- T is a finite set of state transitions of the form $q \rightarrow^{a}_{\varphi, \Lambda} q'$ where $q, q' \in Q$, $a \in \Sigma$, $\Lambda \subseteq [k]$ (called a clock reset) and φ is a clock constraint defined by $\varphi := tt \mid x_i \bowtie c \mid \varphi \wedge \varphi \mid \neg\varphi$ $(i \in [k],\ c \in \mathbb{N},\ \bowtie \in \{<, =, >\})$,
- $\mathrm{wtt} : T \to \mathbb{R}_{\geq 0}$, $\mathrm{wtd} : Q \to (\mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0})$ such that for each $q \in Q$ there is a constant $w_q \in \mathbb{N}$ satisfying $\mathrm{wtd}(q)(d) = w_q \cdot d$.

For $\theta \in \Theta_k$ and $d \in \mathbb{R}_{\geq 0}$, we define the assignment $(\theta + d) \in \Theta_k$ by $(\theta + d)(i) = \theta(i) + d$ for each $i \in [k]$. For a transition $q \rightarrow^{a}_{\varphi, \Lambda} q' \in T$, an assignment $\theta \in \Theta_k$, a data value $d \in \mathbb{R}_{\geq 0}$, we call $(q, \theta) \vdash_{t,d} (q', (\theta + d)[\Lambda \leftarrow 0])$ a switch (with delay d) if $\theta + d \models \varphi$. A run, the weights of a switch, a run and a word of $\mathcal{T}$, the language and the series recognized by $\mathcal{T}$ are defined in the same way as those of WRA.

In this section, we redefine WTA as a subclass of WRA based on Lemma 5.1 of [5] that every k-WTA can be simulated by a $(k + 1)$-WRA by using one extra register to keep the current time instant (in particular, a clock reset can be simulated by loading the current time to the corresponding register). An input data word $w = (a_1, d_1)(a_2, d_2) \ldots (a_n, d_n)$ to a WTA means a_i occurs at time instant d_i $(i \in [n])$. In every switch, an input data value is loaded to the last register x_{k+1} so that x_{k+1} remembers when the latest symbol a_i occurred. The guard formula of every transition requires that an input data value is always not less than x_{k+1} to guarantee that $d_1 \leq d_2 \leq \ldots \leq d_n$.

For a binary relation $\bowtie$ over $\mathbb{R}_{\geq 0}$ and $c \in \mathbb{N}$, let $\bowtie c$ be the binary relation defined as $\bowtie c = \{(r, r') \mid r, r' \in \mathbb{R}_{\geq 0},\ r' - r \bowtie c\}$. Note that $(\theta, d) \models x_i^{\bowtie c}$ means $d - \theta(i) \bowtie c$, not $\theta(i) - d \bowtie c$. We let the data structure $\mathbb{D}^{\mathrm{timed}} = \langle \mathbb{R}_{\geq 0}, \{\bowtie c \mid \bowtie \in \{<, =, >\}, c \in \mathbb{N}\} \rangle$ with the initial value $\perp = 0$.

Definition 9 *([5]).* A k-clock weighted timed automaton (abbreviated as k-WTA) over Σ is a $(k + 1)$-WRA $\mathcal{A}^{\mathrm{timed}} = (Q, Q_0, T, Q_f, (\mathrm{wtt}, \mathrm{wtd}))$ over Σ, $\mathbb{D}^{\mathrm{timed}}$ and $\mathcal{S}_{\mathrm{trpc}}$ where

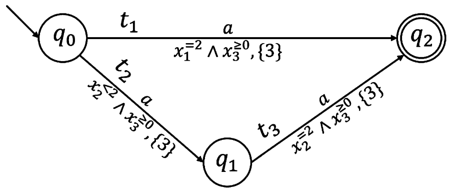

q_0
q_1
q_2
t_1
a
$x_1^{=2} \wedge x_3^{\geq 0}, \{3\}$
t_2
a
$x_2^{<2} \wedge x_3^{\geq 0}, \{3\}$
t_3
a
$x_2^{=2} \wedge x_3^{\geq 0}, \{3\}$

Fig. 3. WTA $\mathcal{A}_1^{\text{timed}}$.

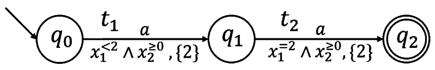

q_0
t_1
a
$x_1^{<2} \wedge x_2^{\geq 0}, \{2\}$
q_1
t_2
a
$x_1^{=2} \wedge x_2^{\geq 0}, \{2\}$
q_2

Fig. 4. WTA $\mathcal{A}_2^{\text{timed}}$.

- $A_b^{\text{timed}} = (Q, Q_0, T, Q_f)$ is a $(k+1)$-RA (called the base k-TA of $\mathcal{A}^{\text{timed}}$) such that for each transition $q \rightarrow^{a}_{\varphi,\Lambda} q' \in T$, $\varphi = \varphi' \wedge x_{k+1}^{\geq 0}$ for some $\varphi' \in F_k$,
- wtt is a function from T to $\mathbb{N}$, and
- for each $q \in Q$, a constant natural number $w_q \in \mathbb{N}$ is specified and for each transition $t = q \rightarrow^{a}_{\varphi,\Lambda} q' \in T$ and $d, d' \in \mathbb{R}_{\geq 0}$,

$$\mathrm{wtd}(t, j)(d, d') = 0 \qquad (j \in [k]),$$
$$\mathrm{wtd}(t, k+1)(d, d') = w_q \cdot (d' - d).$$

$L(A_b^{\text{timed}})$ is the timed language recognized by A_b^{timed} and $[\![\mathcal{A}^{\text{timed}}]\!]$ is the timed series recognized by $\mathcal{A}^{\text{timed}}$. □

By the above definition, the weight of a switch $(q, \theta) \vdash_{t,d} (q', \theta')$ is $\mathrm{wtt}(t) + \mathrm{wtd}(t, k+1)(\theta(k+1), d) = \mathrm{wtt}(t) + w_q(d - \theta(k+1))$. Intuitively, wtt represents the cost of executing t and $w_q(d - \theta(k+1))$ is the cost of time consumption at state q. (Remember that $\theta(k+1)$ is the time at which the latest event occurred.) As in the case of k-WRA, we define the optimal run problem for k-WTA as follows.

Definition 10 *(The optimal run problem).* Input: k-WTA $\mathcal{A}^{\text{timed}}$
Output: The infimum of $\{\mathrm{wt}(\rho) \mid \exists w \in (\Sigma \times \mathbb{R}_{\geq 0})^+.\ \rho \in \mathrm{Run}_{\mathcal{A}^{\text{timed}}}(w)\}$

Example 5 *([4]).* Let $\mathcal{A}_1^{\text{timed}}$ be a 2-WTA shown in Fig. 3 where $w_{q_0} = 3$, $w_{q_1} = 1$, $w_{q_2} = 0$, $\mathrm{wtt}(t_j) = 1$ $(j \in [3])$. Let $A_{1,b}^{\text{timed}}$ be the base 2-TA of $\mathcal{A}_1^{\text{timed}}$. Then, $L(A_{1,b}^{\text{timed}}) = \{(a, 2)\} \cup \{(a, d)(a, 2) \mid 0 \leq d < 2\}$. $\rho_1 \in \mathrm{Run}_{\mathcal{A}_1^{\text{timed}}}((a, 2))$ is unique and $\mathrm{wt}(\rho_1) = \mathrm{wtd}(t_1, 3)(0, 2) + \mathrm{wtt}(t_1) = 3 \cdot 2 + 1 = 7$. For each $w_d = (a, d)(a, 2)$ where $0 \leq d < 2$, $\rho_d \in \mathrm{Run}_{\mathcal{A}_1^{\text{timed}}}(w_d)$ is unique and $\mathrm{wt}(\rho_d) = \mathrm{wtd}(t_2, 3)(0, d) + \mathrm{wtt}(t_2) + \mathrm{wtd}(t_3, 3)(d, 2) + \mathrm{wtt}(t_3) = 3d + 1 + (2 - d) + 1 = 4 + 2d$. We have $\inf\{\mathrm{wt}(\rho) \mid \exists w \in (\Sigma \times \mathbb{R}_{\geq 0})^+.\ \rho \in \mathrm{Run}_{\mathcal{A}_1^{\text{timed}}}(w)\} = 4$.

Example 6 *([4]).* Let $\mathcal{A}_2^{\text{timed}}$ be a 1-WTA shown in Fig. 4 where $w_{q_0} = 1$, $w_{q_1} = 2$, $w_{q_2} = 0$, $\mathrm{wtt}(t_1) = \mathrm{wtt}(t_2) = 1$. Let $A_{2,b}^{\text{timed}}$ be the base 1-TA of $\mathcal{A}_2^{\text{timed}}$. Then, $L(A_{2,b}^{\text{timed}}) = \{(a, 2 - \xi)(a, 2) \mid 0 < \xi \leq 2\}$. For each $w_\xi = (a, 2 - \xi)(a, 2)$ where $0 < \xi \leq 2$, $\rho_\xi \in \mathrm{Run}_{\mathcal{A}_2^{\text{timed}}}(w_\xi)$ is unique and

$$\mathrm{wt}(\rho_\xi) = \mathrm{wtd}(t_1, 2)(0, 2 - \xi) + \mathrm{wtt}(t_1) + \mathrm{wtd}(t_2, 2)(2 - \xi, 2) + \mathrm{wtt}(t_2)$$
$$= (1 \cdot (2 - \xi)) + 1 + (2 \cdot \xi) + 1 = 4 + \xi.$$

Hence, $\inf\{\mathrm{wt}(\rho) \mid \exists w \in (\Sigma \times \mathbb{R}_{\geq 0})^+.\ \rho \in \mathrm{Run}_{\mathcal{A}_2^{\text{timed}}}(w)\} = 4$. □

In [4], an algorithm that solves the optimal run problem for WTA is proposed by extending the region construction for TA. Region construction is a well-known method to divide the infinite set of IDs of TA into a finite set of regions where two IDs in a same region are indistinguishable (or bisimilar) with respect to any transition and time progress. In [4], a sub-region is defined as a refinement of a region by distinguishing $x < y$ and $x \lesssim y$ where the distance of x and y is large in the former case while the distance is very (arbitrarily) small in the latter case. This distinction is needed because it may happen that there is no $\rho_{\min} \in \mathrm{Run}_{\mathcal{A}}(w)$ such that $\mathrm{wt}(\rho_{\min}) \leq \mathrm{wt}(\rho)$ for every $\rho \in \mathrm{Run}_{\mathcal{A}}(w)$, but the infimum of $\{\mathrm{wt}(\rho) \mid \rho \in \mathrm{Run}_{\mathcal{A}}(w)\}$ exists.

An edge-weighted directed graph, called the sub-region graph G is constructed from a given WTA and a minimum weight path of G is computed by any existing graph algorithm, which corresponds to a solution to the optimal run problem for the WTA.

The method proposed in this paper can also compute an optimal run of a WTA by distinguishing $<$ and $\lesssim$, which was first proved in [4].

Proposition 1. *[4] The optimal run problem for k-WTA $\mathcal{A}^{\mathrm{timed}} = (Q, Q_0, T, Q_f, (\mathrm{wtt}, \mathrm{wtd}))$ over Σ is solvable in time exponential in k, $c_{\max}$ and polynomial in $|Q|$, $|\Sigma|$ where $c_{\max}$ is the largest natural number appearing in the guard formula of a transition in T.*

Proof. Let $\mathcal{R}^{\mathrm{BL}}$ be the collection of relations $\{\bowtie c \mid \bowtie \in \{\lesssim, =, \gtrsim\}, c \in \mathbb{N}, c \le c_{\max}\} \backslash \{\lesssim 0\}$. We redefine the data structure for WTA as $\mathbb{D}^{\mathrm{timed,BL}} = \langle \mathbb{R}_{\ge 0}, \mathcal{R}^{\mathrm{BL}} \rangle$. A boundary region is a region specified by at least one constraint using $=$ and no constraints using $\lesssim$ or $\gtrsim$. A limit region is a region specified by at least one constraint using $\lesssim$ or $\gtrsim$.[1] Since the guard formula of any transition of WTA is a linear constraint on the contents of registers, it suffices to consider only the boundary regions and limit regions to compute the solution of the optimal run problem for WTA by the basic property of linear programming. Though the detailed discussion on linear programming is beyond the scope of this paper, we briefly explain the reason why it is enough to consider only boundary and limit regions. (Please see [4] for the details.) Let $t_i = q_{i-1} \to^{a_i}_{\varphi_i, \Lambda_i} q_i$ be transitions for $i \in [n]$ and $\rho = (q_0, \theta_0) \vdash_{t_1, d_1} \cdots \vdash_{t_n, d_n} (q_n, \theta_n)$ be a run. We want to minimize the following object function among possible combinations of data values $d_1, \dots, d_n$ appearing in the above run ρ:

$$\begin{aligned} \mathrm{wt}(\rho) &= \sum_{i=1}^{n} (\mathrm{wtt}(t_i) + \sum_{j=1}^{k+1} \mathrm{wtd}(t_i, j)(\theta_{i-1}(j), d_i)) \\ &= \sum_{i=1}^{n} (\mathrm{wtt}(t_i) + w_{q_{i-1}} \cdot (d_i - \theta_{i-1}(k+1)) \end{aligned} \tag{1}$$

The constraint of this optimization problem for a run ρ is the conjunction of the guard formula φ_i of t_i ($i \in [n]$), which is a Boolean combination of atomic formulas of the form

$$d_i - \theta_{i-1}(j) \bowtie c_{ij} \ \ (i \in [n], j \in [k], c_{ij} \in \mathbb{N}). \tag{2}$$

Note that $\theta_0(j) = \perp$ and $\theta_i(j)$ is equal to either $\theta_{i-1}(j)$ or d_i. Therefore, both of the object function (1) and the constraint (2) are linear combinations of $d_1, \dots, d_n$ and it suffices to consider the boundary and limit regions to obtain the optimal value for given $t_1, \dots, t_n$.

This implies weighted simulation and weight computability if we replace every $<$ and $>$ with $\lesssim$ and $\gtrsim$, respectively, and use $\mathbb{D}^{\mathrm{timed,BL}}$ instead of $\mathbb{D}^{\mathrm{timed}}$. Weight computability holds because $(d_i - \theta_{i-1}(k+1))$ is a constant value for a boundary region and is a value that is arbitrary close to a constant value for a limit region where the constant can be directly obtained from the constraint corresponding to these boundary or limit regions. $\square$

Example 7. Let us revisit Example 6. First, we replace every $<$ and $>$ with $\lesssim$ and $\gtrsim$, respectively, and consider its normal form. Since $c_{\max} = 2$, $\mathcal{R}^{\mathrm{BL}} = \{=0, \gtrsim 0, \lesssim 1, =1, \gtrsim 1, \lesssim 2, =2, \gtrsim 2\}$. After simplifications by using properties of the total order on $\mathbb{N}$, we have the following eight register types to be considered in this example: $\gamma_1 : x_2 - x_1 = 0$, $\gamma_2 : x_2 - x_1 \gtrsim 0$, $\gamma_3 : x_2 - x_1 \lesssim 1$, $\gamma_4 : x_2 - x_1 = 1$, $\gamma_5 : x_2 - x_1 \gtrsim 1$, $\gamma_6 : x_2 - x_1 \lesssim 2$, $\gamma_7 : x_2 - x_1 = 2$, $\gamma_8 : x_2 - x_1 \gtrsim 2$. Note that by the above specification, $\gamma_m(2,1)(R)$ and $\gamma_m(i,i)(R)$ ($m \in [8]$, $i \in [2]$, $R \in \mathcal{R}^{\mathrm{BL}}$) are uniquely determined and not described. $\mathcal{A}_2^{\mathrm{timed}}$ is transformed to $\mathcal{A}'_2 = (\{(q_i, \gamma_j) \mid i \in \{0,1,2\},\ \ j \in [8]\}, \{(q_0, \gamma_1)\}, T', \{(q_2, \gamma_7)\}, (\mathrm{wtt}', \mathrm{wtd}'))$ where T' consists of the following transitions:

$$\begin{aligned} &(q_0, \gamma_1) \to^{a}_{x_1^{=0}, \{2\}} (q_1, \gamma_1), \ \ (q_0, \gamma_1) \to^{a}_{x_1^{\gtrsim 0}, \{2\}} (q_1, \gamma_2), \\ &(q_0, \gamma_1) \to^{a}_{x_1^{\lesssim 1}, \{2\}} (q_1, \gamma_3), \ \ (q_0, \gamma_1) \to^{a}_{x_1^{=1}, \{2\}} (q_1, \gamma_4), \\ &(q_0, \gamma_1) \to^{a}_{x_1^{\gtrsim 1}, \{2\}} (q_1, \gamma_5), \ \ (q_0, \gamma_1) \to^{a}_{x_1^{\lesssim 2}, \{2\}} (q_1, \gamma_6), \\ &(q_1, \gamma_j) \to^{a}_{x_1^{=2}, \{2\}} (q_2, \gamma_7) \quad (j \in [6]) \end{aligned}$$

and wtt′, wtd′ are defined accordingly. Note that $(\theta, d) \models \mathrm{in}^{=0} \wedge \neg \mathrm{in}^{R} \wedge \neg \mathrm{in}^{R^{-1}}$ for $R \in \mathcal{R}^{\mathrm{BL}} \backslash \{=0\}$ and an input data value is always loaded to x_2 (the previous data in x_2 is overwritten), and hence constraints on x_2 and an input data value are not needed in the guard formulas. We have the following six kinds of runs, each of which corresponds to one of the above six transitions from (q_0, γ) followed by the last transition, which have the following weights:

[1] Boundary regions and limit regions are also known as corner-point regions in [11].

$$\begin{aligned}
\mathrm{wt}(\rho_1) &= 1\cdot 0+1+2\cdot 2+1 &&= 6,\\
\mathrm{wt}(\rho_2) &= 1\cdot \xi+1+2\cdot(2-\xi)+1 &&= 6-\xi,\\
\mathrm{wt}(\rho_3) &= 1\cdot(1-\xi)+1+2\cdot(1+\xi)+1 &&= 5+\xi,\\
\mathrm{wt}(\rho_4) &= 1\cdot 1+1+2\cdot 1+1 &&= 5,\\
\mathrm{wt}(\rho_5) &= 1\cdot(1+\xi)+1+2\cdot(1-\xi)+1 &&= 5-\xi,\\
\mathrm{wt}(\rho_6) &= 1\cdot(2-\xi)+1+2\cdot\xi+1 &&= 4+\xi
\end{aligned}$$

for small $\xi > 0$. Hence, the solution of the optimal run problem for this example is 4, which is realized by ρ_6 by $\xi \to 0$. □

8. Conclusion

In this paper, we discussed the optimal run problem for weighted register automata (WRA). We first introduced register type to WRA and provided a transformation from a given WRA into a normal form such that the register types before and after each transition are uniquely determined. Because the decision problem related to the optimal run problem is undecidable, we proposed a sufficient condition called weighted simulation and weight computability for the problem to become decidable. Then, we introduced the subclass of WRA, whose weight is determined only by whether the binary relations between the content of the register and the input data value hold or not. Lastly, we illustrated computing the optimal run of weighted timed automata as an example. Investigating the problem for semirings other than the tropical reals is an interesting future study.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The authors thank the two reviewers for their insightful comments on the earlier versions of the paper. This work was supported by JSPS KAKENHI Grant Number JP19H04083.

References

[1] S. Almagor, M. Cadilhac, F. Mazowiecki, G.A. Pérez, Weak cost register automata are still powerful, in: Developments in Language Theory - 22nd International Conference, DLT 2018, 2018, pp. 83–95.
[2] R. Alur, L. D'Antoni, J.V. Deshmukh, M. Raghothaman, Y. Yuan, Regular functions and cost register automata, in: 28th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2013, 2013, pp. 13–22.
[3] R. Alur, D.L. Dill, A theory of timed automata, Theor. Comput. Sci. 126 (2) (1994) 183–235, https://doi.org/10.1016/0304-3975(94)90010-8.
[4] R. Alur, S. La Torre, G.J. Pappas, Optimal paths in weighted timed automata, in: Hybrid Systems: Computation and Control, 4th International Workshop, HSCC 2001, 2001, pp. 49–62.
[5] P. Babari, M. Droste, V. Perevoshchikov, Weighted register automata and weighted logic on data words, in: Theoretical Aspects of Computing - ICTAC 2016 - 13th International Colloquium, 2016, pp. 370–384.
[6] P. Babari, M. Droste, V. Perevoshchikov, Weighted register automata and weighted logic on data words, Theor. Comput. Sci. 744 (2018) 3–21, https://doi.org/10.1016/j.tcs.2018.01.004.
[7] G. Behrmann, A. Fehnker, T. Hune, K.G. Larsen, P. Pettersson, J. Romijn, F.W. Vaandrager, Minimum-cost reachability for priced timed automata, in: Hybrid Systems: Computation and Control, 4th International Workshop, HSCC 2001, 2001, pp. 147–161.
[8] M. Bojańczyk, C. David, A. Muscholl, T. Schwentick, L. Segoufin, Two-variable logic on data words, ACM Trans. Comput. Log. 12 (4) (Jul 2011) 27:1–27:26, https://doi.org/10.1145/1970398.1970403.
[9] M. Bojańczyk, B. Klin, S. Lasota, Automata theory in nominal sets, Log. Methods Comput. Sci. 10 (3) (Aug 2014), https://doi.org/10.2168/LMCS-10(3:4)2014.
[10] P. Bouyer, A logical characterization of data languages, Inf. Process. Lett. 84 (2) (Oct 2002) 75–85, https://doi.org/10.1016/S0020-0190(02)00229-6.
[11] P. Bouyer, E. Brinksma, K.G. Larsen, Staying alive as cheaply as possible, in: Hybrid Systems: Computation and Control, 7th International Workshop, HSCC 2004, in: Lecture Notes in Computer Science, vol. 2993, Springer, 2004, pp. 203–218.
[12] E.Y. Cheng, M. Kaminski, Context-free languages over infinite alphabets, Acta Inform. 35 (3) (Mar 1998) 245–267, https://doi.org/10.1007/s002360050120.
[13] S. Demri, R. Lazić, LTL with the freeze quantifier and register automata, ACM Trans. Comput. Log. 10 (3) (Apr 2009) 16:1–16:30, https://doi.org/10.1145/1507244.1507246.
[14] S. Demri, R. Lazić, D. Nowak, On the freeze quantifier in constraint LTL: decidability and complexity, Inf. Comput. 205 (1) (Jan 2007) 2–24, https://doi.org/10.1016/j.ic.2006.08.003.
[15] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman, 1979, pp. 167–170.
[16] M. Kaminski, N. Francez, Finite-memory automata, Theor. Comput. Sci. 134 (2) (1994) 329–363, https://doi.org/10.1016/0304-3975(94)90242-9.
[17] L. Libkin, T. Tan, D. Vrgoč, Regular expressions for data words, J. Comput. Syst. Sci. 81 (7) (2015) 1278–1297, https://doi.org/10.1016/j.jcss.2015.03.005.
[18] L. Libkin, D. Vrgoč, Regular path queries on graphs with data, in: 15th International Conference on Database Theory, ICDT '12, 2012, pp. 74–85.
[19] J. Lygeros, C. Tomlin, S. Sastry, Controllers for reachability specifications for hybrid systems, Automatica 35 (3) (1999) 349–370, https://doi.org/10.1016/S0005-1098(98)00193-9.
[20] F. Neven, T. Schwentick, V. Vianu, Finite state machines for strings over infinite alphabets, ACM Trans. Comput. Log. 5 (3) (Jul 2004) 403–435, https://doi.org/10.1145/1013560.1013562.
[21] H. Sakamoto, D. Ikeda, Intractability of decision problems for finite-memory automata, Theor. Comput. Sci. 231 (2) (2000) 297–308, https://doi.org/10.1016/S0304-3975(99)00105-X.

[22] H. Seki, R. Yoshimura, Y. Takata, Optimal run problem for weighted register automata, in: 16th International Colloquium on Theoretical Aspects of Computing, ICTAC 2019, in: LNCS, vol. 11884, 2019, pp. 91–110.
[23] R. Senda, Y. Takata, H. Seki, Complexity results on register context-free grammars and register tree automata, in: 15th International Colloquium on Theoretical Aspects of Computing, ICTAC 2018, in: LNCS, vol. 11187, 2018, pp. 415–434.
[24] R. Senda, Y. Takata, H. Seki, Generalized register context-free grammars, in: Language and Automata Theory and Applications, LATA 2019, in: LNCS, vol. 11417, 2019, pp. 259–271.