Optimal Run Problem for Weighted Register Automata

Hiroyuki Seki[1(✉)], Reo Yoshimura[1], and Yoshiaki Takata[2]

[1] Nagoya University, Nagoya, Japan
seki@i.nagoya-u.ac.jp, yoshimura.reo@sqlab.jp
[2] Kochi University of Technology, Kami, Japan
takata.yoshiaki@kochi-tech.ac.jp

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 optimum 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 [12] 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 has been analyzed [11,18]. Also, [13] 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

R. M. Hierons and M. Mosbah (Eds.): ICTAC 2019, LNCS 11884, pp. 91–110, 2019.
https://doi.org/10.1007/978-3-030-32505-3_6

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), a data word in a systematic way. Closure properties of the data series recognized by WRA are discussed and an MSO logical counterpart of WRA is proposed and studied in depth in [5]. However, decidability and complexity of basic problems on 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 [4,15]. 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 [20] as an abstraction 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 a 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 state 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 proposed in [20]. 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.

Finally, we discuss the optimal run problem for weighted timed automata (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 constrains of TA are linear. If we restrict the register types to those corresponding to boundary regions and limit regions, weighted simulation and weight computability hold where the directed graph constructed in our paper corresponds to the subregion graph in [4].

Related Work. Register automata (RA) were proposed by Kaminsky and Francez [12] 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 [11,18]. In [10], 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 [10,19,20].

As extensions of finite automata other than RA, data automata [9], pebble automata (PA) [16] and nominal automata (NA) [8] are known. Libkin and Vrgoč [14] 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 [17].

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 TA, 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 [15].

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 subclass 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 $\varphi := tt \mid x_i^R \mid x_i^{R^{-1}} \mid \mathrm{in}^R \mid \varphi \wedge \varphi \mid \neg\varphi \quad (i \in [k], R \in \mathcal{R})$. 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 $ff \equiv \neg tt$, $\varphi_1 \vee \varphi_2 \equiv \neg(\neg\varphi_1 \wedge \neg\varphi_2)$.

Definition 1 ([12,13]). *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 $\mathrm{label}(t)$ the second component a of t. The description length of a k-RA $A = (Q, Q_0, T, Q_f)$ is defined as $\|A\| = |Q| + |T| \max\{(\log |Q| + k) + \|\varphi\| \mid q \to^a_{\varphi,\Lambda} q' \in T\}$, where $\|\varphi\|$ is the description length of φ, defined in a usual way.

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 $\perp$ ($\perp \in D$). The 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 the 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 $\mathrm{label}(\rho) = (\mathrm{label}(t_1), d_1) \ldots (\mathrm{label}(t_n), d_n)$ and ρ is called a run of $\mathrm{label}(\rho)$ 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 $L = L(A)$.

Example 1. Let $\Sigma = \{a\}$, $\mathcal{R} = \{<, =, >\}$. An example of 2-RA A_1 is shown in Fig. 1 where $\perp = 0$. 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 $L(A_1) = \{(a, d_1) \ldots (a, d_n) \in (D \times \Sigma)^+ \mid i, j \in [n], i < j, d_i = d_j$ and for $k = i + 1, ..., j - 1, d_i \neq d_k\}$.

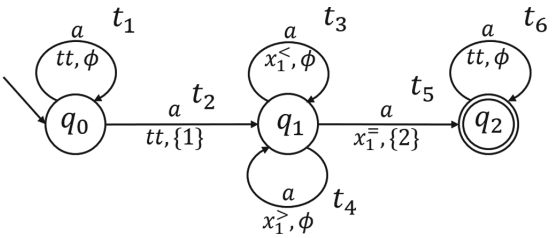

Fig. 1. RA A_1

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 $\mathrm{wtd} : (T \times [k]) \to ((D \times D) \to S)$.* □

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

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

We assume that there are constants $W_1, W_2 \in \mathbb{R}_{\geq 0}$ such that for any $t \in T$, $\mathrm{wtt}(t)$ can be computed in W_1 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 W_2 time. We define the description length of a k-WRA $\mathcal{A}$ as $\|\mathcal{A}\| = \|A_b\|$ where A_b is the base RA of $\mathcal{A}$.[1]

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 $\mathcal{R}_{\mathrm{trpc}} = (\mathbb{R}_{\geq 0} \cup \{\infty\}, \min, +, \infty, 0)$, 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 $d_{i+1}, \cdots, d_{j-1}$ 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 $k = i+1, ..., j-1, d_i \neq d_k\}$.

3 Decision Problems

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

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 $\|\mathcal{A}\| + |w|$.

[1] We do not include the size of the weight part because of the assumption that the computation of the weights of a single transition and a single register can be done in constant time.

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 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). By the assumption on complexity of computing weights of WRA, $\mathrm{wt}(c \vdash_{t,d} c')$ can be computed in $O(W_1 + W_2 k)$ time. Thus, for any run $\rho \in \mathrm{Run}_{\mathcal{A}}(w)$, the weight $\mathrm{wt}(\rho)$ can be computed in $O((W_1 + W_2 k)|w|)$ 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 an RA. Hence the run weight computation problem is NP-hard because the membership problem for RA is NP-complete [13]. □

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 #SAT: How many different variable assignments will satisfy a given general boolean 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 $\Sigma, \langle D, \mathcal{R}\rangle, \mathcal{N}$ 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 $Q = \{q_i \mid i \in [n]\} \cup \{q_{k,l} \mid k \in [m], l \in [2]\} \cup \{q'_k \mid k \in [m]\} \cup \{q_\mathrm{r}, q_f\}$, $Q_0 = \{q_1\}$ 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.

$$q_1 \to^a_{tt,\{1\}} q_2,\ q_1 \to^a_{tt,\emptyset} q_2,\ \ldots,\ q_n \to^a_{tt,\{n\}} q_{1,1},\ q_n \to^a_{tt,\emptyset} q_{1,1}.$$

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

$$\begin{aligned}
&q_{k,1} \to^a_{x_i^=,\emptyset} q'_k, \quad q_{k,1} \to^a_{x_i^\neq,\emptyset} q_{k,2} && \text{if } y_{k,1} = z_i,\\
&q_{k,1} \to^a_{x_i^\neq,\emptyset} q'_k, \quad q_{k,1} \to^a_{x_i^=,\emptyset} q_{k,2}, && \text{if } y_{k,1} = \overline{z_i},\\
&q_{k,2} \to^a_{x_i^=,\emptyset} q_{k+1,1}, \quad q_{k,2} \to^a_{x_i^\neq,\emptyset} q_\mathrm{r}, && \text{if } y_{k,2} = z_i,\\
&q_{k,2} \to^a_{x_i^\neq,\emptyset} q_{k+1,1}, \quad q_{k,2} \to^a_{x_i^=,\emptyset} q_\mathrm{r}, && \text{if } y_{k,2} = \overline{z_i},\\
&q'_k \to^a_{tt,\emptyset} q_{k+1,1}
\end{aligned}$$

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. □

Lemma 2. *The data word weight computation problem for k-WRA is PSPACE-solvable. When the semiring is $\mathcal{N}$,* wtt *is bounded and* $\mathrm{wtd}(t,j)(d_1,d_2) = 1$ *for every $t \in T$ and $j \in [k]$ and $d_1, d_2 \in D$ 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. From a k-WRA $\mathcal{A} = (Q, Q_0, T, Q_f, (\mathrm{wtt}, \mathrm{wtd}))$ over $\Sigma, \langle D, \mathcal{R}\rangle, \mathcal{N}$, we construct k-WRA $\mathcal{A}' = (Q', Q'_0, T', Q'_f, (\mathrm{wtt}', \mathrm{wtd}'))$ such that $[\![\mathcal{A}']\!] = [\![\mathcal{A}]\!]$ by dividing each run in $\mathcal{A}$ into several runs whose weights are 1. For $M = \max\{\mathrm{wtt}(t) \mid t \in T\}$, we introduce new states $q_1, \ldots, q_M$ not included in Q. Note that M is a constant by the assumption. The set of the states of $\mathcal{A}'$ is $Q' = \{(q, q_i) \mid q \in Q, i \in [M]\}$, and the set of transitions is $T' = \{(q, q_i) \to^a_{\varphi,\Lambda} (q', q_j) \mid t = q \to^a_{\varphi,\Lambda} q' \in T, \mathrm{wtt}(t) = m, i \in [M], j \in [m]\}$. Also, let $Q'_0 = \{(q_I, q_1) \mid q_I \in Q_0\}$, and $Q'_f = \{(q_f, q_i) \mid q_f \in Q_f, i \in [M]\}$. 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 $\bot = (\varepsilon, \varepsilon) \in D$ 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 $(x, y)\mathrm{EQ}(x', y') \Leftrightarrow (x = y)$.
- $T = \{q_0 \to^a_{x_1^{R_i}, \{1\}} q,\ q \to^a_{x_1^{R_i}, \{1\}} q \mid i \in [m]\} \cup \{q \to^a_{x_1^{\mathrm{EQ}}, \emptyset} q_f\}$.
- $\mathrm{wtt}(t) = \mathrm{wtd}(t, 1)(d_1, d_2) = 1$ 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 Sect. 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 ([20]). *A register type (of k registers) for a data structure $\langle D, \mathcal{R}\rangle$ is an arbitrary function $\gamma : ([k] \times [k]) \to (\mathcal{R} \to \mathbb{B})$. 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 $\forall R \in \mathcal{R}\big[(\forall i, j \in [k].\gamma_0(i,j)(R) = 1) \Leftrightarrow ((\bot, \bot) \in R)\big]$. $Q'_f = Q_f \times \Gamma_k$. T' is the smallest set of transitions $t' = (p, \gamma) \to^a_{\varphi', \Lambda} (q, \gamma')$ where

$$t = p \to^a_{\varphi,\Lambda} q \in T,\ \gamma, \gamma' \in \Gamma_k,\ \varphi' = \varphi \wedge \textstyle\prod_{R\in\mathcal{R}}(\prod_{i=1}^{k} \alpha_i^R \wedge \beta_i^R) \wedge \delta^R,$$
$$\alpha_i^R \in \{x_i^R, \neg x_i^R\}, \beta_i^R \in \{x_i^{R^{-1}}, \neg x_i^{R^{-1}}\}, \delta^R \in \{\mathrm{in}^R, \neg \mathrm{in}^R\},\ \varphi' \not\equiv \mathit{ff} \text{ and}$$
$$\text{if } \theta \in \Theta_k,\ d \in D,\ \theta : \gamma,\ (\theta, d) \models \varphi',\ (p, \theta) \vdash_{t,d} (q, \theta[\Lambda \leftarrow d]) \text{ then}$$
$$\theta[\Lambda \leftarrow d] : \gamma'.$$

In the above definition, φ' says that in addition to φ, whether the contents of the i-th register and an input data value d satisfies R (resp. d is reflexive on R) is exactly determined by α_i^R and β_i^R (resp. by δ^R). Furthermore, an input data value is loaded to the registers specified by Λ when t' is applied. 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 $t = p \to^{a}_{x_1^R,\{2\}} q$ and register type γ such that $\gamma(i,j)(R) = 1$ ($(i,j) = (1,1),(1,2),(2,2)$), $\gamma(2,1)(R) = 0$. If we merge transitions whose target states are the same, then we have the following four transitions:

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

where

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

Theorem 4. *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*.

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 the initial ID to an accepting ID of a given WRA. We assume the tropical semiring $\mathcal{R}_{\mathrm{trpc}}$ (see Example 2) because by $\mathcal{R}_{\mathrm{trpc}}$ 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* $\Sigma, \langle D, \mathcal{R}\rangle, \mathcal{R}_{\mathrm{trpc}}$
Output: The infimum of $\{\mathrm{wt}(\rho) \mid \exists w \in (\Sigma \times D)^+.\rho \in \mathrm{Run}_{\mathcal{A}}(w)\}$ □

By Theorem 4 and the definition of the problem, the following property holds.

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* $\mathcal{A}$*. 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 $t' = (p, \gamma) \to^{a}_{\varphi', \Lambda} (q, \gamma') \in T'$ satisfies the following condition: for every $\theta \in \Theta_k$ such that $\theta : \gamma$, $\inf\{\mathrm{wt}(((p, \gamma), \theta) \vdash_{t',d} ((q, \gamma'), \theta[\Lambda \leftarrow d])) \mid d \in D, \theta[\Lambda \leftarrow d] : \gamma'\}$ takes a same value[2]. 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 $\|\mathcal{A}\|$. 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 because a binary relation appearing in the guard of a transition may be undecidable. 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.

These assumptions also make the following two problems related to the weight realizability problems decidable.

Definition 7 (The weight bounding problems)
Input: a k-WRA $\mathcal{A}$ *over* Σ, $\langle D, \mathcal{R}\rangle$, $\mathcal{R}_{\mathrm{trpc}}$ *and a weight* $s \in \mathbb{R}_{\geq 0}$
(The run weight bounding problem) $\exists w.\exists \rho \in \mathrm{Run}_{\mathcal{A}}(w).\mathrm{wt}(\rho) \leq s$?
(The data word weight bounding problem) $\exists w.[\![\mathcal{A}]\!](w) \leq s$?
The input size for both problems is $\|\mathcal{A}\|$.

[2] If there is no such a switch $(p, \theta) \vdash_{t',d} (q, \theta[\Lambda \leftarrow d])$ for any $d \in D$, we define the infimum as ∞.

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

Proof (PSPACE-solvability). Let $\mathcal{A} = (Q, Q_0, T, Q_f, \mathrm{wt})$ be a k-WRA over Σ, $\langle D, \mathcal{R}\rangle$ and $\mathcal{S}$ and $s \in \mathbb{R}_{\geq 0}$. Let $\mathcal{A}' = (Q', Q'_0, T', Q'_f, \mathrm{wt}')$ be the normal form k-WRA obtained from $\mathcal{A}$. By weighted simulation, the number of IDs of $\mathcal{A}'$ that must be examined is not more than $|Q'|(k+1)^k$ by a similar reason discussed in [13]. Therefore, it is enough to check whether $\mathrm{wt}(\rho) \leq s$ for every run ρ of w whose length is at most $|Q'|(k+1)^k$. By the proof of Theorem 1, computing the weight of a run can be done in polynomial time. Also, the space needed to simulate a run of an input data word of length $|Q'|(k+1)^k$ is $\log(|Q'|(k+1)^k) = k\log(k+1) + \log|Q'|$. Because $Q' = Q \times \Gamma_k$ holds, $|Q'| \in O(|Q|2^{k^2|\mathcal{R}|})$. Hence, the space complexity is $O(k\log k + \log|Q| + k^2|\mathcal{R}|)$, which is a polynomial order of k, $|Q|$ and $|\mathcal{R}|$. Consequently, 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 the given semiring value is $s = 0$, the run weight realizability 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 [13], the run weight bounding problem is PSPACE-hard.

Theorem 6. *The data word weight bounding problem of k-WRA over Σ, $\langle D, \mathcal{R}\rangle$, $\mathcal{R}_{\mathrm{trpc}}$ is PSPACE-complete if weighted simulation and weight computability hold.*

Proof. Let $\mathcal{A}$ be a given k-WRA and s be a semiring value.

(PSPACE-solvability). We only need to take the sum of the weights of all the runs of w in $\mathcal{A}$ to compute $[\![\mathcal{A}]\!](w)$, which needs only $O(1)$ additional space. Therefore, this problem can be solved in PSPACE.

(PSPACE-hardness). The run weight bounding problem is PSPACE-complete by Theorem 5, and so this problem is PSPACE-hard.

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 $\mathcal{R}_{\mathrm{trpc}}$ 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 $s_{t,\gamma,\varphi'} = (p,\gamma) \to^{a}_{\varphi',\Lambda} (q,\gamma') \in T'$ 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.

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 $\mathrm{wt}(\pi)$.

Theorem 7. *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. For a path π in G starting with the initial state and ending with a final state of $\mathcal{A}'$, there is a run ρ in $\mathcal{A}'$ such that $\mathrm{wt}(\rho) = \mathrm{wt}(\pi)$. Conversely, for a run ρ in $\mathcal{A}'$, there is a path π in G such that $\mathrm{wt}(\pi) = \mathrm{wt}(\rho)$.*

Proof. Let $\pi = e_1 e_2 \cdots e_n$ be a path in G starting with the initial state and ending with a final state of $\mathcal{A}'$ where $e_i \in E$ $(i \in [n])$. By the construction of G, for the i-th edge $e_i = (v_{i-1}, v_i, s_i, m_i)$ of π $(i \in [n])$, the third component s_i can be written as $s_i = s_{t_i,\gamma_{i-1},\varphi'_i} = (p_{i-1}, \gamma_{i-1}) \to^{a_i}_{\varphi'_i,\Lambda} (p_i, \gamma_i) \in T'$ (p_0 is the initial state and p_n is a final state) and $m_i = \mathrm{wt}(s_{t_i,\gamma_{i-1},\varphi'_i}) = \inf\{\mathrm{wt}((p_{i-1}, \theta_{i-1}) \vdash_{s_{t_i,\gamma_{i-1},\varphi'_i},d_i} (p_i, \theta_{i-1}[\Lambda \leftarrow d_i])) \mid d_i \in D\}$ for some $\theta_{i-1} \in \Theta_k$ such that $\theta_{i-1}[\Lambda \leftarrow d_i] : \gamma_i$. Note that (p_0, θ_0) is the initial ID. By weighted simulation, there is a run $\rho = (p_0, \theta'_0) \vdash_{s_{t_1,\gamma_0,\varphi'_1},d'_1} (p_1, \theta'_1) \vdash_{s_{t_2,\gamma_1,\varphi'_2},d'_2} \cdots \vdash_{s_{t_n,\gamma_{n-1},\varphi'_n},d'_n} (p_n, \theta'_n)$ of some data word $(a_1, d'_1) \cdots (a_n, d'_n)$ where $\theta'_0 = \theta_0$ and a_i is the second component of t_i. Also, it is easy to see $\mathrm{wt}(\pi) = \mathrm{wt}(\rho)$. The converse direction holds by the construction of G. □

By Theorems 4 and 7, 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 Theorem 4. In this way, we can easily reconstruct the run in $\mathcal{A}$ that provide 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 8. *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 $|V| = |Q'|$, $|E| = |T'|$.

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 $\mathcal{A}'_2$. 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:

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

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 (Fig. 2).

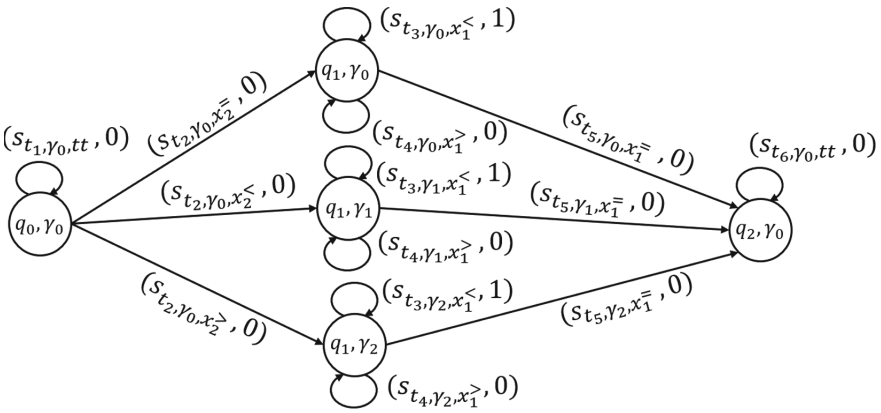

Fig. 2. The directed graph G_1 for $\mathcal{A}_2$ of Example 2.

6 Weighted Timed Automata

Weighted timed automata (WTA) are an extension of timed automata (TA) by introducing the weight to TA. In this subsection, we directly define 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 $d_1 \le d_2 \le \ldots \le d_n$.

For a binary relation $\bowtie$ over $\mathbb{R}_{\ge 0}$ and $c \in \mathbb{N}$, let $\bowtie c$ be the binary relation defined as $\bowtie c = \{(r, r') \mid r, r' \in \mathbb{R}_{\ge 0},\ r' - r \bowtie c\}$. 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 $\mathbb{D}^{\text{timed}} = \langle \mathbb{R}_{\ge 0}, \{\bowtie c \mid \bowtie \in \{<, =, >\}, c \in \mathbb{N}\}\rangle$ with the initial value $\perp = 0$.

Definition 8 ([5]). *A k-register weighted timed automaton (abbreviated as k-WTA) over Σ is a k-WRA $\mathcal{A}^{\mathrm{timed}} = (Q, Q_0, T, Q_f, (\mathrm{wtt}, \mathrm{wtd}))$ over Σ, $\mathbb{D}^{\mathrm{timed}}$ and $\mathcal{R}_{\mathrm{trpc}}$ where*

- *$A_b^{\mathrm{timed}} = (Q, Q_0, T, Q_f)$ is a k-RA (called the base k-TA of $\mathcal{A}^{\mathrm{timed}}$) such that for each transition $q \to^a_{\varphi,\Lambda} q' \in T$, $\varphi = \varphi' \wedge x_k^{\geq 0}$ 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 $t = q \to^a_{\varphi,\Lambda} q' \in T$ and $d, d' \in \mathbb{R}_{\geq 0}$,*

$$\mathrm{wtd}(t, j)(d, d') = 0 \quad (j \in [k-1]), \tag{1}$$
$$\mathrm{wtd}(t, k)(d, d') = w_q(d' - d). \tag{2}$$

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

By the above definition, the weight of a switch $(q, \theta) \vdash_{t,d} (q', \theta')$ is $\mathrm{wtt}(t) + \mathrm{wtd}(t, k)(\theta(k), d) = \mathrm{wtt}(t) + w_q(d - \theta(k))$. Intuitively, wtt represents the cost of executing t and $w_q(d - \theta(k))$ is the cost of time consumption at state q. (Remember that $\theta(k)$ 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 9 (The optimal run problem)
Input: k-WTA $\mathcal{A}^{\mathrm{timed}}$
Output: The infimum of $\{\mathrm{wt}(\rho) \mid \exists w \in (\Sigma \times \mathbb{R}_{\geq 0})^+.\ \rho \in \mathrm{Run}_{\mathcal{A}^{\mathrm{timed}}}(w)\}$

Example 5. ([4]). Let $\mathcal{A}_1^{\mathrm{timed}}$ be a 3-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 $A_{1,b}^{\mathrm{timed}}$ be the base k-TA of $\mathcal{A}_1^{\mathrm{timed}}$. Then, $L(A_{1,b}^{\mathrm{timed}}) = \{(a, 2)\} \cup \{(a, d)(a, 2) \mid 0 \leq d < 2\}$. $\rho_1 \in \mathrm{Run}_{\mathcal{A}_1^{\mathrm{timed}}}((a, 2))$ 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 $0 \leq d < 2$, $\rho_d \in \mathrm{Run}_{\mathcal{A}_1^{\mathrm{timed}}}(w_d)$ 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 $\inf\{\mathrm{wt}(\rho) \mid \exists w \in (\Sigma \times D)^+.\ \rho \in \mathrm{Run}_{\mathcal{A}_1^{\mathrm{timed}}}(w)\} = 4$.

Example 6. ([4]). Let $\mathcal{A}_2^{\mathrm{timed}}$ be a 2-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 $A_{2,b}^{\mathrm{timed}}$ be the base k-TA of $\mathcal{A}_2^{\mathrm{timed}}$. Then, $L(A_{2,b}^{\mathrm{timed}}) = \{(a, 2 - \xi)(a, 2) \mid 0 < \xi \leq 2\}$. For each $w_\xi = (a, 2 - \xi)(a, 2)$ where $0 < \xi \leq 2$, $\rho_\xi \in \mathrm{Run}_{\mathcal{A}_2^{\mathrm{timed}}}(w_\xi)$ is unique and

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

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

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

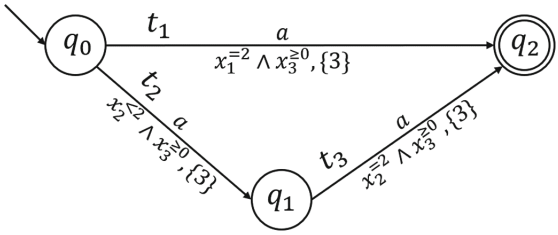

Fig. 3. WTA $\mathcal{A}_1^{\text{timed}}$

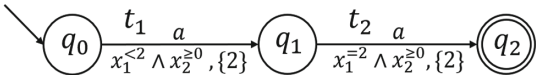

Fig. 4. WTA $\mathcal{A}_2^{\text{timed}}$

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 run that has the minimum weight but there are infinite number of runs whose weights has the infimum as shown in Example 5 (see [4] for details). 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 $\lesssim$ as follows. Let $c_{\max}$ be the largest natural number appearing in the guard formula of some transition in a given WTA and let $\mathcal{R}^{\text{BL}}$ be the collection of relations $\{\bowtie c \mid \bowtie \in \{\lesssim, =, \gtrsim\}, c \in \mathbb{N}, c \leq c_{\max}\} \backslash \{\lesssim 0\}$. We redefine the data structure for WTA as $\mathbb{D}^{\text{timed,BL}} = \langle \mathbb{R}_{\geq 0}, \mathcal{R}^{\text{BL}} \rangle$. A boundary region is a region specified by at least one constraints using $=$ and no constraints using $\lesssim$ or $\gtrsim$. A limit region is a region specified by at least one constraints using $\lesssim$ or $\gtrsim$. 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 (see [4] for example). This implies weighted simulation and weight computability if we replace every $<$ and $>$ with $\lesssim$ and $\gtrsim$, respectively, and use $\mathbb{D}^{\text{timed,BL}}$ instead of $\mathbb{D}^{\text{timed}}$.

Example 7. Let us revisit Example 6. First, we replace every $<$ and $>$ with $\lesssim$ and $\gtrsim$, respectively, and consider its normal form. Since $c_{\max} = 2$, $\mathcal{R}^{\text{BL}} = \{= 0, \gtrsim 0, \lesssim 1, = 1, \gtrsim 1, \lesssim 2, = 2, \gtrsim 2\}$. 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 $\gamma_m(i,i)(R)$ ($m \in [8]$, $i = 1, 2$, $R \in \mathcal{R}^{\mathrm{BL}}$) are uniquely determined and not described. $\mathcal{A}_2^{\mathrm{timed}}$ is transformed to $\mathcal{A}_2' = (\{(q_i, \gamma_j) \mid i = 0, 1, 2,\ j \in [8]\}, \{(q_0, \gamma_1)\}, T', \{(q_2, \gamma_7)\}, (\mathrm{wtt}', \mathrm{wtd}'))$ where T' consists of the following transitions:

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

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: $\mathrm{wt}(\rho_1) = 6$, $\mathrm{wt}(\rho_1) = 6 - \xi$, $\mathrm{wt}(\rho_1) = 5 + \xi$, $\mathrm{wt}(\rho_1) = 5$, $\mathrm{wt}(\rho_1) = 5 - \xi$, $\mathrm{wt}(\rho_1) = 4 + \xi$ for small $\xi > 0$. Hence, the solution of the optimal run problem for this example is 4, which is realized by ρ_6 by $\xi \to 0$. □

7 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. 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.

Acknowledgements. The authors thank the reviewers for providing valuable comments to the paper. This work was supported by JSPS KAKENHI Grant Number JP19H04083.

References

1. Almagor, S., Cadilhac, M., Mazowiecki, F., Pérez, G.A.: Weak cost register automata are still powerful. In: Hoshi, M., Seki, S. (eds.) DLT 2018. LNCS, vol. 11088, pp. 83–95. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98654-8_7

2. Alur, R., D'Antoni, L., Deshmukh, J.V., Raghothaman, M., Yuan, Y.: Regular functions and cost register automata. In: 28th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2013, New Orleans, 25–28 June 2013, pp. 13–22 (2013). https://doi.org/10.1109/LICS.2013.65
3. Alur, R., Dill, D.L.: A theory of timed automata. Theor. Comput. Sci. **126**(2), 183–235 (1994). https://doi.org/10.1016/0304-3975(94)90010-8
4. Alur, R., La Torre, S., Pappas, G.J.: Optimal paths in weighted timed automata. In: Di Benedetto, M.D., Sangiovanni-Vincentelli, A. (eds.) HSCC 2001. LNCS, vol. 2034, pp. 49–62. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45351-2_8
5. Babari, P., Droste, M., Perevoshchikov, V.: Weighted register automata and weighted logic on data words. In: Sampaio, A., Wang, F. (eds.) ICTAC 2016. LNCS, vol. 9965, pp. 370–384. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46750-4_21
6. Babari, P., Droste, M., Perevoshchikov, V.: Weighted register automata and weighted logic on data words. Theor. Comput. Sci. **744**, 3–21 (2018). https://doi.org/10.1016/j.tcs.2018.01.004
7. Behrmann, G., Fehnker, A., Hune, T., Larsen, K., Pettersson, P., Romijn, J., Vaandrager, F.: Minimum-cost reachability for priced time automata. In: Di Benedetto, M.D., Sangiovanni-Vincentelli, A. (eds.) HSCC 2001. LNCS, vol. 2034, pp. 147–161. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45351-2_15
8. Bojańczyk, M., Klin, B., Lasota, S.: Automata theory in nominal sets. Logical Methods Comput. Sci. **10**(3) (2014). https://doi.org/10.2168/LMCS-10(3:4)2014
9. Bouyer, P.: A logical characterization of data languages. Inf. Process. Lett. **84**(2), 75–85 (2002). https://doi.org/10.1016/S0020-0190(02)00229-6
10. Cheng, E.Y., Kaminski, M.: Context-free languages over infinite alphabets. Acta Informatica **35**(3), 245–267 (1998). https://doi.org/10.1007/s002360050120
11. Demri, S., Lazić, R.: LTL with the freeze quantifier and register automata. ACM Trans. Comput. Log. **10**(3), 16:1–16:30 (2009). https://doi.org/10.1145/1507244.1507246
12. Kaminski, M., Francez, N.: Finite-memory automata. Theoret. Comput. Sci. **134**(2), 329–363 (1994). https://doi.org/10.1016/0304-3975(94)90242-9
13. Libkin, L., Tan, T., Vrgoč, D.: Regular expressions for data words. J. Comput. Syst. Sci. **81**(7), 1278–1297 (2015). https://doi.org/10.1016/j.jcss.2015.03.005
14. Libkin, L., Vrgoč, D.: Regular path queries on graphs with data. In: 15th International Conference on Database Theory (ICDT 2012), pp. 74–85 (2012). https://doi.org/10.1145/2274576.2274585
15. Lygeros, J., Tomlin, C., Sastry, S.: Controllers for reachability specifications for hybrid systems. Automatica **35**(3), 349–370 (1999). https://doi.org/10.1016/S0005-1098(98)00193-9
16. Milo, T., Suciu, D., Vianu, V.: Typechecking for XML transformers. In: 19th ACM Symposium on Principles of Database Systems (PODS 2000), pp. 11–22 (2000). https://doi.org/10.1145/335168.335171
17. Neven, F., Schwentick, T., Vianu, V.: Finite state machines for strings over infinite alphabets. ACM Trans. Comput. Log. **5**(3), 403–435 (2004). https://doi.org/10.1145/1013560.1013562
18. Sakamoto, H., Ikeda, D.: Intractability of decision problems for finite-memory automata. Theor. Comput. Sci. **231**(2), 297–308 (2000). https://doi.org/10.1016/S0304-3975(99)00105-X

19. Senda, R., Takata, Y., Seki, H.: Complexity results on register context-free grammars and register tree automata. In: Fischer, B., Uustalu, T. (eds.) ICTAC 2018. LNCS, vol. 11187, pp. 415–434. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-02508-3_22
20. Senda, R., Takata, Y., Seki, H.: Generalized register context-free grammars. In: Martín-Vide, C., Okhotin, A., Shapira, D. (eds.) LATA 2019. LNCS, vol. 11417, pp. 259–271. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-13435-8_19