Synchronized Recursive Timed Automata

Yuya Uezato[1] and Yasuhiko Minamide[2]

[1] University of Tsukuba uezato@score.cs.tsukuba.ac.jp
[2] Tokyo Institute of Technology minamide@is.titech.ac.jp

Abstract. We present *synchronized recursive timed automata (SRTA)* that extend timed automata with a stack. Each frame of a stack is composed of rational-valued clocks, and SRTA *synchronously* increase the values of all the clocks within the stack. Our main contribution is to show that the reachability problem of SRTA is ExpTime-complete. This decidability contrasts with the undecidability for *recursive timed automata (RTA)* introduced by Trivedi and Wojtczak, and Benerecetti et al. Unlike SRTA, the frames below the top are frozen during the computation at the top frame in RTA.
Our construction of the decidability proof is based on the region abstraction for *dense timed pushdown automata (TPDA)* of Abdulla et al. to accommodate together diagonal constraints and fractional constraints of SRTA. Since SRTA can be seen as an extension of TPDA with diagonal and fractional constraints, our result enlarges the decidable class of pushdown-extensions of timed automata.

1 Introduction

The paper presents a new pushdown-extension of timed automata *synchronized recursive timed automata (SRTA)*, and we study its expressiveness and the decidability of the reachability problem. Timed automata are a model of real-time systems, and recently several pushdown-extensions of timed automata have been introduced [12, 3, 1, 8]. Among these pushdown-extensions, our formalization of SRTA has novel constraints *fractional constraints*—formulae of the form $\{x\} = 0$ and $\{x\} < \{y\}$—for checking fractional parts of clocks. These fractional constraints play important roles. First, fractional constraints enlarge the language class of (decidable) pushdown-extensions of timed automata *timed pushdown automata (TPDA)* of Abdulla et al. [1] and TPDA *with diagonal constraints* of Clemente and Lasota [8]. Indeed, we show that the following SRTA language L_{ex} cannot be recognized by any TPDA or TPDA with diagonal constraints because of lack of fractional constraints:

$$L_{\mathrm{ex}} \triangleq \{(a,t_1)(a,t_2)\dots(a,t_n)(b,t'_n)\dots(b,t'_2)(b,t'_1) : t'_i - t_i \in \mathbb{N}\}\,.$$

Next, fractional constraints are needed to achieve the theoretical result: For any SRTA, we can remove *diagonal constraints*—formulae of the form $x - y = k$—while preserving the language. Removal of diagonal constraints is one of important results in the theory of timed automata [2, 4], and recently Clemente and Lasota showed that in the context of TPDA [8].

Timed automata are a model of real-time systems that includes rational-valued clocks where a configuration $\langle q, \eta \rangle$ is a pair of a control location q and a clock valuation $\eta : X \to \mathbb{Q}^+$ from clocks to the non-negative rationals. In timed automata, *timed transitions* evolve the values of all the clocks at the same rate: $\langle q, \eta \rangle \overset{\delta}{\rightsquigarrow} \langle q, \eta + \delta \rangle$. Despite the unboundedness and denseness of rationals, the reachability problem of timed automata was shown decidable by the region abstraction technique in [2].

The two equivalent models, *recursive timed automata (RTA)* and *timed recursive state machines*, were independently introduced by Trivedi and Wojtczak [12] and Benerecetti et al. [3]. A configuration of RTA $\langle q, \langle \gamma_1, \eta_1 \rangle \dots \langle \gamma_n, \eta_n \rangle \rangle$ is a pair of a location q and a stack where each frame is a pair $\langle \gamma_i, \eta_i \rangle$ of a symbol γ_i and a valuation $\eta_i : X \to \mathbb{Q}^+$. In RTA, timed transitions evolve the values of the clocks only at the top frame: $\langle q, \langle \gamma_1, \eta_1 \rangle \dots \langle \gamma_n, \eta_n \rangle \rangle \overset{\delta}{\rightsquigarrow} \langle q, \langle \gamma_1, \eta_1 \rangle \dots \langle \gamma_n, \eta_n + \delta \rangle \rangle$. Unfortunately, the reachability problem of RTA is undecidable because RTA can simulate two-counter machines [12, 3].

Abdulla et al. introduced *dense timed pushdown automata (TPDA)* [1], and recently Clemente and Lasota extended TPDA to allow diagonal constraints. A configuration of TPDA $\langle q, \eta, \langle \gamma_1, r_1 \rangle \dots \langle \gamma_n, r_n \rangle \rangle$ is a triple of a location q, a valuation of clocks $\eta : X \to \mathbb{Q}^+$, and a timed stack where each element $\langle \gamma_i, r_i \rangle$ is a pair of a symbol γ_i and its age $r_i \in \mathbb{Q}^+$. TPDA differ from RTA in the following point: In TPDA, timed transitions evolve *synchronously* the values of all the clocks within the stack at the same rate: $\langle q, \eta, \langle \gamma_1, r_1 \rangle \dots \langle \gamma_n, r_n \rangle \rangle \overset{\delta}{\rightsquigarrow} \langle q, \eta + \delta, \langle \gamma_1, r_1 + \delta \rangle \dots \langle \gamma_n, r_n + \delta \rangle \rangle$. Surprisingly, Abdulla et al. showed the reachability problem of TPDA is decidable and ExpTime-complete [1]. To show this, they designed a region abstraction for pushdown-extensions of timed automata.

Our SRTA are described as *synchronized* RTA; thus a configuration is the same as RTA. But, timed transitions synchronously evolve the values of all the clocks within the stack: $\langle q, \langle \gamma_1, \eta_1 \rangle \dots \langle \gamma_n, \eta_n \rangle \rangle \overset{\delta}{\rightsquigarrow} \langle q, \langle \gamma_1, \eta_1 + \delta \rangle \dots \langle \gamma_n, \eta_n + \delta \rangle \rangle$. Compared to TPDA, the formalization of SRTA provides both diagonal constraints and fractional constraints. These constraints make SRTA more expressive than TPDA (with diagonal constraints). Even though SRTA extend TPDA, we show that *the reachability problem of SRTA remains* ExpTime*-complete*. Our decidability proof is separated into two stages.

At the first stage, we translate SRTA into SRTA without diagonal constraints by effectively using fractional constraints. In TPDA, Clemente and Lasota showed that TPDA with diagonal constraints collapse to TPDA with an untimed stack whose configurations are $\langle q, \eta, \gamma_1 \dots \gamma_n \rangle$ in [8]. This implies that adding diagonal constraints does not enlarge the language class. *However*, we cannot apply their untiming technique to SRTA because the above mentioned language L_{ex} requires unboundedly many clocks, and this contrasts with Clemente's result.

At the second stage, we adapt the region abstraction of Abdulla et al. [1] to show the ExpTime-completeness of the reachability problem of SRTA without diagonal constraints. Interestingly, our fractional constraints are obtained by investigating the region abstraction of Abdulla et al. [1], and thus, our construction is based on their region abstraction. We find out that Abdulla's proof structure

is essentially a *backward-forward simulation* of Lynch and Vaandrager [11], and this mixed simulation makes their proof involved. From this insight, we introduce an intermediate semantics to separate the mixed simulation into two simple simulations, and this makes entire proof easy to follow.

Concrete Valuations. The set of non-negative rationals $\mathbb{Q}^+$ is defined by: $\mathbb{Q}^+ \triangleq \{r \in \mathbb{Q} : r \geq 0\}$. For a rational $r \in \mathbb{Q}^+$, we use $\lfloor r \rfloor$ and $\{r\}$ to denote the integral and fractional part of r, respectively: e.g., $\lfloor 1.5 \rfloor = 1$ and $\{1.5\} = 0.5$.

Let X be a clock set. A function $\eta : X \to \mathbb{Q}^+$ is called a *concrete valuation* on X and we write X_V for the set of valuations on X. We define basic operations:

$$\eta[x \coloneqq r](y) \triangleq \begin{cases} r & \text{if } y = x \\ \eta(y) & \text{otherwise,} \end{cases} \quad \eta[x \coloneqq y] \triangleq \eta[x \coloneqq \eta(y)], \quad (\eta + r)(y) \triangleq \eta(y) + r,$$

where $x, y \in X$ and $r \in \mathbb{Q}^+$. The zero valuation on X is defined by: $\mathbf{0}_X(x) \triangleq 0$ for $x \in X$. For $\eta \in X_V$ and $Y \subseteq X$, we write $\eta|Y \in Y_V$ to denote the restriction of η to Y. We define the ordering $\eta \leq \eta'$ on valuations by: $\eta \leq \eta'$ if $\exists r \in \mathbb{Q}^+ . \eta' = \eta + r$.

Pushdown Systems. A pushdown system (PDS) is a triple $(Q, \Gamma, \hookrightarrow)$ where Q is a finite set of control locations, Γ is a (possibly infinite) stack alphabet, and $\hookrightarrow \subseteq (Q \times \Gamma^*) \times (Q \times \Gamma^*)$ is a set of transition rules. A configuration is a pair $\langle q, w \rangle$ of a location $q \in Q$ and a stack $w \in \Gamma^*$. The one-step transition $\langle q, w\, v \rangle \to \langle q', w\, v' \rangle$ is defined if $\langle q, v \rangle \hookrightarrow \langle q', v' \rangle$. We also write $w \to w'$ by omitting locations if the locations are irrelevant. A PDS is called finite-PDS if the set of transition rules is finite. Otherwise, it is called infinite-PDS. The reachability problem from q_{init} to q_{final} decides if $\langle q_{\text{init}}, \epsilon \rangle \to^* \langle q_{\text{final}}, w \rangle$ holds for some stack w, and the reachability problem of finite-PDS is in PTIME [7, 5, 9].

2 Synchronized Recursive Timed Automata

First, we introduce *synchronized recursive timed automata (SRTA)* where the values of all the clocks in the stack are increased *synchronously* at the same rate. Next, we study the expressiveness of SRTA by brief comparisons with recursive timed automata and timed pushdown automata. Finally, we see the overview of our decidability proof.

Clock Constraints. We write $I \in \mathbb{I}$ for an interval: $\mathbb{I} \triangleq \{(i, j), [i, j] : i, j \in \mathbb{N}\}$. Let X be a clock set. Then, the set Φ_X of clock constraints is given by:

$$\varphi ::= x \in I \mid x \bowtie y \mid \{x\} = 0 \mid \{x\} \bowtie \{y\} \mid \varphi \wedge \varphi \mid \neg\varphi$$
$$\text{where } x, y \in X,\ I \in \mathbb{I},\ \bowtie \in \{<, =, >\}.$$

For $\varphi \in \Phi_X$ and $\eta \in X_V$, we write $\eta \models \varphi$ if φ holds when clocks are replaced by values of η: e.g., $\eta \models x \in I$ if $\eta(x) \in I$, $\eta \models \{x\} = 0$ if $\{\eta(x)\} = 0$. The fractional constraints $\{x\} = 0$ and $\{x\} \bowtie \{y\}$ are novel and are used to recognize the language L_{ex} and remove diagonal constraints later on.

Definition 1 (Synchronized Recursive Timed Automata). A synchronized recursive timed automaton is a tuple $\mathfrak{S} = (Q, q_{\text{init}}, q_{\text{final}}, \Sigma, \Gamma, \mathcal{X}, \Delta)$ where Q is a finite set of control locations, q_{init} and q_{final} are the initial and accepting

locations respectively, Σ is a finite input alphabet, Γ is a finite set of stack symbols, $\mathcal{X}$ is a finite set of clocks, and $\Delta \subseteq Q \times (\Sigma \cup \{\epsilon\}) \times Op \times Q$ is a finite set of *discrete* transition rules. The operations $\tau \in Op$ are given by:

$$\tau ::= \mathsf{push}(\gamma, X) \mid \mathsf{pop}(\gamma, X) \mid x \leftarrow I \mid \mathsf{check}(\varphi)$$
$$\text{where } \gamma \in \Gamma, X \subseteq \mathcal{X}, x \in \mathcal{X}, I \in \mathbb{I}, \text{ and } \varphi \in \Phi_{\mathcal{X}}.$$

We define the standard semantics STND of SRTA as a transition system.

Definition 2 (Semantics STND). A configuration is a pair $\langle q, w \rangle$ of a location q and a stack w where each frame $\langle \gamma, \eta \rangle$ consists of a stack symbol γ and a concrete valuation $\eta \in \mathcal{X}_V$. The set of configurations of STND is $Q \times (\Gamma \times \mathcal{X}_V)^*$.

For $\tau \in Op$, we define a discrete transition $w \xrightarrow{\tau} w'$ for $w, w' \in (\Gamma \times \mathcal{X}_V)^*$ by case analysis on τ:

$$\frac{\eta_2 = \mathbf{0}_{\mathcal{X}}[X \coloneqq \eta_1]}{w\langle \gamma, \eta_1 \rangle \to w\langle \gamma, \eta_1 \rangle \langle \gamma', \eta_2 \rangle}\ \mathsf{push}(\gamma', X) \qquad \frac{r \in I \quad \eta' = \eta[x \coloneqq r]}{w\langle \gamma, \eta \rangle \to w\langle \gamma, \eta' \rangle}\ x \leftarrow I$$

$$\frac{\eta_1' = \eta_1[X \coloneqq \eta_2]}{w\langle \gamma, \eta_1 \rangle \langle \gamma', \eta_2 \rangle \to w\langle \gamma, \eta_1' \rangle}\ \mathsf{pop}(\gamma', X) \qquad \frac{\eta \models \varphi}{w\langle \gamma, \eta \rangle \to w\langle \gamma, \eta \rangle}\ \mathsf{check}(\varphi)$$

where $\eta[\{x_1, \ldots, x_n\} \coloneqq \eta'] \triangleq \eta[x_1 \coloneqq \eta'(x_1), \ldots, x_n \coloneqq \eta'(x_n)]$.

In addition to discrete transitions, we allow *timed* transitions:

$$\langle \gamma_1, \eta_1 \rangle \langle \gamma_2, \eta_2 \rangle \ldots \langle \gamma_n, \eta_n \rangle \overset{\delta}{\rightsquigarrow} \langle \gamma_1, \eta_1 + \delta \rangle \langle \gamma_2, \eta_2 + \delta \rangle \ldots \langle \gamma_n, \eta_n + \delta \rangle$$

where $\delta \in \mathbb{Q}^+$. These transitions for a stack are extended to configurations: $\langle q, w \rangle \xrightarrow{\alpha} \langle q', w' \rangle$ if $w \xrightarrow{\tau} w'$ for some $\langle q, \alpha, \tau, q' \rangle \in \Delta$ and $\langle q, w \rangle \overset{\delta}{\rightsquigarrow} \langle q, w' \rangle$ if $w \overset{\delta}{\rightsquigarrow} w'$.

Timed Languages. A run π is a finite alternating sequence of timed and discrete transitions. From a run $\pi = c_0 \overset{\delta_0}{\rightsquigarrow} c_0' \xrightarrow{\alpha_0} c_1 \overset{\delta_1}{\rightsquigarrow} c_1' \xrightarrow{\alpha_1} \cdots \overset{\delta_n}{\rightsquigarrow} c_n \xrightarrow{\alpha_n} c_n'$, we define the timed trace $\mathsf{tt}(\pi) \triangleq (\alpha_0, \delta_0)(\alpha_1, \delta_0 + \delta_1) \ldots (\alpha_n, \Sigma_{i=0}^n \delta_i) \in ((\Sigma \cup \{\epsilon\}) \times \mathbb{Q}^+)^*$ and the timed word $\mathsf{tw}(\pi) \in (\Sigma \times \mathbb{Q}^+)^*$ by removing all the (ϵ, t) pairs from $\mathsf{tt}(\pi)$. The timed language of $\mathfrak{S}$ is defined by the runs from q_{init} to q_{final}:

$$\mathcal{L}(\mathfrak{S}) \triangleq \{\mathsf{tw}(\pi) : \pi = \langle q_{\text{init}}, \langle \bot, \mathbf{0}_{\mathcal{X}} \rangle \rangle \rightsquigarrow \cdots \to \langle q_{\text{final}}, w \rangle\}.$$

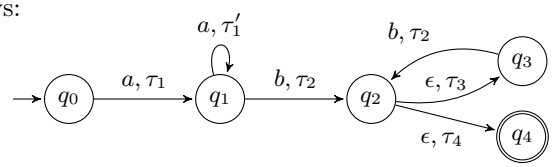
(For the initial configuration $\langle q_{\text{init}}, \langle \bot, \mathbf{0}_{\mathcal{X}} \rangle \rangle$, the special stack symbol $\bot \in \Gamma$ is needed.)

Timed Language Example. We consider the following timed language:

$$L_{\text{ex}} \triangleq \{(a, t_1)(a, t_2) \ldots (a, t_n)(b, t_n') \ldots (b, t_2')(b, t_1') : t_i' - t_i \in \mathbb{N}\}.$$

Note that if we forget the time stamps from L_{ex} then the language $\{a^n b^n : n \geq 1\}$ is a typical context-free language.

We consider a SRTA $(\{q_0, \ldots, q_4\}, q_0, q_4, \{a, b\}, \{\bot, \natural, \sharp\}, \{x\}, \Delta)$ where Δ is defined as follows:



$\tau_1 = \mathsf{push}(\natural, \emptyset)$, $\tau_1' = \mathsf{push}(\sharp, \emptyset)$, $\tau_2 = \mathsf{check}(\{x\} = 0)$, $\tau_3 = \mathsf{pop}(\sharp, \emptyset)$, $\tau_4 = \mathsf{pop}(\natural, \emptyset)$.

Let us consider the timed word $(a, 0.1)(a, 1.2)(b, 2.2)(b, 3.1) \in L_{\mathrm{ex}}$ and the following run that accepts the word:

$$\langle q_0, \langle \bot, 0\rangle\rangle \overset{0.1}{\rightsquigarrow} \overset{a}{\rightarrow} \langle q_1, \langle \bot, 0.1\rangle\langle \natural, 0\rangle\rangle \overset{1.1}{\rightsquigarrow} \overset{a}{\rightarrow} \langle q_1, \langle \bot, 1.2\rangle\langle \natural, 1.1\rangle\langle \sharp, 0\rangle\rangle \overset{1.0}{\rightsquigarrow}$$
$$\langle q_1, \langle \bot, 2.2\rangle\langle \natural, 2.1\rangle\langle \sharp, 1\rangle\rangle \xrightarrow[\mathsf{check}(\{x\}=0)]{b} \langle q_2, \langle \bot, 2.2\rangle\langle \natural, 2.1\rangle\langle \sharp, 1\rangle\rangle \overset{0.5}{\rightsquigarrow} \overset{\epsilon}{\rightarrow}$$
$$\langle q_3, \langle \bot, 2.7\rangle\langle \natural, 2.6\rangle\rangle \overset{0.4}{\rightsquigarrow} \overset{b}{\rightarrow} \langle q_2, \langle \bot, 3.1\rangle\langle \natural, 3\rangle\rangle \overset{0}{\rightsquigarrow} \overset{\epsilon}{\rightarrow} \langle q_4, \langle \bot, 3.1\rangle\rangle.$$

The action τ_2 (i.e., $\mathsf{check}(\{x\} = 0)$) checks if the fractional part of $t'_i - t_i$ is zero, hence it excludes a run such that $\langle q_0, \langle \bot, 0\rangle\rangle \overset{0.1}{\rightsquigarrow} \overset{a}{\rightarrow} \overset{0.2}{\rightsquigarrow} \langle q_1, \langle \bot, 0.3\rangle\langle \natural, 0.2\rangle\rangle \overset{b}{\not\rightarrow} \langle q_2, \langle \bot, 0.3\rangle\langle \natural, 0.2\rangle\rangle$.

Simulating Diagonal Constraints $x - y \bowtie k$. In SRTA, every update $x \leftarrow I$ is bounded because I is an interval. This enables us to encode diagonal constraints of the form $x - y \bowtie k$ where $k \in \mathbb{Z}$.

Let us see an idea of encoding the constraint $x - y \bowtie 1$. We prepare an auxiliary clock y_{+1} for denoting $y + 1$ and check $x \bowtie y_{+1}$ instead of $x - y \bowtie 1$. In order to keep $y + 1 = y_{+1}$, when we update $y \leftarrow (i, i + 1)$, we also execute $y_{+1} \leftarrow (i + 1, i + 2)$ and $\mathsf{check}(\{y\} = \{y_{+1}\})$. In the case of $y \leftarrow [i, i]$, we do $y_{+1} \leftarrow [i+1, i+1]$. Since any updates $y \leftarrow I$ can be decomposed into the forms of $y \leftarrow (i, i + 1)$ and $y \leftarrow [j, j]$ by nondeterminism of SRTA, our decidability result extends even if we consider general diagonal constraints of the form $x - y \bowtie k$.

Then, this decidability result is analogous to one of Bouyer et al. for timed automata with bounded updates and diagonal constraints [6].

Compare to Recursive Timed Automata. The formulation of recursive timed automata (RTA) [12, 3] differs from SRTA in timed transitions: RTA increase only the top of a stack: $\langle q, \langle \gamma_1, \eta_1\rangle \dots \langle \gamma_n, \eta_n\rangle\rangle \overset{\delta}{\rightsquigarrow} \langle q, \langle \gamma_1, \eta_1\rangle \dots \langle \gamma_n, \eta_n + \delta\rangle\rangle$ where $\delta \in \mathbb{Q}^+$. The difference of timed transitions between SRTA and RTA is crucial because RTA can simulate two-counter machines [12, 3] by effectively using the timed transitions.

Krishna et al. considered the subset of RTA called $\mathrm{RTA}_{\mathrm{RN}}$ in [10] and showed that the reachability problem of $\mathrm{RTA}_{\mathrm{RN}}$ is decidable. $\mathrm{RTA}_{\mathrm{RN}}$ are subsumed by our SRTA because $\mathrm{RTA}_{\mathrm{RN}}$ are SRTA without diagonal and fractional constraints. They adapted Abdulla's construction and proof to $\mathrm{RTA}_{\mathrm{RN}}$ by closely following the details. On the other hand, we give a simpler proof by restructuring Abdulla's proof.

Compare to Timed Pushdown Automata. Timed pushdown automata of Abdulla et al. [1] are a pushdown extension of timed automata. Clemente and Lasota [8] equipped TPDA with diagonal constraints and showed that the expressiveness of TPDA with diagonal constraints is equivalent to that of TPDA with respect to languages.

Let us briefly see the formulation of Clemente and Lasota. The constraints $\psi \in \Psi_X$ in their system are given as follows:

$$\psi ::= x \bowtie k \mid x - y \bowtie k \mid \psi \wedge \psi \quad \text{where } x, y \in X \text{ and } k \in \mathbb{Z}.$$

Since there are no fractional constraints ($\{x\} = 0$ or $\{x\} \bowtie \{y\}$), we cannot inspect the fractional parts of clocks. A TPDA is a tuple $(Q, q_{\mathrm{init}}, q_{\mathrm{final}}, \Sigma, \Gamma, \mathcal{X}, \Delta)$

and a configuration $\langle q, \mathtt{X}, \langle \gamma_1, r_1 \rangle \langle \gamma_2, r_2 \rangle \dots \langle \gamma_n, r_n \rangle \rangle$ is a triple of a location q, a (global) valuation $\mathtt{X}$ on $\mathcal{X}$, and a stack where $\langle \gamma_i, r_i \rangle \in \Gamma \times \mathbb{Q}^+$. There are four kinds of discrete operations for Δ:

$\mathsf{push}(\gamma) : \langle p, \mathtt{X}, w \rangle \to \langle q, \mathtt{X}, w\langle \gamma, 0 \rangle \rangle$, $\quad \mathsf{reset}(x) : \langle p, \mathtt{X}, w \rangle \to \langle q, \mathtt{X}[x \coloneqq 0], w \rangle$,
$\mathsf{pop}(\gamma, \psi') : \langle p, \mathtt{X}, w\langle \gamma, r \rangle \rangle \to \langle q, \mathtt{X}, w \rangle$ if $\mathtt{X} \uplus \{z \mapsto r\} \models \psi'$ where $\psi' \in \Psi_{\mathcal{X} \uplus \{z\}}$,
$\mathsf{check}(\psi) : \langle p, \mathtt{X}, w \rangle \to \langle q, \mathtt{X}, w \rangle$ if $\mathtt{X} \models \psi$ where $\psi \in \Psi_{\mathcal{X}}$.

Since a valuation $\mathtt{X}$ of TPDA is simulated by using the value-copying mechanism of $\mathbf{push}(\gamma, \mathcal{X})$ and $\mathbf{pop}(\gamma, \mathcal{X})$ in SRTA, we obtain the following result.

Theorem 1. *For a TPDA $\mathcal{T} = (Q, q_{init}, q_{final}, \Sigma, \Gamma, \mathcal{X}, \Delta)$, we can build a SRTA $\mathfrak{S} = (Q', q'_{init}, q'_{final}, \Sigma, \Gamma \cup \{\bot\}, \mathcal{X} \cup \{z, \mathtt{x}\}, \Delta')$ such that $\mathcal{L}(\mathcal{T}) = \mathcal{L}(\mathfrak{S})$.*

Proof (Sketch). A push transition $\langle p, \mathtt{X}, \epsilon \rangle \xrightarrow{\alpha, \mathsf{push}(\gamma)} \langle q, \mathtt{X}, \langle \gamma, 0 \rangle \rangle$ is simulated by $\langle p, \langle \bot, \eta \rangle \rangle \xrightarrow{\alpha, \mathbf{push}(\gamma, \mathcal{X})} \langle q, \langle \bot, \eta \rangle \langle \gamma, \eta' \rangle \rangle$ where $\mathtt{X}(x) = \eta(x)$ for all $x \in \mathcal{X}$.

To simulate a pop transition $\langle p, \mathtt{X}, \langle \gamma, r \rangle \rangle \xrightarrow{\alpha, \mathsf{pop}(\gamma, \psi')} \langle q, \mathtt{X}, \epsilon \rangle$ atomically in $\mathfrak{S}$, we use the extra clock $\mathtt{x}$ as follows:

$$\langle p, \langle \bot, \eta \rangle \langle \gamma, \eta' \rangle \rangle \xrightarrow{\epsilon, \mathtt{x} \leftarrow [0,0]} \langle p', \langle \bot, \eta \rangle \langle \gamma, \eta'[\mathtt{x} \coloneqq 0] \rangle \rangle \overset{\delta_1}{\rightsquigarrow} \xrightarrow{\epsilon, \mathsf{check}(\psi')} \overset{\delta_2}{\rightsquigarrow} \xrightarrow{\alpha, \mathbf{pop}(\gamma, \mathcal{X} \cup \{\mathtt{x}\})}$$
$$\langle q', \langle \bot, \eta'' \rangle \rangle \overset{\delta_3}{\rightsquigarrow} \xrightarrow{\epsilon, \mathsf{check}(\mathtt{x} \in [0,0])} \langle q, \langle \bot, \eta'' \rangle \rangle$$

where $\mathtt{X}(x) = \eta'(x)$ for $x \in \mathcal{X}$ and $\eta'(z) = r$. By using the clock $\mathtt{x}$ as a stopwatch, we ensure the atomicity of these transitions and $\delta_1 = \delta_2 = \delta_3 = 0.0$. □

Furthermore SRTA have the major advantage over TPDA, namely we can inspect the fractional parts of clocks by fractional constraints. Indeed, the language class of SRTA is strictly larger than TPDA with diagonal constraints.

Theorem 2. *The above timed language L_{ex} cannot be recognized by TPDA with diagonal constraints.*

Intuitively, unboundedly many clocks are needed to keep the exact fractional values to recognize the language L_{ex}. The proof of this theorem can be found in the long version of the paper [13].

This suggests that fractional constraints play a crucial role in pushdown extensions of timed automata. Interestingly, the constraints are obtained by studying Abdulla's proof [1]. Unlike standard regions of timed automata, Abdulla's regions carry the fractional part ordering of clocks even their values are beyond the bound that the maximal constant appears in interval constraints.

As an overview of the rest of the paper, we see the proof of our main theorem.

Main Theorem. *The reachability problem of SRTA, which decides if there is a run from $\langle q_{init}, \langle \bot, \mathbf{0} \rangle \rangle$ to $\langle q_{final}, w \rangle$ for some stack w, is* ExpTime*-complete.*

Proof. The ExpTime-hardness is shown from the result of Abdulla et al. that the reachability problem of TPDA is ExpTime-hard [1] and the above Theorem 1.

Next, to show the reachability problem is decidable and in ExpTime, we build the finite-PDS semantics Digi through Section 3 and Section 4:

$$\text{SRTA} \xrightarrow{\text{Thm 3}} \text{SRTA without comparisons } x \bowtie y \xrightarrow{\text{Thm 4}} \text{Digi}$$

where each step preserves the reachability and especially Theorem 3 states that we can safely remove clock comparisons while preserving languages of SRTA.

The obtained finite-PDS DIGI is basically equivalent to the *symbolic pushdown automaton* of Abdulla et al. [1] and they proved the reachability problem of that is in EXPTIME by using the PTIME algorithm for the reachability problem of finite-PDS [7, 5, 9]. Hence, the reachability problem of SRTA is in EXPTIME. □

3 Language-Preserving Removal of Comparison Constraints

We show that comparison constraints $x \bowtie y$ can be removed from SRTA without losing its expressiveness. Namely, from a given SRTA $\mathfrak{S}$, we construct a SRTA $\mathfrak{S}'$ without comparison constraints such that $\mathcal{L}(\mathfrak{S}) = \mathcal{L}(\mathfrak{S}')$.

We say that a SRTA $\mathfrak{S} = (Q, q_{\text{init}}, q_{\text{final}}, \Sigma, \Gamma, \mathcal{X}, \Delta)$ is M-bounded if $\mathsf{M} \geq j$ holds for any intervals (i, j) and $[i, j]$ in Δ. As a running example of this section, we consider the following run of a 2-bounded SRTA:

$$\eta_1 \xrightarrow{\mathsf{push}(\{u,z\})} \eta_1\,\eta_2 \xrightarrow{y \leftarrow [1,2]} \eta_1\,\eta_2' \overset{1.3}{\rightsquigarrow} \eta_1'\,\eta_2'' \xrightarrow{u \leftarrow [0,1]} \eta_1'\,\eta_2''' \xrightarrow{\mathsf{pop}(\{u,y,z\})} \eta_3$$

where

$$\begin{aligned}
&\eta_1 = \{x_1 \mapsto 0.1; x_2, u \mapsto 1.2; z \mapsto 2.6\}, && \eta_1' = \{x_1 \mapsto 1.4; x_2, u \mapsto 2.5; z \mapsto 3.9\},\\
&\eta_2 = \{y \mapsto 0.0; u \mapsto 1.2; z \mapsto 2.6\}, && \eta_2' = \{u \mapsto 1.2; y \mapsto 1.4; z \mapsto 2.6\},\\
&\eta_2'' = \{u \mapsto 2.5; y \mapsto 2.7; z \mapsto 3.9\}, && \eta_2''' = \{u \mapsto 0.3; y \mapsto 2.7; z \mapsto 3.9\}.
\end{aligned}$$

From the definition, $\eta_3 = \{u \mapsto 0.3; x_1 \mapsto 1.4; x_2 \mapsto 2.5; y \mapsto 2.7; z \mapsto 3.9\}$. For the sake of readability, we only write relevant clocks for an explanation as above. Also, we omit zero timed transitions $\overset{0.0}{\rightsquigarrow}$, locations, input alphabet, and stack symbols.

Our basic idea is to encode the liner order between clocks into a stack symbol: e.g., the linear order of η_1 is represented symbolically by $x_1 < \lceil x_2, u \rfloor < z$ as a stack symbol. Hence, the above run is encoded as follows:

$$\nu_1 \xrightarrow{\mathsf{push}(\{u,z\})} \nu_1\,\nu_2 \xrightarrow{y \leftarrow [1,2]} \nu_1\,\nu_2' \overset{1.3}{\rightsquigarrow} \nu_1'\,\nu_2'' \xrightarrow{u \leftarrow [0,1]} \nu_1'\,\nu_2''' \xrightarrow{\mathsf{pop}(\{u,y,z\})} \nu_3$$

where

$$\begin{aligned}
&\nu_1 = \big(\eta_1, x_1 < \lceil x_2, u \rfloor < z\big), \quad \nu_2 = \big(\eta_2, y < u < z\big), \quad \nu_2' = \big(\eta_2', u < y < z\big),\\
&\nu_1' = \big(\eta_1', x_1 < \lceil x_2, u \rfloor < z\big), \quad \nu_2'' = \big(\eta_2'', u < y < z\big), \quad \nu_2''' = \big(\eta_2''', u < y < z\big).
\end{aligned}$$

For this encoding we do the following calculation at each step:

1. At $\mathsf{push}(\{u, z\})$, we extract the order of u and z in ν_1 and pass $u < z$ to ν_2.
2. At update $y \leftarrow [1, 2]$, first we actually perform $y \leftarrow [1, 2]$ and set $y \mapsto 1.4$, so we obtain $(\eta_2', y < u < z)$. Next, we reconstruct the correct order $u < y$ of y and u in η_2'. *Since our updates $y \leftarrow [i, j]$ or $y \leftarrow (i, j)$ are M-bounded (i.e., $j \leq 2$)*, we can calculate the correct order by using M-bounded interval constraints and fractional constraints. After $\mathsf{check}(u \in (1, 2))$, $\mathsf{check}(y \in (1, 2))$, and $\mathsf{check}(\{u\} < \{y\})$, we find out $u < y$.
3. At time transition $\overset{1.3}{\rightsquigarrow}$, we do not need to modify any orderings.
4. At update $u \leftarrow [0, 1]$, we also perform $u \leftarrow [0, 1]$ first and next we reconstruct the correct ordering of η_2'''.

Finally, we consider the $\mathbf{pop}(\{u,y,z\})$ transition. As above, first we actually perform $\mathbf{pop}(\{u,y,z\})$ and obtain $(\eta_3, \nu_1' : (x_1 < \lfloor x_2, u \rfloor < z)\ \&\ \nu_2''' : (u < y < z))$. However we have no ways to determine the correct ordering $u < x_1 < x_2 < y < z$ because both $\eta_3(x_2)$ and $\eta_3(y)$ are larger than $\mathsf{M} = 2$ and $x_2 < y$ cannot be understood with 2-bounded interval constraints. Of course, if we take $\mathsf{M} = 3$ then this matter is solved. But, this ad-hoc solution fails when $\nu_1\,\nu_2' \overset{2.3}{\rightsquigarrow} \nu_1'\,\nu_2''$.

To solve this, we introduce *auxiliary clocks* $\dot{\mathsf{C}}_i$ and $\dot{\mathsf{c}}_j$ as follows:

$$\lambda_1 \xrightarrow{\mathbf{push}(\{u,z\})} \lambda_1\,\lambda_2 \xrightarrow{y \leftarrow [1,2]} \lambda_1\,\lambda_2' \overset{1.3}{\rightsquigarrow} \lambda_1'\,\lambda_2'' \xrightarrow{u \leftarrow [0,1]} \lambda_1'\,\lambda_2''' \xrightarrow{\mathbf{pop}(\{u,y,z\})} \lambda_3$$

where

$$\begin{aligned}
\lambda_1 &= \big(\eta_1 \ \cup\ \{\dot{\mathsf{C}}_0 \mapsto 0.0; \dot{\mathsf{C}}_1 \mapsto 1.0; \dot{\mathsf{C}}_2 \mapsto 2.0\},\ \dot{\mathsf{C}}_0 < x_1 < \dot{\mathsf{C}}_1 < \lfloor x_2, u \rfloor < \dot{\mathsf{C}}_2 < z\big),\\
\lambda_2 &= \big(\eta_2 \ \cup\ \{\dot{\mathsf{c}}_0 \mapsto 0.0; \dot{\mathsf{c}}_1 \mapsto 1.0; \dot{\mathsf{c}}_2 \mapsto 2.0\},\ \lfloor y, \dot{\mathsf{c}}_0 \rfloor < \dot{\mathsf{c}}_1 < u < \dot{\mathsf{c}}_2 < z\big),\\
\lambda_2' &= \big(\eta_2' \ \cup\ \{\dot{\mathsf{c}}_0 \mapsto 0.0; \dot{\mathsf{c}}_1 \mapsto 1.0; \dot{\mathsf{c}}_2 \mapsto 2.0\},\ \dot{\mathsf{c}}_0 < \dot{\mathsf{c}}_1 < u < y < \dot{\mathsf{c}}_2 < z\big),\\
\lambda_1' &= \big(\eta_1' \ \cup\ \{\dot{\mathsf{C}}_0 \mapsto 1.3; \dot{\mathsf{C}}_1 \mapsto 2.3; \dot{\mathsf{C}}_2 \mapsto 3.3\},\ \dot{\mathsf{C}}_0 < x_1 < \dot{\mathsf{C}}_1 < \lfloor x_2, u \rfloor < \dot{\mathsf{C}}_2 < z\big),\\
\lambda_2'' &= \big(\eta_2'' \ \cup\ \{\dot{\mathsf{c}}_0 \mapsto 1.3; \dot{\mathsf{c}}_1 \mapsto 2.3; \dot{\mathsf{c}}_2 \mapsto 3.3\},\ \dot{\mathsf{c}}_0 < \dot{\mathsf{c}}_1 < u < y < \dot{\mathsf{c}}_2 < z\big),\\
\lambda_2''' &= \big(\eta_2''' \cup\ \{\dot{\mathsf{c}}_0 \mapsto 1.3; \dot{\mathsf{c}}_1 \mapsto 2.3; \dot{\mathsf{c}}_2 \mapsto 3.3\},\ u < \dot{\mathsf{c}}_0 < \dot{\mathsf{c}}_1 < y < \dot{\mathsf{c}}_2 < z\big).
\end{aligned}$$

When taking $\mathbf{push}(\{u,z\})$, we set the clocks $\dot{\mathsf{C}}_i$ by $\dot{\mathsf{C}}_i \leftarrow [i,i]$ in the next frame λ_1 and also set the clocks $\dot{\mathsf{c}}_i$ by $\dot{\mathsf{c}}_i \leftarrow [i,i]$ in the top frame λ_2. Hence, $\dot{\mathsf{C}}_i$ of λ_1 and $\dot{\mathsf{c}}_i$ of λ_2 have the same values. We require two kinds of the auxiliary clocks $\dot{\mathsf{C}}$ and $\dot{\mathsf{c}}$ because if we push a new frame on top of the current frame λ_2, we also set $\dot{\mathsf{C}}_i \leftarrow [i,i]$ in λ_2. To compute the correct ordering in λ_3 at $\mathbf{pop}(\{u,y,z\})$, the auxiliary clocks $\dot{\mathsf{C}}$ of λ_1' and $\dot{\mathsf{c}}$ of λ_2''' behave as *separators* as follows.

Determine $x_2 < y$. With the auxiliary clocks, we determine $x_2 < y$. Performing $\mathbf{pop}(\{u,y,z\})$ makes $(\eta_3 \cup \{\dot{\mathsf{C}}_0 \mapsto 1.3, \dot{\mathsf{C}}_1 \mapsto 2.3, \dot{\mathsf{C}}_2 \mapsto 3.3\}, \lambda_1' : \boldsymbol{o}_1\ \&\ \lambda_2''' : \boldsymbol{o}_2)$ where $\boldsymbol{o}_1 = \dot{\mathsf{C}}_0 < x_1 < \dot{\mathsf{C}}_1 < \lfloor x_2, u \rfloor < \dot{\mathsf{C}}_2 < z$ and $\boldsymbol{o}_2 = u < \dot{\mathsf{c}}_0 < \dot{\mathsf{c}}_1 < y < \dot{\mathsf{c}}_2 < z$.

It is easily understood that $\dot{\mathsf{C}}_1 < x_2 < \dot{\mathsf{C}}_2$ and $\dot{\mathsf{C}}_1 < y < \dot{\mathsf{C}}_2$ from $\boldsymbol{o}_1$ and $\boldsymbol{o}_2$. We also obtain $\{\dot{\mathsf{C}}_1\} < \{x_2\} < \{y\}$ by using fractional constraints because of $\{\dot{\mathsf{C}}_1\} = 0.3$, $\{x_2\} = 0.5$, and $\{y\} = 0.7$. Then $\dot{\mathsf{C}}_1 < x_2 < y < \dot{\mathsf{C}}_2$ follows from: 1) the fractional part ordering $\{\dot{\mathsf{C}}_1\} < \{x_2\} < \{y\}$, 2) x_2 and y are in between $\dot{\mathsf{C}}_1$ and $\dot{\mathsf{C}}_2$, and 3) *the fact* $\dot{\mathsf{C}}_1 + 1.0 = \dot{\mathsf{C}}_2$ obtained by the construction.

Treating clocks $< \dot{\mathsf{C}}_0$ or $> \dot{\mathsf{C}}_{\mathsf{M}}$. From the above argument, in general, we can reconstruct the correct ordering of clocks between $\dot{\mathsf{C}}_0$ and $\dot{\mathsf{C}}_{\mathsf{M}}$. Here we consider the other clocks: 1) clocks that are smaller than $\dot{\mathsf{C}}_0$ and 2) clocks that are larger than $\dot{\mathsf{C}}_{\mathsf{M}}$.

(1) We consider $u < \dot{\mathsf{c}}_0$ in $\boldsymbol{o}_2$. This implies that u *was updated after* $\mathbf{push}(\{u,z\})$ because the only way to make a clock smaller than $\dot{\mathsf{c}}_0$ is updating. Hence, we take $x < \dot{\mathsf{C}}_0$ in λ_3 if $x < \dot{\mathsf{c}}_0$ in $\boldsymbol{o}_2$. And also we take $x \lhd x'$ in λ_3 if $x \lhd x' < \dot{\mathsf{c}}_0$ in $\boldsymbol{o}_2$ where $\lhd \in \{<, =\}$. As the result, we obtain $u < \dot{\mathsf{C}}_0$ in λ_3.

(2) We consider $\dot{\mathsf{c}}_{\mathsf{M}} < z$ in $\boldsymbol{o}_2$. This states that z *was copied by* $\mathbf{push}(\{u,z\})$ *and never updated* because our updates are bounded by $\mathsf{M}=2$ and the bounded updates cannot make a clock larger than $\dot{\mathsf{c}}_{\mathsf{M}}$. Thus, we take $\dot{\mathsf{C}}_{\mathsf{M}} < x$ in λ_3 if $\dot{\mathsf{c}}_{\mathsf{M}} < x$ in $\boldsymbol{o}_2$ and $x \lhd x'$ in λ_3 if $\dot{\mathsf{C}}_{\mathsf{M}} < x \lhd x'$ in $\boldsymbol{o}_1$. As the result, we obtain $\dot{\mathsf{C}}_{\mathsf{M}} < z$ in λ_3. Finally, we find out $u < \dot{\mathsf{C}}_0 < x_1 < \dot{\mathsf{C}}_1 < x_2 < y < \dot{\mathsf{C}}_2 < z$ and it reflects the correct ordering $u < x_1 < x_2 < y < z$ in η_3.

In general, when performing $\mathbf{pop}(X)$ for $(\eta_1, \boldsymbol{o}_1)$ and $(\eta_2, \boldsymbol{o}_2)$, we build the ordering $\boldsymbol{o}_3$ of $\eta_3\,(= \eta_1[X \coloneqq \eta_2])$ in the following steps from 1. to 4.:

We write $\mathcal{X}_{\dot{\mathsf{C}}}$ for $\mathcal{X} \cup \{\dot{\mathsf{C}}_i, \dot{\mathsf{C}}_i : i \in [0..\mathsf{M}]\}$ and Y for $\{y \in X : y \leq \dot{\mathsf{C}}_{\mathsf{M}} \text{ in } \boldsymbol{o}_2\}$.

1. For $x_1, x_2 \in \mathcal{X}_{\dot{\mathsf{C}}} \setminus Y$, if $x_1 \bowtie x_2$ in $\boldsymbol{o}_1$ then add $x_1 \bowtie x_2$ to $\boldsymbol{o}_3$.
2. For $y \in Y$, if $y \bowtie \dot{\mathsf{C}}_i$ in $\boldsymbol{o}_2$, then add $y \bowtie \dot{\mathsf{C}}_i$ to $\boldsymbol{o}_3$.
3. For $z_1, z_2 \in \mathcal{X}_{\dot{\mathsf{C}}}$ such that $\dot{\mathsf{C}}_i < z_1 < \dot{\mathsf{C}}_{i+1}$ and $\dot{\mathsf{C}}_i < z_2 < \dot{\mathsf{C}}_{i+1}$ in $\boldsymbol{o}_3$,
   - add $z_1 < z_2$ to $\boldsymbol{o}_3$ if $\eta_3 \models \{\dot{\mathsf{C}}_i\} < \{z_1\} < \{z_2\}$, $\eta_3 \models \{z_2\} < \{\dot{\mathsf{C}}_i\} < \{z_1\}$, or $\eta_3 \models \{z_1\} < \{z_2\} < \{\dot{\mathsf{C}}_i\}$.
   - add $z_1 = z_2$ to $\boldsymbol{o}_3$ if $\eta_3 \models \{z_1\} = \{z_2\}$.
4. For $y_1, y_2 \in Y$, if $y_1 \lhd y_2 < \dot{\mathsf{C}}_0$ in $\boldsymbol{o}_2$, then add $y_1 \lhd y_2$ in $\boldsymbol{o}_3$ where $\lhd \in \{<, =\}$.

Note that the computation of $\boldsymbol{o}_3$ only requires $\boldsymbol{o}_1$, $\boldsymbol{o}_2$, and fractional constraints. Then the lemma below holds for well-formed simulating stacks.

A stack $(\eta_1, \boldsymbol{o}_1)(\eta_2, \boldsymbol{o}_2) \dots (\eta_n, \boldsymbol{o}_n)$ is a well-formed simulating stack if

- For any $i \in [1..n]$, $\eta_i \models x \bowtie y$ iff $x \bowtie y$ in $\boldsymbol{o}_i$;
- For any $i \in [1..(n-1)]$, $\eta_i(\dot{\mathsf{C}}_j) = \eta_{i+1}(\dot{\mathsf{C}}_j)$, $\quad \eta_i(x) = \eta_{i+1}(x)$ if $\dot{\mathsf{C}}_{\mathsf{M}} < x$ in $\boldsymbol{o}_{i+1}$, and $\dot{\mathsf{C}}_0$ is the smallest in η_i.

Lemma 1. *Let $w\,(\eta_1, \boldsymbol{o}_1)\,(\eta_2, \boldsymbol{o}_2)$ be a well-formed simulating stack. The simulated $\mathbf{pop}(X)$ transition $w\,(\eta_1, \boldsymbol{o}_1)\,(\eta_2, \boldsymbol{o}_2) \xrightarrow{\mathbf{pop}(X)} w\,(\eta_1[X \coloneqq \eta_2], \boldsymbol{o}_3)$ (where $\boldsymbol{o}_3$ is obtained by the above steps) preserves well-formedness of the stack.*

Since well-formedness is also preserved under the other transitions, the main result of the present section follows.

Theorem 3. *From a SRTA $\mathfrak{S}$, we can build a SRTA $\mathfrak{S}'$ without comparison constraints such that $\mathcal{L}(\mathfrak{S}) = \mathcal{L}(\mathfrak{S}')$. The size of locations and stack symbols of $\mathfrak{S}'$ are exponential in $|\mathcal{X}|$ and M of $\mathfrak{S}$. However, the size of clocks of $\mathfrak{S}'$ is linear in M of $\mathfrak{S}$.*

Proof (Sketch). In simulated transitions of **pop**, we use the ordering $\boldsymbol{o}_2$ in the top frame and $\boldsymbol{o}_1$ in the next frame within a stack at the same time. This operation, however, is not allowed in the formalization of SRTA. Hence we use extended locations $q^{\boldsymbol{o}}$ with a symbolic ordering $\boldsymbol{o}$ to realize transitions as follows:

a **push** transition: $\langle q_0, w\langle\gamma_1, \eta_1, \boldsymbol{o}_1\rangle\rangle \xrightarrow{\alpha_1, \mathbf{push}(\gamma_2, X)} \langle q_1, w\langle\gamma_1, \eta_1, \boldsymbol{o}_1\rangle\langle\gamma_2, \eta_2, \boldsymbol{o}_2\rangle\rangle$

is realized by

$$\langle q_0^{\boldsymbol{o}_1}, w'\langle(\gamma_1, \boldsymbol{o}), \eta_1\rangle\rangle \xrightarrow{\alpha_1, \mathbf{push}((\gamma_2, \boldsymbol{o}_1), X)} \langle q_1^{\boldsymbol{o}_2}, w'\langle(\gamma_1, \boldsymbol{o}), \eta_1\rangle\langle(\gamma_2, \boldsymbol{o}_1), \eta_2\rangle\rangle.$$

Also, a pop transition

$$\langle q_2, w\langle\gamma_1, \eta_1, \boldsymbol{o}_1\rangle\langle\gamma_2, \eta_2, \boldsymbol{o}_2\rangle\rangle \xrightarrow{\alpha_2, \mathbf{pop}(\gamma_2, X)} \langle q_3, w\langle\gamma_1, \eta_1[X \coloneqq \eta_2], \boldsymbol{o}_3\rangle\rangle$$

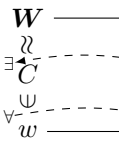
is realized by

$$\langle q_2^{\boldsymbol{o}_2}, w'\langle(\gamma_1, \boldsymbol{o}), \eta_1\rangle\langle(\gamma_2, \boldsymbol{o}_1), \eta_2\rangle\rangle \xrightarrow{\alpha_2, \mathbf{pop}((\gamma_2, \boldsymbol{o}_1), X)} \langle q_3^{\boldsymbol{o}_3}, w'\langle(\gamma_1, \boldsymbol{o}), \eta_1[X \coloneqq \eta_2]\rangle\rangle.$$

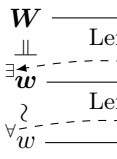
To compute $\boldsymbol{o}_3$ correctly by using fractional constraints, we need multi-step ϵ-transitions that are performed atomically. To ensure this atomicity, we again employ the technique in the proof of Theorem 1. □

4 Collapsed and Digital Semantics for Reachability Problem

Based on the result of the previous section, hereafter we consider *SRTA without comparison constraints*. In this section, we consider three techniques and combine them to translate the standard semantics Stnd into a finite-PDS semantics Digi via an infinite-PDS semantics Coll. We compare Abdulla's and our proofs of the soundness property that for any transition in the abstract semantics Digi, there is a corresponding transition in the concrete semantics Stnd.



The proof of *Lemma 4* of Abdulla et al. [1] can be summarized schematically as the left diagram: if $\boldsymbol{W} \to \boldsymbol{W}'$ and $C' \approx \boldsymbol{W}'$, then there exists C such that for all $w \in C$ there exists $w' \in C'$ with $w \to w'$. We find out that this elaborate simulation is called *backward-forward simulations* in Lynch and Vaandrager [11]. It is a source of complications in their proof to simultaneously handle the backward direction (choosing C from C') and the forward direction (finding w' from $w \in C$). In addition, the stack correspondence $\approx$ was defined *indirectly* through a flatten operator, and it is another source of complications in their proof. For example, the operator `flat` flattens a stack of Stnd $\eta_1\eta_2\eta_3$ to a single valuation η where $\eta = \eta_1^{(1)} \uplus \eta_2^{(2)} \uplus \eta_3^{(3)}$ is uniquely obtained by introducing $x^{(i)}$ for x at i-th frame: $\eta_i^{(i)}(x^{(i)}) \triangleq \eta_i(x)$. However, for a stack of the abstract semantics Digi, `flat` behaves *non*deterministically because as we will see later on we dismiss exact fractional values to obtain a finite-PDS. Then there are many ways to arrange clocks in a linear order.



In contrast, we clearly solve these problems as Lemma 3 and 6 by considering the intermediate semantics Coll. This allows us to completely separate the above mixed simulation into two simple simulations and *directly* define correspondences $\sim$ and $\models$ in a componentwise manner.

We use the following run of Stnd as a running example of this section:

$$\eta_1\,\eta_2 \xrightarrow{\mathsf{push}(\emptyset)} \eta_1\,\eta_2\,\mathbf{0}_{\mathcal{X}} \xrightarrow{x\leftarrow[1,2]} \eta_1\,\eta_2\,\eta_3 \overset{2.0}{\rightsquigarrow} \eta_1'\,\eta_2'\,\eta_3' \xrightarrow{\mathsf{pop}(\{x\})} \eta_1'\,\eta_4 \qquad \text{where}$$

$\eta_1 = \{x \mapsto 0.5\}$, $\eta_2 = \{x \mapsto 2.0\}$, $\eta_3 = \{x \mapsto 1.5\}$, $\eta_i' = \eta_i + 2$, $\eta_4 = \eta_3' = \{x \mapsto 3.5\}$.

4.1 Collapsed Semantics

Removing the unboundedness. Since we consider SRTA without comparison constraints, we can safely collapse the integral parts of clocks which are larger than M where $\mathsf{M} \geq \max\{j : (i,j)$ or $[i,j]$ appears in interval constraints$\}$. For example, if $\mathsf{M} = 2$, we cannot distinguish $\{x \mapsto 2.5; y \mapsto 2.6\}$ and $\{x \mapsto 3.5; y \mapsto 4.6\}$ by any constraints. The above run is collapsed as follows (if $\mathsf{M} = 2$):

$$\lambda_1\,\lambda_2 \xrightarrow{\mathsf{push}(\emptyset)} \lambda_1\,\lambda_2\,\mathbf{0}_{\mathcal{X}} \xrightarrow{x\leftarrow[1,2]} \lambda_1\,\lambda_2\,\lambda_3 \overset{2.0}{\rightsquigarrow} \lambda_1'\,\lambda_2'\,\lambda_3' \xrightarrow{\mathsf{pop}(\{x\})} \lambda_1'\,\lambda_4 \quad \text{where}$$

$\lambda_i = \eta_i\,(i = 1,2,3)$, $\lambda_1' = \{x \mapsto \infty.5\}, \lambda_2' = \{x \mapsto \infty.0\}$, $\lambda_3' = \{x \mapsto \infty.5\}$, $\lambda_4 = \lambda_3'$.

Definition 3. We define the collapse function to formalize the above argument:

$$\mathcal{C} : \mathbb{Q}^+ \to \big(\{0,1,\ldots,\mathsf{M},\infty\}\big) \times \big(\mathbb{Q}^+ \cap [0,1)\big); \quad \mathcal{C}(r) \triangleq \begin{cases} (\lfloor r \rfloor, \{r\}) & \text{if } r \leq \mathsf{M} \\ (\infty, \{r\}) & \text{otherwise.} \end{cases}$$

We write $v.r$ to denote (v, r). Moreover, $\lfloor v.r \rfloor$ and $\{v.r\}$ denote v and r, respectively. For a concrete valuation η on X, we define the *collapsed valuation of* η by $\mathcal{C}(\eta)(x) \triangleq \mathcal{C}(\eta(x))$. We use Greek letters $\lambda, \ldots$ to denote a collapsed valuation.

Proposition 1. *Let η_1 and η_2 be concrete valuations on X. If $\mathcal{C}(\eta_1) = \mathcal{C}(\eta_2)$,*
Validity. *$\eta_1 \models \varphi$ iff $\eta_2 \models \varphi$ for any constraint φ.*
Copying. *$\mathcal{C}(\eta_1[x \coloneqq y]) = \mathcal{C}(\eta_2[x \coloneqq y])$ for any $x, y \in X$.*
Restriction. *$\mathcal{C}(\eta_1|Y) = \mathcal{C}(\eta_2|Y)$ for any $Y \subseteq X$.*
Updating. *$\mathcal{C}(\eta_1[x \coloneqq r]) = \mathcal{C}(\eta_2[x \coloneqq r])$ for any $x \in X$ and $r \in [0, \mathsf{M}]$.*
Evolve. *$\mathcal{C}(\eta_1 + \delta) = \mathcal{C}(\eta_2 + \delta)$ for any $\delta \in \mathbb{Q}^+$.*

By Proposition 1, we define several notions for collapsed valuations as follows. Let X be a clock set, η and λ be concrete and collapsed valuations on X, respectively, such that $\mathcal{C}(\eta) = \lambda$. For a constraint φ, we write $\lambda \models \varphi$ if $\eta \models \varphi$. Then $\lambda \models \varphi$ is well-defined because Proposition 1 ensures that the result does not depend on the choice of a witness η for λ. We also define copying $\lambda[x \coloneqq y]$, restriction $\lambda|Y$, updating $\lambda[x \coloneqq r]$, and evolve $\lambda + \delta$ in the same way.

We define a quasi-ordering for collapsed valuations. Let λ, λ' be collapsed valuations and η be a concrete valuation such that $\mathcal{C}(\eta) = \lambda$. We write $\lambda \preccurlyeq \lambda'$ if there exists η' such that $\eta \leq \eta'$ and $\mathcal{C}(\eta') = \lambda'$.

Removing entire stack modifications. Collapsed valuations are effective to reduce the unboundedness of the nonnegative rational numbers. However, they are ineffective to reduce entire stack modifications of timed transitions in STND and translate STND into an *infinite-PDS* semantics.

To obtain a corresponding infinite-PDS semantics, we adopt the **lazy time elapsing** technique of Abdulla et al. [1]. Then the above collapsed run is simulated as follows:

$$\lambda_1 \xrightarrow{\mathsf{push}(\emptyset)} \lambda_1\, \lambda_2 \xrightarrow{x \leftarrow [1,2]} \lambda_1\, \lambda_2'\, \lambda_2'' \overset{2.0}{\rightsquigarrow} \lambda_1\, \lambda_2''\, \lambda_2'' \xrightarrow{\mathsf{pop}(\{x\})} \lambda,$$

where $\lambda_1 = \{\underset{\bullet}{x} \mapsto 0.5; \dot{x} \mapsto 2.0\}$, $\lambda_2 = \{\underset{\bullet}{x} \mapsto 2.0; \dot{x} \mapsto 0.0\}$, $\lambda_2' = \{\underset{\bullet}{x} \mapsto 2.0; \dot{x} \mapsto 1.5\}$, and $\lambda_2'' = \{\underset{\bullet}{x} \mapsto \infty.0; \dot{x} \mapsto \infty.5\}$.

Although we do not evolve the frames below the top frame during the timed transition, we *lazily* evolve λ_1 when performing the $\mathsf{pop}(\{x\})$ transition. To correctly evolve λ_1, we use the marked clocks $\dot{x}$ of λ_1 and $\underset{\bullet}{x}$ of λ_2'' and increase $\lambda_1 + \delta$ until they are compatible: $\lambda_1(\dot{x}) + \delta = \lambda_2''(\underset{\bullet}{x})$.

However, there are two possibilities for compatibility:

- $\delta_1 = 1.0$: $\lambda_1 + \delta_1 = \{\underset{\bullet}{x} \mapsto 1.5; \dot{x} \mapsto \infty.0\}$ is compatible with λ_2''.
- $\delta_2 = 2.0$: $\lambda_1 + \delta_2 = \{\underset{\bullet}{x} \mapsto \infty.5; \dot{x} \mapsto \infty.0\}$ is compatible with λ_2''.

The ambiguity happens because we collapse the integral parts of clocks. In order to overcome this problem, we use the reference clock C and it is inserted as the value 0.0 when a **push** transition is taken as follows:

$$\begin{aligned}
&\Lambda_1 \xrightarrow{\mathsf{push}(\emptyset)} \Lambda_1^{\mathrm{reset}}\, \Lambda_2 \xrightarrow{x \leftarrow [1,2]} \Lambda_1^{\mathrm{reset}}\, \Lambda_2'\, \Lambda_2'' \overset{2.0}{\rightsquigarrow} \Lambda_1^{\mathrm{reset}}\, \Lambda_2''\, \Lambda_2'' \xrightarrow{\mathsf{pop}(\{x\})} \Lambda,\\
&\Lambda_1^{\mathrm{reset}} = \{\underset{\bullet}{x} \mapsto 0.5;\ \dot{x} \mapsto 2.0;\ \dot{\mathsf{C}} \mapsto 0.0\},\ \ \Lambda_2 = \{\underset{\bullet}{x} \mapsto 2.0;\ \dot{x} \mapsto 0.0;\ \underset{\bullet}{\mathsf{C}} \mapsto 0.0\},\\
&\Lambda_2' = \{\underset{\bullet}{x} \mapsto 2.0;\ \dot{x} \mapsto 1.5;\ \underset{\bullet}{\mathsf{C}} \mapsto 0.0\},\ \ \Lambda_2'' = \{\underset{\bullet}{x} \mapsto \infty.0;\ \dot{x} \mapsto \infty.5;\ \underset{\bullet}{\mathsf{C}} \mapsto 2.0\}.
\end{aligned}$$

The clock C enables us to find the correct corresponding valuation Λ_1' of $\Lambda_1^{\mathrm{reset}}$ as $\Lambda_1' = \Lambda_1^{\mathrm{reset}} + 2.0$ (c.f. Lemma 2) and it also appeared in Abdulla's construction.

To formalize the above lazy time elapsing technique, we define the notion of clock marking and extended clock set.

Definition 4. Let X be a clock set. We define the marked clock sets $\dot{X}$ and $\underset{\bullet}{X}$ of X by marking every clock x as $\dot{x}$ and $\underset{\bullet}{x}$, respectively. For a valuation $\eta \in X_V$, the renamed valuation $\dot{\eta} \in \dot{X}_V$ is defined by $\dot{\eta}(\dot{x}) \triangleq \eta(x)$ for all $x \in X$. We also define the renamed valuation $\underset{\bullet}{\eta} \in \underset{\bullet}{X}_V$. Furthermore, for a constraint $\varphi \in \Phi_X$, we define $\dot{\varphi} \in \Phi_{\dot{X}}$ by renaming every clock x in φ to $\dot{x}$.

We extend the clock set $\mathcal{X}$ to $\mathbb{X}$ by $\mathbb{X} \triangleq \mathcal{X} \cup \{\mathfrak{C}\}$. We use $\mathbb{C}$ for the set of collapsed valuations on $\underset{\bullet}{\mathbb{X}} \cup \dot{\mathbb{X}}$ and use capital Greek letters $\Lambda, \ldots$ to denote a collapsed valuation in $\mathbb{C}$.

Let Λ_1 and Λ_2 be collapsed valuations on $\underset{\bullet}{\mathbb{X}} \cup \dot{\mathbb{X}}$. Then,

- the two valuations are **compatible** $\Lambda_1 \parallel \Lambda_2$ if $\Lambda_1(\dot{x}) = \Lambda_2(\underset{\bullet}{x})$ for all $x \in \mathbb{X}$.
- If two valuations are compatible, then the **glued** valuation $\Lambda_1 \oplus \Lambda_2 \in \mathbb{C}$ is defined by $(\Lambda_1 \oplus \Lambda_2)(\underset{\bullet}{x}) \triangleq \Lambda_1(\underset{\bullet}{x})$ and $(\Lambda_1 \oplus \Lambda_2)(\dot{x}) \triangleq \Lambda_2(\dot{x})$ for $x \in \mathbb{X}$.

Collapsed valuations lead to the collapsed semantics Coll, which removes the unboundedness of rationals and entire stack modifications of Stnd.

Definition 5 (Collapsed Semantics). We define the infinite-PDS $(Q, \Gamma \times \mathbb{C}, \hookrightarrow)$ where $\langle q, \boldsymbol{w} \rangle \hookrightarrow \langle q', \boldsymbol{w}' \rangle$ if there is $\langle q, \tau, q' \rangle \in \Delta$ and $\boldsymbol{w} \overset{\tau}{\hookrightarrow} \boldsymbol{w}'$.

For $\tau \in Op$, we define the action $\boldsymbol{w} \overset{\tau}{\hookrightarrow} \boldsymbol{w}'$ by case analysis on τ as follows:

$$\frac{\Lambda_1^{\mathrm{r}} = \Lambda_1[\dot{\mathfrak{C}} \coloneqq 0] \quad \Lambda_1^{\mathrm{r}} \parallel \Lambda_2 \quad \Lambda_2 | \dot{\mathbb{X}} = \mathcal{C}(\mathbf{0}_{\underset{\bullet}{\mathbb{X}}})}{\langle \gamma, \Lambda_1 \rangle \hookrightarrow \langle \gamma, \Lambda_1^{\mathrm{r}} \rangle \langle \gamma', \mathcal{U}(X, \Lambda_2) \rangle}\ \mathsf{push}(\gamma', X) \qquad \frac{r \in I \quad \Lambda' = \Lambda[\dot{x} \coloneqq r]}{\langle \gamma, \Lambda \rangle \hookrightarrow \langle \gamma, \Lambda' \rangle}\ x \leftarrow I$$

$$\frac{\Lambda_1 \preccurlyeq \Lambda_1' \quad \Lambda_1' \parallel \Lambda_2 \quad \Lambda = \Lambda_1' \oplus \mathcal{U}(\mathcal{X} \setminus X, \Lambda_2)}{\langle \gamma, \Lambda_1 \rangle \langle \gamma', \Lambda_2 \rangle \hookrightarrow \langle \gamma, \Lambda \rangle}\ \mathsf{pop}(\gamma', X) \qquad \frac{\Lambda \models \dot{\varphi}}{\langle \gamma, \Lambda \rangle \hookrightarrow \langle \gamma, \Lambda \rangle}\ \mathsf{check}(\varphi)$$

where $\mathcal{U}(\{x_1, \ldots, x_n\}, \Lambda) \triangleq \Lambda[\dot{x}_1 \coloneqq \underset{\bullet}{x}_1, \ldots, \dot{x}_n \coloneqq \underset{\bullet}{x}_n]$ and this intuitively means that copying the values of clocks x_i in the next to the top frame into the top frame. Hence, we use $\mathcal{U}(X, \Lambda_2)$ to define $\mathsf{push}(\gamma', X)$. Also, from the fact that $\eta_1\, \eta_2 \to \eta_1[X \coloneqq \eta_2]$ is equal to $\eta_1\, \eta_2 \to \eta_2[(\mathcal{X} \setminus X) \coloneqq \eta_1]$, we employ $\mathcal{U}(\mathcal{X} \setminus X, \Lambda_2)$ to define $\mathsf{pop}(\gamma', X)$ for fitting the definition of gluing $\oplus$.

In addition, the rules $\langle q, \langle \gamma, \Lambda \rangle \rangle \hookrightarrow \langle q, \langle \gamma, \Lambda' \rangle \rangle$ are added for all $q \in Q$, $\gamma \in \Gamma$, and $\Lambda \preccurlyeq \Lambda'$ to reflect timed transitions in Stnd.

A stack $\langle \gamma_1, \Lambda_1 \rangle \langle \gamma_2, \Lambda_2 \rangle \ldots \langle \gamma_n, \Lambda_n \rangle$ is *well-formed* WF if for all $i \in [1..(n-1)]$

- $\Lambda_i \models \dot{\mathfrak{C}} \in [0, 0]$ and there exists Λ_i' such that $\Lambda_i \preccurlyeq \Lambda_i'$ and $\Lambda_i' \parallel \Lambda_{i+1}$.

It can be easily shown that transitions preserve well-formedness. As we mentioned above, the condition $\Lambda_i \models \dot{\mathfrak{C}} \in [0, 0]$ of the well-formedness is key to ensure the following property and the determinacy of **pop** transitions.

Lemma 2. *If* $\mathsf{WF}(\boldsymbol{w} \langle \gamma_1, \Lambda_1 \rangle \langle \gamma_2, \Lambda_2 \rangle)$*, then there exists the* unique Λ_1' *such that* $\Lambda \preccurlyeq \Lambda_1'$ *and* $\Lambda_1' \parallel \Lambda_2$.

This defines the stack correspondence $w \sim \boldsymbol{w}$ of Stnd and Coll with $\mathsf{WF}(\boldsymbol{w})$:

- $\langle \gamma, \eta \rangle \sim \langle \gamma, \Lambda \rangle$ if $\mathcal{C}(\dot{\eta}) = \Lambda | \dot{\mathcal{X}}$.

– $w\langle\gamma_1,\eta_1\rangle\langle\gamma_2,\eta_2\rangle \sim \boldsymbol{w}\langle\gamma_1,\Lambda_1\rangle\langle\gamma_2,\Lambda_2\rangle$ if $\mathcal{C}(\dot{\eta}_1 \cup \dot{\eta}_2) = \Lambda_2|(\dot{\mathcal{X}} \cup \dot{\mathcal{X}})$ and $w\langle\gamma_1,\eta_1\rangle \sim \boldsymbol{w}\langle\gamma_1,\Lambda_1'\rangle$ where Λ_1' is *uniquely* determined from Λ_1 and Λ_2 by Lemma 2.

This correspondence forms a bisimulation of STND and COLL.

Lemma 3. *Let $\langle q, w\rangle$ and $\langle q, \boldsymbol{w}\rangle$ be configurations of* STND *and* COLL*, respectively, with $w \sim \boldsymbol{w}$ and $\mathsf{WF}(\boldsymbol{w})$. If $\langle q, w\rangle \twoheadrightarrow \langle q', w'\rangle$, then there exists $\boldsymbol{w}'$ such that $\langle q, \boldsymbol{w}\rangle \to \langle q', \boldsymbol{w}'\rangle$ and $w' \sim \boldsymbol{w}'$. Conversely, if $\langle q, \boldsymbol{w}\rangle \to \langle q', \boldsymbol{w}'\rangle$, then there exists $\langle q', w'\rangle$ such that $\langle q, w\rangle \twoheadrightarrow \langle q', w'\rangle$ and $w' \sim \boldsymbol{w}'$. We use $c_1 \twoheadrightarrow c_2$ to denote a timed transition $c_1 \overset{\delta}{\rightsquigarrow} c_2$ or discrete transition $c_1 \overset{\alpha}{\to} c_2$ in* STND*.*

4.2 Digital Valuations and Finite-PDS Semantics

The COLL semantics cannot be formalized as finite-PDS for the denseness of rationals. To remove the denseness, we define digital valuations and give the digital semantics DIGI. The definition is based on *regions* of Abdulla et al. in [1].

Definition 6 (Digital Valuations). Let X be a clock set. A sequence of sets $\boldsymbol{d} = d_0\, d_1 \ldots d_n$, where $d_i \subseteq X \times \{0, \ldots, \mathsf{M}, \infty\}$, is a *digital valuation* on X if $\boldsymbol{d}$ satisfies the following conditions:

– Every clock in X appears in $\boldsymbol{d}$ exactly once.
– Except d_0, all the sets d_i are not empty: $d_i \neq \emptyset$ for all $i \in [\mathbf{1}..n]$.
– The constant M only appears at d_0: if $(x, \mathsf{M}) \in d_i$, then $i = 0$.

Let λ be a collapsed valuations on X. We write $\lambda \models \boldsymbol{d}$ if the following hold:

– $\boldsymbol{d}$ reflects collapsed integrals: for all $x \in X$, $(x, \lfloor\lambda(x)\rfloor) \in d_i$ holds for some i.
– For all $x \in X$, $\{\lambda(x)\} = 0.0$ iff x is in d_0.
– Fractional order: $\{\lambda(x)\} \bowtie \{\lambda(y)\}$ iff x is in d_i and y is in d_j for some $i \bowtie j$.

The realization relation $\models$ is **functional**: for a collapsed valuation λ, there exists the unique digital valuation $\mathcal{D}(\lambda)$ such that $\lambda \models \mathcal{D}(\lambda)$.

Let us see an example with $\mathsf{M} = 1$:

$$\underset{\top\!\top}{\{x \mapsto 0.0; y \mapsto 0.3\}} \preccurlyeq \underset{\top\!\top}{\{x \mapsto 0.5; y \mapsto 0.8\}} \preccurlyeq \underset{\top\!\top}{\{x \mapsto 0.7; y \mapsto 1.0\}} \preccurlyeq \underset{\top\!\top}{\{x \mapsto 0.9; y \mapsto \infty.2\}}$$
$$\{(x,0)\}\{(y,0)\} \;\preceq\; \emptyset\{(x,0)\}\{(y,0)\} \;\preceq\; \{(y,1)\}\{(x,0)\} \;\preceq\; \emptyset\{(y,\infty)\}\{(x,0)\}$$

The relation $\boldsymbol{d} \preceq \boldsymbol{d'}$ and other operations are defined just as collapsed valuations.

Proposition 2. *Let λ_1 and λ_2 be collapsed valuations on X. If $\mathcal{D}(\lambda_1) = \mathcal{D}(\lambda_2)$,*

Validity. *$\lambda_1 \models \varphi$ iff $\lambda_2 \models \varphi$ for any constraint φ.*
Copying. *$\mathcal{D}(\lambda_1[x \coloneqq y]) = \mathcal{D}(\lambda_2[x \coloneqq y])$ for any $x, y \in X$.*
Restriction. *$\mathcal{D}(\lambda_1|Y) = \mathcal{D}(\lambda_2|Y)$ for any $Y \subseteq X$.*
Integer Update. *$\mathcal{D}(\lambda_1[x \coloneqq n]) = \mathcal{D}(\lambda_2[x \coloneqq n])$ for any $x \in X$ and $n \in [0..\mathsf{M}]$.*
Elapse. *If $\lambda_1 \preccurlyeq \lambda_1'$, then there exists λ_2' such that $\lambda_2 \preccurlyeq \lambda_2'$ and $\mathcal{D}(\lambda_1') = \mathcal{D}(\lambda_2')$.*

We define *validity* $\boldsymbol{d} \models \varphi$, *copying* $\boldsymbol{d}[x \coloneqq y]$, *restriction* $\boldsymbol{d}|Y$, and *quasi-ordering* $\boldsymbol{d} \preceq \boldsymbol{d'}$ similarly as collapsed valuations. Moreover, we define discrete updates $\boldsymbol{d}[x \coloneqq n]$ for $x \in X$ and $n \in [0..\mathsf{M}]$ by: $\boldsymbol{d}[x \coloneqq n] \triangleq \mathcal{D}(\lambda[x \coloneqq n])$ where λ is a witness $\lambda \models \boldsymbol{d}$. These are well-defined by Proposition 2. We define the *update* $\boldsymbol{d}[x \leftarrow I] \triangleq \{\, \mathcal{D}(\,\lambda[x \coloneqq r]\,) : r \in I, \lambda \models \boldsymbol{d} \,\}$ for a clock x and an interval I.

Lemma 4. *If $\boldsymbol{d} \preceq \boldsymbol{d'}$ and $\lambda' \models \boldsymbol{d'}$, then there is λ such that $\lambda \preccurlyeq \lambda'$ and $\lambda \models \boldsymbol{d}$.*

This lemma is crucial for the backward simulation lemma, Lemma 6, and peculiar to collapsed valuations. Indeed, this fails if we consider $\eta \models \boldsymbol{d}$ of concrete and digital valuations. Let us consider $\boldsymbol{d} = \{(y,1)\}\{(x,0)\} \preceq \emptyset\{(y,\infty)\}\{(x,0)\} = \boldsymbol{d'}$ and take $\eta' = \{x \mapsto 0.9; y \mapsto 2.2\}$ for $\eta' \models \boldsymbol{d'}$. There are no concrete valuations η such that $\eta \leq \eta'$ and $\eta \models \boldsymbol{d}$ because $y - x < 1$ in $\boldsymbol{d}$ but $y - x = 1.3$ in η'.

Digital Semantics. We use $\boldsymbol{D}, \ldots$ to denote a digital valuation on $\mathbb{X} \cup \dot{\mathbb{X}}$. As the semantics Coll, we define the compatibility and gluing as follows:

- We write $\boldsymbol{D}_1 \parallel \boldsymbol{D}_2$ if $\exists \Lambda_1, \Lambda_2.\ \Lambda_1 \models \boldsymbol{D}_1,\ \Lambda_2 \models \boldsymbol{D}_2$, and $\Lambda_1 \parallel \Lambda_2$;
- The glued valuations are defined by:

$$\boldsymbol{D}_1 \oplus \boldsymbol{D}_2 \triangleq \{\mathcal{D}(\Lambda) : \Lambda_1 \models \boldsymbol{D}_1,\ \Lambda_2 \models \boldsymbol{D}_2,\ \Lambda_1 \parallel \Lambda_2,\ \Lambda \in \Lambda_1 \oplus \Lambda_2\}\,.$$

Non-determinism Example. We revisit our running example to see the essential non-determinism of the gluing in **pop**.

$$\boldsymbol{D}_1 \xrightarrow{\mathsf{push}(\emptyset)} \boldsymbol{D}_1^{\mathrm{r}}\,\boldsymbol{D}_2 \xrightarrow{x \leftarrow [1,2]} \boldsymbol{D}_1^{\mathrm{r}}\,\boldsymbol{D}_2' \overset{2.0}{\rightsquigarrow} \boldsymbol{D}_1^{\mathrm{r}}\,\boldsymbol{D}_2'' \xrightarrow{\mathsf{pop}(\{x\})} \boldsymbol{D},$$

$$\boldsymbol{D}_1^{\mathrm{r}} = \mathcal{D}(\Lambda_1^{\mathrm{reset}}) = \{(\dot{\complement}, 0), (\dot{x}, 2)\}\{(\underset{\cdot}{x}, 0)\},\quad \boldsymbol{D}_2 = \{(\underset{\cdot}{\complement}, 0), (\dot{x}, 0), (\underset{\cdot}{x}, 2)\},$$

$$\boldsymbol{D}_2' = \{(\underset{\cdot}{\complement}, 0), (\underset{\cdot}{x}, 2)\}\{(\dot{x}, 1)\},\qquad \boldsymbol{D}_2'' = \{(\underset{\cdot}{\complement}, 2), (\underset{\cdot}{x}, \infty)\}\{(\dot{x}, \infty)\}.$$

To perform $\mathbf{pop}(\{x\})$, we compute $\boldsymbol{D}_1'$ such that $\boldsymbol{D}_1^{\mathrm{r}} \preceq \boldsymbol{D}_1'$ and $\boldsymbol{D}_1' \parallel \boldsymbol{D}_2''$ and obtain $\boldsymbol{D}_1' = \{(\dot{\complement}, 2), (\dot{x}, \infty)\}\{(\underset{\cdot}{x}, \infty)\}$. Then,

$$\begin{aligned}\boldsymbol{D}_1' \oplus \mathcal{U}(\emptyset, \boldsymbol{D}_2'') &= \big(\{(\dot{\complement}, 2), (\dot{x}, \infty)\}\{(\underset{\cdot}{x}, \infty)\}\big) \oplus \big(\{(\underset{\cdot}{\complement}, 2), (\underset{\cdot}{x}, \infty)\}\{(\dot{x}, \infty)\}\big) \\ &= \{\ \emptyset\{(\underset{\cdot}{x}, \infty), (\dot{x}, \infty)\},\ \emptyset\{(\underset{\cdot}{x}, \infty)\}\{(\dot{x}, \infty)\},\ \emptyset\{(\dot{x}, \infty)\}\{(\underset{\cdot}{x}, \infty)\}\ \}\,.\end{aligned}$$

Namely there are three choices for $\boldsymbol{D}$ in the order of $\dot{x}$ and $\underset{\cdot}{x}$ because we dismiss the fractional values from digital valuations to remove the denseness of rationals.

Digital valuations lead to the digital semantics Digi as *finite-PDS*. Since the definition is given by the same way as the Coll semantics, we give it in the long version [13]. The definition of the well-formedness $\mathsf{WF}(\boldsymbol{W})$ is also omitted.

Lemma 5. *The following properties hold for well-formed stacks.*
- *If $\mathsf{WF}(\boldsymbol{W})$ and $\langle q, \boldsymbol{W}\rangle \to \langle q', \boldsymbol{W'}\rangle$, then $\boldsymbol{W'}$ is also well-formed.*
- *If $\mathsf{WF}(\boldsymbol{W})$, then there exists $\boldsymbol{w}$ such that $\mathsf{WF}(\boldsymbol{w})$ and $\boldsymbol{w} \models \boldsymbol{W}$.*
  The realization $\Lambda_1 \ldots \Lambda_n \models \boldsymbol{D}_1 \ldots \boldsymbol{D}_n$ holds if $\Lambda_i \models \boldsymbol{D}_i$ for all $i \in [1..n]$.

Let $\boldsymbol{W}$ be a well-formed stack, then there is a well-formed stack $\mathsf{WF}(\boldsymbol{w})$ such that $\boldsymbol{w} \models \boldsymbol{W}$ by Lemma 5. Since digital valuations are an abstraction of collapsed valuations, if $\langle q, \boldsymbol{w}\rangle \to \langle q', \boldsymbol{w'}\rangle$ then there exists $\boldsymbol{W'}$ such that $\langle q, \boldsymbol{W}\rangle \to \langle q', \boldsymbol{W'}\rangle$ and $\boldsymbol{w'} \models \boldsymbol{W'}$. By contrast, the counterpart does not hold for the nondeterminism of **pop** rule in Digi (c.f. the above example). However, we can show the following backward-direction simulation by Lemma 4.

Lemma 6. *If $\mathsf{WF}(\boldsymbol{W})$, $\langle q, \boldsymbol{W}\rangle \to \langle q', \boldsymbol{W'}\rangle$, $\boldsymbol{w'} \models \boldsymbol{W'}$, and $\mathsf{WF}(\boldsymbol{w'})$, then there exists a well-formed stack $\boldsymbol{w}$ such that $\langle q, \boldsymbol{w}\rangle \to \langle q', \boldsymbol{w'}\rangle$ and $\boldsymbol{w} \models \boldsymbol{W}$.*

Finally, Lemma 3 and 6 imply our main theorem.

Theorem 4. *The following are equivalent:*
1) In Stnd, there is a run from $\langle q_{init}, \langle \bot, \mathbf{0}_{\mathcal{X}}\rangle\rangle$ to $\langle q_{final}, w\rangle$ for some stack w;
2) In Digi, there exists $\boldsymbol{W}$ such that $\langle q_{init}, \langle \bot, (\mathcal{D} \circ \mathcal{C})(\mathbf{0}_{\mathbb{X}} \cup \mathbf{0}_{\dot{\mathbb{X}}})\rangle\rangle \to^* \langle q_{final}, \boldsymbol{W}\rangle$.

5 Conclusion and Future Works

We have studied synchronized recursive timed automata (SRTA) and shown that the reachability problem of SRTA is ExpTime-complete. Our SRTA are described from the two perspectives: 1) SRTA are a variant of recursive timed automata (RTA) of Trivedi and Wojtczak, and Benerecetti et al. [12, 3] because SRTA are obtained by synchronizing timed transitions of RTA, 2) SRTA extend timed pushdown automata of Abdulla et al. [1] because SRTA are obtained by adding bounded updates ($x \leftarrow [i,j]$ and $x \leftarrow (i,j)$), diagonal constraints, and fractional constraints to their automata. We have also introduced an intermediate semantics to simplify our decidability proof of the reachability problem.

In the formalization of SRTA, we adopt bounded updates. Since our updates are performed within an interval, we can simulate diagonal constraints $x - y \bowtie k$ in Section 2 by using comparisons $x \bowtie y$ and fractional constraints $\{x\} \bowtie \{y\}$. As already proved by Bouyer et al. in [6], the presence of both *un*bounded updates $x \leftarrow [i,\infty)$ and diagonal constraints enables timed automata to simulate two-counter machines. However, up to the authors' knowledge, the combination of unbounded updates, comparison constraints, and fractional constraints has not been studied yet. We think that this combination further enlarges the decidable class of pushdown-extensions of timed automata.

Acknowledgement. This work was supported by JSPS KAKENHI Grant Number 15J01843 and 15K00087.

References

1. Abdulla, P., Atig, M., Stenman, J.: Dense-timed pushdown automata. In: LICS '12. pp. 35–44. IEEE (2012)
2. Alur, R., Dill, D.: A theory of timed automata. TCS. 126(2), 183–235 (1994)
3. Benerecetti, M., Minopoli, S., Peron, A.: Analysis of timed recursive state machines. In: TIME. pp. 61–68. IEEE (2010)
4. Bérard, B., Diekert, V., Gastin, P., Petit, A.: Characterization of the expressive power of silent transitions in timed automata. Fundam. Inf. 36(2), 145–182 (1998)
5. Bouajjani, A., Esparza, J., Maler, O.: Reachability analysis of pushdown automata: Application to model-checking. In: CONCUR '97, pp. 135–150. Springer (1997)
6. Bouyer, P., Dufourd, C., Fleury, E., Petit, A.: Updatable timed automata. TCS 321(2–3), 291–345 (2004)
7. Büchi, J.: Regular canonical systems. Arch. Math. Logik Grundlag 6, 91–111 (1964)
8. Clemente, L., Lasota, S.: Timed pushdown automata revisited. In: LICS '15. pp. 738–749. IEEE (2015)
9. Finkel, A., Willems, B., Wolper, P.: A direct symbolic approach to model checking pushdown systems. In: INFINITY '97, pp. 27–37. Elsevier (1997)
10. Krishna, S., Manasa, L., Trivedi, A.: What's decidable about recursive hybrid automata? In: HSCC '15. pp. 31–40. ACM (2015)
11. Lynch, N., Vaandrager, F.: Forward and backward simulations. Inf. & Comp. 121(2), 214–233 (1995)
12. Trivedi, A., Wojtczak, D.: Recursive timed automata. In: ATVA '10, pp. 306–324. Springer (2010)
13. Uezato, Y., Minamide, Y.: Synchronized recursive timed automata. `http://score.cs.tsukuba.ac.jp/~uezato/SRTA.pdf`, (2015)