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 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 extends 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: From a given SRTA, we can remove *diagonal constraints*—formulae of the form $x - y = 1$—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 nonnegative 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 includes 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 a 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 the Clemente's result.

At a 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 nonnegative 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 its stack alphabet 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 it is well-known that 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 features and these are used to recognize the language L_{ex} and treat 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:

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

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 $w \overset{\delta}{\rightsquigarrow} w'$.

Definition 3 (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}:

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

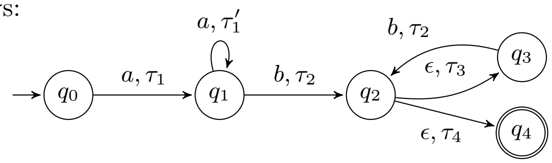
(For the initial configuration $\langle \bot, \mathbf{0}_{\mathcal{X}}\rangle$, we require the special stack symbol $\bot$ in Γ.)

Timed Language Example. We consider the following timed language:

$$L_{\text{ex}} \triangleq \{(a, t_1)(a, t_2) \ldots (a, t_n)(b, t_n') \ldots (b, t_2')(b, t_1') : \delta_i \in \mathbb{N} \text{ and } \delta_i \geq \delta_j \text{ if } i < j\}\,,$$

where $\delta_i = t_i' - t_i$. 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:



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

First, we consider the timed word $(a, 0.1)(a, 1.2)(b, 2.2)(b, 3.1) \in L_{\text{ex}}$ and it is accepted by the following run:

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

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 $\langle q_0, \langle \bot, 0\rangle\rangle \overset{0.1}{\rightsquigarrow}\xrightarrow{a}\overset{0.2}{\rightsquigarrow} \langle q_1, \langle \bot, 0.3\rangle\langle \natural, 0.2\rangle\rangle \overset{b}{\not\to} \langle q_2, \langle \bot, 0.3\rangle\langle \natural, 0.2\rangle\rangle$.

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 extra 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 using the timed transitions effectively.

Krishna et al. considered the subset of RTA called RTA_{RN} in [10] and showed that the reachability problem of RTA_{RN} is decidable. RTA_{RN} are completely subsumed by our SRTA because RTA_{RN} are SRTA without diagonal and fractional constraints. Although their proof closely followed that of Abdulla et al. [1] and depended on the details of Abdulla's construction, we adapt the Abdulla's construction to simplify our proof.

Compare to Timed Pushdown Automata. Timed pushdown automata of Abdulla et al. [1] are a pushdown extension of timed automata. Recently, 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 $(Q, q_{\text{init}}, q_{\text{final}}, \Sigma, \Gamma, \mathcal{X}, \Delta)$

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:

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

Since a valuation $\mathtt{X}$ of TPDA is simulated by using the value-copying mechanism of $\textbf{push}(\gamma, \mathcal{X})$ and $\textbf{pop}(\gamma, \mathcal{X})$ 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 $\mathfrak{S} = (Q', q'_{init}, q'_{final}, \Sigma, \Gamma \uplus \{\bot\}, \mathcal{X} \uplus \{z, \mathtt{x}\}, \Delta')$ 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 $\langle p, \langle\bot, \eta\rangle\rangle \xrightarrow{\alpha, \textbf{push}(\gamma, \mathcal{X})} \langle q, \langle\bot, \eta\rangle\langle\gamma, \eta'\rangle\rangle$ where $\mathtt{X}(x) = \eta(x)$ for all $x \in \mathcal{X}$.

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

$$\langle p, \langle\bot, \eta\rangle\langle\gamma, \eta'\rangle\rangle \xrightarrow{\epsilon, \mathtt{x} \leftarrow [0,0]} \langle p', \langle\bot, \eta\rangle\langle\gamma, \eta'[\mathtt{x} \coloneqq 0]\rangle\rangle \overset{\delta_1}{\rightsquigarrow} \xrightarrow{\epsilon, \textbf{check}(\psi)} \overset{\delta_2}{\rightsquigarrow} \xrightarrow{\alpha, \textbf{pop}(\mathcal{X} \uplus \{\mathtt{x}\})}$$
$$\langle q', \langle\bot, \eta''\rangle\rangle \overset{\delta_3}{\rightsquigarrow} \xrightarrow{\epsilon, \textbf{check}(\mathtt{x} \in [0,0])} \langle q, \langle\bot, \eta''\rangle\rangle \text{ where } \mathtt{X}(x) = \eta'(x) = \eta''(x) \text{ for } x \in \mathcal{X}.$$

By using the clock $\mathtt{x}$ as a stopwatch, we ensure that there are no interfering timed 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}. See the appendix for details of the proof of this theorem.

This suggests that fractional constraints play a crucial role in pushdown extensions of timed automata. Interestingly, the constraints are obtained by studying the 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:

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

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. 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 also 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, for 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 interval (i, j) or $[i, j]$ 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., a linear order of η_1' is represented symbolically by $x_1 < \wr x_2, u \int < z$ 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

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

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 modify any orderings.
4. At update $u \leftarrow [0, 1]$, we also perform $u \leftarrow [1, 2]$ first and next we reconstruct the correct ordering of η_2'''.

Finally, we consider the $\textbf{pop}(\{u,y,z\})$ transition. As above, first we actually perform $\textbf{pop}(\{u,y,z\})$ and obtain $(\eta_3, \nu_1' : (x_1 < \wr x_2, u \int < z)\ \&\ \nu_2''' : (u < y < z))$. 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 is very ad-hoc and does also fail 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 $\underset{\cdot}{\mathsf{C}}_j$ as follows:

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

where

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

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

Determine $x_2 < y$. With the auxiliary clocks, we try to determine $x_2 < y$. Performing $\textbf{pop}(\{u,y,z\})$ makes $(\eta_3 \uplus \{\dot{\mathsf{C}}_0 \mapsto 1.3, \dot{\mathsf{C}}_1 \mapsto 2.3, \dot{\mathsf{C}}_2 \mapsto 3.3\}, \lambda_1' : \boldsymbol{o}_1\ \&\ \lambda_2''' : \boldsymbol{o}_2)$ where $\boldsymbol{o}_1 = \dot{\mathsf{C}}_0 < x_1 < \dot{\mathsf{C}}_1 < \wr x_2, u \int < \dot{\mathsf{C}}_2 < z$ and $\boldsymbol{o}_2 = u < \underset{\cdot}{\mathsf{C}}_0 < \underset{\cdot}{\mathsf{C}}_1 < y < \underset{\cdot}{\mathsf{C}}_2 < z$.

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 are smaller than $\dot{\mathsf{C}}_0$ and 2) clocks are larger than $\dot{\mathsf{C}}_{\mathsf{M}}$.

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

(2) We consider $\underset{\cdot}{\mathsf{C}}_{\mathsf{M}} < z$ in $\boldsymbol{o}_2$. This states that *z was copied by* $\boldsymbol{\mathit{push}}(\{u,z\})$ *and never updated* because our updates are bounded by $\mathsf{M} = 2$ and the bounded updates cannot make a clock larger than $\underset{\cdot}{\mathsf{C}}_{\mathsf{M}}$. Thus, we take $\dot{\mathsf{C}}_{\mathsf{M}} < x$ in λ_3 if $\dot{\mathsf{C}}_{\mathsf{M}} < x$ in $\boldsymbol{o}_1$ 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 I. to IV.:

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

I. For $x_1, x_2 \in \mathcal{X}_{\mathtt{C}} \setminus Y$, if $x_1 \bowtie x_2$ in $\boldsymbol{o}_1$ then add $x_1 \bowtie x_2$ to $\boldsymbol{o}_3$.
II. For $y \in Y$, if $y \bowtie \underset{\bullet}{\mathtt{C}}_i$ in $\boldsymbol{o}_2$, then add $y \bowtie \dot{\mathtt{C}}_i$ to $\boldsymbol{o}_3$.
III. For $z_1, z_2 \in \mathcal{X}_{\mathtt{C}}$ such that $\dot{\mathtt{C}}_i < z_1 < \dot{\mathtt{C}}_{i+1}$ and $\dot{\mathtt{C}}_i < z_2 < \dot{\mathtt{C}}_{i+1}$ in $\boldsymbol{o}_3$,
   – add $z_1 < z_2$ to $\boldsymbol{o}_3$ if $\eta_3 \models \{\dot{\mathtt{C}}_i\} < \{z_1\} < \{z_2\}$, $\eta_3 \models \{z_2\} < \{\dot{\mathtt{C}}_i\} < \{z_1\}$, or $\eta_3 \models \{z_1\} < \{z_2\} < \{\dot{\mathtt{C}}_i\}$.
   – add $z_1 = z_2$ to $\boldsymbol{o}_3$ if $\eta_3 \models \{z_1\} = \{z_2\}$.
IV. For $y_1, y_2 \in Y$, if $y_1 \lhd y_2 < \underset{\bullet}{\mathtt{C}}_0$ in $\boldsymbol{o}_2$, then add $y_1 \lhd y_2$ in $\boldsymbol{o}_3$ where $\lhd \in \{<, =\}$.

We remark 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 $(\eta_1, \boldsymbol{o}_1)(\eta_2, \boldsymbol{o}_2) \ldots (\eta_n, \boldsymbol{o}_n)$ 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 $i \in [1..(n-1)]$, $\eta_i(\dot{\mathtt{C}}_j) = \eta_{i+1}(\underset{\bullet}{\mathtt{C}}_j)$, $\quad \eta_i(x) = \eta_{i+1}(x)$ if $\dot{\mathtt{C}}_{\mathsf{M}} < x$ in $\boldsymbol{o}_{i+1}$, and $\dot{\mathtt{C}}_0$ 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 $\boldsymbol{pop}(X)$ transition $w\,(\eta_1, \boldsymbol{o}_1)\,(\eta_2, \boldsymbol{o}_2) \xrightarrow{\boldsymbol{pop}(X)} w\,(\eta_1[X \coloneqq \eta_2], \boldsymbol{o}_3)$ (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. *For a given SRTA $\mathfrak{S}$, we can build a SRTA $\mathfrak{S}'$ without comparison constraints such that $\mathcal{L}(\mathfrak{S}) = \mathcal{L}(\mathfrak{S}')$. To determine and keep correct orderings, 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 $\mathbf{pop}$, 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 $\mathbf{push}$ 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

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

More precisely, the one step of the above transitions should be decomposed into multi-step ϵ-transitions that are performed atomically. To ensure this atomicity, we again employ the technique appearing in the proof of Theorem 1. $\square$

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. Before describing our construction, we compare the approaches of Abdulla et al. and ours to show that a reachability in the abstract semantics DIGI is also possible in the concrete semantics STND.

$$\begin{tikzcd}
\boldsymbol{W} \arrow[r] \arrow[d, phantom, "\wr\wr"] & \boldsymbol{W'} \arrow[d, phantom, "\wr\wr"] \\
{}^{\exists}C \arrow[d, phantom, "\Cup"] & {}^{\forall}C' \arrow[l, dashed] \arrow[d, phantom, "\Cup"] \\
{}^{\forall}w \arrow[r] \arrow[r, dashed, bend left] & {}^{\exists}w'
\end{tikzcd}$$

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.

$$\begin{tikzcd}
\boldsymbol{W} \arrow[r] \arrow[d, phantom, "\rotatebox{90}{$\models$}"] & \boldsymbol{W'} \arrow[d, phantom, "\rotatebox{90}{$\models$}"] \arrow[dl, phantom, "\text{Lem 6}"] \\
{}^{\exists}\boldsymbol{w} \arrow[r] \arrow[d, phantom, "\wr"] & {}^{\forall}\boldsymbol{w'} \arrow[l, dashed, bend right] \arrow[d, phantom, "\wr"] \arrow[dl, phantom, "\text{Lem 3}"] \\
{}^{\forall}w \arrow[r] \arrow[r, dashed, bend left] & {}^{\exists}w'
\end{tikzcd}$$

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. Finally, these make the entire proof easy to follow.

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

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

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

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 $\mathsf{M} \geq \max\{j : (i,j) \text{ or } [i,j] \text{ appears in interval constraints}\}$. For example, if $\mathsf{M} = 2$, we cannot distinguish $\{x \mapsto 2.5; y \mapsto 2.6\}$ and $\{x \mapsto 3.5; y \mapsto 4.9\}$ by any constraints. The above run is collapsed as follows (if $\mathsf{M} = 2$):

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

$\lambda_1 = \{x \mapsto 0.5\}$, $\lambda_2 = \{x \mapsto 2.0\}$, $\lambda_3 = \{x \mapsto 1.5\}$, $\lambda_i' = \lambda_i + 2$, and $\lambda_4 = \{x \mapsto \infty.5\}$.

Definition 4. 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 except comparison constraints.*
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:

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

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

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 $\overset{\bullet}{x}$ of λ_1 and $\underset{\bullet}{x}$ of λ_2'' and increase $\lambda_1 + \delta$ until they are compatible: $\lambda_1(\overset{\bullet}{x}) + \delta = \lambda_2''(\underset{\bullet}{x})$.

However, there are two possibilities for compatibility:

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

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

The clock C enables us to find the correct corresponding valuation Λ_1' of Λ_1^{reset} as $\Lambda_1' = \Lambda_1^{\text{reset}} + 2.0$ (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 5. 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 $\mathbb{X} \triangleq \mathcal{X} \cup \{\mathtt{C}\}$. 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 6 (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:

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

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

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 $\textsf{push}(\gamma', X)$. Also, by using 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 use this alternative way $\mathcal{U}(\mathcal{X} \setminus X, \Lambda_2)$ to define $\textsf{pop}(\gamma', X)$ 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* $\textsf{WF}$ if for all $i \in [1..(n-1)]$

- $\Lambda_i \models \dot{\mathtt{C}} \in [0, 0]$ 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 $\Lambda_i \models \dot{\mathtt{C}} \in [0, 0]$ of the well-formedness is key to ensure the following property and the determinacy of **pop** transitions.

Lemma 2. *If* $\textsf{WF}(\boldsymbol{w}\langle \gamma_1, \Lambda_1\rangle\langle \gamma_2, \Lambda_2\rangle)$*, 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 $\textsf{WF}(\boldsymbol{w})$:

- $\langle \gamma, \eta\rangle \sim \langle \gamma, \Lambda\rangle$ 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 $\mathcal{C}(\underset{\bullet}{\eta_1}\cupdot\dot{\eta}_2) = \Lambda_2|(\underset{\bullet}{\mathcal{X}} \cupdot \dot{\mathcal{X}})$ 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 $\mathit{WF}(\boldsymbol{w})$. 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 used $c_1 \twoheadrightarrow c_2$ to denote a timed transition $c_1 \overset{\delta}{\rightsquigarrow} c_2$ or discrete transition $c_1 \xrightarrow{\alpha} c_2$ 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 7 (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 $\mathsf{M} = 1$:

$$\begin{array}{ccccccc}
\{x \mapsto 0.0; y \mapsto 0.3\} & \preccurlyeq & \{x \mapsto 0.5; y \mapsto 0.8\} & \preccurlyeq & \{x \mapsto 0.7; y \mapsto 1.0\} & \preccurlyeq & \{x \mapsto 0.9; y \mapsto \infty.2\} \\
\rotatebox{-90}{$\models$} & & \rotatebox{-90}{$\models$} & & \rotatebox{-90}{$\models$} & & \rotatebox{-90}{$\models$} \\
\{(x,0)\}\{(y,0)\} & \preceq & \emptyset\{(x,0)\}\{(y,0)\} & \preceq & \{(y,1)\}\{(x,0)\} & \preceq & \emptyset\{(y,\infty)\}\{(x,0)\}
\end{array}$$

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 except comparison constraints.*
Copying. *$\mathcal{D}(\lambda_1[x := y]) = \mathcal{D}(\lambda_2[x := y])$ 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* $\boldsymbol{d}[x := y]$, *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 $y - x = 1.2$ in η'.

Digital Semantics. We use $\boldsymbol{D}, \dots$ to denote a digital valuation on $\underset{\bullet}{\mathbb{X}} \cup \dot{\mathbb{X}}$. 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**.

$$\begin{aligned}
&\boldsymbol{D}_1 \xrightarrow{\textsf{push}(\emptyset)} \boldsymbol{D}_1^{\mathrm{r}}\,\boldsymbol{D}_2 \xrightarrow{x \leftarrow [1,2]} \boldsymbol{D}_1^{\mathrm{r}}\,\boldsymbol{D}_2' \overset{2.0}{\rightsquigarrow} \boldsymbol{D}_1^{\mathrm{r}}\,\boldsymbol{D}_2'' \xrightarrow{\textsf{pop}(\{x\})} \boldsymbol{D},\\
&\boldsymbol{D}_1^{\mathrm{r}} = \mathcal{D}(\Lambda_1^{\mathrm{reset}}) = \{(\dot{\mathtt{C}},0),(\dot{x},2)\}\{(\underset{\bullet}{x},0)\},\quad \boldsymbol{D}_2 = \{(\underset{\bullet}{\mathtt{C}},0),(\dot{x},0),(\underset{\bullet}{x},2)\},\\
&\boldsymbol{D}_2' = \{(\underset{\bullet}{\mathtt{C}},0),(\underset{\bullet}{x},2)\}\{(\dot{x},1)\},\qquad\qquad\quad \boldsymbol{D}_2'' = \{(\underset{\bullet}{\mathtt{C}},2),(\underset{\bullet}{x},\infty)\}\{(\dot{x},\infty)\}.
\end{aligned}$$

To perform $\underset{\bullet}{\textsf{pop}}(\{x\})$, we should compute $\boldsymbol{D}_1'$ such that $\boldsymbol{D}_1^{\mathrm{r}} \preceq \boldsymbol{D}_1'$ and $\boldsymbol{D}_1' \parallel \boldsymbol{D}_2''$ and $\boldsymbol{D}_1' = \{(\dot{\mathtt{C}},2),(\dot{x},\infty)\}\{(\underset{\bullet}{x},\infty)\}$ can be easily obtained. Then,

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

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

Digital valuations lead 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 appendix. 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 $\Lambda_1 \dots \Lambda_n \models \boldsymbol{D}_1 \dots \boldsymbol{D}_n$ 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 lemma 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 $\langle q_{init}, \langle \bot, (\mathcal{D} \circ \mathcal{C})(\underset{\bullet}{\mathbf{0}}_{\mathbb{X}} \cup \dot{\mathbf{0}}_{\mathbb{X}})\rangle\rangle \to^* \langle q_{final}, \boldsymbol{W}\rangle$.*

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 and given a two-stage construction to show the decidability of the reachability problem. The intermediate semantics separated the involved backward-forward simulation of Abdulla et al. into two simple simulations, and this made our proof easy to follow.

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 comparison $x \bowtie y$ 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, considering the combination of bounded 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 and try to clarify the decidability of the reachability problem for such the combination.

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)

A Proof of Theorem 2

We show the following theorem:

Theorem 2. The following language cannot be recognized by any TPDA with diagonal constraints:

$L_{\text{ex}} \triangleq \{(a,t_1)(a,t_2)\ldots(a,t_n)(b,t'_n)\ldots(b,t'_2)(b,t'_1) : \delta_i \in \mathbb{N} \text{ and } \delta_i \geq \delta_j \text{ if } i < j\}$,

where $\delta_i = t'_i - t_i$.

By the result of Clemente and Lasota in [8], we can translate any TPDA with diagonal constraints $\mathcal{T}$ into a corresponding TPDA with an untimed stack $\mathcal{T}_u$ where a configuration of $\mathcal{T}_u$ is a triple $\langle q, \mathtt{X}, \gamma_1\,\gamma_2\,\ldots\,\gamma_n\rangle$. Then $\mathcal{T}_u$ can be seen as a timed automaton with a stack. We first show that the above language L_{ex} cannot be recognized by any (ordinary) timed automata and next, we extend an argument for timed automata to show that L_{ex} cannot be recognized by any TPDA with an untimed stack.

Timed automata and n-clock language

A timed automaton is a tuple $\mathcal{A} = (Q, q_{\text{init}}, F, \Sigma, X, \Delta)$ where

- Q is a finite set of locations;
- q_{init} is the initial location; $F \subseteq Q$ is the final locations;
- Σ is a finite input alphabet;
- X is a finite set of clocks;
- Δ is a finite set of (discrete) transition rules.

Each transition rule is a form of $p \xrightarrow{\alpha,\psi,Y} q \in \Delta$ where p and q are in Q, $\alpha \in \Sigma \cup \{\epsilon\}$, $Y \subseteq X$ is a subset of X, and $\psi \in \Psi_X$ is a diagonal constraint. Recall the definition of Ψ_X:

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

A configuration $\langle q, \nu\rangle$ is a pair of a location q and a valuation $\nu : X \to \mathbb{Q}^+$ on X. For a configuration $\langle p, \nu\rangle$, we write $\langle p, \nu\rangle \xrightarrow{\alpha} \langle q, \nu'\rangle$ where $\nu' = \nu[Y := 0]$ if $\nu \models \psi$ and there is a rule $p \xrightarrow{\alpha,\psi,Y} q \in \Delta$. We have another kind of transitions, timed transitions. Each timed transition increases the values of all the clock of a configuration by $\delta \in \mathbb{Q}^+$: $\langle p, \nu\rangle \overset{\delta}{\rightsquigarrow} \langle p, \nu'\rangle$ where $\nu' \triangleq \nu + \delta$.

Runs π and timed words $\mathrm{tw}(\pi)$ of $\mathcal{A}$ are defined by the same manner of TPDA. The language of $\mathcal{A}$ is defined by the runs from q_{init} to q_{final}:

$$\mathcal{L}(\mathcal{A}) \triangleq \{\mathrm{tw}(\pi) : \pi = \langle q_{\text{init}}, \mathbf{0}_X\rangle \rightsquigarrow \cdots \to \langle q_{\text{final}}, \nu\rangle,\ q_{\text{final}} \in F\}.$$

Let $\mathcal{A} = (Q, q_{\text{init}}, F, \Sigma, X, \Delta)$ be a timed automaton $\mathcal{A}$. If the number of the clocks in $\mathcal{A}$ is n, $|X| = n$, then we call $\mathcal{A}$ a n-clock timed automaton. Furthermore, if a language L is recognized by a n-clock timed automaton but not by any m-clock timed automata for $m < n$, then we call L n-clock language.

Below we will define n-clock languages $L_1, L_2, \ldots$ and see $L_{\text{ex}} = \bigcup_{i=1} L_i$. These results immediately imply that L_{ex} cannot be recognized by any timed automata.

A.1 n-clock timed languages: $L_1, L_2, L_3, \ldots$

We prepare technical lemmas for indistinguishable configurations of timed automata. Let $\langle q, \nu \rangle$ and $\langle q', \nu' \rangle$ be configurations of a timed automaton. We write $\langle q, \nu \rangle \cong \langle q', \nu' \rangle$ if these satisfy the following conditions:

- $q = q'$;
- For all $x, y \in X$, $\nu(x) \bowtie \nu(y)$ iff $\nu'(x) \bowtie \nu'(y)$.
- For all $x \in X$ and $k \in \mathbb{N}$, $\nu(x) \bowtie k$ iff $\nu'(x) \bowtie k$.

It can be verified that if $\langle q, \nu \rangle \cong \langle q, \nu' \rangle$ then there are no constraints $\psi \in \Psi_X$ such that $\nu \models \psi$ and $\nu' \not\models \psi$. Furthermore, we obtain the property about reachability.

Lemma A1. *Let c_1 and c_2 be indistinguishable configurations, $c_1 \cong c_2$.*

- *If $c_1 \xrightarrow{\alpha} c_1'$, then there exists c_2' such that $c_2 \xrightarrow{\alpha} c_2'$ and $c_1' \cong c_2'$.*
- *If $c_1 \overset{\delta}{\rightsquigarrow} c_1'$, then there exists δ' such that $c_2 \overset{\delta'}{\rightsquigarrow} c_2'$ and $c_1' \cong c_2'$.*

This immediately implies that if a location q is reachable from c_1, then c_2 also reaches q.

Lemma A2. *If $\langle q, \nu \rangle \overset{\delta}{\rightsquigarrow} \langle q, \nu' \rangle$ and there are no clocks x such that $\{\nu'(x)\} = 0.0$, then there is δ' such that $0 < \delta' < 1$, $\langle q, \nu \rangle \overset{\delta + \delta'}{\rightsquigarrow} \langle q, \nu'' \rangle$, and $\langle q, \nu' \rangle \cong \langle q, \nu'' \rangle$.*

Proof. It suffices to take $\delta' = \frac{M}{2}$ where $M = \max\{1.0 - \{\nu'(x)\} : x \in X\}$.

L_1 : one-clock timed language

$$L_1 \triangleq \{(a, t_1)(b, t_1') : t_1' - t_1 \in \mathbb{N}\}.$$

For example, $(a, 0.0)(b, 1.0), (a, 0.3)(b.2.3), (a, 1.8)(b, 5.8) \in L_1$.
But $(a, 0.0)(b, 1.2), (a, 0.3)(b, 2.4), (a, 1.8)(b, 5.9) \notin L_1$. We can show that this language is recognized by a *one-clock* timed automaton, and clearly this language cannot be recognized by any zero-clock timed automata.

L_2 : two-clock timed language

The next language L_2 cannot be recognized by any *one-clock* timed automata.

$$L_2 \triangleq \{(a_1, t_1)(a_2, t_2)(b_2, t_2')(b_1, t_1') : \delta_1, \delta_2 \in \mathbb{N},\ \delta_1 \geq \delta_2\},$$

where $\delta_i = t_i' - t_i$. For example, $(a_1, 0.6)(a_2, 1.2)(b_2, 3.2)(b_1, 5.6) \in L_2$ because of $5.6 - 0.6, 3.2 - 1.2 \in \mathbb{N}$. But $(a_1, 0.6)(a_2, 1.2)(b_2, 3.2)(b_1, 5.5) \notin L_2$ because of $5.5 - 0.6 \notin L_2$.

Proposition A1. *This language is not a* one-clock *language.*

Proof. Intuitively, we should remember the exact fractional parts of ages of a_1 and a_2. We use the term age of a symbol a_i to denote the time passed after we receive it from the input stream. However, since there is only a single clock in one-clock timed automata, we cannot keep both of the values. This implies L_2 is not a one-clock language.

Here, we assume that there is a one-clock automaton $\mathcal{A}$ and write x for its clock. Let us consider the following accepting run π such that $\mathrm{tw}(\pi) \in L_2$:

$$\langle q_{\mathrm{init}}, \mathbf{0}_X \rangle \overset{\delta_1}{\rightsquigarrow} \xrightarrow{a_1} \cdots \overset{\delta_i}{\rightsquigarrow} \xrightarrow{a_2} \overset{\delta_{i+1}}{\rightsquigarrow} \cdots \overset{\delta_j}{\rightsquigarrow} \langle q, \nu \rangle \xrightarrow{b_2} \langle q', \nu' \rangle \overset{\delta_{j+1}}{\rightsquigarrow} \cdots \overset{\delta_k}{\rightsquigarrow} \xrightarrow{b_1} \overset{\delta_{k+1}}{\rightsquigarrow} \cdots \xrightarrow{\epsilon} \langle q_f, \nu_f \rangle.$$

Especially, we choose π so that π satisfies: $0 < \delta_{j+1} + \delta_{j+2} + \cdots + \delta_k < 1$. We can do this because L_2 contains a word $(a_1, 0.0)(a_2, 0.5)(b_2, 1.5)(b_1, 2.0)$ with $0 < 2.0 - 1.5 < 1$. This condition will be used in the last part of the proof.

$$\langle q_{\mathrm{init}}, \mathbf{0}_X \rangle \overset{\delta_1}{\rightsquigarrow} \xrightarrow{a_1} \cdots \overset{\delta_i}{\rightsquigarrow} \xrightarrow{a_2} \overset{\delta_{i+1}}{\rightsquigarrow} \cdots \overset{\delta_j}{\rightsquigarrow} \langle q, \nu \rangle \xrightarrow{b_2} \langle q', \nu' \rangle \underbrace{\overset{\delta_{j+1}}{\rightsquigarrow} \cdots \overset{\delta_k}{\rightsquigarrow}}_{\in (0,1)} \xrightarrow{b_1} \overset{\delta_{k+1}}{\rightsquigarrow} \cdots \xrightarrow{\epsilon} \langle q_f, \nu_f \rangle.$$

The proof is showing the following at reading b_2:

1. There exists a clock x such that x keeps the exact fractional parts of the age of a_2. Namely, $\{\nu(x)\} = 0.0$.
2. There exists a clock y such that y keeps the exact fractional parts of the age of a_1. Namely, $\{\nu(y)\} = \delta_1 + \delta_2 + \cdots + \delta_j = 1.0 - (\delta_{j+1} + \delta_{j+2} + \cdots + \delta_k)$.
3. By $0 < \delta_{j+1} + \delta_{j+2} + \cdots + \delta_k < 1$, we cannot achieve both in a one-clock timed automaton.

If $\{\nu(x)\} \neq 0.0$, we can find δ from Lemma A2 such that

$$\pi' = c_0 \overset{\delta_1}{\rightsquigarrow} \xrightarrow{a_1} \cdots \overset{\delta_i}{\rightsquigarrow} \xrightarrow{a_2} \overset{\delta_{i+1}}{\rightsquigarrow} \cdots \overset{\delta_j + \delta}{\rightsquigarrow} \langle q, \mu \rangle \xrightarrow{b_2} \langle q', \mu' \rangle$$

where $\langle q, \nu \rangle \cong \langle q, \mu \rangle$ and $\langle q', \nu' \rangle \cong \langle q', \mu' \rangle$. By Lemma A1, $\mathcal{A}$ reach q_f from $\langle q', \mu' \rangle$ and accepts $\mathrm{tw}(\pi')$. However, $\mathrm{tw}(\pi') \notin L_2$ because of $0 < \delta < 1$.

Next, we consider the case $\{\nu(x)\} = 0.0$ and focus on $\langle q, \nu \rangle \xrightarrow{b_2} \underbrace{\cdots}_{\in (0,1)} \xrightarrow{b_1} c$ as follows:

$$\langle q, \nu \rangle \xrightarrow{b_2} \langle q', \nu' \rangle \overset{\delta_{j+1}}{\rightsquigarrow} \cdots \xrightarrow{\epsilon} \langle p, \vartheta \rangle \overset{\delta_l}{\rightsquigarrow} \langle p, \vartheta' \rangle \xrightarrow{\epsilon} \overset{0.0}{\rightsquigarrow} \xrightarrow{\epsilon} \overset{0.0}{\rightsquigarrow} \cdots \xrightarrow{b_1} \langle q_b, \nu_b \rangle$$

where $\delta_l > 0$ and $\overset{\delta_l}{\rightsquigarrow}$ is the last non-zero timed transition among $\langle q, \nu \rangle \xrightarrow{b_2} \cdots \xrightarrow{b_1} \langle q_b, \nu_b \rangle$. If $\{\vartheta'(x)\} \neq 0.0$, from Lemma A2, we can find δ such that:

$$\pi'' = \langle q, \nu \rangle \xrightarrow{b_2} \langle q', \nu' \rangle \overset{\delta_{j+1}}{\rightsquigarrow} \cdots \xrightarrow{\epsilon} \langle p, \vartheta \rangle \overset{\delta_l + \delta}{\rightsquigarrow} \langle p, \vartheta' \rangle \xrightarrow{\epsilon} \overset{0.0}{\rightsquigarrow} \xrightarrow{\epsilon} \overset{0.0}{\rightsquigarrow} \cdots \xrightarrow{b_1} \langle q_b, \nu'_b \rangle.$$

By Lemma A1, we obtain $\langle q_b, \nu_b \rangle \cong \langle q_b, \nu'_b \rangle$ and $\mathcal{A}$ reach q_f from $\langle q_b, \nu'_b \rangle$. However $\mathrm{tw}(\pi'')$ is a word of $\mathcal{A}$, $\mathrm{tw}(\pi'') \notin L_2$ because of $0 < \delta < 1$.

Finally, we consider the case $\{\vartheta'(x)\} = 0.0$. Now the following hold for $\langle q', \nu' \rangle \overset{\delta_{j+1}}{\rightsquigarrow} \cdots \xrightarrow{\epsilon} \langle p, \vartheta \rangle \overset{\delta_l}{\rightsquigarrow} \langle p, \vartheta' \rangle$:

$$0 < \delta_{j+1} + \delta_{j+2} + \cdots + \delta_l < 1, \quad \delta_l \neq 0.0, \quad \text{and} \quad \{\vartheta'(x)\} = 0.0.$$

These mean that $\mathcal{A}$ never resets x among $\langle q', \nu' \rangle \overset{\delta_{j+1}}{\rightsquigarrow} \cdots \overset{\delta_l}{\rightsquigarrow} \langle p, \vartheta' \rangle$. Indeed, if $\mathcal{A}$ resets x among the computation, then $\{\vartheta'(x)\} \neq 0.0$ and it contradicts to $\{\vartheta'(x)\} = 0.0$. Furthermore, we obtain $\{\nu'(x)\} = 1.0 - (\delta_{j+1} + \delta_{j+2} + \cdots + \delta_l)$. But, this implies $\{\nu(x)\} \neq 0.0$ and makes a contradiction because of $\{\nu(x)\} = 0.0$. $\square$

The last technical argument is summarized as follows.

Lemma A3. *Let $\mathcal{A}$ be a timed automata and $\langle q, \nu \rangle \overset{\delta_1}{\rightsquigarrow} \cdots \overset{\delta_n}{\rightsquigarrow} \langle q', \nu' \rangle$ be a computation.*

$$\text{If} \begin{cases} \delta_1 + \delta_2 + \cdots + \delta_n < 1, \\ \{\nu'(x)\} = 0.0, \\ \delta_n > 0, \end{cases} \qquad \text{then } \{\nu(x)\} = 1 - (\delta_1 + \delta_2 + \cdots + \delta_n).$$

L_3 : three-clock timed language

The next language L_3 cannot be recognized by any *two-clock* timed automata.

$$L_3 \triangleq \{(a_1, t_1)(a_2, t_2)(a_3, t_3)(b_3, t'_3)(b_2, t'_2)(b_1, t'_1) : \delta_1, \delta_2, \delta_3 \in \mathbb{N},\ \delta_1 \geq \delta_2 \geq \delta_3\},$$

where $\delta_i = t'_i - t_i$.

Proposition A2. *This language is not a* two-clock *language.*

Proof. We show this proposition by the same argument of Proposition A1. Let $\mathcal{A}$ be a two-clock automaton accepting L_3 and we use x and y to denote the clocks of $\mathcal{A}$. We can choose the following accepting run:

$$\langle q_{\text{init}}, \mathbf{0} \rangle \overset{\delta_1}{\rightsquigarrow} \cdots \xrightarrow{a_1} \cdots \xrightarrow{a_2} \cdots \xrightarrow{a_3} \cdots \overset{\delta_n}{\rightsquigarrow} \langle q, \nu \rangle \xrightarrow{b_3} \underbrace{\underbrace{\cdots}_{d \in (0,1)} \xrightarrow{b_2} \cdots}_{d' \in (0,1)} \xrightarrow{b_1} \cdots \xrightarrow{\epsilon} \langle q_f, \nu_f \rangle$$

where $d < d'$ because L_3 contains $(a_1, 0)(a_2, 0.5)(a_3, 0.25)(b_3, 1.25)(b_2, 1.5)(b_1, 2)$. As the same as the proof of Proposition A1, at least one clock z satisfies $\{\nu(z)\} = 0.0$. Here we assume x satisfies $\{\nu(x)\} = 0.0$.

Next, we further analysis the computation among $\langle q, \nu \rangle \xrightarrow{b_3} \cdots \xrightarrow{\epsilon} \langle q_f, \nu_f \rangle$.

$$\langle q, \nu \rangle \xrightarrow{b_3} \overset{\delta_{n+1}}{\rightsquigarrow} \cdots \overset{\delta_{n+m}}{\rightsquigarrow} \langle q', \nu' \rangle \xrightarrow{\epsilon} \overset{0.0}{\rightsquigarrow} \xrightarrow{\epsilon} \overset{0.0}{\rightsquigarrow} \cdots \xrightarrow{b_2} \langle q_{b_2}, \nu_{b_2} \rangle \overset{\delta_k}{\rightsquigarrow} \cdots \xrightarrow{\epsilon} \langle q_f, \nu_f \rangle$$

where $\delta_{n+m} > 0.0$. By the same argument of Proposition A1, a clock z that satisfies $\{\nu'(z)\} = 0.0$ is needed. Furthermore, from Lemma A3, $\{\nu(z)\} = 1 - (\delta_{n+1} + \delta_{n+2} + \cdots + \delta_{n+m})$ and $\{\nu(z)\} \neq 0.0$. Since we have only two clocks, here we have to use y for this z.

Finally, we consider the following part:

$$\langle q, \nu \rangle \xrightarrow{b_3} \overset{\delta_{n+1}}{\rightsquigarrow} \cdots \xrightarrow{b_2} \langle q_{b_2}, \nu_{b_2} \rangle \overset{\delta_k}{\rightsquigarrow} \cdots \overset{\delta_{k+l}}{\rightsquigarrow} \langle q'', \nu'' \rangle \xrightarrow{\epsilon} \overset{0.0}{\rightsquigarrow} \xrightarrow{\epsilon} \overset{0.0}{\rightsquigarrow} \cdots \xrightarrow{b_1} \langle q_{b_1}, \nu_{b_1} \rangle$$

where $\delta_{k+l} > 0.0$. Again, by applying the same argument of a_2 and b_2 to a_1 and b_1, a clock z is required such that z satisfies $\{\nu''(z)\} = 0.0$. Furthermore, Lemma A3 implies $\{\nu(z)\} = 1 - (\delta_{n+1} + \cdots + \delta_{n+m}) - (\delta_k + \delta_{k+1} + \cdots + \delta_{k+l})$. However, this requirement cannot be satisfied because we already used x and y for $\{\nu(x)\} = 0.0$ and $\{\nu(y)\} = 1 - (\delta_{n+1} + \cdots + \delta_{n+m})$. ☐

The rest of n-clock languages: L_4, L_5, ...

We define the other n-clock languages $L_4, L_5, \ldots$ in a similar way as L_1, L_2, and L_3. Then each language L_n is a n-clock language.

To apply the same argument of Proposition A1 and A2, we have to ensure a run of the following form:

$$\pi = c_0 \overset{\delta_1}{\rightsquigarrow} \cdots \xrightarrow{a_1} \cdots \xrightarrow{a_n} \cdots \xrightarrow{b_n} \underbrace{\cdots}_{\in(0,1)} \xrightarrow{b_1} \cdots \xrightarrow{\epsilon} \langle q_{\mathrm{final}}, \nu\rangle$$

where $\mathrm{tw}(\pi) \in L_n$. From the definition of L_n, we have the following word in L_n:

$$(a_1, t_1)(a_2, t_2) \cdots (a_n, t_n)(a_{n+1}, t_{n+1})(b_{n+1}, t'_{n+1})(b_n, t'_n) \cdots (b_2, t'_2)(b_1, t'_1)$$

where

- $t_1 = 0.0$ and $t'_{n+1} - t_{n+1} = 1.0$;
- $t_{i+1} - t_i = 1 - {}^1/_{2^i}$ for all $i \in [1..n]$;
- $t'_i - t'_{i+1} = {}^1/_{2^i}$ for all $i \in [1..n]$.

For example,

- $(a_1, 0)\,(a_2, {}^1/_2)\,(b_2, {}^3/_2)\,(b_1, 2) \in L_2$.
- $(a_1, 0)\,(a_2, {}^2/_4)\,(a_3, {}^5/_4)\,(b_3, {}^9/_4)\,(b_2, {}^{10}/_4)\,(b_1, 3) \in L_3$.
- $(a_1, 0)\,(a_2, {}^4/_8)\,(a_3, {}^{10}/_8)\,(a_4, {}^{17}/_8)\,(b_4, {}^{25}/_8)\,(b_3, {}^{26}/_8)\,(b_2, {}^{28}/_8)\,(b_1, 4) \in L_4$.

  $\vdots$

As summary, we obtain the lemma.

Lemma A4. *The language L_n is a n-clock language.*

L_{ex} : unboundedly many clocks timed language

We revisit

$L_{\mathrm{ex}} \triangleq \{(a, t_1)(a, t_2) \dots (a, t_n)(b, t'_n) \dots (b, t'_2)(b, t'_1) : \delta_i \in \mathbb{N} \text{ and } \delta_i \geq \delta_j \text{ if } i < j\}$,

where $\delta_i = t'_i - t_i$. This language can be seen as $L_{\mathrm{ex}} = \bigcup_{i=1} L_i$, and thus this language cannot be recognized by any timed automata.

Theorem A1. *The language L_{ex} is not timed automata.*

By using the Clemente's result—each TPDA with diagonal constraints is translated into a corresponding TPDA with an untimed stack—, we show that L_{ex} cannot be recognized by any TPDA with diagonal constraints.

We briefly review the definition of TPDA with an untimed stack. TPDA with an untimed stack $\mathcal{T}_u$ is a tuple $(Q, q_{\mathrm{init}}, q_{\mathrm{final}}, \Sigma, \Gamma, \mathcal{X}, \Delta)$ and a configuration $\langle q, \mathtt{X}, \gamma_1\,\gamma_2 \dots \gamma_n\rangle$ is a triple of a location q, a valuation $\mathtt{X}$ on $\mathcal{X}$, and a stack where $\gamma_i \in \Gamma$. Note that there are no rational values in a stack.

There are four kinds of discrete operations for Δ:

$$\begin{aligned}
&\mathsf{push}(\gamma) : \langle p, \mathtt{X}, w\rangle \to \langle q, \mathtt{X}, w\,\gamma\rangle,\\
&\mathsf{reset}(Y) : \langle p, \mathtt{X}, w\rangle \to \langle q, \mathtt{X}[Y \coloneqq 0], w\rangle \text{ where } Y \subseteq \mathcal{X},\\
&\mathsf{pop}(\gamma, \psi) : \langle p, \mathtt{X}, w\,\gamma\rangle \to \langle q, \mathtt{X}, w\rangle \text{ if } \mathtt{X} \models \psi \text{ where } \psi \in \Psi_{\mathcal{X}},\\
&\mathsf{check}(\psi) : \langle p, \mathtt{X}, w\rangle \to \langle q, \mathtt{X}, w\rangle \text{ if } \mathtt{X} \models \psi \text{ where } \psi \in \Psi_{\mathcal{X}}.
\end{aligned}$$

Recall the constraints $\psi \in \Psi_X$ in their system are given as follows:

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

As timed automata, we define the notion of indistinguishability and write $\langle q, \mathtt{X}, w\rangle \cong \langle q', \mathtt{X}', w'\rangle$ if the following are satisfied:

- $q = q'$ and $w = w'$.
- For all $x, y \in X$, $\mathtt{X}(x) \bowtie \mathtt{X}(y)$ iff $\mathtt{X}'(x) \bowtie \mathtt{X}'(y)$.
- For all $x \in X$ and $k \in \mathbb{N}$, $\mathtt{X}(x) \bowtie k$ iff $\mathtt{X}'(x) \bowtie k$.

Then, the similar lemmas of Lemma A1, Lemma A2, and Lemma A3 hold for TPDA with an untimed stack. Since, by these lemmas, our proofs of Proposition A1, Proposition A2, and Lemma A4 are also effective for TPDA with an untimed stack, we obtain the following lemma

Lemma A5. *A language L_n cannot be recognized by any m-clock TPDA with an untimed stack for $m < n$.*

This lemma immediately implies our theorem.
Theorem 2. The following language cannot be recognized by any TPDA with diagonal constraints:

$$L_{\text{ex}} \triangleq \{(a, t_1)(a, t_2) \dots (a, t_n)(b, t'_n) \dots (b, t'_2)(b, t'_1) : \delta_i \in \mathbb{N} \text{ and } \delta_i \geq \delta_j \text{ if } i < j\},$$

where $\delta_i = t'_i - t_i$.

B Proof of Lemma 2

Lemma 2. *If $\mathit{WF}(\boldsymbol{w}\langle\gamma_1, \Lambda_1\rangle\langle\gamma_2, \Lambda_2\rangle)$, then there exists the* unique *Λ'_1 such that $\Lambda \preccurlyeq \Lambda'_1$ and $\Lambda'_1 \parallel \Lambda_2$.*

Proof. From the well-formedness, $\Lambda_1(\dot{\mathtt{C}}) = 0.0$ is clear. Then we show the following claim:

$$\text{if } \Lambda_1 \preccurlyeq \Lambda'_1,\ \Lambda_1 \preccurlyeq \Lambda''_1, \text{ and } \Lambda'_1(\dot{\mathtt{C}}) = \Lambda''_1(\dot{\mathtt{C}}), \text{ then } \Lambda'_1 = \Lambda''_1.$$

Note that $\Lambda \preccurlyeq \Lambda'$ iff $\exists \delta. \Lambda + \delta = \Lambda'$ is trivial from the definition.

Case $\lfloor \Lambda'_1(\dot{\mathtt{C}}) \rfloor \neq \infty$: Then $\Lambda'_1(\dot{\mathtt{C}}) = \Lambda''_1(\dot{\mathtt{C}}) = (i, r)$ holds where $i \in [0..\mathsf{M}]$ and $r \in [0, 1)$. This means that $\Lambda'_1 = \Lambda_1 + (i + r)$ and $\Lambda''_1 = \Lambda_1 + (i + r)$.

Case $\lfloor \Lambda'_1(\dot{\mathtt{C}}) \rfloor = \infty$: Then $\Lambda'_1(\dot{\mathtt{C}}) = \Lambda''_1(\dot{\mathtt{C}}) = (\infty, r)$ holds where $r \in [0, 1)$. In contrast to the above case, in general, there maybe exist two distinct rationals δ_1 and δ_2 such that $\Lambda'_1 = \Lambda_1 + \delta_1$, $\Lambda''_1 = \Lambda_1 + \delta_2$. (Here, we assume $\delta_1 > \delta_2$.)
The following is easily verified:
  - $\delta_1 - \delta_2 \in \mathbb{N}$ because $\{\Lambda'_1(\dot{\mathtt{C}})\} = \{\Lambda''_1(\dot{\mathtt{C}})\}$.

Hence, $\{\Lambda'_1(x)\} = \{\Lambda''_1(x)\}$ for any $x \in \underset{\bullet}{\mathbb{X}} \cup \dot{\mathbb{X}}$.
The condition $\Lambda_1(\dot{\mathtt{C}}) = 0.0$ means that $\dot{\mathtt{C}}$ is the minimum clock in Λ_1. Now, $\lfloor \Lambda'_1(\dot{\mathtt{C}}) \rfloor = \lfloor \Lambda''_1(\dot{\mathtt{C}}) \rfloor = \infty$, hence any other clocks are also collapsed: $\lfloor \Lambda'_1(x) \rfloor = \lfloor \Lambda''_1(x) \rfloor = \infty$ for any $x \in \underset{\bullet}{\mathbb{X}} \cup \dot{\mathbb{X}}$.
By combining these $\{\Lambda'_1(x)\} = \{\Lambda''_1(x)\}$ and $\lfloor \Lambda'_1(x) \rfloor = \lfloor \Lambda''_1(x) \rfloor = \infty$, we obtain $\Lambda'_1 = \Lambda''_1$.

From the assumption $\mathsf{WF}(\boldsymbol{w}\langle\gamma_1, \Lambda_1\rangle\langle\gamma_2, \Lambda_2\rangle)$, there exists Λ'_1 such that $\Lambda_1 \preccurlyeq \Lambda'_1$ and $\Lambda'_1 \parallel \Lambda_2$. To show the uniqueness of Λ'_1, we assume a collapsed valuation Λ''_1 such that $\Lambda_1 \preccurlyeq \Lambda''_1$ and $\Lambda''_1 \parallel \Lambda_2$. From the definition, $\Lambda'_1(\dot{\mathtt{C}}) = \Lambda_2(\underset{\bullet}{\mathtt{C}}) = \Lambda''_1(\dot{\mathtt{C}})$. Then, applying the above claim, we obtain $\Lambda'_1 = \Lambda''_1$. □

C Proof of Lemma 4

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

Proof. First, we define the time-passage quasi-ordering for digital valuations in a different manner. For digital valuations $\boldsymbol{d}$ and $\boldsymbol{d'}$, the time-passage relation $\boldsymbol{d} \vdash \boldsymbol{d'}$ is defined as follows:

Small Elapse $d_0\, d_1 \dots d_n \vdash \emptyset\, d'_0\, d_1 \dots d_n$ if $d_0 \neq \emptyset$ and d'_0 satisfies the following:

- If $(x, \mathsf{M}) \in d_0$, then $(x, \infty) \in d'_0$; and
- If $(x, v) \in d_0$ and $v \neq \mathsf{M}$, then $(x, v) \in d'_0$.

Carry $\emptyset\, d_1 \dots d_{n-1}\, d_n \vdash d'_n\, d_1 \dots d_{n-1}$ if d'_n satisfies the following:

- If $(x, v) \in d_n$ and $v < \mathsf{M}$, then $(x, v+1) \in d'_n$; and
- If $(x, v) \in d_n$ and $v = \infty$, then $(x, \infty) \in d'_n$.

Remark: $(x, \mathsf{M}) \in d_n$ never happens from the condition "the constant M only appears at d_0".

We write $\boldsymbol{d} \vdash^* \boldsymbol{d'}$ for the reflexive transitive closure of $\vdash$. Then the following claim is easily checked.

Claim (A). For any digital valuations $\boldsymbol{d}$ and $\boldsymbol{d'}$, $\boldsymbol{d} \preceq \boldsymbol{d'}$ iff $\boldsymbol{d} \vdash^* \boldsymbol{d'}$.

It suffices to show the following property.

Claim (B). If $\boldsymbol{d} \vdash \boldsymbol{d'}$ and $\lambda' \models \boldsymbol{d'}$, then there exists λ such that $\lambda \preccurlyeq \lambda'$ and $\lambda \models \boldsymbol{d}$.

Let η' be a concrete valuation such that $\mathcal{C}(\eta') = \lambda'$. This claim is shown by case analysis on $\boldsymbol{d} \vdash \boldsymbol{d'}$.

Case $d_0 d_1 \dots d_n \vdash \emptyset d'_0 d_1 \dots d_n$: From the definition, there exists a clock z such that $z \in d_0$. We write δ to denote $\lambda'(z)$ and define λ as follows:

$$\lambda(x) = (i, r) \iff \langle x, i \rangle \in \boldsymbol{d} \wedge \lambda'(x) = (i', r + \delta).$$

(We used $\langle x, i \rangle \in \boldsymbol{d}$ instead of $\exists k \in [0..n].\, (x, i) \in d_k$.)
Then, it is easily verified that $\lambda \models \boldsymbol{d}$, $\lambda + \delta = \lambda'$, and $\lambda \preccurlyeq \lambda'$.

Case $\emptyset d_1 \dots d_{n-1} d_n \vdash d'_n d_1 \dots d_{n-1}$: From the definition of digital valuations, there exists a clock z such that $z \in d_1$. We write δ for $\frac{\lambda'(z)}{2}$ and define λ as follows:

$$\lambda(x) = (i, r) \iff \langle x, i \rangle \in \boldsymbol{d} \wedge \lambda'(x) = (i', \{r + \delta\}).$$

(We used $\langle x, i \rangle \in \boldsymbol{d}$ instead of $\exists k \in [1..n].\, (x, i) \in d_k$.)
Then, it is easily verified that $\lambda \models \boldsymbol{d}$, $\lambda + \delta = \lambda'$, and $\lambda \preccurlyeq \lambda'$.

Finally, **Claim** (A) and (B) conclude Lemma 4. □

D Missing definitions and proof of Section 4.2

We use $\mathbb{D}$ for the set of digital valuations on $\underset{\bullet}{\mathbb{X}} \cup \dot{\mathbb{X}}$

Definition (Digital Semantics Digi). We define the *finite*-PDS $(Q, \Gamma \times \mathbb{D}, \hookrightarrow)$ where $\langle q, \boldsymbol{W} \rangle \hookrightarrow \langle q', \boldsymbol{W'} \rangle$ if there exists $\langle q, \tau, q' \rangle \in \Delta$ 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:

$$\frac{\boldsymbol{D}_1^{\mathrm{r}} = \boldsymbol{D}_1[\dot{\mathsf{C}} \coloneqq 0] \quad \boldsymbol{D}_1^{\mathrm{r}} \parallel \boldsymbol{D}_2 \quad \boldsymbol{D}_2|\dot{\mathbb{X}} = (\mathcal{D} \circ \mathcal{C})(\dot{\mathbf{0}}_{\mathbb{X}})}{\langle \gamma, \boldsymbol{D}_1 \rangle \hookrightarrow \langle \gamma, \boldsymbol{D}_1^{\mathrm{r}} \rangle \, \langle \gamma', \mathcal{U}(X, \boldsymbol{D}_2) \rangle} \ \textsf{push}(\gamma', X) \qquad \frac{\boldsymbol{D}' \in \boldsymbol{D}[\dot{x} \leftarrow I]}{\langle \gamma, \boldsymbol{D} \rangle \hookrightarrow \langle \gamma, \boldsymbol{D}' \rangle} \ x \leftarrow I$$

$$\frac{\boldsymbol{D}_1 \preceq \boldsymbol{D}_1' \quad \boldsymbol{D}_1' \parallel \boldsymbol{D}_2 \quad \boldsymbol{D} \in \boldsymbol{D}_1' \oplus \mathcal{U}(\mathcal{X} \setminus X, \boldsymbol{D}_2)}{\langle \gamma, \boldsymbol{D}_1 \rangle \, \langle \gamma', \boldsymbol{D}_2 \rangle \hookrightarrow \langle \gamma, \boldsymbol{D} \rangle} \ \textsf{pop}(\gamma', X) \qquad \frac{\boldsymbol{D} \models \dot{\varphi}}{\langle \gamma, \boldsymbol{D} \rangle \hookrightarrow \langle \gamma, \boldsymbol{D} \rangle} \ \textsf{check}(\varphi)$$

In addition, the rules $\langle q, \langle \gamma, \boldsymbol{D} \rangle \rangle \hookrightarrow \langle q, \langle \gamma, \boldsymbol{D'} \rangle \rangle$ are added for all $q \in Q$, $\gamma \in \Gamma$, and $\boldsymbol{D} \preceq \boldsymbol{D'}$ to reflect timed transitions in Stnd.

As the Coll semantics, we define the well-formedness $\mathsf{WF}(\boldsymbol{W})$.
A stack $\langle \gamma_1, \boldsymbol{D}_1 \rangle \langle \gamma_2, \boldsymbol{D}_2 \rangle \ldots \langle \gamma_n, \boldsymbol{D}_n \rangle$ is *well-formed* WF if for all $i \in [1..(n-1)]$
– $\boldsymbol{D}_i \models \dot{\mathsf{C}} \in [0, 0]$ and there exists $\boldsymbol{D'_i}$ such that $\boldsymbol{D}_i \preceq \boldsymbol{D'_i}$ and $\boldsymbol{D'_i} \parallel \boldsymbol{D}_{i+1}$.

Next, we show the backward simulation lemma Lemma 6 below.

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 $\boldsymbol{w} \models \boldsymbol{W}$.*

D.1 Proof of Lemma 6

To show this lemma, we prepare technical notations.

- We define two renamings $\mathring{\varsigma} : \dot{\mathbb{X}} \to \mathbb{X}$ and $\underset{\circ}{\varsigma} : \underset{\bullet}{\mathbb{X}} \to \mathbb{X}$ naturally.
- Let $\boldsymbol{d} = d_0 d_1 \ldots d_n$ be a digital valuation on $\dot{\mathbb{X}}$. Then renamed digital valuation $\mathring{\varsigma}(\boldsymbol{d})$ on $\mathbb{X}$ is defined by naturally: $\mathring{\varsigma}(\boldsymbol{d}) \triangleq \mathring{\varsigma}(d_0)\mathring{\varsigma}(d_1) \ldots \mathring{\varsigma}(d_n)$.
- For $\boldsymbol{d}$ be a digital valuation on $\underset{\bullet}{\mathbb{X}}$, we also define $\underset{\circ}{\varsigma}(\boldsymbol{d})$ in a similar way.

Let $\boldsymbol{D}$ be a digital valuation on $\dot{\mathbb{X}} \cup \underset{\bullet}{\mathbb{X}}$. By using these unmarking notations, we define two extracted digital valuations $\mathring{\boldsymbol{D}}$ and $\underset{\circ}{\boldsymbol{D}}$: $\mathring{\boldsymbol{D}} \triangleq \mathring{\varsigma}(\boldsymbol{D}|\dot{\mathbb{X}})$ and $\underset{\circ}{\boldsymbol{D}} \triangleq \underset{\circ}{\varsigma}(\boldsymbol{D}|\underset{\bullet}{\mathbb{X}})$.

As an example, let us consider the following digital valuation

$$\boldsymbol{D} = \{(\dot{\mathsf{C}}, 0)\}\{(\dot{x}, 1), (\underset{\bullet}{y}, 3)\}\{(\underset{\bullet}{\mathsf{C}}, 4)\}\{(\dot{y}, 2), (\underset{\bullet}{x}, \infty)\}.$$

Then $\mathring{\boldsymbol{D}} = \{(\mathsf{C}, 0)\}\{(x, 1)\}\{(y, 2)\}$ and $\underset{\circ}{\boldsymbol{D}} = \emptyset\,\{(y, 3)\}\{(\mathsf{C}, 4)\}\{(x, \infty)\}$.

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 $\boldsymbol{w} \models \boldsymbol{W}$.*

Proof. We consider the case **time** : $\langle q, \boldsymbol{W} \rangle \to \langle q, \boldsymbol{W'} \rangle$ because the other rules **push**, **check**, and $x \leftarrow I$ are not difficult and the rule **pop** is proved by the same argument as here.
If $\boldsymbol{W} = \boldsymbol{D}$, then $\langle q, \langle \gamma, \boldsymbol{D} \rangle \rangle \to \langle q, \langle \gamma, \boldsymbol{D'} \rangle \rangle$ where $\boldsymbol{D} \preceq \boldsymbol{D'}$. This case is immediate from Lemma 4.

Otherwise, $\langle q, \boldsymbol{W}\langle \gamma_1, \boldsymbol{D}_1\rangle\langle \gamma_2, \boldsymbol{D}_2\rangle\rangle \to \langle q, \boldsymbol{W}\langle \gamma_1, \boldsymbol{D}_1\rangle\langle \gamma_2, \boldsymbol{D}_2'\rangle\rangle$ where $\boldsymbol{D}_2 \preceq \boldsymbol{D}_2'$. By Lemma 5, there is a well-formed stack such that $\boldsymbol{w}\,\langle \gamma_1 \Lambda_1\rangle\,\langle \gamma_2, \Lambda_2'\rangle$ and:

$$\boldsymbol{w}\,\Lambda_1\,\Lambda_2' \models \boldsymbol{W}\boldsymbol{D}_1\boldsymbol{D}_2'.$$

Since Λ_1 are collapsed valuations on $\dot{\mathbb{X}} \cup \underset{\bullet}{\mathbb{X}}$, we can decompose Λ_1 as $\Lambda_1 = \underset{\bullet}{\lambda}_0 \cup \dot{\lambda}_1$ where λ_0 and λ_1 are collapsed valuations on $\mathbb{X}$. For the sake of readability, we rewrite this as $\Lambda_1 = \langle \lambda_0, \lambda_1\rangle$ and then the above expression is also rewritten as follows:

$$\boldsymbol{w}\,\langle \lambda_0, \lambda_1\rangle\,\langle \lambda_1'', \lambda_2'\rangle \models \boldsymbol{W}\boldsymbol{D}_1\boldsymbol{D}_2'.$$

Since $\boldsymbol{D}_2 \preceq \boldsymbol{D}_2'$, by Lemma 4, we can find $\langle \lambda_1', \lambda_2\rangle$ such that $\langle \lambda_1', \lambda_2\rangle \preccurlyeq \langle \lambda_1'', \lambda_2'\rangle$ and $\langle \lambda_1', \lambda_2\rangle \models \boldsymbol{D}_2$. Then, $\boldsymbol{w}\langle \lambda_0, \lambda_1\rangle\langle \lambda_1', \lambda_2\rangle \models \boldsymbol{W}\boldsymbol{D}_1\boldsymbol{D}_2$ is clear.

Next, we show that $\boldsymbol{w}\langle \lambda_0, \lambda_1\rangle\langle \lambda_1', \lambda_2\rangle$ is a well-formed stack. From the well-formedness $\mathsf{WF}(\boldsymbol{w}\langle \lambda_0, \lambda_1\rangle\langle \lambda_1'', \lambda_2'\rangle)$, $\lambda_1(\dot{\mathtt{C}}) = 0.0$ and $\mathsf{WF}(\boldsymbol{w}\langle \lambda_0, \lambda_1\rangle)$ is clear. Thus it suffices to show $\lambda_1 \preccurlyeq \lambda_1'$.

The reference clock $\mathtt{C}$ plays an important role to show $\lambda_1 \preccurlyeq \lambda_1'$. The following diagram holds:

$$\begin{array}{ccccc} \lambda_1 & & \lambda_1' & & \lambda_1'' \\ \rotatebox{-90}{$\models$} & & \rotatebox{-90}{$\models$} & & \rotatebox{-90}{$\models$} \\ \boldsymbol{d} & \preceq & \boldsymbol{d'} & \preceq & \boldsymbol{d''} \end{array} \wedge \lambda_1 \preccurlyeq \lambda_1'' \wedge \lambda_1' \preccurlyeq \lambda_1'' \wedge \mathtt{C} \in_0 \boldsymbol{d},$$

where $\boldsymbol{d} = \mathring{\boldsymbol{D}}_1$, $\boldsymbol{d'} = \underset{\circ}{\boldsymbol{D}}_2$, $\boldsymbol{d''} = \underset{\circ}{\boldsymbol{D}}{}_2'$. We write $\mathtt{C} \in_0 \boldsymbol{d}$ to denote that $\boldsymbol{d} = d_0 d_1 \dots d_n$ and $(\mathtt{C}, 0) \in d_0$.

- $\boldsymbol{d} \preceq \boldsymbol{d'}$ and $\boldsymbol{d'} \preceq \boldsymbol{d''}$ follow from $\mathsf{WF}(\boldsymbol{W}\boldsymbol{D_1}\boldsymbol{D_2})$ and $\boldsymbol{D_2} \preceq \boldsymbol{D_2'}$, respectively.
- $\lambda_1 \preccurlyeq \lambda_1''$ and $\lambda_1' \preccurlyeq \lambda_1''$ follow from $\mathsf{WF}(\boldsymbol{w}\langle \lambda_0, \lambda_1\rangle\langle \lambda_1'', \lambda_2\rangle)$ and $\langle \lambda_1', \lambda_2\rangle \preccurlyeq \langle \lambda_1'', \lambda_2'\rangle$, respectively.
- $\mathtt{C} \in_0 \boldsymbol{d}$ follows from $\mathsf{WF}(\boldsymbol{W}\boldsymbol{D_1}\boldsymbol{D_2})$.

By Lemma D1 (this lemma will be shown in the next subsection), $\lambda_1 \preccurlyeq \lambda_1'$. □

D.2 The Role of the Reference Clock in Digitized Semantics

Our main statement of this appendix is the following:

$$\begin{array}{ccccc} \lambda & & \lambda' & & \lambda'' \\ \rotatebox{-90}{$\models$} & & \rotatebox{-90}{$\models$} & & \rotatebox{-90}{$\models$} \\ \boldsymbol{d} & \preceq & \boldsymbol{d'} & \preceq & \boldsymbol{d''} \end{array} \wedge \lambda \preccurlyeq \lambda'' \wedge \lambda' \preccurlyeq \lambda'' \wedge \big(\exists x. x \in_0 \boldsymbol{d}\big) \implies \begin{array}{ccccc} \lambda & \preccurlyeq & \lambda' & \preccurlyeq & \lambda'' \\ \rotatebox{-90}{$\models$} & & \rotatebox{-90}{$\models$} & & \rotatebox{-90}{$\models$} \\ \boldsymbol{d} & \preceq & \boldsymbol{d'} & \preceq & \boldsymbol{d''} \end{array}$$

From the definition of the quasi-ordering $\preccurlyeq$, the following desired property fails: If $\lambda \preccurlyeq \lambda''$ and $\lambda' \preccurlyeq \lambda''$, then $\lambda \preccurlyeq \lambda'$ or $\lambda' \preccurlyeq \lambda$. Consider:

$$\lambda = \{x \mapsto 0, y \mapsto 1\}, \lambda' = \{x \mapsto 0, y \mapsto 2\}, \lambda'' = \{x \mapsto \infty, y \mapsto \infty\}.$$

Since we collapse the domain, distance information of integral parts are lost. However, the conditions $\lambda \models \boldsymbol{d}$ recover the integral parts information.

We prepare several notations for the proof of this lemma.

Let λ and λ' be collapsed valuations on X.

$$
\begin{array}{lcl}
\{\lambda\}(x) & \triangleq & \{\lambda(x)\}, \\
\lfloor\lambda\rfloor(x) & \triangleq & \lfloor\lambda(x)\rfloor, \\
\lambda \doteqdot \lambda' & \iff & (\exists r.\, \{\lambda\} = \{\lambda' + r\}) \wedge (\exists r'.\, \{\lambda + r'\} = \{\lambda'\}).
\end{array}
$$

It is easily checked that this relation is an equivalence relation.

Proposition D1. *Let λ and λ' be collapsed valuations on X, and $\boldsymbol{d}$ be a digital valuation on X. Then the following properties hold clearly.*

1. *If $\lambda \preccurlyeq \lambda'$, then $\lambda \doteqdot \lambda'$.*
2. *If $\{\lambda\} = \{\lambda'\}$ and $\lfloor\lambda\rfloor = \lfloor\lambda'\rfloor$, then $\lambda = \lambda'$.*
3. *If $\lambda \models \boldsymbol{d}$ and $\lambda' \models \boldsymbol{d}$, then $\lfloor\lambda\rfloor = \lfloor\lambda'\rfloor$.*
4. *If $\lambda \doteqdot \lambda'$ and $\exists x.\, \{\lambda\}(x) = \{\lambda'\}(x)$, then $\{\lambda\} = \{\lambda'\}$.*

Proposition D2.

$$
\begin{array}{ccc}
\lambda & \preceq & \lambda' \\
\top\!\!\top & & \top\!\!\top \\
\boldsymbol{d} & = & \boldsymbol{d}
\end{array}
\wedge \big(\exists x. x \in_0 \boldsymbol{d}\big) \implies \lambda = \lambda'.
$$

Proof. $\lambda \models \boldsymbol{d}$ and $\lambda' \models \boldsymbol{d}$ ensures that $\lfloor\lambda\rfloor = \lfloor\lambda'\rfloor$.

From $\lambda \preccurlyeq \lambda'$, $\lambda \doteqdot \lambda'$. Since there exists a clock x such that $x \in_0 \boldsymbol{d}$, $\{\lambda(x)\} = \{\lambda'(x)\} = 0.0$. By this fact, we obtain $\{\lambda\} = \{\lambda'\}$ and $\lambda = \lambda'$. □

Proposition D3.

$$
\begin{array}{ccc}
\lambda & \doteqdot & \lambda' \\
\top\!\!\top & & \top\!\!\top \\
\boldsymbol{d} & \vdash & \boldsymbol{d'}
\end{array}
\implies \lambda \preccurlyeq \lambda'.
$$

Proof. We proceed by case analysis on $\boldsymbol{d} \vdash \boldsymbol{d'}$.

Case $\emptyset\, d_1 \cdots d_{n-1}\, d_n \vdash d'_n\, d_1 \cdots d_{n-1}$**:** There exists a clock x such that x is in d_n. We write δ for $1.0 - \{\lambda(x)\}$. Then, $\lambda' = \lambda + \delta$ is easily verified and hence $\lambda \preccurlyeq \lambda'$.

Case $d_0\, d_1 \cdots d_n \vdash \emptyset\, d'_0\, d_1 \cdots d_n$**:** Now, there exists a clock $x \in d_0$. We write δ for $\{\lambda'(x)\}$. Then, $\lambda = \lambda + \delta$ is easily verified and hence $\lambda \preccurlyeq \lambda'$.

□

Proposition D4.

$$
\begin{array}{ccc}
\lambda & \doteqdot & \lambda' \\
\top\!\!\top & & \top\!\!\top \\
\boldsymbol{d} & \vdash^+ & \boldsymbol{d'}
\end{array}
\implies \lambda \preccurlyeq \lambda'.
$$

Proof. Base case $\boldsymbol{d} \vdash \boldsymbol{d'}$ is shown by Proposition D3.

We consider the induction step:

$$
\begin{array}{ccccc}
\lambda & & & & \lambda' \\
\top\!\!\top & & & & \top\!\!\top \\
\boldsymbol{d} & \vdash & \boldsymbol{d''} & \vdash^+ & \boldsymbol{d'}
\end{array}
\wedge \lambda \doteqdot \lambda' \implies \lambda \preccurlyeq \lambda'.
$$

From Lemma 4, there exists a valuation λ'' such that:

$$\begin{array}{ccccc} \lambda & \preccurlyeq & \lambda'' & & \lambda' \\ \text{⫪} & & \text{⫪} & & \text{⫪} \\ \boldsymbol{d} & \vdash & \boldsymbol{d''} & \vdash^{+} & \boldsymbol{d'} \end{array}.$$

Then, $\lambda'' \doteqdot \lambda'$ from $\lambda \preccurlyeq \lambda''$ and $\lambda \doteqdot \lambda'$. So, we can apply the induction hypothesis and obtain $\lambda'' \preccurlyeq \lambda'$. Finally, from the transitivity of the quasi-ordering $\preccurlyeq$, we obtain $\lambda \preccurlyeq \lambda'$. □

Lemma D1.

$$\begin{array}{ccccc} \lambda & & \lambda' & & \lambda'' \\ \text{⫪} & & \text{⫪} & & \text{⫪} \\ \boldsymbol{d} & \preceq & \boldsymbol{d'} & \preceq & \boldsymbol{d''} \end{array} \wedge \lambda \preccurlyeq \lambda'' \wedge \lambda' \preccurlyeq \lambda'' \wedge \big(\exists x. x \in_0 \boldsymbol{d}\big) \implies \begin{array}{ccccc} \lambda & \preccurlyeq & \lambda' & \preccurlyeq & \lambda'' \\ \text{⫪} & & \text{⫪} & & \text{⫪} \\ \boldsymbol{d} & \preceq & \boldsymbol{d'} & \preceq & \boldsymbol{d''} \end{array}$$

Proof. From the assumption $\lambda \preccurlyeq \lambda''$ and $\lambda' \preccurlyeq \lambda''$, $\lambda \doteqdot \lambda' \doteqdot \lambda''$ holds.

First, we consider $\boldsymbol{d} = \boldsymbol{d'}$. Then, this case is immediate from Proposition D2.

Next, we consider $\boldsymbol{d} \neq \boldsymbol{d'}$ (i.e., $\boldsymbol{d} \vdash^{+} \boldsymbol{d'}$). Then, this case is immediate from Proposition D4. □