Configuration Reachability Analysis of Synchronized Recursive Timed Automata

Yuya Uezato Yasuhiko Minamide

We present synchronized recursive timed automata (SRTA) that extend timed automata with a stack in which each frame contains multiple real valued clocks. SRTA are an extension of dense-timed pushdown automata (TPDA) of Abdulla et al. As with TPDA, timed transitions of an SRTA synchronously increase the values of all the clocks within its stack at the same rate. Our contribution is to show the decidability of the configuration reachability problem of SRTA. On the basis of a thorough study of the proof structure of Abdulla et al., we present a simpler and more modular proof that applies several abstractions to the concrete semantics and consists of their forward and backward simulations. Our proof enables to extend their decidability result of the location reachability problem to the configuration reachability problem of SRTA. We use the region designed by Abdulla et al. for TPDA instead of the conventional region in the theory of timed automata to establish a key technical lemma.

1 Introduction

Timed automata and pushdown automata are models of real-time systems and recursive programs, respectively. Both the classes are a fundamental model for software model checking due to the decidability of their reachability problem. Several pushdown-extensions of timed automata have been introduced [6] [19] [4] [1] [9] from a theoretical point of view and the decidability of their reachability problem has been studied.

This paper presents a new pushdown-extension of timed automata, *synchronized recursive timed automata (SRTA)*, and we study the decidability of the configuration reachability problem. In comparison with previous pushdown-extensions of timed automata, SRTA have novel constraints, *fractional constraints*—formulae of the form $\{x\} = 0$ and $\{x\} < \{y\}$ ($\{x\}$ is the fractional part of a clock x)—for checking the fractional parts of clocks. These fractional constraints play the important role: fractional constraints enlarge the language class of the pushdown-extension of timed automata called *dense-timed pushdown automata (TPDA)* of Abdulla et al. [1] (or equivalently, *TPDA with diagonal constraints* of Clemente and Lasota [9]). Indeed, the following SRTA language L_{ex} cannot be recognized by any TPDA or TPDA with diagonal constraints due to the lack of fractional constraints:

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

Timed automata are a model of real-time systems that includes real-valued clocks where a configuration $\langle q, \nu \rangle$ is a pair of a control location q and a clock valuation $\nu : \mathcal{X} \to \mathbb{R}_+$ from a finite set of clocks $\mathcal{X}$ to the set of non-negative real numbers $\mathbb{R}_+$. On timed automata, timed transitions evolve the values of all the clocks at the same rate: $\langle q, \nu \rangle \overset{\delta}{\rightsquigarrow} \langle q, \nu + \delta \rangle$ where $\delta \in \mathbb{R}_+$ and $(\nu + \delta)(x) = \nu(x) + \delta$ holds for all clock $x \in \mathcal{X}$. Despite the unboundedness and denseness of real numbers, the location reachability problem of timed automata was shown decidable by the region abstrac-

同期型再帰的時間オートマトンの到達可能性解析

上里友弥, 筑波大学システム情報工学研究科, Department of Computer Science, University of Tsukuba.

南出靖彦, 東京工業大学情報理工学院数理・計算科学系, Department of Mathematical and Computing Science, Tokyo Institute of Technology.

コンピュータソフトウェア, Vol.35, No.1 (2018), pp.140–168.
[研究論文] 2017 年 1 月 3 日受付.
本論文は LPAR-20 の発表論文 “Synchronized Recursive Timed Automata” をもとに発展させたものである.

tion technique in [2] [3].

The two equivalent models, *recursive timed automata (RTA)* and *timed recursive state machines*, were independently introduced by Trivedi and Wojtczak [19] and Benerecetti, Minopoli, and Peron [4] [5]. A configuration of RTA $\langle q, \langle \gamma_1, \nu_1 \rangle \dots \langle \gamma_n, \nu_n \rangle \rangle$ is a pair of a location q and a stack where each frame is a pair $\langle \gamma_i, \nu_i \rangle$ of a symbol γ_i and a valuation $\nu_i : \mathcal{X} \to \mathbb{R}_+$. Timed transitions of RTA evolve the values of the clocks only at the top frame: $\langle q, \langle \gamma_1, \nu_1 \rangle \dots \langle \gamma_n, \nu_n \rangle \rangle \overset{\delta}{\rightsquigarrow} \langle q, \langle \gamma_1, \nu_1 \rangle \dots \langle \gamma_n, \nu_n + \delta \rangle \rangle$. Unfortunately, the reachability problem of RTA is undecidable because RTA can simulate two-counter machines [19] [4] [5].

Abdulla, Atig, and Stenman introduced *dense-timed pushdown automata (TPDA)* [1], and recently Clemente and Lasota extended TPDA to allow diagonal constraints [9]. A configuration of TPDA $\langle q, \nu, \langle \gamma_1, k_1 \rangle \dots \langle \gamma_n, k_n \rangle \rangle$ is a triple of a location q, a valuation $\nu : \mathcal{X} \to \mathbb{R}_+$, and a timed stack where each element $\langle \gamma_i, k_i \rangle$ is a pair of a symbol γ_i and its age $k_i \in \mathbb{R}_+$. TPDA differ from RTA in the point that timed transitions of TPDA synchronously evolve the values of all the clocks within a configuration at the same rate:

$$\begin{aligned} &\langle q, \nu, \langle \gamma_1, k_1 \rangle \dots \langle \gamma_n, k_n \rangle \rangle \\ \overset{\delta}{\rightsquigarrow}\ &\langle q, \nu + \delta, \langle \gamma_1, k_1 + \delta \rangle \dots \langle \gamma_n, k_n + \delta \rangle \rangle. \end{aligned}$$

Abdulla et al. showed the location reachability problem of TPDA is decidable and ExpTime-complete. For a given TPDA and control location q, the location reachability problem $\mathbf{c}_{\text{init}} \to_?^* q$ decides whether we can reach a configuration $\langle q, \nu, w \rangle$ for some valuation ν and stack w from the initial configuration $\mathbf{c}_{\text{init}}$. In order to show the decidability of the location reachability problem, they designed a region abstraction for pushdown-extensions of timed automata.

Our SRTA are defined as synchronized RTA with fractional constraints. Configurations of SRTA are the same as RTA $\langle q, \langle \gamma_1, \nu_1 \rangle \dots \langle \gamma_n, \nu_n \rangle \rangle$; however, timed transitions of SRTA synchronously evolve the values of all the clocks within a stack:

$$\begin{aligned} &\langle q, \langle \gamma_1, \nu_1 \rangle \dots \langle \gamma_n, \nu_n \rangle \rangle \\ \overset{\delta}{\rightsquigarrow}\ &\langle q, \langle \gamma_1, \nu_1 + \delta \rangle \dots \langle \gamma_n, \nu_n + \delta \rangle \rangle. \end{aligned}$$

The fractional constraints of SRTA make SRTA more expressive than TPDA (with diagonal constraints). Even though SRTA extend TPDA, the location reachability problem of SRTA is decidable. Furthermore, we show the decidability of the configuration reachability problem of SRTA. For a given SRTA and configuration $\langle q, w \rangle$, the configuration reachability problem $\mathbf{c}_{\text{init}} \to_?^* \langle q, w \rangle$ decides whether we can reach the configuration $\langle q, w \rangle$ from the initial configuration $\mathbf{c}_{\text{init}}$. Although the configuration reachability problem of pushdown automata is straightforwardly reduced to the location reachability problem of them, such a reduction does not hold on SRTA due to the unboundedness and denseness of real numbers. Indeed, the configuration reachability problem of TPDA was not considered by Abdulla et al. in [1].

Our decidability proof of the configuration reachability problem is organized as follows. The reader will find the detailed overview of our proof in Section 4. In Section 3, we introduce SRTA and give the standard semantics of SRTA. In order to reduce the configuration reachability problem of the standard semantics to that of a semantics defined as a pushdown system, we need to remove the entire stack modification of timed transitions $\langle q, \langle \gamma_1, \nu_1 \rangle \dots \langle \gamma_n, \nu_n \rangle \rangle \overset{\delta}{\rightsquigarrow} \langle q, \langle \gamma_1, \nu_1 + \delta \rangle \dots \langle \gamma_n, \nu_n + \delta \rangle \rangle$ and the unboundedness and denseness of real numbers. In Section 5, to remove the entire stack modification of timed transitions, we use the technique called lazy time elapsing that was introduced by Abdulla et al. to show the decidability of the location reachability problem of TPDA in [1]. In Section 6, we remove the unboundedness of real numbers by introducing collapsed real numbers. In Section 7, we remove the denseness of real numbers with the formalization of the region of Abdulla et al. Through Section 5 to 7, we can give a semantics defined as a pushdown system that corresponds to the standard semantics and it leads to the decidability of the configuration reachability problem of SRTA.

This article extends our LPAR paper [20] with the following new contributions:

- We consider the configuration reachability problem rather than the location reachability problem. We find out that a backward simulation is key to the configuration reachability problem and will explain this precisely in Section 4. 2.
- We give proofs to important technical propositions and lemmas, which are marked by the symbol ♣. (In the previous version, we omitted

all the proofs). Furthermore, we formalize all the important technical propositions and lemmas with the proof assistant Coq. The reader can find the HTML document generated by the Coq system in [22].

- We compare the conventional region of timed automata given by Alur and Dill in [3] and the region designed by Abdulla et al. for TPDA in Section 8. Through this comparison, we find out that a key technical lemma fails on the region of Alur and Dill.

To simplify the formalization of SRTA and focus on the configuration reachability analysis, we do not consider the following materials that are investigated in the previous version:

- SRTA with diagonal constraints: this extension corresponds to TPDA with diagonal constraints.
- The proof of the fact that the above SRTA language L_{ex} cannot be recognized by any TPDA.

The reader is referred to the long version of the previous work for these two topics [21].

2 Preliminaries

The set of non-negative real numbers $\mathbb{R}_+$ is defined by: $\mathbb{R}_+ \triangleq \{r \in \mathbb{R} : r \geq 0\}$. For a real number $r \in \mathbb{R}_+$, 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 $\mathcal{X}$ be a finite set of clocks. A function $\nu : \mathcal{X} \to \mathbb{R}_+$ is called a *concrete valuation* (or simply *valuation*) on $\mathcal{X}$. We define basic operations:

$$\begin{aligned} (\nu[x := r])(y) &\triangleq \begin{cases} r & \text{if } y = x \\ \nu(y) & \text{otherwise,} \end{cases} \\ \nu[x := y] &\triangleq \nu[x := \nu(y)], \\ (\nu + r)(y) &\triangleq \nu(y) + r, \end{aligned}$$

where $x, y \in \mathcal{X}$ and $r \in \mathbb{R}_+$. The zero valuation on $\mathcal{X}$ is defined by: $\mathbf{0}_{\mathcal{X}}(x) \triangleq 0$ for $x \in \mathcal{X}$. We omit the subscript $\mathcal{X}$ and simply write $\mathbf{0}$ if the domain $\mathcal{X}$ is clear from the context. For a valuation $\nu : \mathcal{X} \to \mathbb{R}_+$ and subset $Y \subseteq \mathcal{X}$, we write $\nu \restriction Y$ to denote the restriction of ν to Y. We define the ordering $\nu \leq \nu'$ on valuations by: $\nu \leq \nu'$ if $\exists r \in \mathbb{R}_+ . \nu' = \nu + r$.

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 $w \in \Gamma^*$. A 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 a *finite* PDS if its set of transition rules is finite. Otherwise, it is called an *infinite* PDS. Note that our formalization allows multiple popping rather than single element popping at each single move. A PDS is called a *normalized* PDS if the set of transition rules $\hookrightarrow$ is a subset of $(Q \times \Gamma) \times (Q \times \Gamma^*)$.

For given configurations $\mathbf{c}_1$ and $\mathbf{c}_2$, the configuration reachability problem asks if $\mathbf{c}_1 \to^* \mathbf{c}_2$ holds. The configuration reachability problem of finite normalized PDS is in PTIME [7] [8] [13]. Since we can translate a finite PDS to the corresponding finite normalized PDS while preserving the configuration reachability, the configuration reachability problem of finite PDS is also decidable.

3 Synchronized Recursive Timed Automata

We introduce synchronized recursive timed automata (SRTA) and define the standard semantics of SRTA called STND.

Clock Constraints.

Let $\mathbb{I} \triangleq \{(i, j), [i, j] : i, j \in \mathbb{N}\}$ be the set of intervals and $\mathcal{X}$ be a finite set of clocks. The set $\Phi_{\mathcal{X}}$ of clock constraints is given by:

$$\varphi ::= x \in I \mid \{x\} = 0 \mid \{x\} \bowtie \{y\} \mid \varphi \wedge \varphi \mid \neg\varphi$$

where $x, y \in \mathcal{X}$, $I \in \mathbb{I}$, and $\bowtie \in \{<, =, >\}$.

For a constraint $\varphi \in \Phi_{\mathcal{X}}$ and valuation $\nu : \mathcal{X} \to \mathbb{R}_+$, we write $\nu \models \varphi$ if φ holds when clocks are replaced by values of ν: e.g., $\nu \models x \in I$ if $\nu(x) \in I$, $\nu \models \{x\} = 0$ if $\{\nu(x)\} = 0$. The fractional constraints $\{x\} = 0$ and $\{x\} \bowtie \{y\}$ are a novel feature against previous pushdown-extensions of timed automata.

Definition 1 (Synchronized Recursive Timed Automata)**.** A synchronized recursive timed automaton (SRTA) is a tuple $\mathcal{A} = (Q, q_{\mathrm{init}}, F, \Sigma, \Gamma, \mathcal{X}, \Delta)$ where Q is a finite set of control locations, q_{init} is the initial location, $F \subseteq Q$ is a set of final locations, Σ is a finite input alphabet, Γ is a finite set of stack symbols, $\mathcal{X}$ is a finite set of clocks, and $\Delta \subseteq Q \times (\Sigma \cup \{\epsilon\}) \times \mathit{Op} \times Q$ is a finite set of *discrete transition rules*. The operations $\tau \in \mathit{Op}$ are given by:

$$\tau ::= \mathsf{push}(\gamma) \mid \mathsf{pop}(\gamma) \mid \mathsf{dig}(x, y) \mid x \leftarrow I \mid \mathsf{check}(\varphi)$$

where $\gamma \in \Gamma, x, y \in \mathcal{X}, I \in \mathbb{I}$, and $\varphi \in \Phi_{\mathcal{X}}$. ◇

We define the standard semantics Stnd of SRTA as an infinite transition system.

Definition 2 (Semantics Stnd)**.** A configuration is a pair $\langle q, w\rangle$ of a location q and a stack w in which each frame $\langle \gamma, \nu\rangle$ consists of a stack symbol γ and a concrete valuation $\nu : \mathcal{X} \to \mathbb{R}_+$. The set of configurations of Stnd is $Q \times (\Gamma \times (\mathcal{X} \to \mathbb{R}_+))^*$.

For an operation $\tau \in \mathit{Op}$, we define a *discrete transition* $w \xrightarrow{\tau} w'$ for $w, w' \in (\Gamma \times (\mathcal{X} \to \mathbb{R}_+))^*$ by case analysis on τ as follows:

$$w \langle \gamma_1, \nu_1\rangle \xrightarrow{\mathsf{push}(\gamma_2)} w \langle \gamma_1, \nu_1\rangle \langle \gamma_2, \mathbf{0}_{\mathcal{X}}\rangle,$$

$$w \langle \gamma_1, \nu_1\rangle \langle \gamma_2, \nu_2\rangle \xrightarrow{\mathsf{pop}(\gamma_2)} w \langle \gamma_1, \nu_2\rangle,$$

$$\frac{\nu_2' = \nu_2[x := \nu_1(y)]}{w \langle \gamma_1, \nu_1\rangle \langle \gamma_2, \nu_2\rangle \xrightarrow{\mathsf{dig}(x,y)} w \langle \gamma_1, \nu_1\rangle \langle \gamma_2, \nu_2'\rangle} ,$$

$$\frac{r \in I \quad \nu' = \nu[x := r]}{w \langle \gamma, \nu\rangle \xrightarrow{x \leftarrow I} w \langle \gamma, \nu'\rangle} ,$$

$$\frac{\nu \models \varphi}{w \langle \gamma, \nu\rangle \xrightarrow{\mathsf{check}(\varphi)} w \langle \gamma, \nu\rangle} .$$

We note that the **pop**-rule removes the top frame and puts ν_2 to the next frame as follows:

$$\begin{tikzcd}[column sep=tiny] w\langle\gamma_1, \arrow[rrrr, bend right] & \nu_1\rangle\langle\gamma_2, & \nu_2\rangle \arrow[rrr, bend left] & \to w\langle & \gamma_1, & \nu_2\rangle. \end{tikzcd}$$

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

$$\frac{|w| \geq 1 \quad \delta \in \mathbb{R}_+}{w \overset{\delta}{\rightsquigarrow} w + \delta}\ \mathsf{time}$$

where $w + \delta$ is defined as follows:

$$\begin{aligned} &(\langle \gamma_1, \nu_1\rangle\langle \gamma_2, \nu_2\rangle \ldots \langle \gamma_n, \nu_n\rangle) + \delta \\ \triangleq\ &\langle \gamma_1, \nu_1 + \delta\rangle\langle \gamma_2, \nu_2 + \delta\rangle \ldots \langle \gamma_n, \nu_n + \delta\rangle. \end{aligned}$$

These transitions for stacks are extended to configurations as follows:

- $\langle q, w\rangle \xrightarrow{\alpha} \langle q', w'\rangle$ if $w \xrightarrow{\tau} w'$ for some rule $q \xrightarrow{\alpha}_\tau q' \in \Delta$. We write $q \xrightarrow{\alpha}_\tau q' \in \Delta$ instead of $\langle q, \alpha, \tau, q'\rangle \in \Delta$.
- $\langle q, w\rangle \overset{\delta}{\rightsquigarrow} \langle q, w'\rangle$ if $w \overset{\delta}{\rightsquigarrow} w'$ for $\delta \in \mathbb{R}_+$.

◇

By exploiting the **dig** rule, we can implement useful transition rules, which naturally appeared in the existing models: timed recursive state machines and recursive timed automata [4] [19]. We call an SRTA equipped with the following extended push and pop rules an *extended* SRTA:

$$\frac{\nu_2 = \mathbf{0}_{\mathcal{X}}[X := \nu_1]}{w \langle \gamma_1, \nu_1\rangle \to w \langle \gamma_1, \nu_1\rangle \langle \gamma_2, \nu_2\rangle}\ \mathsf{push}(\gamma_2, X)$$

$$\frac{\nu_2' = \nu_2[X := \nu_1]}{w \langle \gamma_1, \nu_1\rangle \langle \gamma_2, \nu_2\rangle \to w \langle \gamma_1, \nu_2'\rangle}\ \mathsf{pop}(\gamma_2, X)$$

where X is a subset of $\mathcal{X}$ and $\nu[\{x_1, \ldots, x_n\} := \nu']$ is defined as follows:

$$\begin{aligned} &\nu[\{x_1, x_2, \ldots, x_n\} := \nu'] \\ &\triangleq \nu[x_1 := \nu'(x_1)][x_2 := \nu'(x_2)] \cdots [x_n := \nu'(x_n)]. \end{aligned}$$

For any extended SRTA, we can construct the corresponding normal SRTA by replacing each extended transition rule with appropriate **dig** rules.

We define the timed language of an SRTA and see an SRTA language example.

Definition 3 (Timed Languages)**.** A *run* π is a finite alternating sequence of timed and discrete transitions. For a run $\pi = c_0 \overset{\delta_0}{\rightsquigarrow} c_0' \xrightarrow{\alpha_0} c_1 \overset{\delta_1}{\rightsquigarrow} c_1' \xrightarrow{\alpha_1} \cdots \overset{\delta_n}{\rightsquigarrow} c_n' \xrightarrow{\alpha_n} c_{n+1}$, we define the *timed trace* $\mathsf{tt}(\pi) \triangleq (\alpha_0, \delta_0)(\alpha_1, \delta_0 + \delta_1) \ldots (\alpha_n, \Sigma_{i=0}^n \delta_i) \in ((\Sigma \cup \{\epsilon\}) \times \mathbb{R}_+)^*$ and the *timed word* $\mathsf{tw}(\pi) \in (\Sigma \times \mathbb{R}_+)^*$ by removing all the (ϵ, t) pairs from $\mathsf{tt}(\pi)$. The timed language of an SRTA $\mathcal{A}$, $L(\mathcal{A})$, is defined as follows:

$$\begin{aligned} &L(\mathcal{A}) \triangleq \{\mathsf{tw}(\pi) : \\ &\quad \pi = \langle q_{\text{init}}, \langle \bot, \mathbf{0}\rangle\rangle \rightsquigarrow \cdots \to \langle q, w\rangle, q \in F\}. \end{aligned}$$

Note: For the initial configuration $\langle q_{\text{init}}, \langle \bot, \mathbf{0}\rangle\rangle$, we require the special stack symbol $\bot$ in Γ. ◇

Timed Language Example.

We consider the following timed language:

$$\begin{aligned} L_{\text{ex}} \triangleq &\{(a, t_1)(a, t_2) \ldots (a, t_n)(b, t_n') \ldots (b, t_2')(b, t_1') \\ &: t_i' - t_i \in \mathbb{N},\ n \geq 1\}. \end{aligned}$$

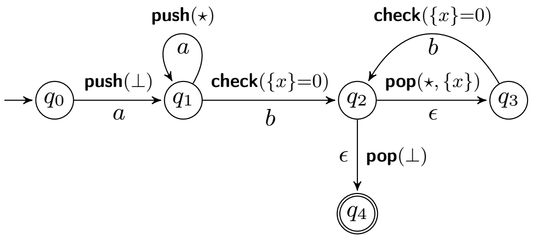
Note that if we forget the time stamps from L_{ex} then the language $\{a^n b^n : n \geq 1\}$ is a typical context-free language.

To accept the above timed language, let us consider an extended SRTA $\mathcal{A}_{\text{ex}} = (\{q_0, \ldots, q_4\}, q_0, \{q_4\}, \{a, b\}, \{\bot, \star\}, \{x\}, \Delta)$ where Δ is defined

$$\langle q_0, \langle \bot, 0\rangle\rangle \overset{0.1}{\rightsquigarrow} \langle q_0, \langle \bot, 0.1\rangle\rangle \xrightarrow{a} \langle q_1, \langle \bot, 0.1\rangle\langle \bot, 0\rangle\rangle \overset{1.1}{\rightsquigarrow} \langle q_1, \langle \bot, 1.2\rangle\langle \bot, 1.1\rangle\rangle \xrightarrow{a} \langle q_1, \langle \bot, 1.2\rangle\langle \bot, 1.1\rangle\langle \star, 0\rangle\rangle \overset{1.0}{\rightsquigarrow}$$
$$\langle q_1, \langle \bot, 2.2\rangle\langle \bot, 2.1\rangle\langle \star, 1\rangle\rangle \xrightarrow{b} \langle q_2, \langle \bot, 2.2\rangle\langle \bot, 2.1\rangle\langle \star, 1\rangle\rangle \overset{0.5}{\rightsquigarrow} \langle q_2, \langle \bot, 2.7\rangle\langle \bot, 2.6\rangle\langle \star, 1.5\rangle\rangle \xrightarrow{\epsilon}$$
$$\langle q_3, \langle \bot, 2.7\rangle\langle \bot, 2.6\rangle\rangle \overset{0.4}{\rightsquigarrow} \langle q_3, \langle \bot, 3.1\rangle\langle \bot, 3\rangle\rangle \xrightarrow{b} \langle q_2, \langle \bot, 3.1\rangle\langle \bot, 3\rangle\rangle \overset{0}{\rightsquigarrow} \langle q_2, \langle \bot, 3.1\rangle\langle \bot, 3\rangle\rangle \xrightarrow{\epsilon} \langle q_4, \langle \bot, 3.1\rangle\rangle.$$

Fig. 1　An acceptable run for $(a, 0.1)(a, 1.2)(b, 2.2)(b, 3.1)$**.**

as follows:



SRTA $\mathcal{A}_{\text{ex}}$ accepts L_{ex} and Fig. 1 shows how SRTA $\mathcal{A}_{\text{ex}}$ accepts the following timed word:

$$(a, 0.1)(a, 1.2)(b, 2.2)(b, 3.1) \in L_{\text{ex}}.$$

The fractional constraint **check**$(\{x\} = 0)$ checks if the fractional part of $t'_i - t_i$ is zero (i.e., $t'_i - t_i \in_? \mathbb{N}$) and is key to excluding runs τ such that $\mathsf{tw}(\tau) \notin L_{\text{ex}}$. For example,

$$\tau = \langle q_0, \langle \bot, 0\rangle\rangle \overset{0.1}{\rightsquigarrow} \langle q_0, \langle \bot, 0.1\rangle\rangle \xrightarrow{a}$$
$$\langle q_1, \langle \bot, 0.1\rangle\langle \bot, 0\rangle\rangle \overset{0.2}{\rightsquigarrow} \langle q_1, \langle \bot, 0.3\rangle\langle \bot, 0.2\rangle\rangle \not\xrightarrow{b}$$
$$\cancel{\langle q_2, \langle \bot, 0.3\rangle\langle \bot, 0.2\rangle\rangle}$$

and $\mathsf{tw}(\tau) = (a, 0.1)(b, 0.3) \notin L_{\text{ex}}$.

4　Overview of Decidability Proof of Configuration Reachability Problem

As an overview of the rest of the present paper, we outline our proof of the decidability of the configuration reachability problem of SRTA.

Configuration Reachability Problem.

For a configuration $\langle q, w\rangle$, the configuration reachability problem $\langle q_{\text{init}}, \langle \bot, \mathbf{0}\rangle\rangle \to^*_? \langle q, w\rangle$ decides if there is a run from the initial configuration $\langle q_{\text{init}}, \langle \bot, \mathbf{0}\rangle\rangle$ to $\langle q, w\rangle$.

The following is our main result:

Main Result (Corollary 1)

The configuration reachability problem of SRTA is decidable.

To show this, we build a semantics called the digitized semantics Digi that can be defined as a finite PDS through Section 5 and Section 7:

$$\textsc{Stnd} \xrightarrow{\text{Sec.5}} \text{Lazy Semantics } \textsc{Lazy}$$
$$\xrightarrow{\text{Sec.6}} \text{Collapsed Semantics } \textsc{Coll}$$
$$\xrightarrow{\text{Sec.7}} \text{Digitized Semantics } \textsc{Digi}.$$

These translations allow reducing the configuration reachability problem of the standard semantics to the configuration reachability problem of the digitized semantics. Since the configuration reachability problem of finite PDS is decidable, we obtain the decidability of the configuration reachability problem of SRTA. We note that our construction is based on the construction of Abdulla et al. in [1] and the finally obtained finite PDS is basically equivalent to their *symbolic pushdown automaton.*

Let us see the idea of each translation and explain the reason why our approach works well for the configuration reachability problem.

4.1　Idea of Each Semantics

Our aim is to reduce the reachability problem of the standard semantics Stnd to the corresponding reachability problem of the digitized semantics Digi. To this end, we remove the following three problems at each step:

1. The entire stack modification of timed transitions:
$$\langle \gamma_1, \nu_1\rangle\langle \gamma_2, \nu_2\rangle \dots \langle \gamma_n, \nu_n\rangle$$
$$\overset{\delta}{\rightsquigarrow} \langle \gamma_1, \nu_1 + \delta\rangle\langle \gamma_2, \nu_2 + \delta\rangle \dots \langle \gamma_n, \nu_n + \delta\rangle.$$
2. The unboundedness of real numbers.
3. The denseness of real numbers.

Removing Entire Stack Modification: Lazy Semantics

To simulate the entire stack modification of timed transitions by pushdown systems that only allow to modify finitely (boundedly) many elements of a stack, we use the technique called *lazy time elapsing* that was developed by Abdulla et al. in [1].

To show the idea of the technique, let us consider the following transitions:

$$\nu_1\,\nu_2\,\nu_3 \overset{2.0}{\rightsquigarrow} \nu_1'\,\nu_2'\,\nu_3' \xrightarrow{\textsf{dig}(x,x)} \nu_1'\,\nu_2'\,\nu_3'' \xrightarrow{\textsf{pop}} \nu_1'\,\nu_3''$$

where

$$\nu_1=\{x\mapsto 0.5\}, \nu_2=\{x\mapsto 2.0\}, \nu_3=\{x\mapsto 1.5\},$$
$$\nu_i'=\nu_i+2.0,\ \nu_3''=\{x\mapsto 4.0\}.$$

Our simulation idea is to

- keep the correct top and next to the top frames rather than the entire stack for the **check** and **dig** rules; and
- reconstruct a new frame that reflects the correct information when performing **pop** transitions.

We introduce the notion of clock marking and valuation pairing to describe this idea. We pair two valuations ν_i and ν_{i+1} and make the paired valuation μ_{i+1} as follows:

1. Mark ν_{i+1} and obtain the marked valuation $\dot{\nu}_{i+1}:\dot{\mathcal{X}}\to\mathbb{R}_+$ as $\dot{\nu}_{i+1}(\dot{x})\triangleq\nu_{i+1}(x)$.
2. In the same way, we mark and obtain the marked valuation $\underset{\bullet}{\nu}_i:\underset{\bullet}{\mathcal{X}}\to\mathbb{R}_+$.
3. Finally, we obtain $\mu:\underset{\bullet}{\mathcal{X}}\cup\dot{\mathcal{X}}\to\mathbb{R}_+$ by gluing them $\mu_{i+1}\triangleq\underset{\bullet}{\nu}_i\cup\dot{\nu}_{i+1}$.

We relate the 1-height stack ν_1 to a 1-height stack μ_1 of a paired valuation such that $\nu_1(x)=\mu_1(\dot{x})$ as follows:

$$\nu_1=\{x\mapsto 0.5\}\models\left\{\underset{\bullet}{x}\mapsto r;\ \dot{x}\mapsto 0.5\right\},$$

where the value of $\underset{\bullet}{x}$ is irrelevant. We also relate the 3-height stack $\nu_1\nu_2\nu_3$ to the following 3-height stack $\mu_1\mu_2\mu_3$:

$$\begin{array}{c}\nu_3=\{x\mapsto 1.5\}\\\hline \nu_2=\{x\mapsto 2.0\}\\\hline \nu_1=\{x\mapsto 0.5\}\end{array}\models\begin{array}{c}\mu_3=\left\{\underset{\bullet}{x}\mapsto 2.0;\ \dot{x}\mapsto 1.5\right\}\\\hline \mu_2=\left\{\underset{\bullet}{x}\mapsto 0.5;\ \dot{x}\mapsto 2.0\right\}\\\hline \mu_1=\left\{\underset{\bullet}{x}\mapsto\ \ r;\ \dot{x}\mapsto 0.5\right\},\end{array}$$

where the top frame μ_3 has the correct information of the top frame ν_3 and next to the top frame ν_2, the frame μ_2 corresponds to the frames ν_2 and ν_1, and the frame μ_1 corresponds to the frame ν_1.

On the basis of valuation pairing, we simulate the entire stack modification $\nu_1\nu_2\nu_3\overset{2.0}{\rightsquigarrow}\nu_1'\nu_2'\nu_3'$ by only evolving the top frame as follows ($\mu_1\mu_2\mu_3\overset{2.0}{\rightsquigarrow}\mu_1\mu_2\mu_3'$):

$$\begin{array}{c}\left\{\underset{\bullet}{x}\mapsto 2.0;\ \dot{x}\mapsto 1.5\right\}\\\hline \left\{\underset{\bullet}{x}\mapsto 0.5;\ \dot{x}\mapsto 2.0\right\}\\\hline \left\{\underset{\bullet}{x}\mapsto\ \ r;\ \dot{x}\mapsto 0.5\right\}\end{array}\overset{2.0}{\rightsquigarrow}\begin{array}{c}\left\{\underset{\bullet}{x}\mapsto 4.0;\ \dot{x}\mapsto 3.5\right\}\\\hline \left\{\underset{\bullet}{x}\mapsto 0.5;\ \dot{x}\mapsto 2.0\right\}\\\hline \left\{\underset{\bullet}{x}\mapsto\ \ r;\ \dot{x}\mapsto 0.5\right\}.\end{array}$$

Although the obtained stack does not match the stack $\nu_1'\nu_2'\nu_3'$, the top frame μ_3' matches the top frame ν_3' and the next to the top frame ν_2'. Therefore, we can safely simulate the **dig** transition $\nu_1'\nu_2'\nu_3'\xrightarrow{\textsf{dig}(x,x)}\nu_1'\nu_2'\nu_3''$ as follows ($\mu_1\mu_2\mu_3'\xrightarrow{\dot{x}:=\underset{\bullet}{x}}\mu_1\mu_2\mu_3''$):

$$\begin{array}{c}\left\{\underset{\bullet}{x}\mapsto 4.0;\ \dot{x}\mapsto 3.5\right\}\\\hline \left\{\underset{\bullet}{x}\mapsto 0.5;\ \dot{x}\mapsto 2.0\right\}\\\hline \left\{\underset{\bullet}{x}\mapsto\ \ r;\ \dot{x}\mapsto 0.5\right\}\end{array}\xrightarrow{\dot{x}:=\underset{\bullet}{x}}\begin{array}{c}\left\{\underset{\bullet}{x}\mapsto 4.0;\ \dot{x}\mapsto 4.0\right\}\\\hline \left\{\underset{\bullet}{x}\mapsto 0.5;\ \dot{x}\mapsto 2.0\right\}\\\hline \left\{\underset{\bullet}{x}\mapsto\ \ r;\ \dot{x}\mapsto 0.5\right\}.\end{array}$$

In order to simulate $\nu_1'\nu_2'\nu_3''\xrightarrow{\textsf{pop}}\nu_1'\nu_3''$, we evolve the next to the top frame μ_2 until $\mu_2+\delta$ matches μ_3 or formally $(\mu_2+\delta)(\dot{x})=\mu_3(\underset{\bullet}{x})$ holds:

$$\begin{array}{c}\left\{\underline{\underset{\bullet}{x}\mapsto 4.0};\ \dot{x}\mapsto 4.0\right\}\\\hline \left\{\underset{\bullet}{x}\mapsto 0.5;\ \overline{\dot{x}\mapsto 2.0}\right\}\\\hline \left\{\underset{\bullet}{x}\mapsto\ \ r;\ \dot{x}\mapsto 0.5\right\}\end{array}\to\begin{array}{c}\left\{\underline{\underset{\bullet}{x}\mapsto 4.0};\ \dot{x}\mapsto 4.0\right\}\\\hline \left\{\underset{\bullet}{x}\mapsto 2.5;\ \overline{\dot{x}\mapsto 4.0}\right\}\\\hline \left\{\underset{\bullet}{x}\mapsto\ \ r;\ \dot{x}\mapsto 0.5\right\}.\end{array}$$

This operation reconstructs the frame $\{\underset{\bullet}{x}\mapsto 2.5;\ \dot{x}\mapsto 4.0\}$ that has the correct information of ν_1' and ν_2'. We compose the frames μ_3'' and $\mu_2+2.0$ as follows:

$$\begin{array}{c}\left\{\underline{\underset{\bullet}{x}\mapsto 4.0};\ \dot{x}\mapsto 4.0\right\}\\\hline \left\{\underset{\bullet}{x}\mapsto 2.5;\ \overline{\dot{x}\mapsto 4.0}\right\}\\\hline \left\{\underset{\bullet}{x}\mapsto\ \ r;\ \dot{x}\mapsto 0.5\right\}.\end{array}\to\begin{array}{c}\left\{\underset{\bullet}{x}\mapsto 2.5;\ \dot{x}\mapsto 4.0\right\}\\\hline \left\{\underset{\bullet}{x}\mapsto\ \ r;\ \dot{x}\mapsto 0.5\right\}.\end{array}$$

After simulating the **pop** transition, the top frame $\left\{\underset{\bullet}{x}\mapsto 2.5;\ \dot{x}\mapsto 4.0\right\}$ matches $\nu_1'\nu_3''$.

It is worth noting that, to simulate **pop** transitions, we need to operate the top and the next to the top frames at once. For this purpose, multiple pop transition rules $\langle q,\alpha\rangle\hookrightarrow\langle q,\beta\rangle$ where $\alpha\in\Gamma^+$ are allowed in our formalization of pushdown systems.

Considering stacks of paired valuations is enough to simulate the standard semantics by an infinite pushdown system. However, for technical reasons, we introduce *reference clocks* along with the lazy elapsing technique. We use a fresh clock called a *reference clock* $\mathtt{C}$. Such a clock is accessed and reset to 0.0 only when we push a new frame as follows:

$$\left\{\underset{\bullet}{\mathtt{C}}\mapsto r_1;\ \underset{\bullet}{x}\mapsto 0.5;\ \dot{\mathtt{C}}\mapsto r_2;\ \dot{x}\mapsto 2.0\right\}\xrightarrow{\textsf{push}}$$
$$\begin{array}{c}\left\{\underset{\bullet}{\mathtt{C}}\mapsto 0.0;\ \underset{\bullet}{x}\mapsto 2.0;\ \dot{\mathtt{C}}\mapsto 0.0;\ \dot{x}\mapsto 0.0\right\}\\\hline \left\{\underset{\bullet}{\mathtt{C}}\mapsto r_1;\ \underset{\bullet}{x}\mapsto 0.5;\ \dot{\mathtt{C}}\mapsto 0.0;\ \dot{x}\mapsto 2.0\right\}.\end{array}$$

The introduction of reference clocks does not interfere with the above idea of the lazy elapsing technique. The reader can find the reason why we need reference clocks in Section 6. 2, the remark after Lemma 5, and Section 7. 5.

In Section 5, we show the simulation between the standard semantics and the lazy semantics in the both directions:

Lemma 1 :
$$\begin{tikzcd} \boldsymbol{s} \arrow[r, "\textsc{Stnd}"'] & \boldsymbol{s'} & & \boldsymbol{s} \arrow[r, "\textsc{Stnd}"'] & \boldsymbol{s'} \\ \top & & \Longrightarrow & \top & \top \\ \boldsymbol{l} & & & \boldsymbol{l} \arrow[r, "\textsc{Lazy}"'] & \exists \boldsymbol{l'}, \end{tikzcd}$$

Lemma 2 :
$$\begin{tikzcd} \boldsymbol{s} & & & \boldsymbol{s} \arrow[r, "\textsc{Stnd}"'] & \exists\boldsymbol{s'} \\ \top & & \Longrightarrow & \top & \top \\ \boldsymbol{l} \arrow[r, "\textsc{Lazy}"'] & \boldsymbol{l'} & & \boldsymbol{l} \arrow[r, "\textsc{Lazy}"'] & \boldsymbol{l'}. \end{tikzcd}$$

Removing the Unboundedness of Real Numbers: Collapsed Semantics Coll

Since an SRTA $\mathcal{A}$ has only finitely many constraints (i,j) and $[i,j]$, if two real numbers $r_1, r_2 \in \mathbb{R}_+$ are sufficiently large, then we cannot distinguish them by any constraints of $\mathcal{A}$. To formalize this, we introduce an upper-bound constant M:

$$\mathsf{M} \triangleq \max\{\ j : (i,j) \text{ or } [i,j] \text{ appears in an interval constraint of } \mathcal{A}\ \} + 1.$$

We say a real number r is *sufficiently large* if $r \geq \mathsf{M}$. For two sufficiently large real numbers $r_1, r_2 \geq \mathsf{M}$, the following property holds:

$$r_1 \in I \iff r_2 \in I \text{ for any interval } I \text{ in } \mathcal{A}.$$

On the other hand, since we have fractional constraints $\{x\} = 0$ and $\{x\} \bowtie \{y\}$, we need to record the fractional parts of sufficiently large real numbers. For example, if $r_1, r_2 \geq \mathsf{M}$, $\{r_1\} = 0.0$, and $\{r_2\} = 0.5$, then $\{x \mapsto r_1\} \models (\{x\} = 0)$ but $\{x \mapsto r_2\} \not\models (\{x\} = 0)$.

In order to remove the unboundedness of real numbers, on the basis of the above argument, we define the collapsed real numbers $\mathbb{C}$ as follows:

$$\mathbb{C} \triangleq \big([0..(\mathsf{M}-1)] \cup \{\infty\}\big) \times [0,1)$$

and define the collapsing function $\mathcal{C} : \mathbb{R}_+ \to \mathbb{C}$ as follows:

$$\mathcal{C}(r) \quad \triangleq \quad \begin{cases} (\infty\ , \{r\}) & \text{if } r \geq \mathsf{M} \\ (\lfloor r \rfloor, \{r\}) & \text{if } r < \mathsf{M} \end{cases}$$

In Section 6, we will show the simulation between the lazy and collapsed semantics in the both directions:

Lemma 4 :
$$\begin{tikzcd} \boldsymbol{l} \arrow[r, "\textsc{Lazy}"'] & \boldsymbol{l'} & & \boldsymbol{l} \arrow[r, "\textsc{Lazy}"'] & \boldsymbol{l'} \\ \top & & \Longrightarrow & \top & \top \\ \boldsymbol{c} & & & \boldsymbol{c} \arrow[r, "\textsc{Coll}"'] & \exists \boldsymbol{c'}, \end{tikzcd}$$

Lemma 5 :
$$\begin{tikzcd} \boldsymbol{l} & & & \boldsymbol{l} \arrow[r, "\textsc{Lazy}"'] & \exists\boldsymbol{l'} \\ \top & & \Longrightarrow & \top & \top \\ \boldsymbol{c} \arrow[r, "\textsc{Coll}"'] & \boldsymbol{c'} & & \boldsymbol{c} \arrow[r, "\textsc{Coll}"'] & \boldsymbol{c'}. \end{tikzcd}$$

Removing the Denseness of Real Numbers: Digitized Semantics Digi.

Finally, we remove the denseness of real numbers from the collapsed domain $\mathbb{C}$ by using the region of Abdulla et al. that is designed for their TPDA. In the present paper, to distinguish their region and the conventional region of the theory of timed automata, we call their regions *digital valuations*. A digital valuation is obtained from a collapsed valuation by abstracting the fractional parts of the valuation into the corresponding ordering. For example, we abstract the following collapsed valuation

$$\mu = \{a \mapsto 1.0;\ b \mapsto 2.3;\ c \mapsto 3.7;\ d \mapsto \infty.3\}$$

into the digital valuation

$$\boldsymbol{d} = \{(a,1)\}_0\, \{(b,2),(d,\infty)\}\, \{(c,3)\} \qquad (\mu \models \boldsymbol{d})$$

and $\boldsymbol{d}$ means the following:

- The term $\{(a,1)\}_0$ means that the integral part of a is 1 and the fractional part of a is 0.0 (i.e., the value of a is 1.0).
- The term $\{(b,2),(d,\infty)\}$ means that the fractional parts of b and d are the same. Furthermore, the fractional parts of b and d are strictly larger than 0.0 because they do not belong to $\{\ldots\}_0$.
- The order $\{(b,2),(d,\infty)\}\,\{(c,3)\}$ means that the fractional part of b and d is strictly smaller than that of c.

In the theory of timed automata (without the stack), the region is an appropriate abstraction of the collapsed valuations. Indeed, when considering timed automata rather than SRTA, we have the forward simulation between the collapsed and digitized semantics in the both directions as follows:

For TA:
$$\begin{tikzcd} \boldsymbol{c} & & & \boldsymbol{c} \arrow[r, "\textsc{Coll}"'] & \exists\boldsymbol{c'} \\ \top & & \Longrightarrow & \top & \top \\ \boldsymbol{d} \arrow[r, "\textsc{Digi}"'] & \boldsymbol{d'} & & \boldsymbol{d} \arrow[r, "\textsc{Digi}"'] & \boldsymbol{d'}. \end{tikzcd}$$

Lemma 7:
$$\begin{tikzcd} \boldsymbol{c} \arrow[r, "\textsc{Coll}"'] & \boldsymbol{c'} & & \boldsymbol{c} \arrow[r, "\textsc{Coll}"'] & \boldsymbol{c'} \\ \top & & \Longrightarrow & \top & \top \\ \boldsymbol{d} & & & \boldsymbol{d} \arrow[r, "\textsc{Digi}"'] & \exists \boldsymbol{d'}. \end{tikzcd}$$

However, on SRTA, the former diagram does not hold for the unavoidable nondeterminacy of **pop** transitions. For example, let us consider the following **pop**-transition:

$$\begin{tikzcd}[column sep=small] \dfrac{\left\{\underset{\bullet}{x} \mapsto 1.0;\ \dot{x} \mapsto 2.2\right\}}{\left\{\dot{x} \mapsto 1.0;\ \underset{\bullet}{x} \mapsto 0.4\right\}} \arrow[r, "\textbf{pop}", "\textsc{Coll}"'] \arrow[d, phantom, "⫪" description] & \left\{\underset{\bullet}{x} \mapsto 0.4;\ \dot{x} \mapsto 2.2\right\} \\ \dfrac{\boldsymbol{d}_2 = \left\{(\underset{\bullet}{x},1)\right\}_0\{(\dot{x},2)\}}{\boldsymbol{d}_1 = \{(\dot{x},1)\}_0\left\{(\underset{\bullet}{x},0)\right\}} \arrow[r, "\textbf{pop}", "\textsc{Digi}"'] & \{\boldsymbol{d}, \boldsymbol{d}', \boldsymbol{d}''\}\,. \end{tikzcd}$$

We need to decide the order of $\underset{\bullet}{x} \in \boldsymbol{d}_1$ and $\dot{x} \in \boldsymbol{d}_2$ when composing $\boldsymbol{d}_1$ and $\boldsymbol{d}_2$. However, since there is no information about their fractional parts, we generate all the possibilities as follows:

$$\begin{aligned} \boldsymbol{d} &= \{\}_0 \left\{(\underset{\bullet}{x},0),(\dot{x},2)\right\}, \\ \boldsymbol{d}' &= \{\}_0 \left\{(\underset{\bullet}{x},0)\right\}\{(\dot{x},2)\}, \\ \boldsymbol{d}'' &= \{\}_0 \{(\dot{x},2)\}\left\{(\underset{\bullet}{x},0)\right\}. \end{aligned}$$

On the other hand, the **pop** transition of the collapsed semantics behaves deterministically; therefore, the collapsed semantics cannot capture the nondeterministic behavior of the **pop** transition of the digitized semantics.

Although we give up the forward simulation, the following *backward* simulation holds:

Lemma 8 :
$$\begin{tikzcd} & \boldsymbol{c}' \arrow[d, phantom, "⫪" description] & & \exists\boldsymbol{c} \arrow[r, "\textsc{Coll}"'] \arrow[d, phantom, "⫪" description] & \boldsymbol{c}' \arrow[d, phantom, "⫪" description] \\ \boldsymbol{d} \arrow[r, "\textsc{Digi}"'] & \boldsymbol{d}' & \Longrightarrow & \boldsymbol{d} \arrow[r, "\textsc{Digi}"'] & \boldsymbol{d}'. \end{tikzcd}$$

The backward simulation naturally solves the above determinacy vs. nondeterminacy problem. Furthermore, as we will see below, the backward simulation is key to establishing the decidability of the configuration reachability problem.

4.2 Backward Simulation in Configuration Reachability Problem

We see how to use the backward simulation lemma (Lemma 8) in the configuration reachability problem. Let us consider the following configuration reachability problem:

$$\begin{aligned} &\langle q_{\text{init}}, \langle \bot, \{x \mapsto 0.0;\ y \mapsto 0.0\}\rangle\rangle \\ \xrightarrow[\textsc{Stnd}]{}^{*}\ &\langle q, \langle \bot, \{x \mapsto 2.71;\ y \mapsto 3.14\}\rangle\rangle. \end{aligned} \qquad (\star)$$

Hereafter we confirm that the above problem is equivalent to find a digital valuation $\boldsymbol{d}$ that satisfies $\{x \mapsto 2.71; y \mapsto 3.14\} \models \boldsymbol{d}$ and the following:

$$\begin{aligned} &\langle q_{\text{init}}, \langle \bot, \{(\underset{\bullet}{x},0),(\underset{\bullet}{y},0),(\dot{x},0),(\dot{y},0)\}_0\rangle\rangle \\ \xrightarrow[\textsc{Digi}]{}^{*}\ &\langle q, \langle \bot, \boldsymbol{d}\rangle\rangle. \end{aligned} \qquad (\sharp)$$

For simplicity, we do not consider reference clocks in this example and assume $\mathsf{M} \geq 4$.

Let us assume the following digital valuation $\boldsymbol{d}$ satisfies $(\sharp)$:

$$\boldsymbol{d} = \{\}_0 \{(\dot{y},3)\}\{(\dot{x},2)\}\{(\underset{\bullet}{x},\infty),(\underset{\bullet}{y},\infty)\}.$$

We consider the following collapsed valuation μ such that $\mu \models \boldsymbol{d}$:

$$\mu = \left\{\begin{array}{c} \dot{x} \mapsto 2.71;\ \dot{y} \mapsto 3.14; \\ \hdashline \underset{\bullet}{x} \mapsto \infty.9;\ \underset{\bullet}{y} \mapsto \infty.9 \end{array}\right\} \models \boldsymbol{d}.$$

The backward simulation lemma, Lemma 8, ensures the following run:

$$\langle q_{\text{init}}, \langle \bot, \left\{\begin{array}{c} \dot{x} \mapsto 0.0;\ \dot{y} \mapsto 0.0; \\ \hdashline \underset{\bullet}{x} \mapsto 0.0;\ \underset{\bullet}{y} \mapsto 0.0 \end{array}\right\}\rangle\rangle \xrightarrow[\textsc{Coll}]{}^{*} \langle q, \langle \bot, \mu\rangle\rangle.$$

Sequentially applying the forward simulation lemmas, Lemma 5 and 2, to this run, we obtain the above run $(\star)$. Therefore, to solve $(\star)$, it suffices to find a digital valuation $\boldsymbol{d}$ such that $\{x \mapsto 2.71; y \mapsto 3.14\} \models \boldsymbol{d}$ and solve $(\sharp)$. Indeed, we will use the same argument in the proof of our main theorem (Theorem 1).

In the above argument, using the backward simulation is key and we cannot replace it by the forward simulation of the digitized semantics by the collapsed semantics. (As we have seen above, we cannot forwardly simulate the semantics $\textsc{Digi}$ by the semantics $\textsc{Coll}$ due to the nondeterminacy of **pop**-transitions.) Even if we could apply the forward simulation to the run $(\sharp)$, then we may obtain the following run:

$$\begin{aligned} &\langle q_{\text{init}}, \langle \bot, \left\{\begin{array}{c} \dot{x} \mapsto 0.0;\ \dot{y} \mapsto 0.0; \\ \hdashline \underset{\bullet}{x} \mapsto 0.0;\ \underset{\bullet}{y} \mapsto 0.0 \end{array}\right\}\rangle\rangle \xrightarrow[\textsc{Coll}]{}^{*} \\ &\langle q, \langle \bot, \mu' = \left\{\begin{array}{c} \dot{x} \mapsto 2.34;\ \dot{y} \mapsto 3.09; \\ \hdashline \underset{\bullet}{x} \mapsto \infty.9;\ \underset{\bullet}{y} \mapsto \infty.9 \end{array}\right\}\rangle\rangle \end{aligned}$$

where $\mu' \models \boldsymbol{d}$. Since the forward simulation only ensures a run to $\langle q, \langle \bot, \mu'\rangle\rangle$ where $\mu' \models \boldsymbol{d}$, we cannot show the existence of a run to $\langle q, \langle \bot, \mu\rangle\rangle$ where $\mu(\dot{x}) = 2.71$ and $\mu(\dot{y}) = 3.14$. If we apply Lemma 5 and 6 to this run. then we only obtain the following run that differs from $(\star)$:

$$\begin{aligned} &\langle q_{\text{init}}, \langle \bot, \{x \mapsto 0.0;\ y \mapsto 0.0\}\rangle\rangle \\ \xrightarrow[\textsc{Stnd}]{}^{*}\ &\langle q, \langle \bot, \{x \mapsto 2.34;\ y \mapsto 3.09\}\rangle\rangle. \end{aligned}$$

4.3 Comparing Proof of Abdulla et al. and Ours

We review the proof of *Lemma 4* of Abdulla et al. in [1], which enables us to reduce the location reachability problem of the standard semantics to the location reachability problem of the digitized semantics. The proof structure of their lemma can be summarized schematically as the following diagram by using our notation:

$$\begin{tikzcd}
\boldsymbol{W} \arrow[r, "\textsc{Digi}"] \arrow[d, phantom, "\wr"] & \boldsymbol{W}' \arrow[d, phantom, "\wr"] \\
{}^{\exists}C \arrow[d, phantom, "\wr\!\!\mid\!\mid"] & {}^{\forall}C' \arrow[l, dashed, bend right] \arrow[d, phantom, "\wr\!\!\mid\!\mid"] \\
{}^{\forall}w \arrow[r, "\textsc{Stnd}"] \arrow[r, dashed, bend left] & {}^{\exists}w'
\end{tikzcd}.$$

This diagram says that if $\boldsymbol{W} \xrightarrow[\textsc{Digi}]{}{}^{*} \boldsymbol{W'}$ and $C' \approx \boldsymbol{W'}$, then there is C such that for all stack $w \cong C$ there exists $w' \cong C'$ such that $w \xrightarrow[\textsc{Stnd}]{}{}^{*} w'$. Let us explain the definition of the relation $\approx$ and $\cong$. Informally, $C \approx \boldsymbol{W}$ means that a valuation C is obtained by flattening a stack $\boldsymbol{W}$. Let us consider the following stack:

$$\boldsymbol{W} = \cfrac{\boldsymbol{d}_3 = \{(y,1)\}_0\,\{(x,2)\}}{\cfrac{\boldsymbol{d}_2 = \{\}_0\,\{(x,3)(y,4)\}}{\boldsymbol{d}_1 = \{\}_0\,\{(x,1)\}\,\{(y,5)\}\,.}}$$

The following is one of the valuations obtained by flattening $\boldsymbol{W}$:

$$C = \left\{ \begin{array}{l} y^{(3)} \mapsto 1.0;\ x^{(1)} \mapsto 1.1;\ x^{(3)} \mapsto 2.4; \\ x^{(2)} \mapsto 3.6;\ y^{(2)} \mapsto 4.6;\ y^{(1)} \mapsto 5.9 \end{array} \right\}.$$

The tag of each clock $x^{(i)}$ or $y^{(j)}$ points to the frame where the clock comes from: for example, the clock $y^{(3)}$ and $x^{(1)}$ comes from the digital valuation $\boldsymbol{d}_3$ and $\boldsymbol{d}_1$ of $\boldsymbol{W}$, respectively. Informally, $w \cong C$ means that a stack w matches a valuation C or w is isomorphic to C. For the above flatten valuation C, there exists the unique stack w that matches C:

$$w = \cfrac{\nu_3 = \{y \mapsto 1.0;\ x \mapsto 2.4\}}{\cfrac{\nu_2 = \{x \mapsto 3.6;\ y \mapsto 4.6\}}{\nu_1 = \{x \mapsto 1.1;\ y \mapsto 5.9\}}}$$

because we can reconstruct the stack w from the tag information of C.

They directly bridged the two semantics, the standard semantics and digitized semantics. As a result, their simulation requires an elaborated form; we find out that their elaborate simulation is called a *backward-forward simulation* in Lynch and Vaandrager [16]. 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 relation $\approx$ is not defined by a componentwise manner and it is another source of their elaborated proof.

In contrast, we clearly solve these problems as Lemmas 2, 5, and 8 by considering the intermediate semantics Lazy and Coll.

$$\begin{tikzcd}
\boldsymbol{W} \arrow[r, "\textsc{Digi}"] \arrow[d, phantom, "\models"] \arrow[r, phantom, "\text{Lem 8}"', shift right=4] & \boldsymbol{W}' \arrow[d, phantom, "\models"] \\
{}^{\exists}\boldsymbol{w} \arrow[r, "\textsc{Coll}"] \arrow[d, phantom, "\models"] \arrow[r, phantom, "\text{Lem 5}"', shift right=4] & {}^{\forall}\boldsymbol{w}' \arrow[l, dashed, bend right] \arrow[d, phantom, "\models"] \\
{}^{\forall}\omega \arrow[r, "\textsc{Lazy}"] \arrow[r, dashed, bend left] \arrow[d, phantom, "\models"] \arrow[r, phantom, "\text{Lem 2}"', shift right=4] & {}^{\exists}\omega' \arrow[d, phantom, "\models"] \\
{}^{\forall}w \arrow[r, "\textsc{Stnd}"] \arrow[r, dashed, bend left] & {}^{\exists}w'
\end{tikzcd}$$

This allows us to separate the above mixed simulation into three simple simulations and define correspondences $\models$ in a componentwise manner. Finally, these make the entire proof structure easy to understand.

5 Lazy Semantics: Removing Entire Stack Modification

We define notations to formalize the lazy time elapsing technique.

Definition 4 (Clock Marking)**.** Let $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$ be a finite clock set. We use $\dot{\mathcal{X}}$ to denote the marked set $\{\dot{x}_1, \dot{x}_2, \ldots, \dot{x}_n\}$ and $\underset{\cdot}{\mathcal{X}}$ to denote the marked set $\{\underset{\cdot}{x}_1, \underset{\cdot}{x}_2, \ldots, \underset{\cdot}{x}_n\}$.

Let $\nu : \mathcal{X} \to \mathbb{R}_+$ be a valuation on $\mathcal{X}$. We write $\dot{\nu}$ to denote the marked valuation $\dot{\nu} : \dot{\mathcal{X}} \to \mathbb{R}_+$ defined by $\dot{\nu}(\dot{x}) \triangleq \nu(x)$. We also write $\underset{\cdot}{\nu}$ for the marked valuation on $\underset{\cdot}{\mathcal{X}}$.

For a constraint φ on $\mathcal{X}$, we use $\dot{\varphi}$ to denote the corresponding marked constraint on $\dot{\mathcal{X}}$. For example, if $\varphi = \big(\{x\} = 0 \wedge \{x\} < \{y\}\big)$, then $\dot{\varphi} = \big(\{\dot{x}\} = 0 \wedge \{\dot{x}\} < \{\dot{y}\}\big)$. ♢

Definition 5 (Compatibility and Composition)**.** Let μ_1 and μ_2 be valuations on $\underset{\cdot}{\mathcal{X}} \cup \dot{\mathcal{X}}$.

The valuation μ_1 is *compatible* with the valuation μ_2 if $\mu_1(\dot{x}) = \mu_2(\underset{\cdot}{x})$ for all $x \in \mathcal{X}$ and we write $\mu_1 \mathbin{/\!/} \mu_2$.

If a valuation μ_1 is compatible with a valuation μ_2, the composed valuation $\mu_1 \odot \mu_2$ is defined as follows:

$$(\mu_1 \odot \mu_2)(\underset{\cdot}{x}) = \mu_1(\underset{\cdot}{x}), \quad (\mu_1 \odot \mu_2)(\dot{x}) = \mu_2(\dot{x}).$$

♢

As we have explained in the previous section, we need a reference clock to justify a simulation between the collapsed and digitized semantics.

Definition 6. Let $\mathcal{X}$ be a finite clock set. We write $\mathcal{X}_{\mathtt{C}}$ to denote the clock set $\mathcal{X} \cup \{\mathtt{C}\}$ extended by a reference clock $\mathtt{C}$.

Let ν_1 and ν_2 be valuations on $\mathcal{X}$ and μ be a valuation on $\underset{\bullet}{\mathcal{X}}_{\mathtt{C}} \cup \dot{\mathcal{X}}_{\mathtt{C}}$. If $\left(\underset{\bullet}{\nu_1} \cup \dot{\nu_2}\right) = \mu \restriction (\underset{\bullet}{\mathcal{X}} \cup \dot{\mathcal{X}})$, then we write $\langle \nu_1, \nu_2 \rangle \models \mu$. $\diamondsuit$

For example, let us simulate the following transitions with our notations:

$$\nu_1 \xrightarrow{\textsf{push}} \nu_1\nu_2 \overset{2.0}{\rightsquigarrow} \nu_1'\nu_2' \xrightarrow{\textsf{pop}} \nu_2'$$

where

$$\nu_1 = \{x \mapsto 1.5\},\ \nu_2 = \{x \mapsto 0.0\},\ \nu_i' = \nu_i + 2.$$

We start from the following paired valuation μ_1:

$$\mu_1 = \left\{\begin{array}{c} \dot{x} \mapsto 1.5;\ \dot{\mathtt{C}} \mapsto r_2 \\ \hdashline \underset{\bullet}{x} \mapsto r_x;\ \underset{\bullet}{\mathtt{C}} \mapsto r_1 \end{array}\right\}.$$

A 1-height stack μ corresponds to a 1-height stack ν if $\dot{\nu} = \mu \restriction \dot{\mathcal{X}}$ holds. Therefore, the above 1-height stack μ_1 corresponds to the 1-height stack ν_1.

We simulate the first transition $\nu_1 \xrightarrow{\textsf{push}} \nu_1\nu_2$ as follows ($\mu_1 \to \mu_1'\mu_2$):

$$\left\{\begin{array}{c} \dot{x} \mapsto 1.5;\ \dot{\mathtt{C}} \mapsto r_2 \\ \hdashline \underset{\bullet}{x} \mapsto r_x;\ \underset{\bullet}{\mathtt{C}} \mapsto r_1 \end{array}\right\} \to \dfrac{\left\{\begin{array}{c} \dot{x} \mapsto 0.0;\ \dot{\mathtt{C}} \mapsto 0.0 \\ \hdashline \underset{\bullet}{x} \mapsto 1.5;\ \underset{\bullet}{\mathtt{C}} \mapsto 0.0 \end{array}\right\}}{\left\{\begin{array}{c} \dot{x} \mapsto 1.5;\ \dot{\mathtt{C}} \mapsto 0.0 \\ \hdashline \underset{\bullet}{x} \mapsto r_x;\ \underset{\bullet}{\mathtt{C}} \mapsto r_1 \end{array}\right\}}.$$

When pushing a new frame, we reset the clock $\dot{\mathtt{C}}$ of the current top frame to 0.0. This resetting is important to establish the backward simulation lemma, Lemma 8, in Section 7.5. We assign the values of the marked clocks $\dot{y}$ of the current top frame to the corresponding marked clocks $\underset{\bullet}{y}$ of the new frame to be pushed. As the result, the new top frame μ_2 reflects the information of the top ν_2 and next to the top ν_1 frames in the standard semantics.

We simulate the second transition $\nu_1\nu_2 \overset{2.0}{\rightsquigarrow} \nu_1'\nu_2'$ as follows ($\mu_1'\mu_2 \to \mu_1'\mu_2'$):

$$\dfrac{\left\{\begin{array}{c} \dot{x} \mapsto 0.0;\ \dot{\mathtt{C}} \mapsto 0.0 \\ \hdashline \underset{\bullet}{x} \mapsto 1.5;\ \underset{\bullet}{\mathtt{C}} \mapsto 0.0 \end{array}\right\}}{\left\{\begin{array}{c} \dot{x} \mapsto 1.5;\ \dot{\mathtt{C}} \mapsto 0.0 \\ \hdashline \underset{\bullet}{x} \mapsto r_x;\ \underset{\bullet}{\mathtt{C}} \mapsto r_1 \end{array}\right\}} \to \dfrac{\left\{\begin{array}{c} \dot{x} \mapsto 2.0;\ \dot{\mathtt{C}} \mapsto 2.0 \\ \hdashline \underset{\bullet}{x} \mapsto 3.5;\ \underset{\bullet}{\mathtt{C}} \mapsto 2.0 \end{array}\right\}}{\left\{\begin{array}{c} \dot{x} \mapsto 1.5;\ \dot{\mathtt{C}} \mapsto 0.0 \\ \hdashline \underset{\bullet}{x} \mapsto r_x;\ \underset{\bullet}{\mathtt{C}} \mapsto r_1 \end{array}\right\}}.$$

When performing timed transitions, we only modify the top frame in a stack as above.

We simulate the last transition $\nu_1'\nu_2' \xrightarrow{\textsf{pop}} \nu_2'$. First, we adjust μ_1' to be matched with the real frame ν_1', and it is formalized by evolving μ_1' until $\mu_1' + \delta \mathrel{/\!/} \mu_2'$. For this case, $\delta = 2.0$ and this corresponds to adding the time (2.0) elapsed after μ_1' was covered by the new top frame μ_2. Next, we compose the two valuations $\mu_1' + 2.0$ and μ_2' as follows:

$$\dfrac{\left\{\begin{array}{c} \dot{x} \mapsto 2.0;\ \dot{\mathtt{C}} \mapsto 2.0; \\ \hdashline \underset{\bullet}{x} \mapsto 3.5;\ \underset{\bullet}{\mathtt{C}} \mapsto 2.0 \end{array}\right\}}{\left\{\begin{array}{c} \dot{x} \mapsto 3.5;\ \dot{\mathtt{C}} \mapsto 2.0; \\ \hdashline \underset{\bullet}{x} \mapsto r_x + 2.0;\ \underset{\bullet}{\mathtt{C}} \mapsto r_1 + 2.0 \end{array}\right\}} \overset{\odot}{\Rightarrow}$$

$$\left\{\begin{array}{c} \dot{x} \mapsto 2.0;\ \dot{\mathtt{C}} \mapsto 2.0; \\ \hdashline \underset{\bullet}{x} \mapsto r_x + 2.0;\ \underset{\bullet}{\mathtt{C}} \mapsto r_1 + 2.0 \end{array}\right\}.$$

Since $((\mu_1' + 2.0) \odot \mu_2')(\dot{x}) = 2.0 = \nu_2'(x)$, the composed valuation reflects the real top frame ν_2'.

5.1 Lazy Semantics Lazy

On the basis of the above description, we formalize the lazy semantics Lazy of SRTA.

Definition 7 (Lazy Semantics Lazy)**.** Let $\mathcal{A} = (Q, q_{\text{init}}, F, \Sigma, \Gamma, \mathcal{X}, \Delta)$ be an SRTA.

We define the infinite-PDS $(Q, \Gamma \times (\underset{\bullet}{\mathcal{X}}_{\mathtt{C}} \cup \dot{\mathcal{X}}_{\mathtt{C}} \to \mathbb{R}_+), \hookrightarrow_d \cup \hookrightarrow_t)$ where discrete transition rules $\hookrightarrow_d$ and time elapsing transition rules $\hookrightarrow_t$ are defined as follows:

- A discrete transition rule $\langle q, \omega \rangle \hookrightarrow_d \langle q', \omega' \rangle$ is defined if there is $q \xrightarrow{\alpha}_{\tau} q' \in \Delta$ and $\omega \overset{\tau}{\hookrightarrow} \omega'$ is defined by following Fig. 2.
- Time elapsing transition rules $\langle q, \langle \gamma, \mu \rangle \rangle \hookrightarrow_t \langle q, \langle \gamma, \mu' \rangle \rangle$ are defined for all $q \in Q$ and $\gamma \in \Gamma$ if $\mu \leq \mu'$ holds.

$\diamondsuit$

Note that, on the lazy semantics, we cannot always perform **pop**-transitions $\omega\, \mu_1\mu_2 \xrightarrow{\textsf{pop}} \omega\, \mu$ because it requires the existence of a valuation μ_1' such that $\mu_1 \leq \mu_1'$ and $\mu_1' \mathrel{/\!/} \mu_2$. However, this is not an obstacle to simulating the standard semantics because the well-formedness of the stack defined below ensures to perform **pop**-transitions.

Definition 8. Let μ_1 and μ_2 be valuations on $\underset{\bullet}{\mathcal{X}} \cup \dot{\mathcal{X}}$. If there is a valuation μ_1' such that $\mu_1 \leq \mu_1'$ and $\mu_1' \mathrel{/\!/} \mu_2$, then we write $\mu_1 \precsim \mu_2$.

For two valuations μ_1 and μ_2 such that $\mu_1 \precsim \mu_2$, we define $\mu_2 \ominus \mu_1 \in \mathbb{R}_+$ by $\mu_2 \ominus \mu_1 \triangleq \mu_2(\underset{\bullet}{x}) - \mu_1(\dot{x})$ where x is a clock of $\mathcal{X}$. It is well-defined because $\mu_2(\underset{\bullet}{x}) - \mu_1(\dot{x})$ does not depend on the choice of a clock $x \in \mathcal{X}$. $\diamondsuit$

Definition 9 (Well-formed Stack)**.** A stack $\omega = \langle \gamma_1, \mu_1 \rangle \langle \gamma_2, \mu_2 \rangle \ldots \langle \gamma_n, \mu_n \rangle \in \big(\Gamma \times (\underset{\bullet}{\mathcal{X}}_{\mathtt{C}} \cup \dot{\mathcal{X}}_{\mathtt{C}} \to$

$$\frac{\mu_1' = \mu_1[\dot{\mathtt{C}} := 0] \quad \mu_1' \mathbin{/\!/} \mu_2 \quad \mu_2(\dot{x}) = 0\ (\forall x \in \mathcal{X}_{\mathtt{C}})}{\langle \gamma_1, \mu_1 \rangle \xhookrightarrow{\mathsf{push}(\gamma_2)} \langle \gamma_1, \mu_1' \rangle \langle \gamma_2, \mu_2 \rangle}\ \mathsf{push}(\gamma_2)$$

$$\frac{\mu_1 \le \mu_1' \quad \mu_1' \mathbin{/\!/} \mu_2}{\langle \gamma_1, \mu_1 \rangle \langle \gamma_2, \mu_2 \rangle \xhookrightarrow{\mathsf{pop}(\gamma_2)} \langle \gamma_1, \mu_1' \odot \mu_2 \rangle}\ \mathsf{pop}(\gamma_2) \qquad \frac{\mu \models \dot{\varphi}}{\langle \gamma, \mu \rangle \xhookrightarrow{\mathsf{check}(\varphi)} \langle \gamma, \mu \rangle}\ \mathsf{check}(\varphi)$$

$$\frac{\mu_2' = \mu_2[\dot{x} := \underset{\bullet}{y}]}{\langle \gamma_1, \mu_1 \rangle \langle \gamma_2, \mu_2 \rangle \xhookrightarrow{\mathsf{dig}(x,y)} \langle \gamma_1, \mu_1 \rangle \langle \gamma_2, \mu_2' \rangle}\ \mathsf{dig}(x, y) \qquad \frac{r \in I \quad \mu' = \mu[\dot{x} := r]}{\langle \gamma, \mu \rangle \xhookrightarrow{x \leftarrow I} \langle \gamma, \mu' \rangle}\ x \leftarrow I$$

Fig. 2 Definition of actions on Lazy semantics

$\mathbb{R}_+))^+$ is *well-formed* $\mathsf{WF}(\omega)$ if, for all $i \in [1..(n-1)]$, the following holds:

- The marked clock $\dot{\mathtt{C}}$ satisfies $\mu_i(\dot{\mathtt{C}}) = 0$;
- $\mu_i \lessapprox \mu_{i+1}$.

$\diamondsuit$

The following two basic properties are easily shown by definitions.

Proposition 1. *Let μ_1 and μ_2 be valuations such that $\mu_1 \lessapprox \mu_2$. There exists the unique valuation μ_1' such that $\mu_1 \le \mu_1'$ and $\mu_1' \mathbin{/\!/} \mu_2$.*

Proof. It suffices to take μ_1' as $\mu_1' = \mu_1 + (\mu_2 \ominus \mu_1)$. For any other δ ($\delta \ne \mu_2 \ominus \mu_1$), it is clear that $\mu_1 + \delta$ is not compatible with μ_2. $\square$

Below, for $\mu_1 \lessapprox \mu_2$, we write $\mu_1 \lhd \mu_2$ to denote the valuation μ_1' uniquely determined by the above proposition.

Proposition 2 (WF is an invariant)**.** *Let ω be a well-formed stack $\mathsf{WF}(\omega)$.*

If $\langle q, \omega \rangle \to \langle q', \omega' \rangle$, then ω' is also a well-formed stack $\mathsf{WF}(\omega')$.

Proof. We consider the following nontrivial case induced by a **pop** transition:

$$\langle q, \omega\mu_1\mu_2\mu_3 \rangle \to \langle q', \omega\mu_1((\mu_2 \lhd \mu_3) \odot \mu_3) \rangle.$$

Since $\mathsf{WF}(\omega\mu_1\mu_2\mu_3)$, we have $\mu_1 \lessapprox \mu_2$. This and $\mu_2 \le (\mu_2 \lhd \mu_3)$ imply $\mu_1 \lessapprox (\mu_2 \lhd \mu_3)$. It is clear that $(\mu_2 \lhd \mu_3)(x) = ((\mu_2 \lhd \mu_3) \odot \mu_3)(x)$ for any $x \in \underset{\bullet}{\mathcal{X}}_{\mathtt{C}}$. Therefore, we obtain $\mu_1 \lessapprox ((\mu_2 \lhd \mu_3) \odot \mu_3)$. $\square$

We define the notations of stack correspondence and configuration correspondence between the standard and lazy semantics.

Definition 10 (Stack and Configuration Correspondence)**.** Let $w \in \big(\Gamma \times (\mathcal{X} \to \mathbb{R}_+)\big)^+$ be a stack of the semantics Stnd and $\omega \in \big(\Gamma \times (\underset{\bullet}{\mathcal{X}}_{\mathtt{C}} \cup \dot{\mathcal{X}}_{\mathtt{C}} \to \mathbb{R}_+)\big)^+$ be a well-formed stack $\mathsf{WF}(\omega)$ of the semantics Lazy.

The *stack correspondence* relation is inductively defined as follows:

- $\langle \gamma, \nu \rangle \models \langle \gamma, \mu \rangle$ if $\dot{\nu} = \mu \restriction \dot{\mathcal{X}}$.
- $w\langle \gamma_1, \nu_1 \rangle \langle \gamma_2, \nu_2 \rangle \models \omega\langle \gamma_1, \mu_1 \rangle \langle \gamma_2, \mu_2 \rangle$ if
  - $\langle \nu_1, \nu_2 \rangle \models \mu_2$;
  - $w\langle \gamma_1, \nu_1 \rangle \models \omega\langle \gamma_1, \mu_1 \lhd \mu_2 \rangle$.

Let $\langle q, w \rangle$ and $\langle q', \omega \rangle$ be configurations of the semantics Stnd and Lazy, respectively. If $q = q'$ and $w \models \omega$, two configurations are corresponding and we write $\langle q, w \rangle \sim \langle q', \omega \rangle$. $\diamondsuit$

For a given well-formed stack ω of paired valuations, there exists the unique stack w such that $w \models \omega$. This is shown by the following property.

Proposition 3 (Recover the concrete stack from a lazy one)**.** *Let $\omega = \langle \gamma_1, \mu_1 \rangle \langle \gamma_2, \mu_2 \rangle \ldots \langle \gamma_n, \mu_n \rangle$ be a well-formed stack $\mathsf{WF}(\omega)$.*

The stack $w = \langle \gamma_1, \nu_1 \rangle \langle \gamma_2, \nu_2 \rangle \ldots \langle \gamma_n, \nu_n \rangle$ defined as follows is the unique stack that satisfies $w \models \omega$:

- $\dot{\nu}_n = \mu_n \restriction \dot{\mathcal{X}}$.
- *For all $i \in [1..(n-1)]$, $\dot{\nu}_i = \mu_i \restriction \dot{\mathcal{X}} + \sum_{j=i}^{n-1} (\mu_{j+1} \ominus \mu_j)$.*

Proof. We proceed by induction on n.

Case $\omega = \langle \gamma_1, \mu_1 \rangle$**:** The valuation ν_1 that satisfies $\dot{\nu}_1 = \mu_1 \restriction \dot{\mathcal{X}}$ is the unique valuation such that $\langle \gamma_1, \nu_1 \rangle \models \langle \gamma_1, \mu_1 \rangle$.

Case $\omega = \langle \gamma_1, \mu_1 \rangle \ldots \langle \gamma_n, \mu_n \rangle \langle \gamma_{n+1}, \mu_{n+1} \rangle$**:**
Since ω is a well-formed stack, a stack $\omega_n = \langle \gamma_1, \mu_1 \rangle \ldots \langle \gamma_n, \mu_n + (\mu_{n+1} \ominus \mu_n) \rangle$ is also well formed and the induction hypothesis leads to

the unique stack $w_n = \langle\gamma_1,\nu_1\rangle\ldots\langle\gamma_1,\nu_n\rangle$ that satisfies $w_n \models \omega_n$ and the following:

$$\begin{aligned}\dot{\nu}_n &= \mu_n \restriction \dot{\mathcal{X}} + (\mu_{n+1} \ominus \mu_n)\\ &= \mu_n \restriction \dot{\mathcal{X}} + \sum_{j=n}^{n}(\mu_{j+1} \ominus \mu_j),\\ \dot{\nu}_i &= \mu_i \restriction \dot{\mathcal{X}} + (\mu_{n+1} \ominus \mu_n) + \sum_{j=i}^{n-1}(\mu_{j+1} \ominus \mu_j)\\ &= \mu_i \restriction \dot{\mathcal{X}} + \sum_{j=i}^{n}(\mu_{j+1} \ominus \mu_j).\end{aligned}$$

Let ν_{n+1} be the unique valuation such that $\dot{\nu}_{n+1} = \mu_{n+1} \restriction \dot{\mathcal{X}}$. It is clear that $w_n\langle\gamma_{n+1},\nu_{n+1}\rangle \models \omega$. The uniqueness of the $w_n\langle\gamma_{n+1},\nu_{n+1}\rangle$ comes from the uniqueness of ν_{n+1} and w_n.

□

The above proposition immediately implies the following property that is key to connecting the two semantics Stnd and Lazy.

Proposition 4. *Let $w\langle\gamma,\nu\rangle$ and $\omega\langle\gamma,\mu\rangle$ be stacks such that $w\langle\gamma,\nu\rangle \models \omega\langle\gamma,\mu\rangle$. For any $\delta \in \mathbb{R}_+$, $(w\langle\gamma,\nu\rangle) + \delta \models \omega\langle\gamma,\mu+\delta\rangle$.*

We show the correspondence $\sim$ forms a bisimulation between Stnd and Lazy.

Lemma 1. *Let $\langle q,w\rangle$ and $\langle q,\omega\rangle$ be configurations such that $w \models \omega$.*

- *If there is a timed transition $\langle q,w\rangle \overset{\delta}{\rightsquigarrow} \langle q,w'\rangle$ for some $\delta \in \mathbb{R}_+$, then there exists ω' such that $\langle q,\omega\rangle \to \langle q,\omega'\rangle$ and $w' \models \omega'$.*
- *If there is a discrete transition $\langle q,w\rangle \xrightarrow{\alpha} \langle q',w'\rangle$ for some $\alpha \in \Sigma \cup \{\epsilon\}$, then there exists ω' such that $\langle q,\omega\rangle \to \langle q',\omega'\rangle$ and $w' \models \omega'$.*

$$\begin{tikzcd}\langle q,w\rangle \arrow[r, "\textsc{Stnd}"'] \arrow[d, phantom, "\wr"] & \langle q',w'\rangle & & \langle q,w\rangle \arrow[r, "\textsc{Stnd}"'] \arrow[d, phantom, "\wr"] & \langle q',w'\rangle \arrow[d, phantom, "\wr"] \\ \langle q,\omega\rangle & & \Longrightarrow & \langle q,\omega\rangle \arrow[r, "\textsc{Lazy}"'] & \exists\langle q',\omega'\rangle.\end{tikzcd}$$

Proof. We only consider the case of timed transitions. The other cases **push**, **dig**, $x \leftarrow I$, and **check**(φ) are trivial and the nontrivial case **pop** is shown in the same way as that of timed transitions.

Let us assume $\langle q,w\langle\gamma,\nu\rangle\rangle \overset{\delta}{\rightsquigarrow} \langle q,(w\langle\gamma,\nu\rangle)+\delta\rangle$. From the assumption, we have $\langle q,w\langle\gamma,\nu\rangle\rangle \sim \langle q,\omega\langle\gamma,\mu\rangle\rangle$ for some valuations μ.

It suffices to show $(w\langle\gamma,\nu\rangle) + \delta \models \omega\langle\gamma,\mu+\delta\rangle$. This is clear from Proposition 4. □

Lemma 2. *Let $\langle q,w\rangle$ and $\langle q,\omega\rangle$ be configurations such that $w \models \omega$. We have the following:*

$$\begin{tikzcd}\langle q,w\rangle \arrow[d, phantom, "\wr"] & & & \langle q,w\rangle \arrow[r, "\textsc{Stnd}"'] \arrow[d, phantom, "\wr"] & \exists\langle q',w'\rangle \arrow[d, phantom, "\wr"] \\ \langle q,\omega\rangle \arrow[r, "\textsc{Lazy}"'] & \langle q',\omega'\rangle & \Longrightarrow & \langle q,\omega\rangle \arrow[r, "\textsc{Lazy}"'] & \langle q',\omega'\rangle.\end{tikzcd}$$

Proof. We only consider the case of **pop** transitions. The other cases **push**, **dig**, $x \leftarrow I$, and **check**(φ) are trivial and the nontrivial case, timed transitions, is shown in the same way as the **pop** transitions.

We consider the following transition:

$$\langle q,\omega\langle\gamma_1,\mu_1\rangle\langle\gamma_2,\mu_2\rangle\langle\gamma_3,\mu_3\rangle\rangle \xrightarrow[\textsc{Lazy}]{\textsf{pop}} \langle q',\omega\langle\gamma_1,\mu_1\rangle\langle\gamma_2,(\mu_2 \lhd \mu_3) \odot \mu_3\rangle\rangle.$$

(The following easy case

$$\langle q,\langle\gamma_1,\mu_1\rangle\langle\gamma_2,\mu_2\rangle\rangle \xrightarrow[\textsc{Lazy}]{\textsf{pop}} \langle q',\langle\gamma_1,(\mu_1 \lhd \mu_2) \odot \mu_2\rangle\rangle$$

is shown in the same argument for the above case.)

From the assumption, for some valuations ν_1, ν_2, and ν_3, we have following:

$$(\star):\quad \begin{aligned}&\langle q,w\langle\gamma_1,\nu_1\rangle\langle\gamma_2,\nu_2\rangle\langle\gamma_3,\nu_3\rangle\rangle \sim\\ &\langle q,\omega\langle\gamma_1,\mu_1\rangle\langle\gamma_2,\mu_2\rangle\langle\gamma_3,\mu_3\rangle\rangle.\end{aligned}$$

This implies the following:

$\sharp_1$ $\langle\nu_2,\nu_3\rangle \models \mu_3$.

$\sharp_2$ $w\langle\gamma_1,\nu_1\rangle\langle\gamma_2,\nu_2\rangle \sim \omega\langle\gamma_1,\mu_1\rangle\langle\gamma_2,\mu_2 \lhd \mu_3\rangle$.

Our aim is to show the following:

$$\begin{aligned}&\langle q',w\langle\gamma_1,\nu_1\rangle\langle\gamma_2,\nu_3\rangle\rangle \sim\\ &\langle q',\omega\langle\gamma_1,\mu_1\rangle\langle\gamma_2,(\mu_2 \lhd \mu_3) \odot \mu_3\rangle\rangle.\end{aligned}$$

First, we show $\langle\nu_1,\nu_3\rangle \models (\mu_2 \lhd \mu_3) \odot \mu_3$.

- Since $((\mu_2 \lhd \mu_3) \odot \mu_3)(\dot{x}) = \mu_3(\dot{x})$ and $\mu_3(\dot{x}) = \nu_3(x)$ (this comes from $\sharp_1$), we have $((\mu_2 \lhd \mu_3) \odot \mu_3)(\dot{x}) = \nu_3(x)$.
- Since $((\mu_2 \lhd \mu_3) \odot \mu_3)(\underset{\bullet}{x}) = (\mu_2 \lhd \mu_3)(\underset{\bullet}{x})$ and $(\mu_2 \lhd \mu_3)(\underset{\bullet}{x}) = \nu_1(x)$ (this comes from $\sharp_2$), we have $((\mu_2 \lhd \mu_3) \odot \mu_3)(\underset{\bullet}{x}) = \nu_1(x)$.

Next, we show the following:

$$w\langle\gamma_1,\nu_1\rangle \models \omega\langle\gamma_1,\mu_1 \lhd ((\mu_2 \lhd \mu_3) \odot \mu_3)\rangle.$$

Since $\mu_1 \lhd ((\mu_2 \lhd \mu_3) \odot \mu_3) = \mu_1 \lhd (\mu_2 \lhd \mu_3)$ is clear from the definition of $\lhd$, it suffices to show $w\langle\gamma_1,\nu_1\rangle \models \omega\langle\gamma_1,\mu_1 \lhd (\mu_2 \lhd \mu_3)\rangle$ and it is immediately shown by $\sharp_2$. □

6 Collapsed Semantics

We cannot formalize the lazy semantics as finite PDS for the unboundedness and denseness of real numbers. In this section, we remove the unboundedness of real numbers. We will remove the denseness of real numbers in the next section and give a finite PDS semantics.

6. 1 Removing the Unboundedness

In order to remove the unboundedness of real numbers, we translate a sufficiently large real number into the corresponding collapsed real number. We define an upper-bound constant M for each SRTA and introduce the collapsed domain $\mathbb{C}$.

Definition 11 (Upper-bound constant and Collapsed domain)**.** Let $\mathcal{A}$ be an SRTA and $\mathcal{I}$ be the set of intervals that appear in $\mathcal{A}$. The upper-bound constant M for $\mathcal{A}$ is defined as follows:

$$\mathsf{M} \triangleq \max\{\, j : [i,j] \in \mathcal{I},\ (i,j) \in \mathcal{I}\} + 1.$$

The set of collapsed real numbers $\mathbb{C}$ is defined as follows:

$$\mathbb{C} \triangleq \big([0..(\mathsf{M}-1)] \cup \{\infty\}\big) \times [0,1).$$

The collapsing function $\mathcal{C} : \mathbb{R}_+ \to \mathbb{C}$ is defined as follows:

$$\mathcal{C}(r) \triangleq \begin{cases} (\infty\ , \{r\}) & \text{if } r \geq \mathsf{M}, \\ (\lfloor r \rfloor, \{r\}) & \text{if } r < \mathsf{M}. \end{cases}$$

For a concrete valuation $\nu : \mathcal{X} \to \mathbb{R}_+$, we define the *collapsed valuation of* ν by $\mathcal{C}(\nu)(x) \triangleq \mathcal{C}(\nu(x))$. $\diamondsuit$

We write $v.r$ to denote (v, r). For example, we write 2.6 and $\infty.3$ to denote the collapsed values $(2, 0.6)$ and $(\infty, 0.3)$, respectively. Moreover, $\lfloor v.r \rfloor$ and $\{v.r\}$ denote v and r, respectively. We use Greek letters $\lambda, \ldots$ to denote a collapsed valuation. Especially, we use $\Lambda, \ldots$ to denote a collapsed valuation on a marked clock set $\underset{\bullet}{\mathcal{X}}_{\mathtt{C}} \cup \dot{\mathcal{X}}_{\mathtt{C}}$.

The following basic properties are immediately shown by the above definition.

Proposition 5. *Let ν_1 and ν_2 be valuations on a finite clock set $\mathcal{X}$. If $\mathcal{C}(\nu_1) = \mathcal{C}(\nu_2)$,*

Validity. $\nu_1 \models \varphi$ *iff* $\nu_2 \models \varphi$ *for any constraint* φ.

Copying. $\mathcal{C}(\nu_1[x := y]) = \mathcal{C}(\nu_2[x := y])$ *for any* $x, y \in \mathcal{X}$.

Updating. $\mathcal{C}(\nu_1[x := r]) = \mathcal{C}(\nu_2[x := r])$ *for any* $x \in \mathcal{X}$ *and* $r \in \mathbb{R}_+$.

Evolving. $\mathcal{C}(\nu_1 + \delta) = \mathcal{C}(\nu_2 + \delta)$ *for any* $\delta \in \mathbb{R}_+$.

On the basis of Proposition 5, we define operations for collapsed valuations as follows.

Definition 12. Let $\mathcal{X}$ be a finite clock set, ν and λ be concrete and collapsed valuations on $\mathcal{X}$ such that $\mathcal{C}(\nu) = \lambda$.

- For a constraint φ, we write $\lambda \models \varphi$ if $\nu \models \varphi$.
- For a real number $r \in \mathbb{R}_+$, $\lambda[x := r] \triangleq \mathcal{C}(\nu[x := r])$.
- For clocks $x, y \in \mathcal{X}$, $\lambda[x := y] \triangleq \mathcal{C}(\nu[x := y])$.
- For a real number $\delta \in \mathbb{R}_+$, $\lambda + \delta \triangleq \mathcal{C}(\nu + \delta)$.

$\diamondsuit$

The above definition is well-defined because Proposition 5 ensures that the result does not depend on the choice of a witness ν for λ.

We define a (quasi) ordering $\lambda \preccurlyeq \lambda'$ for collapsed valuations that corresponds to the ordering $\leq$ on concrete valuations.

Definition 13. Let λ and λ' be collapsed valuations. We write $\lambda \preccurlyeq \lambda'$ if there are two concrete valuations ν and ν' such that $\nu \leq \nu'$, $\mathcal{C}(\nu) = \lambda$, and $\mathcal{C}(\nu') = \lambda'$. $\diamondsuit$

This quasi-ordering is not antisymmetric because $\{x \mapsto \infty.0\} \preccurlyeq \{x \mapsto \infty.5\}$ and $\{x \mapsto \infty.5\} \preccurlyeq \{x \mapsto \infty.0\}$ but these are different valuations.

We define the compatibility and composition in the same way as the lazy semantics.

Definition 14. Let Λ_1 and Λ_2 be collapsed valuations on a finite set of clocks $\underset{\bullet}{\mathcal{X}}_{\mathtt{C}} \cup \dot{\mathcal{X}}_{\mathtt{C}}$.

- If $\Lambda_1(\dot{x}) = \Lambda_2(\underset{\bullet}{x})$ for any $x \in \mathcal{X}_{\mathtt{C}}$, then Λ_1 is *compatible* with Λ_2 and we write $\Lambda_1 \mathbin{//} \Lambda_2$.
- If $\Lambda_1 \mathbin{//} \Lambda_2$, then the *composed* collapsed valuation $\Lambda_1 \odot \Lambda_2 : \underset{\bullet}{\mathcal{X}}_{\mathtt{C}} \cup \dot{\mathcal{X}}_{\mathtt{C}} \to \mathbb{C}$ is defined as follows:
  - $(\Lambda_1 \odot \Lambda_2)(\underset{\bullet}{x}) \triangleq \Lambda_1(\underset{\bullet}{x})$ for all $x \in \mathcal{X}_{\mathtt{C}}$.
  - $(\Lambda_1 \odot \Lambda_2)(\dot{x}) \triangleq \Lambda_2(\dot{x})$ for all $x \in \mathcal{X}_{\mathtt{C}}$.

$\diamondsuit$

6. 2 Why Do We Need Reference Clock

Let us consider the following pop transition of the lazy semantics:

$$\frac{\mu_2 = \left\{\begin{array}{c} \dot{x} \mapsto 3.5;\ \dot{\mathtt{C}} \mapsto 2.0; \\ \hdashline \underset{\bullet}{x} \mapsto 4.0;\ \underset{\bullet}{\mathtt{C}} \mapsto 2.0 \end{array}\right\}}{\mu_1 = \left\{\begin{array}{c} \dot{x} \mapsto 2.0;\ \dot{\mathtt{C}} \mapsto 0.0; \\ \hdashline \underset{\bullet}{x} \mapsto 0.5;\ \underset{\bullet}{\mathtt{C}} \mapsto 0.0 \end{array}\right\}} \xrightarrow{\textsf{pop}} \mu$$

where

$$\mu = \left\{\begin{array}{c} \dot{x} \mapsto 3.5;\ \dot{\mathtt{C}} \mapsto 2.0; \\ \hdashline \underset{\bullet}{x} \mapsto 2.5;\ \underset{\bullet}{\mathtt{C}} \mapsto 2.0 \end{array}\right\}.$$

The above run is simulated as follows under

$\mathsf{M} = 3$ in the collapsed semantics:

$$\frac{\Lambda_2 = \left\{\begin{array}{c} \overset{\bullet}{x} \mapsto \infty.5;\ \overset{\bullet}{\mathtt{C}} \mapsto 2.0; \\ \hdashline \underset{\bullet}{x} \mapsto \infty.0;\ \underset{\bullet}{\mathtt{C}} \mapsto 2.0 \end{array}\right\}}{\Lambda_1 = \left\{\begin{array}{c} \overset{\bullet}{x} \mapsto 2.0;\ \overset{\bullet}{\mathtt{C}} \mapsto 0.0; \\ \hdashline \underset{\bullet}{x} \mapsto 0.5;\ \underset{\bullet}{\mathtt{C}} \mapsto 0.0 \end{array}\right\}} \Rightarrow \Lambda$$

where

$$\Lambda = \left\{\begin{array}{c} \overset{\bullet}{x} \mapsto \infty.5;\ \overset{\bullet}{\mathtt{C}} \mapsto 2.0; \\ \hdashline \underset{\bullet}{x} \mapsto 2.5;\ \underset{\bullet}{\mathtt{C}} \mapsto 2.0 \end{array}\right\}.$$

In order to compute Λ from Λ_1 and Λ_2, we evolve Λ_1 to Λ_1' ($\Lambda_1 \preccurlyeq \Lambda_1'$) until $\Lambda_1' \mathbin{//} \Lambda_2$ and compose them as $\Lambda = \Lambda_1' \odot \Lambda_2$. For this case, Λ_1' is uniquely determined as $\Lambda_1' = \Lambda_1 + 2.0$ by the reference clocks $\overset{\bullet}{\mathtt{C}}$ of Λ_1 and $\underset{\bullet}{\mathtt{C}}$ of Λ_2. In general, the presence of reference clocks ensures the uniqueness of such Λ_1' (see Proposition 6 and Lemma 3).

On the other hand, if we drop reference clocks, then the above **pop** transition behaves nondeterministically. Let us consider the following stack that is obtained by removing the reference clocks from the above transition:

$$\frac{\lambda_2 = \left\{\underset{\bullet}{x} \mapsto \infty.0;\ \overset{\bullet}{x} \mapsto \infty.5\right\}}{\lambda_1 = \left\{\underset{\bullet}{x} \mapsto 0.5;\ \overset{\bullet}{x} \mapsto 2.0\right\}}.$$

Due to the absence of reference clocks, there are the following three possibilities about λ_1' such that $\lambda_1 \preccurlyeq \lambda_1'$ and $\lambda_1' \mathbin{//} \lambda_2$:

- $\lambda_1 + 1.0 = \left\{\underset{\bullet}{x} \mapsto 1.5;\quad \overset{\bullet}{x} \mapsto \infty.0\right\} \mathbin{//} \lambda_2$.
- $\lambda_1 + 2.0 = \left\{\underset{\bullet}{x} \mapsto 2.5;\quad \overset{\bullet}{x} \mapsto \infty.0\right\} \mathbin{//} \lambda_2$.
- $\lambda_1 + 3.0 = \left\{\underset{\bullet}{x} \mapsto \infty.5;\quad \overset{\bullet}{x} \mapsto \infty.0\right\} \mathbin{//} \lambda_2$.

Thus, we have the following nondeterministic transition:

$$\lambda_1\lambda_2 \xrightarrow{\textsf{pop}} \{\lambda_1 + 1 \odot \lambda_2, \lambda_1 + 2 \odot \lambda_2, \lambda_1 + 3 \odot \lambda_2\}.$$

Since the valuation μ only justifies $(\lambda_1 + 2.0) \odot \lambda_2$ for $\mathcal{C}(\mu)(\underset{\bullet}{x}) = 2.5$, the lazy semantics without reference clocks cannot simulate the collapsed semantics.

We show a key property to ensure the uniqueness of **pop** transitions.

Proposition 6 (♣). *Let λ, λ', λ'' be collapsed valuations such that $\lambda \preccurlyeq \lambda'$ and $\lambda \preccurlyeq \lambda''$.*

If the following conditions hold, then $\lambda' = \lambda''$: there is a clock $\mathtt{C}$ such that

($\sharp_1$) $\lambda(\mathtt{C}) = 0.0$*; and*

($\sharp_2$) $\lambda'(\mathtt{C}) = \lambda''(\mathtt{C})$.

Proof. We proceed by case analysis on $\lfloor \lambda'(\mathtt{C}) \rfloor$:

Case $\lfloor \lambda'(\mathtt{C}) \rfloor \neq \infty$: The condition ($\sharp_2$) implies $\lambda'(\mathtt{C}) = \lambda''(\mathtt{C}) = (i, r)$ for some $i \neq \infty$ and $r \in [0, 1)$. This means that $\lambda' = \lambda + (i, r)$ and $\lambda'' = \lambda + (i, r)$. Hence $\lambda' = \lambda''$.

Case $\lfloor \lambda'(\mathtt{C}) \rfloor = \infty$: The condition ($\sharp_2$) implies $\lambda'(\mathtt{C}) = \lambda''(\mathtt{C}) = (\infty, r)$ for some $r \in [0, 1)$. In contrast to the above case, in general, there may exist two distinct real numbers δ_1 and δ_2 such that $\lambda' = \lambda + \delta_1$ and $\lambda'' = \lambda + \delta_2$. Here, we assume $\delta_1 > \delta_2$.

It suffices to show $\{\lambda'(x)\} = \{\lambda''(x)\}$ and $\lfloor \lambda'(x) \rfloor = \lfloor \lambda''(x) \rfloor$ for any clock x.

$\{\lambda'(x)\} = \{\lambda''(x)\}$: We have $\delta_1 - \delta_2 \in \mathbb{N}$ because $\{\lambda'(\mathtt{C})\} = \{\lambda''(\mathtt{C})\}$. This implies $\{\lambda'(x)\} = \{\lambda''(x)\}$ for any clock x.

$\lfloor \lambda'(x) \rfloor = \lfloor \lambda''(x) \rfloor$: The condition ($\sharp_1$) implies that any clocks are collapsed: $\lfloor \lambda'(x) \rfloor = \lfloor \lambda''(x) \rfloor = \infty$ for any clock x.

□

Remark: We cannot replace the two conditions of Proposition 6 by the following single condition:

($\sharp_3$) There is a clock $\mathtt{C}$ such that $\lambda'(\mathtt{C}) = \lambda''(\mathtt{C})$.

For example, let us consider the following valuations under $\mathsf{M} = 4$:

$$\begin{aligned} \lambda &= \{x \mapsto 0.5;\ \mathtt{C} \mapsto 3.0\}, \\ \lambda' &= \lambda + 1 = \{x \mapsto 1.5;\ \mathtt{C} \mapsto \infty.0\}, \\ \lambda'' &= \lambda + 2 = \{x \mapsto 2.5;\ \mathtt{C} \mapsto \infty.0\}. \end{aligned}$$

We have $\lambda'(\mathtt{C}) = \lambda''(\mathtt{C})$; however, $\lambda' \neq \lambda''$.

6.3 Collapsed Semantics Coll

Collapsed valuations lead to the collapsed semantics Coll, which removes the unboundedness of real numbers from the lazy semantics Lazy.

Definition 15 (Collapsed Semantics). Let $\mathcal{A} = (Q, q_{\text{init}}, F, \Sigma, \Gamma, \mathcal{X}, \Delta)$ be an SRTA.

We define the infinite-PDS $(Q, \Gamma \times (\underset{\bullet}{\mathcal{X}_{\mathtt{C}}} \cup \overset{\bullet}{\mathcal{X}_{\mathtt{C}}} \to \mathbb{C}), \hookrightarrow_d \cup \hookrightarrow_t)$ where discrete transition rules $\hookrightarrow_d$ and time elapsing transition rules $\hookrightarrow_t$ are defined as follows:

- A discrete transition rule $\langle q, \boldsymbol{w} \rangle \hookrightarrow_d \langle q', \boldsymbol{w'} \rangle$ is defined if there is $q \xrightarrow{\alpha}_{\tau} q' \in \Delta$ and $\boldsymbol{w} \overset{\tau}{\hookrightarrow} \boldsymbol{w'}$ is defined by following Fig. 3.
- Time elapsing transition rules $\langle q, \langle \gamma, \Lambda \rangle \rangle \hookrightarrow_t \langle q, \langle \gamma, \Lambda' \rangle \rangle$ are defined for all $q \in Q$ and $\gamma \in \Gamma$ if $\Lambda \preccurlyeq \Lambda'$ holds.

♢

In the same way as the lazy semantics Lazy, we define the notion of the well-formed stack and prove

$$\frac{\Lambda_1' = \Lambda_1[\dot{\mathtt{C}} := 0] \quad \Lambda_1' \mathbin{//} \Lambda_2 \quad \Lambda_2(\dot{x}) = 0\ (\forall x \in \dot{\mathcal{X}}_{\mathtt{C}})}{\langle \gamma_1, \Lambda_1 \rangle \xhookrightarrow{\mathsf{push}(\gamma_2)} \langle \gamma_1, \Lambda_1' \rangle \langle \gamma_2, \Lambda_2 \rangle}\ \mathsf{push}(\gamma_2)$$

$$\frac{\Lambda_1 \preccurlyeq \Lambda_1' \quad \Lambda_1' \mathbin{//} \Lambda_2}{\langle \gamma_1, \Lambda_1 \rangle \langle \gamma_2, \Lambda_2 \rangle \xhookrightarrow{\mathsf{pop}(\gamma_2)} \langle \gamma_1, \Lambda_1 \odot \Lambda_2 \rangle}\ \mathsf{pop}(\gamma_2) \qquad \frac{\Lambda \models \dot{\varphi}}{\langle \gamma, \Lambda \rangle \xhookrightarrow{\mathsf{check}(\varphi)} \langle \gamma, \Lambda \rangle}\ \mathsf{check}(\varphi)$$

$$\frac{\Lambda_2' = \Lambda_2[\dot{x} := \underset{\bullet}{y}]}{\langle \gamma_1, \Lambda_1 \rangle \langle \gamma_2, \Lambda_2 \rangle \xhookrightarrow{\mathsf{dig}(x,y)} \langle \gamma_1, \Lambda_1 \rangle \langle \gamma_2, \Lambda_2' \rangle}\ \mathsf{dig}(x,y) \qquad \frac{r \in I \quad \Lambda' = \Lambda[\dot{x} := r]}{\langle \gamma, \Lambda \rangle \xhookrightarrow{x \leftarrow I} \langle \gamma, \Lambda' \rangle}\ x \leftarrow I$$

Fig. 3 Definition of actions on Coll semantics

some properties about well-formed stacks.

Definition 16. Let Λ_1 and Λ_2 be collapsed valuations on $\underset{\bullet}{\mathcal{X}}_{\mathtt{C}} \cup \dot{\mathcal{X}}_{\mathtt{C}}$. If there is a collapsed valuation Λ_1' such that $\Lambda_1 \preccurlyeq \Lambda_1'$ and $\Lambda_1' \mathbin{//} \Lambda_2$, then we write $\Lambda_1 \precapprox \Lambda_2$. ♢

Definition 17 (Well-formed Stack)**.** A stack $\boldsymbol{w} = \langle \gamma_1, \Lambda_1 \rangle \langle \gamma_2, \Lambda_2 \rangle \dots \langle \gamma_n, \Lambda_n \rangle$ is *well-formed* $\mathsf{WF}(\boldsymbol{w})$ if for all $i \in [1..(n-1)]$

- The marked clock $\dot{\mathtt{C}}$ satisfies $\Lambda_i(\dot{\mathtt{C}}) = 0$; and
- $\Lambda_i \precapprox \Lambda_{i+1}$.

♢

As the lazy semantics, the well-formedness WF forms an invariant.

Proposition 7. *Let $\langle q, \boldsymbol{w} \rangle$ be a configuration where $\boldsymbol{w}$ is a well-formed stack $\mathsf{WF}(\boldsymbol{w})$.*

If $\langle q, \boldsymbol{w} \rangle \to \langle q, \boldsymbol{w'} \rangle$, then $\boldsymbol{w'}$ is also a well-formed stack $\mathsf{WF}(\boldsymbol{w'})$.

Proof. This is shown in the same argument of the proof in Proposition 2. □

As we mentioned above, the presence of reference clocks is key to the following property and the determinacy of **pop** transitions. The following property, which corresponds to Proposition 1, is immediate from Proposition 6.

Lemma 3. *If $\mathsf{WF}(\boldsymbol{w}\langle \gamma_1, \Lambda_1 \rangle \langle \gamma_2, \Lambda_2 \rangle)$, then there exists the* unique *Λ_1' such that $\Lambda \preccurlyeq \Lambda_1'$ and $\Lambda_1' \mathbin{//} \Lambda_2$. We use $\Lambda_1 \lhd \Lambda_2$ for such the unique valuation.*

We define the stack and configuration correspondence between Lazy and Coll.

Definition 18. Let $\omega = \mu_1 \dots \mu_n$ and $\boldsymbol{w} = \Lambda_1 \dots \Lambda_n$ be well-formed stacks $\mathsf{WF}(\omega), \mathsf{WF}(\boldsymbol{w})$ of the lazy and collapsed semantics, respectively.

The *stack correspondence* $\omega \models \boldsymbol{w}$ is defined if $\mathcal{C}(\mu_i) = \Lambda_i$ for all $i \in [1..n]$.

It is naturally extended to configurations: we write $\langle q, \omega \rangle \sim \langle q', \boldsymbol{w} \rangle$ if $q = q'$ and $\omega \models \boldsymbol{w}$. ♢

We show the correspondence $\sim$ forms a bisimulation between Lazy and Coll.

Lemma 4. *Let $\langle q, \omega \rangle$ and $\langle q, \boldsymbol{w} \rangle$ be configurations such that $\omega \models \boldsymbol{w}$.*

$$\begin{tikzcd} \langle q, \omega \rangle \arrow[r, "\text{Lazy}"'] \arrow[d, phantom, "\wr"] & \langle q', \omega' \rangle \\ \langle q, \boldsymbol{w} \rangle & \end{tikzcd} \implies \begin{tikzcd} \langle q, \omega \rangle \arrow[r, "\text{Lazy}"'] \arrow[d, phantom, "\wr"] & \langle q', \omega' \rangle \arrow[d, phantom, "\wr"] \\ \langle q, \boldsymbol{w} \rangle \arrow[r, "\text{Coll}"'] & \exists \langle q', \boldsymbol{w'} \rangle. \end{tikzcd}$$

Proof. Since the stack correspondence relation $\omega \models \boldsymbol{w}$ is defined in a componentwise way, the forward simulation is immediately shown for all the transition rules. □

Lemma 5. *Let $\langle q, \omega \rangle$ and $\langle q, \boldsymbol{w} \rangle$ be configurations such that $\omega \models \boldsymbol{w}$.*

$$\begin{tikzcd} \langle q, \omega \rangle \arrow[d, phantom, "\wr"] & \\ \langle q, \boldsymbol{w} \rangle \arrow[r, "\text{Coll}"'] & \langle q', \boldsymbol{w'} \rangle \end{tikzcd} \implies \begin{tikzcd} \langle q, \omega \rangle \arrow[r, "\text{Lazy}"'] \arrow[d, phantom, "\wr"] & \exists \langle q', \omega' \rangle \arrow[d, phantom, "\wr"] \\ \langle q, \boldsymbol{w} \rangle \arrow[r, "\text{Coll}"'] & \langle q', \boldsymbol{w'} \rangle \end{tikzcd}$$

Proof. We consider the case of **pop**-transitions:

$$\langle q, \boldsymbol{w}\langle \gamma_1, \Lambda_1 \rangle \langle \gamma_2, \Lambda_2 \rangle \rangle \xrightarrow{\mathsf{pop}} \langle q', \boldsymbol{w}\langle \gamma_1, \Lambda_1' \odot \Lambda_2 \rangle \rangle$$

for some Λ_1' such that $\Lambda_1 \preccurlyeq \Lambda_1'$ and $\Lambda_1' \mathbin{//} \Lambda_2$. All the other cases are not difficult. Using Lemma 3, Λ_1' is uniquely determined as $\Lambda_1 \lhd \Lambda_2$. From the assumption, for some valuations, we have the following:

$$(\star) \quad \langle q, \omega \langle \gamma_1, \mu_1 \rangle \langle \gamma_2, \mu_2 \rangle \rangle \sim \langle q, \boldsymbol{w}\langle \gamma_1, \Lambda_1 \rangle \langle \gamma_2, \Lambda_2 \rangle \rangle.$$

We show the following correspondence:

$$\omega \, \langle \gamma_1, (\mu_1 \lhd \mu_2) \odot \mu_2 \rangle \models \boldsymbol{w'}\langle \gamma_1, (\Lambda_1 \lhd \Lambda_2) \odot \Lambda_2 \rangle.$$

It suffices to confirm $\mathcal{C}(\mu_2) = \Lambda_2$ and $\mathcal{C}(\mu_1 \lhd \mu_2) = \Lambda_1 \lhd \Lambda_2$. The former is trivial from ($\star$). To show the latter equation, we use the following simple property:

$$\text{if } \mu_a \mathrel{/\!/} \mu_b, \text{ then } \mathcal{C}(\mu_a) \mathrel{/\!/} \mathcal{C}(\mu_b).$$

We use a real number δ such that $\mu_1 + \delta = \mu_1 \lhd \mu_2$. Since $\mathcal{C}(\mu_1 + \delta) = \Lambda_1 + \delta$, which comes from $\mathcal{C}(\mu_1) = \Lambda_1$, and $\mathcal{C}(\mu_2) = \Lambda_2$, we have $\Lambda_1 + \delta \mathrel{/\!/} \Lambda_2$ from the above property. This compatibility and Lemma 3 imply $\Lambda_1 + \delta = \Lambda_1 \lhd \Lambda_2$. Therefore, our goal is rewritten as $\mathcal{C}(\mu_1 + \delta) = \Lambda_1 + \delta$ and it is already shown. □

Remark: As we have seen in Section 6. 2, we need reference clocks to ensure the uniqueness of **pop**-transitions on the collapsed semantics. However, this does not answer the question why we need to introduce reference clocks at the lazy semantics rather than collapsed semantics. The answer of that question is that the forward simulation does not hold between the lazy semantics without reference clocks and the collapsed semantics. Let us consider the following **pop**-transition under $\mathsf{M} = 2$:

$$\frac{\Lambda_2 = \left\{\begin{array}{c} \dot{\mathtt{C}} \mapsto 1.5;\ \dot{x} \mapsto \infty.5; \\ \hdashline \underset{\bullet}{\mathtt{C}} \mapsto 1.5;\ \underset{\bullet}{x} \mapsto \infty.5 \end{array}\right\}}{\Lambda_1 = \left\{\begin{array}{c} \dot{\mathtt{C}} \mapsto 0.0;\ \dot{x} \mapsto \infty.0; \\ \hdashline \underset{\bullet}{\mathtt{C}} \mapsto 0.0;\ \underset{\bullet}{x} \mapsto 0.0 \end{array}\right\}} \xrightarrow{\textsf{pop}} \Lambda$$

where $\Lambda = (\Lambda_1 + \mathbf{1.5}) \odot \Lambda_2$ and thus it forms the following valuation:

$$\Lambda = \left\{\underset{\bullet}{\mathtt{C}} \mapsto 1.5;\ \underset{\bullet}{x} \mapsto 1.5;\ \dot{\mathtt{C}} \mapsto 1.5;\ \dot{x} \mapsto \infty.5\right\}.$$

If we do not introduce reference clocks at the lazy semantics, the following stack realizes the above stack $\Lambda_1\Lambda_2$:

$$\frac{\mu_2 = \left\{\underset{\bullet}{x} \mapsto 2.5;\ \dot{x} \mapsto 2.5\right\}}{\mu_1 = \left\{\dot{x} \mapsto 2.0;\ \underset{\bullet}{x} \mapsto 0.0\right\}} \models \frac{\Lambda_2}{\Lambda_1}.$$

On the lazy semantics without reference clocks, we have the following **pop**-transition:

$$\mu_1\mu_2 \xrightarrow{\textsf{pop}} (\mu_1 + \mathbf{0.5}) \odot \mu_2 = \left\{\underset{\bullet}{x} \mapsto 0.5;\ \dot{x} \mapsto 2.5\right\}.$$

For the obtained valuations, $(\mu_1 + \mathbf{0.5}) \odot \mu_2 \not\models \Lambda$ because $((\mu_1 + \mathbf{0.5}) \odot \mu_2)(\underset{\bullet}{x}) = 0.5$ and $\Lambda(\underset{\bullet}{x}) = 1.5$. To prevent this problem, we need reference clocks at the lazy semantics rather than at the collapsed semantics.

6. 4 Upper Bound for Configuration Reachability Problem

For an SRTA $\mathcal{A}$, we defined the upper bound constant M as follows:

$$\mathsf{M} \triangleq \max\{\ j\ :\ (i, j) \text{ or } [i, j] \text{ appears in interval constraints of } \mathcal{A}\ \} + 1.$$

This definition is enough to show the decidability of the location reachability problem but not enough to show the decidability of the configuration reachability problem. Let us consider the following SRTA:

$$\begin{tikzcd} \arrow[r] & q_0 \arrow[r, "x \in {[2,2]}"] & q_1 \arrow[r, "x \leftarrow {[0,0]}"] & q_2 \arrow[loop right, "\mathbf{tt}"] \end{tikzcd}$$

where $\mathbf{tt}$ is a tautology such as $\{x\} = \{x\}$. We have $\mathsf{M} = 3$ by the above definition.

Let us consider the following instance of the configuration reachability problem:

$$\langle q_0, \langle \bot, \{x \mapsto 0;\ y \mapsto 0\}\rangle\rangle \xrightarrow[\textsc{Stnd}]{}{}^{*}_{?} \langle q_2, \langle \bot, \{x \mapsto 10;\ y \mapsto 11\}\rangle\rangle.$$

It is clear that any configuration $\langle q_2, \langle \bot, \nu\rangle\rangle$ that is reachable from $\langle q_0, \langle \bot, \langle x \mapsto 0;\ y \mapsto 0\rangle\rangle\rangle$ satisfies $\nu(y) - \nu(x) \geq 2$. Therefore, we cannot reach $\langle q_2, \langle \bot, \{x \mapsto 10;\ y \mapsto 11\}\rangle\rangle$.

Collapsing the above instance, we obtain the following instance:

$$\langle q_0, \langle \bot, \{x \mapsto 0;\ y \mapsto 0\}\rangle\rangle \xrightarrow[\textsc{Coll}]{}{}^{*}_{?} \langle q_2, \langle \bot, \{x \mapsto \infty;\ y \mapsto \infty\}\rangle\rangle. \qquad (\star)$$

Unfortunately, even though we cannot reach the target configuration on the original instance, we can reach the collapsed target configuration as follows:

$$\begin{array}{ll} & \langle q_0,\ \langle \bot,\ \{x \mapsto 0;\ y \mapsto 0\}\rangle\rangle \\ \overset{2.0}{\rightsquigarrow} & \langle q_0,\ \langle \bot,\ \{x \mapsto 2;\ y \mapsto 2\}\rangle\rangle \\ \to & \langle q_1,\ \langle \bot,\ \{x \mapsto 2;\ y \mapsto 2\}\rangle\rangle \\ \overset{0.0}{\rightsquigarrow} & \langle q_1,\ \langle \bot,\ \{x \mapsto 2;\ y \mapsto 2\}\rangle\rangle \\ \to & \langle q_2,\ \langle \bot,\ \{x \mapsto 0;\ y \mapsto 2\}\rangle\rangle \\ \overset{10.0}{\rightsquigarrow} & \langle q_2,\ \langle \bot,\ \{x \mapsto \infty;\ y \mapsto \infty\}\rangle\rangle \end{array}$$

To overcome this problem, it suffices to take a sufficiently large upper bound with respect to a given instance of the configuration reachability problem. To formalize this, let us fix one instance $\langle q_0, \langle \bot, \mathbf{0}\rangle\rangle \xrightarrow[\textsc{Stnd}]{}{}^{*}_{?} \langle q, w\rangle$. For a stack $w = \langle \gamma_1, \nu_1\rangle\langle \gamma_2, \nu_2\rangle \ldots \langle \gamma_n, \nu_n\rangle \in (\Gamma \times (\mathcal{X} \to \mathbb{R}_+))^*$ on the standard semantics, we define the maximum value $\max(w)$ of the real numbers in w as follows:

$$\max(w) \triangleq \max\{\nu_i(x) : i \in [1..n], x \in \mathcal{X}\}.$$

We again take an upper-bound constant $\mathsf{M} \in \mathbb{N}$ so that it could satisfy the following:

$\mathsf{M} \geq \max\{\, j : (i, j) \text{ or } [i, j] \text{ appears in } \mathcal{A}\} + 1,$
$\mathsf{M} \geq \max(w) + 1.$

Under the new definition of upper-bound constant, we have the following adequate simulation properties for the configuration reachability problem. To formalize the statement, we define a stack correspondence between the standard and collapsed semantics in a natural way.

Definition 19. Let w and $\boldsymbol{w}$ be a stack of the standard and collapsed semantics, respectively. If there is a stack ω of the lazy semantics such that $w \models \omega$ and $\omega \models \boldsymbol{w}$, we write $w \models \boldsymbol{w}$. $\diamondsuit$

Lemma 6. *Let $\langle q, w\rangle$ be a configuration of the standard semantics. The following are equivalent:*
(1) $\langle q_{init}, \langle \bot, \mathbf{0}_{\mathcal{X}}\rangle\rangle \xrightarrow[\textsc{Stnd}]{}^{*} \langle q, w\rangle$.
(2) $\langle q_{init}, \langle \bot, \mathbf{0}_{\underset{\bullet}{\mathcal{X}}_{\mathtt{C}} \cup \dot{\mathcal{X}}_{\mathtt{C}}}\rangle\rangle \xrightarrow[\textsc{Coll}]{}^{*} \langle q, \boldsymbol{w}\rangle$ *for some stack $\boldsymbol{w}$ such that $\mathit{WF}(\boldsymbol{w})$ and $w \models \boldsymbol{w}$.*

Proof. The direction (1) $\Rightarrow$ (2) is shown by using the forward simulation Lemmas 1 and 4 sequentially.

We consider the other direction (2) $\Rightarrow$ (1). The assumption $w \models \boldsymbol{w}$ implies the existence of a stack ω such that $w \models \omega \models \boldsymbol{w}$.

First, the forward simulation of Coll by Lazy (Lemma 5) implies the following transition:

$$\langle q_{\text{init}}, \langle \bot, \mathbf{0}_{\underset{\bullet}{\mathcal{X}}_{\mathtt{C}} \cup \dot{\mathcal{X}}_{\mathtt{C}}}\rangle\rangle \xrightarrow[\textsc{Lazy}]{}^{*} \langle q, \omega'\rangle$$

where ω' is a well-formed stack such that $\omega' \models \boldsymbol{w}$.

Next, the forward simulation of Lazy by Stnd (Lemma 2) implies the following transition:

$$\langle q_{\text{init}}, \langle \bot, \mathbf{0}\rangle\rangle \xrightarrow[\textsc{Stnd}]{}^{*} \langle q, w'\rangle$$

where w' is a stack such that $w' \models \omega'$.

We need to show $w = w'$ from $w \models \omega \models \boldsymbol{w}$ and $w' \models \omega' \models \boldsymbol{w}$. However, since $\omega \neq \omega'$ in general, showing $w = w'$ is not trivial. Let us consider the following two stacks under $\mathsf{M} = 4$:

$$\omega = \cfrac{\mu_2 = \left\{\begin{array}{c} \dot{x} \mapsto 2.0;\ \dot{\mathtt{C}} \mapsto 4.5 \\ \hdashline \underset{\bullet}{x} \mapsto 3.5;\ \underset{\bullet}{\mathtt{C}} \mapsto 2.5 \end{array}\right\}}{\mu_1 = \left\{\begin{array}{c} \dot{x} \mapsto 1.0;\ \dot{\mathtt{C}} \mapsto 0.0 \\ \hdashline \underset{\bullet}{x} \mapsto 8.0;\ \underset{\bullet}{\mathtt{C}} \mapsto 4.0 \end{array}\right\}},$$

$$\omega' = \cfrac{\mu'_2 = \left\{\begin{array}{c} \dot{x} \mapsto 2.0;\ \dot{\mathtt{C}} \mapsto 6.5 \\ \hdashline \underset{\bullet}{x} \mapsto 3.5;\ \underset{\bullet}{\mathtt{C}} \mapsto 2.5 \end{array}\right\}}{\mu'_1 = \left\{\begin{array}{c} \dot{x} \mapsto 1.0;\ \dot{\mathtt{C}} \mapsto 0.0 \\ \hdashline \underset{\bullet}{x} \mapsto 6.0;\ \underset{\bullet}{\mathtt{C}} \mapsto 5.0 \end{array}\right\}}$$

where ω and ω' are different and realize the same stack $\mathcal{C}(\mu_1)\mathcal{C}(\mu_2)$ because $\mathcal{C}(\mu_1) = \mathcal{C}(\mu'_1)$ and $\mathcal{C}(\mu_2) = \mathcal{C}(\mu'_2)$. Although $\omega \neq \omega'$, by the definition of M, $\mu_1(\dot{x}) = \mu'_1(\dot{x})$ and $\mu_2(\underset{\bullet}{x}) = \mu'_2(\underset{\bullet}{x})$ and it implies $\mu_2 \ominus \mu_1 = \mu'_2 \ominus \mu'_1 = 2.5$. Therefore, Proposition 3 implies $w = w'$ for the unique stacks w and w' such that $w \models \omega$ and $w' \models \omega$:

$$w = w' = \cfrac{\{x \mapsto 2.0\}}{\{x \mapsto 3.5\}}.$$

We formalize the above argument in the general case to show $w = w'$:

1. If $\omega = \langle \gamma_1, \mu_1\rangle \dots \langle \gamma_n, \mu_n\rangle\langle \gamma_{n+1}, \mu_{n+1}\rangle$, then we can show the following from $w \models \omega$ and Proposition 3:
   - $\mu_1(\dot{x}) < \mathsf{M}$ for all $x \in \mathcal{X}$.
   - $\mu_i(x) < \mathsf{M}$ for all $i \in [2..(n+1)]$ and $x \in \underset{\bullet}{\mathcal{X}} \cup \dot{\mathcal{X}}$.
2. If $\omega' = \langle \gamma_1, \mu'_1\rangle \dots \langle \gamma_n, \mu'_n\rangle\langle \gamma_{n+1}, \mu'_{n+1}\rangle$, then we can show the following from $\omega \models \boldsymbol{w}$ and $\omega' \models \boldsymbol{w}$:
   - $\mu_1(\dot{x}) = \mu'_1(\dot{x}) < \mathsf{M}$ for all $x \in \mathcal{X}$.
   - $\mu_i(x) = \mu'_i(x) < \mathsf{M}$ for all $i \in [2..(n+1)]$ and $x \in \underset{\bullet}{\mathcal{X}} \cup \dot{\mathcal{X}}$.

   These imply $\mu_{i+1} \ominus \mu_i = \mu'_{i+1} \ominus \mu'_i$ for $i \in [1..n]$.
3. Assume $w = \langle \gamma_1, \nu_1\rangle \dots \langle \gamma_{n+1}, \nu_{n+1}\rangle$ and $w' = \langle \gamma_1, \nu'_1\rangle \dots \langle \gamma_{n+1}, \nu'_{n+1}\rangle$. Proposition 3 and $\mu_{i+1} \ominus \mu_i = \mu'_{i+1} \ominus \mu'_i$ for all $i \in [1..n]$ imply $\nu_i = \nu'_i$ for all $i \in [1..n]$. Thus, $w = w'$.

The fact $w = w'$ implies the following transition:

$$\langle q_{\text{init}}, \langle \bot, \mathbf{0}\rangle\rangle \xrightarrow[\textsc{Stnd}]{}^{*} \langle q, w\rangle.$$

$\square$

7 Digital Valuations and Finite-PDS Semantics

The collapsed semantics cannot be formalized as a finite PDS for the denseness of real numbers. We define digital valuations to remove the denseness and the digitized semantics Digi as a finite PDS.

7.1 Digital Valuations

Our definition of digital valuations equals to that of regions given by Abdulla et al. in [1].

Definition 20 (Digital Valuations). Let $\mathcal{X}$ be a

$$\begin{array}{ccccccc}
\{x \mapsto 0.0;\ y \mapsto 1.3\} & \preccurlyeq & \{x \mapsto 0.5;\ y \mapsto 1.8\} & \preccurlyeq & \{x \mapsto 0.7;\ y \mapsto \infty.0\} & \preccurlyeq & \{x \mapsto 0.9;\ y \mapsto \infty.2\} \\
\rotatebox{-90}{$\models$} & & \rotatebox{-90}{$\models$} & & \rotatebox{-90}{$\models$} & & \rotatebox{-90}{$\models$} \\
\{(x,0)\}_0\{(y,1)\} & \vdash & \{\}_0\,\{(x,0)\}\{(y,1)\} & \vdash & \{(y,\infty)\}_0\{(x,0)\} & \vdash & \{\}_0\,\{(y,\infty)\}\{(x,0)\}
\end{array}$$

Fig. 4 An example of the successor relation $\vdash$ with $\mathsf{M} = 2$

finite clock set. A sequence of sets $\boldsymbol{d} = d_0\, d_1 \ldots d_n$, where $d_i \subseteq \mathcal{X} \times \{0, 1, \ldots, \mathsf{M}-1, \infty\}$, is a *digital valuation* on $\mathcal{X}$ if $\boldsymbol{d}$ satisfies the following conditions:

- Every clock in $\mathcal{X}$ appears in $\boldsymbol{d}$ exactly once.
- Except d_0, all the sets d_i are not empty: $d_i \neq \emptyset$ for all $i \in [\mathbf{1}..n]$.

For a clock $x \in \mathcal{X}$ and set d_i of $\boldsymbol{d}$, we write $x \in d_i$ if $(x, v) \in d_i$ for some $v \in \{0, 1, \ldots, \mathsf{M}-1, \infty\}$. $\diamondsuit$

Intuitively, each digital valuation is obtained by forgetting the fractional parts of a collapsed valuation and only keeping the order of the fractional parts of it. On the basis of this intuition, we define a realization relation between collapsed and digital valuations.

Definition 21 (Realization)**.** Let $\mathcal{X}$ be a finite clock set, λ be a collapsed valuation on $\mathcal{X}$, and $\boldsymbol{d} = d_0 d_1 \ldots d_n$ be a digital valuation on $\mathcal{X}$. We write $\lambda \models \boldsymbol{d}$ if the following hold:

- For all $x \in \mathcal{X}$, $(x, \lfloor \lambda(x) \rfloor) \in d_i$ for some i.
- For all $x \in \mathcal{X}$, $\{\lambda(x)\} = 0.0$ iff $x \in d_0$.
- $\{\lambda(x)\} < \{\lambda(y)\}$ iff $x \in d_i$ and $y \in d_j$ for some $i < j$.

$\diamondsuit$

Proposition 8. *The realization relation $\models$ is* functional*: for a collapsed valuation λ, there exists the unique digital valuation $\mathcal{D}(\lambda)$ such that $\lambda \models \mathcal{D}(\lambda)$.*

For the collapsed valuation $\lambda = \{x \mapsto 2.0; y \mapsto 1.3\}$, the corresponding digital valuation is $\mathcal{D}(\lambda) = \{(x, 2)\}_0\, \{(y, 1)\}$. For another collapsed valuation $\lambda' = \{x \mapsto 0.8;\ y \mapsto 1.5\}$, the corresponding digital valuation is $\mathcal{D}(\lambda') = \{\}_0\, \{(y, 1)\}\, \{(x, 0)\}$.

For the special sets d_0 that contain clocks whose fractional parts are 0.0, we use the notation $\{\ldots\}_0$ as above.

We define the successor relation $\boldsymbol{d} \vdash \boldsymbol{d}'$ that corresponds to time elapsing on collapsed valuations.

Definition 22 (Successor)**.** Let $\boldsymbol{d}$ and $\boldsymbol{d}'$ be digital valuations. The valuation $\boldsymbol{d}'$ is the unique successor of $\boldsymbol{d}$ ($\boldsymbol{d} \vdash \boldsymbol{d}'$) if one of the following holds:

Case $\boldsymbol{d} = d_0 d_1 \ldots d_n$ and $d_0 \neq \emptyset$:

$$d_0\, d_1 \ldots d_n \vdash \emptyset\, d_0\, d_1 \ldots d_n.$$

Case $\boldsymbol{d} = \emptyset\, d_1 \ldots d_{n-1} d_n$:

$$\emptyset\, d_1 \ldots d_{n-1}\, d_n \vdash d'_n\, d_1 \ldots d_{n-1},$$

where d'_n satisfies the following: if $(x, k) \in d_n$,

$$\begin{cases} (x, k+1) \in d'_n & \text{if } k < \mathsf{M} - 1, \\ (x, \infty) \quad\ \in d'_n & \text{if } k = \mathsf{M} - 1 \text{ or } k = \infty. \end{cases}$$

We use $\vdash^*$ to denote the reflexive transitive closure of $\vdash$. $\diamondsuit$

Fig. 4 is an example of the successor relation $\vdash$.

On the realization relation between collapsed and digital valuations, the similar properties to Proposition 5 hold.

Proposition 9. *Let λ_1 and λ_2 be collapsed valuations on $\mathcal{X}$. If $\mathcal{D}(\lambda_1) = \mathcal{D}(\lambda_2)$,*

Validity. *$\lambda_1 \models \varphi$ iff $\lambda_2 \models \varphi$ for any constraint φ.*
Copying. *$\mathcal{D}(\lambda_1[x := y]) = \mathcal{D}(\lambda_2[x := y])$ for any $x, y \in \mathcal{X}$.*
Integer Updating. *$\mathcal{D}(\lambda_1[x := n]) = \mathcal{D}(\lambda_2[x := n])$ for any $x \in \mathcal{X}$ and $n \in \{0, 1, \ldots, \mathsf{M}-1\}$.*
Evolving. *If $\lambda_1 \preccurlyeq \lambda'_1$, then there exists λ'_2 such that $\lambda_2 \preccurlyeq \lambda'_2$ and $\mathcal{D}(\lambda'_1) = \mathcal{D}(\lambda'_2)$.*

On the basis of Proposition 9, we define basic operations on the digital valuations in the same way as that of the collapsed valuations.

Definition 23. Let $\boldsymbol{d}$ be a digital valuation and λ be a collapsed valuation that satisfies $\mathcal{D}(\lambda) = \boldsymbol{d}$.

- For a constraint φ, $\boldsymbol{d} \models \varphi$ if $\lambda \models \varphi$.
- For two clocks x, y, $\boldsymbol{d}[x := y] \triangleq \mathcal{D}(\lambda[x := y])$.
- For a natural number n such that $0 \leq n < \mathsf{M}$, $\boldsymbol{d}[x := n] \triangleq \mathcal{D}(\lambda[x := n])$.

$\diamondsuit$

In order to treat updating $x \leftarrow I$ for a clock x and interval I, we need to define nondeterministic updating $\boldsymbol{d}[x \leftarrow I]$ for a digital valuation $\boldsymbol{d}$.

Definition 24 (Nondeterministic Update)**.** Let $\boldsymbol{d}$ be a digital valuation. We define the *update*

$\boldsymbol{d}[x \leftarrow I]$ for a clock x and an interval I as follows:

$$\boldsymbol{d}[x \leftarrow I] \triangleq \{\, \mathcal{D}(\,\lambda[x := r]\,) : r \in I, \lambda \models \boldsymbol{d} \,\}\,.$$

$\diamondsuit$

Example. Let $\boldsymbol{d} = \{(x,0)\}_0\,\{(y,1)\}$ be a digital valuation. The updating $\boldsymbol{d}[x \leftarrow (2,3)]$ generates the following three digital valuations:

$$\boldsymbol{d}[\,x \leftarrow (2,3)\,] = \{\boldsymbol{d}_1, \boldsymbol{d}_2, \boldsymbol{d}_3\}\,,$$

$$\begin{aligned} \boldsymbol{d}_1 &= \{\}_0\,\{(x,2)\}\,\{(y,1)\}\,, \\ \boldsymbol{d}_2 &= \{\}_0\,\{(x,2),(y,1)\}\,, \\ \boldsymbol{d}_3 &= \{\}_0\,\{(y,1)\}\,\{(x,2)\}\,. \end{aligned}$$

7.2 Forward and Backward Simulation of Time Elapsing

For time elapsing, we show two forward simulations and one backward simulation below.

Proposition 10 (♣). *Let λ be a collapsed valuation and $\boldsymbol{d}$ be a digital valuation.*

$$\begin{array}{ccc} \lambda & \preccurlyeq & \lambda' \\ \rotatebox{-90}{$\models$} & & \\ \boldsymbol{d} & & \end{array} \quad \Longrightarrow \quad \begin{array}{ccc} \lambda & \preccurlyeq & \lambda' \\ \rotatebox{-90}{$\models$} & & \rotatebox{-90}{$\models$} \\ \boldsymbol{d} & \vdash^* & \exists \boldsymbol{d}'. \end{array}$$

Sketch. Since this proposition is formalized and verified by Coq, we only sketch the idea of our formalized proof.

Let us consider the following example:

$$\begin{aligned} \lambda &= \{x \mapsto 0.2;\ y \mapsto 1.6;\ z \mapsto 2.9\} \preccurlyeq \\ \lambda' &= \{x \mapsto 1.0;\ y \mapsto 2.4;\ z \mapsto 3.7\}\,, \end{aligned}$$

where $\mathcal{D}(\lambda) = \{\}_0\,\{(x,0)\}\,\{(y,1)\}\,\{(z,2)\}$. We introduce the two auxiliary time elapsing relation $\preccurlyeq_1$ and $\preccurlyeq_2$ that correspond to the two cases of the successor relation of digital valuations:

- $\lambda \preccurlyeq_1 \lambda'$ holds if the following holds:
  - There is no clock x such that $\{\lambda(x)\} = 0.0$;
  - $\lambda' = \lambda + \delta$ where $\delta = 1.0 - \{\lambda(y)\}$ and y is one of the clocks that have the maximal fractional parts for λ.

  For example,

  $$\begin{aligned} &\{x \mapsto 0.2;\ y \mapsto 1.6;\ z \mapsto \underline{2.9}\} \preccurlyeq_1 \\ &\{x \mapsto 0.3;\ y \mapsto 1.7;\ z \mapsto 3.0\}\,. \end{aligned}$$

- $\lambda \preccurlyeq_2 \lambda'$ holds if the following holds:
  - There is a clock x such that $\{\lambda(x)\} = 0.0$;
  - $\lambda' = \lambda + \delta$ where $\delta \lneq 1.0 - \{\lambda(y)\}$ and y is one of the clocks that have the maximal fractional parts for λ.

  For example,

  $$\begin{aligned} &\{x \mapsto 0.3;\ y \mapsto \underline{1.7};\ z \mapsto 3.0\} \preccurlyeq_2 \\ &\{x \mapsto 0.4;\ y \mapsto 1.8;\ z \mapsto 3.1\}\,. \end{aligned}$$

These two elapsing relations have a fine property as follows: if $\lambda \models \boldsymbol{d}$ and either $\lambda \preccurlyeq_1 \lambda'$ or $\lambda \preccurlyeq_2 \lambda'$, then the successor $\boldsymbol{d}'$ of $\boldsymbol{d}$ ($\boldsymbol{d} \vdash \boldsymbol{d}'$) satisfies $\lambda' \models \boldsymbol{d}'$.

By using these two elapsing relations, we can decompose $\lambda \preccurlyeq \lambda'$ as follows:

$$\begin{aligned} \lambda = \lambda_0 &= \{x \mapsto 0.2;\ y \mapsto 1.6;\ z \mapsto 2.9\} \preccurlyeq_1 \\ \lambda_1 &= \{x \mapsto 0.3;\ y \mapsto 1.7;\ z \mapsto 3.0\} \preccurlyeq_2 \\ \lambda_2 &= \{x \mapsto 0.4;\ y \mapsto 1.8;\ z \mapsto 3.1\} \preccurlyeq_1 \\ \lambda_3 &= \{x \mapsto 0.6;\ y \mapsto 2.0;\ z \mapsto 3.3\} \preccurlyeq_2 \\ \lambda_4 &= \{x \mapsto 0.9;\ y \mapsto 2.3;\ z \mapsto 3.6\} \preccurlyeq_1 \\ \lambda' = \lambda_5 &= \{x \mapsto 1.0;\ y \mapsto 2.4;\ z \mapsto 3.7\}\,. \end{aligned}$$

Combining this decomposition and the above property, we conclude the proof. $\square$

Proposition 11 (♣). *Let λ be a collapsed valuation and $\boldsymbol{d}$ be a digital valuation.*

$$\begin{array}{ccc} \lambda & & \\ \rotatebox{-90}{$\models$} & & \\ \boldsymbol{d} & \vdash^* & \boldsymbol{d}' \end{array} \quad \Longrightarrow \quad \begin{array}{ccc} \lambda & \preccurlyeq & \exists\lambda' \\ \rotatebox{-90}{$\models$} & & \rotatebox{-90}{$\models$} \\ \boldsymbol{d} & \vdash^* & \boldsymbol{d}'. \end{array}$$

Sketch. It suffices to show the following 1-step diagram holds:

$$\begin{array}{ccc} \lambda & & \\ \rotatebox{-90}{$\models$} & & \\ \boldsymbol{d} & \vdash & \boldsymbol{d}' \end{array} \quad \Longrightarrow \quad \begin{array}{ccc} \lambda & \preccurlyeq & \exists\lambda' \\ \rotatebox{-90}{$\models$} & & \rotatebox{-90}{$\models$} \\ \boldsymbol{d} & \vdash & \boldsymbol{d}'. \end{array}$$

In order to show this diagram, we do case analysis on $\boldsymbol{d} \vdash \boldsymbol{d}'$.

If $\boldsymbol{d} = \{\cdots\}_0\, d_1 \ldots d_n \vdash \boldsymbol{d}' = \{\}_0\ d_0\, d_1 \ldots d_n$
Since this corresponds to $\preccurlyeq_2$, we take λ' such that $\lambda \preccurlyeq_2 \lambda'$. It is clear that $\lambda' \models \boldsymbol{d}'$.

If $\boldsymbol{d} = \{\}_0\, d_1 \ldots d_{n-1} d_n \vdash \boldsymbol{d}' = \{\cdots\}_0\, d_1 \ldots d_{n-1}$
Since this corresponds to $\preccurlyeq_1$, we take λ' such that $\lambda \preccurlyeq_1 \lambda'$. It is clear that $\lambda' \models \boldsymbol{d}'$.

$\square$

Proposition 12 (♣). *Let λ' be a collapsed valuation and $\boldsymbol{d}$ be a digital valuation.*

$$\begin{array}{ccc} & & \lambda' \\ & & \rotatebox{-90}{$\models$} \\ \boldsymbol{d} & \vdash^* & \boldsymbol{d}' \end{array} \quad \Longrightarrow \quad \begin{array}{ccc} \exists\lambda & \preccurlyeq & \lambda' \\ \rotatebox{-90}{$\models$} & & \rotatebox{-90}{$\models$} \\ \boldsymbol{d} & \vdash^* & \boldsymbol{d}'. \end{array}$$

Sketch. This proposition is shown by the similar argument to Proposition 11. $\square$

Remark. Proposition 12 is crucial to the backward simulation lemma (Lemma 8) and peculiar to collapsed valuations. Indeed, this fails on the realization relation $\nu \models \boldsymbol{d}$ of concrete and digital valuations. Let us consider the following under $\mathsf{M} = 2$:

$$\begin{array}{ccc} & & \{x \mapsto 2.0\} \\ & & \rotatebox{90}{$\models$} \\ \{\}_0\,\{x \mapsto \infty\} & \xrightarrow[\textsc{Digi}]{} & \{x \mapsto \infty\}_0. \end{array}$$

For the backward simulation, we need to find a valuation ν such that $\nu < \{x \mapsto 2.0\}$ and $\nu \models \{\}_0\,\{x \mapsto \infty\}$. However, since $\nu < \{x \mapsto 2.0\}$ requires $\nu(x) < 2.0$, $\nu \models \{\}_0\,\{x \mapsto \infty\}$ never holds.

7.3 Nondeterministic Composition

As the collapsed semantics Coll, we define the compatibility and composition.

Definition 25 (Compatibility and Composition)**.** Let $\boldsymbol{D}_1$ and $\boldsymbol{D}_2$ be digital valuations on a finite marked clock set $\underset{\bullet}{\mathcal{X}}_{\mathsf{C}} \cup \dot{\mathcal{X}}_{\mathsf{C}}$.

- The valuation $\boldsymbol{D}_1$ is *compatible* with the valuation $\boldsymbol{D}_2$ ($\boldsymbol{D}_1 \mathbin{/\!/} \boldsymbol{D}_2$) if there are collapsed valuations Λ_1 and Λ_2 such that $\Lambda_1 \models \boldsymbol{D}_1$, $\Lambda_2 \models \boldsymbol{D}_2$, and $\Lambda_1 \mathbin{/\!/} \Lambda_2$.
- The composed valuations are defined by:

$$\boldsymbol{D}_1 \odot \boldsymbol{D}_2 \triangleq \left\{ \mathcal{D}(\Lambda) : \begin{array}{l} \Lambda_1 \models \boldsymbol{D}_1,\ \Lambda_2 \models \boldsymbol{D}_2, \\ \Lambda_1 \mathbin{/\!/} \Lambda_2,\ \Lambda = \Lambda_1 \odot \Lambda_2 \end{array} \right\}.$$

$\diamondsuit$

Nondeterminacy of ***pop*** *transitions.* Let us consider the following **pop** transition under $\mathsf{M} = 3$:

$$\frac{\boldsymbol{D}_2 = \left\{(\dot{\mathsf{C}}, 2), (\underset{\bullet}{\mathsf{C}}, 2), (\underset{\bullet}{x}, \infty)\right\}_0 \{(\dot{x}, \infty)\}}{\boldsymbol{D}_1 = \left\{(\underset{\bullet}{\mathsf{C}}, 0), (\dot{\mathsf{C}}, 0), (\dot{x}, 2)\right\}_0 \left\{(\underset{\bullet}{x}, 0)\right\}} \xrightarrow{\textsf{pop}} \cdots.$$

When performing the **pop** transition, first we compute $\boldsymbol{D}_1'$ such that $\boldsymbol{D}_1 \vdash^* \boldsymbol{D}_1'$ and $\boldsymbol{D}_1' \mathbin{/\!/} \boldsymbol{D}_2$. For this case, $\boldsymbol{D}_1'$ is uniquely determined as follows:

$$\boldsymbol{D}_1' = \{(\underset{\bullet}{\mathsf{C}}, 2), (\dot{\mathsf{C}}, 2), (\dot{x}, \infty)\}_0\{(\underset{\bullet}{x}, 2)\}.$$

Next we compute the new top frame by composing the two digital valuations $\boldsymbol{D}_1'$ and $\boldsymbol{D}_2$ as $\boldsymbol{D}_1' \odot \boldsymbol{D}_2$:

$$\begin{array}{ll} & \dfrac{\boldsymbol{D}_2 = \left\{(\dot{\mathsf{C}}, 2), (\underset{\bullet}{\mathsf{C}}, 2), (\underset{\bullet}{x}, \infty)\right\}_0 \{(\dot{x}, \infty)\}}{\boldsymbol{D}_1' = \{(\underset{\bullet}{\mathsf{C}}, 2), (\dot{\mathsf{C}}, 2), (\dot{x}, \infty)\}_0\{(\underset{\bullet}{x}, 2)\}} \\ \xrightarrow{\odot} & \{\boldsymbol{D}_3, \boldsymbol{D}_3', \boldsymbol{D}_3''\} \end{array}$$

where

$$\begin{array}{rcl} \boldsymbol{D}_3 & = & \left\{(\underset{\bullet}{\mathsf{C}}, 2), (\dot{\mathsf{C}}, 2)\right\}_0 \{(\underset{\bullet}{x}, 2)\}\{(\dot{x}, \infty)\}, \\ \boldsymbol{D}_3' & = & \left\{(\underset{\bullet}{\mathsf{C}}, 2), (\dot{\mathsf{C}}, 2)\right\}_0 \{(\underset{\bullet}{x}, 2), (\dot{x}, \infty)\}, \\ \boldsymbol{D}_3'' & = & \left\{(\underset{\bullet}{\mathsf{C}}, 2), (\dot{\mathsf{C}}, 2)\right\}_0 \{(\dot{x}, \infty)\}\{(\underset{\bullet}{x}, 2)\}. \end{array}$$

When composing the two digital valuations, we need to decide the order between $\underset{\bullet}{x}$ of $\boldsymbol{D}_1'$ and $\dot{x}$ of $\boldsymbol{D}_2$. However, since we dismiss the fractional values in digital valuations, there are no clues to decide the ordering, and so we generate all the possible orderings as above.

As we have seen in the previous section, pop transitions for well-formed stacks behave deterministically on the collapsed semantics. Hence, the collapsed semantics cannot capture the nondeterminacy of pop transitions of the digitized semantics.

7.4 Digital Semantics

Digital valuations lead to the digitized semantics Digi, which is defined as a finite PDS.

Definition 26 (Digitized Semantics Digi)**.** Let $\mathcal{A} = (Q, q_{\mathrm{init}}, F, \Sigma, \Gamma, \mathcal{X}, \Delta)$ be an SRTA.

We use $\mathbb{D}$ for the finite set of digital valuations on $\underset{\bullet}{\mathcal{X}}_{\mathsf{C}} \cup \dot{\mathcal{X}}_{\mathsf{C}}$.

We define the finite PDS $(Q, \Gamma \times \mathbb{D}, \hookrightarrow_d \cup \hookrightarrow_t)$ where discrete transition rules $\hookrightarrow_d$ and time elapsing transition rules $\hookrightarrow_t$ are defined as follows:

- A discrete transition rule $\langle q, \boldsymbol{W}\rangle \hookrightarrow_d \langle q', \boldsymbol{W'}\rangle$ is defined if there is $q \xrightarrow{\alpha}_{\tau} q' \in \Delta$ and $\boldsymbol{W} \overset{\tau}{\hookrightarrow} \boldsymbol{W'}$ is defined by following Fig. 5.
- Time elapsing transition rules $\langle q, \langle \gamma, \boldsymbol{D}\rangle\rangle \hookrightarrow_t \langle q, \langle \gamma, \boldsymbol{D'}\rangle\rangle$ are defined for all $q \in Q$ and $\gamma \in \Gamma$ if $\boldsymbol{D} \vdash^* \boldsymbol{D'}$ holds.

$\diamondsuit$

In the same way as the lazy and collapsed semantics, we define the stack well-formedness for the digitized semantics.

Definition 27. Let $\boldsymbol{D}_1$ and $\boldsymbol{D}_2$ be digital valuations on a finite marked clock set $\underset{\bullet}{\mathcal{X}}_{\mathsf{C}} \cup \dot{\mathcal{X}}_{\mathsf{C}}$.

If there is a digital valuation $\boldsymbol{D}_1'$ such that $\boldsymbol{D}_1 \vdash^* \boldsymbol{D}_1'$ and $\boldsymbol{D}_1' \mathbin{/\!/} \boldsymbol{D}_2$, then we write $\boldsymbol{D}_1 \precapprox \boldsymbol{D}_2$. $\diamondsuit$

Definition 28 (Well-formed Stack)**.** A stack $\boldsymbol{W} = \langle \gamma_1, \boldsymbol{D}_1\rangle\langle \gamma_2, \boldsymbol{D}_2\rangle \ldots \langle \gamma_n, \boldsymbol{D}_n\rangle$ is *well-formed* $\mathsf{WF}(\boldsymbol{W})$ if for all $i \in [1..(n-1)]$

- $\boldsymbol{D}_i \models \dot{\mathsf{C}} \in [0, 0]$ holds for the clock $\dot{\mathsf{C}}$; and
- $\boldsymbol{D}_i \precapprox \boldsymbol{D}_{i+1}$.

$\diamondsuit$

The well-formedness of the digitized semantics forms an invariant.

Proposition 13. *Let $\boldsymbol{W}$ be a well-formed stack $\mathsf{WF}(\boldsymbol{W})$. If $\langle q, \boldsymbol{W}\rangle \to \langle q', \boldsymbol{W'}\rangle$, then $\mathsf{WF}(\boldsymbol{W'})$.*

$$\frac{\boldsymbol{D}_1[\dot{\mathtt{C}} := 0] \mathbin{//} \boldsymbol{D}_2 \quad \boldsymbol{D}_2 \models \dot{x} \in [0,0] \ (\forall x \in \mathcal{X}_{\mathtt{C}})}{\langle \gamma_1, \boldsymbol{D}_1 \rangle \hookrightarrow \langle \gamma_1, \boldsymbol{D}_1[\dot{\mathtt{C}} := 0] \rangle \, \langle \gamma_2, \boldsymbol{D}_2 \rangle} \ \mathsf{push}(\gamma_2)$$

$$\frac{\boldsymbol{D}_1 \vdash^* \boldsymbol{D}_1' \quad \boldsymbol{D}_1' \mathbin{//} \boldsymbol{D}_2 \quad \boldsymbol{D} \in \boldsymbol{D}_1' \odot \boldsymbol{D}_2}{\langle \gamma_1, \boldsymbol{D}_1 \rangle \, \langle \gamma_2, \boldsymbol{D}_2 \rangle \hookrightarrow \langle \gamma_1, \boldsymbol{D} \rangle} \ \mathsf{pop}(\gamma_2) \qquad \frac{\boldsymbol{D} \models \dot{\varphi}}{\langle \gamma, \boldsymbol{D} \rangle \hookrightarrow \langle \gamma, \boldsymbol{D} \rangle} \ \mathsf{check}(\varphi)$$

$$\frac{\boldsymbol{D}_2' = \boldsymbol{D}_2[\dot{x} := \underset{\bullet}{y}]}{\langle \gamma_1, \boldsymbol{D}_1 \rangle \langle \gamma_2, \boldsymbol{D}_2 \rangle \hookrightarrow \langle \gamma_1, \boldsymbol{D}_1 \rangle \langle \gamma_2, \boldsymbol{D}_2' \rangle} \ \mathsf{dig}(x,y) \qquad \frac{\boldsymbol{D}' \in \boldsymbol{D}[\dot{x} \leftarrow I]}{\langle \gamma, \boldsymbol{D} \rangle \hookrightarrow \langle \gamma, \boldsymbol{D}' \rangle} \ x \leftarrow I$$

Fig. 5 Definition of actions on Digi Semantics

Proof. This is shown in the same argument of the proof in Proposition 2. □

In order to show the forward and backward simulation properties between the collapsed and digitized semantics, we define stack and configuration correspondences.

Definition 29. Let $\boldsymbol{w} = \Lambda_1, \ldots, \Lambda_n$ and $\boldsymbol{W} = \boldsymbol{D}_1, \ldots, \boldsymbol{D}_n$ be well-formed stacks $\mathsf{WF}(\boldsymbol{w})$ and $\mathsf{WF}(\boldsymbol{W})$ of the collapsed and digitized semantics, respectively.

The *stack correspondence* $\boldsymbol{w} \models \boldsymbol{W}$ holds if $\Lambda_i \models \boldsymbol{D}_i$ for all $i \in [1..n]$.

It is naturally extended to configurations: we write $\langle q, \boldsymbol{w} \rangle \sim \langle q', \boldsymbol{W} \rangle$ if $q = q'$ and $\boldsymbol{w} \models \boldsymbol{W}$. ♢

The following forward simulation of Coll by Digi can be shown easily.

Lemma 7. *Let $\langle q, \boldsymbol{w} \rangle$ and $\langle q, \boldsymbol{W} \rangle$ be configurations such that $\boldsymbol{w} \models \boldsymbol{W}$.*

$$\begin{tikzcd}
\langle q, \boldsymbol{w} \rangle \arrow[r, "\textsc{Coll}"'] \arrow[d, phantom, "\wr"] & \langle q', \boldsymbol{w}' \rangle & & \langle q, \boldsymbol{w} \rangle \arrow[r, "\textsc{Coll}"'] \arrow[d, phantom, "\wr"] & \langle q', \boldsymbol{w}' \rangle \arrow[d, phantom, "\wr"] \\
\langle q, \boldsymbol{W} \rangle & & \Longrightarrow & \langle q, \boldsymbol{W} \rangle \arrow[r, "\textsc{Digi}"'] & \exists \langle q', \boldsymbol{W}' \rangle.
\end{tikzcd}$$

Proof. We consider the following nontrivial case:

$$\begin{aligned} & \langle q\ , \boldsymbol{w} \langle \gamma_1, \Lambda_1 \rangle \langle \gamma_2, \Lambda_2 \rangle \langle \gamma_3, \Lambda_3 \rangle \rangle \\ \xrightarrow{\mathsf{pop}(\gamma_3)} \ & \langle q', \boldsymbol{w} \langle \gamma_1, \Lambda_1 \rangle \langle \gamma_2, (\Lambda_2 \lhd \Lambda_3) \odot \Lambda_3 \rangle \rangle. \end{aligned}$$

Our assumption can be rewritten as follows:

$$\begin{gathered} \langle q\ , \boldsymbol{w} \langle \gamma_1, \Lambda_1 \rangle \langle \gamma_2, \Lambda_2 \rangle \langle \gamma_3, \Lambda_3 \rangle \rangle \\ \wr \\ \langle q\ , \boldsymbol{W} \langle \gamma_1, \boldsymbol{D}_1 \rangle \langle \gamma_2, \boldsymbol{D}_2 \rangle \langle \gamma_3, \boldsymbol{D}_3 \rangle \rangle. \end{gathered}$$

Since $\Lambda_2 \preccurlyeq (\Lambda_2 \lhd \Lambda_3)$, we apply Proposition 10 to $\Lambda_2 \models \boldsymbol{D}_2$ and obtain $\boldsymbol{D}_2'$ such that $\boldsymbol{D}_2 \vdash^* \boldsymbol{D}_2'$ and $(\Lambda_2 \lhd \Lambda_3) \models \boldsymbol{D}_2'$. It is clear that $\boldsymbol{D}_2' \mathbin{//} \boldsymbol{D}_3$ from the definition of $//$.

Since $\Lambda_2 \lhd \Lambda_3 \models \boldsymbol{D}_2'$ and $\Lambda_3 \models \boldsymbol{D}_3$, the definition of $\odot$ implies that there is a digital valuation $\boldsymbol{D}$ such that $\boldsymbol{D} \in \boldsymbol{D}_2' \odot \boldsymbol{D}_3$ and $(\Lambda_2 \lhd \Lambda_3) \odot \Lambda_3 \models \boldsymbol{D}$. This leads to the following transition:

$$\begin{aligned} & \langle q\ , \boldsymbol{W} \langle \gamma_1, \boldsymbol{D}_1 \rangle \langle \gamma_2, \boldsymbol{D}_2 \rangle \langle \gamma_3, \boldsymbol{D}_3 \rangle \rangle \\ \xrightarrow{\mathsf{pop}(\gamma_3)} \ & \langle q', \boldsymbol{W} \langle \gamma_1, \boldsymbol{D}_1 \rangle \langle \gamma_2, \boldsymbol{D} \rangle \rangle. \end{aligned}$$

Now $(\Lambda_2 \lhd \Lambda_3) \odot \Lambda_3 \models \boldsymbol{D}$ concludes the proof. □

7.5 Backward Simulation

Compared to the forward simulation of Coll by Digi, the counterpart does not hold for the nondeterminacy of **pop** rules in Digi as we have stated in Section 7.3. However, we can show the backward simulation by Proposition 12.

Lemma 8. *Let $\langle q', \boldsymbol{w}' \rangle$ and $\langle q', \boldsymbol{W}' \rangle$ be configurations such that $\boldsymbol{w}' \models \boldsymbol{W}'$.*

$$\begin{tikzcd}
 & \langle q', \boldsymbol{w}' \rangle \arrow[d, phantom, "\wr"] & & \exists \langle q, \boldsymbol{w} \rangle \arrow[r, "\textsc{Coll}"'] \arrow[d, phantom, "\wr"] & \langle q', \boldsymbol{w}' \rangle \arrow[d, phantom, "\wr"] \\
\langle q, \boldsymbol{W} \rangle \arrow[r, "\textsc{Digi}"'] & \langle q', \boldsymbol{W}' \rangle & \Longrightarrow & \langle q, \boldsymbol{W} \rangle \arrow[r, "\textsc{Digi}"'] & \langle q', \boldsymbol{W}' \rangle.
\end{tikzcd}$$

Proof. We consider the two cases **push** and **time**. The cases **dig**, $x \leftarrow I$, and $\mathsf{check}(\varphi)$ are trivial and the case **pop** is shown in the same argument of the proof of the case **time**.

Proof of Case push

We consider the following **push** transition:

$$\begin{aligned} & \langle q\ , \boldsymbol{W} \langle \gamma_1, \boldsymbol{D}_1 \rangle \rangle \\ \xrightarrow{\mathsf{push}(\gamma_2)} \ & \langle q', \boldsymbol{W} \langle \gamma_1, \boldsymbol{D}_1[\dot{\mathtt{C}} := 0] \rangle \langle \gamma_2, \boldsymbol{D}_2 \rangle \rangle. \end{aligned}$$

Our assumption is rewritten as follows:

$$\begin{gathered} \langle q', \boldsymbol{w} \langle \gamma_1, \Lambda_1' \rangle \langle \gamma_2, \Lambda_2 \rangle \rangle \\ \wr \\ \langle q', \boldsymbol{W} \langle \gamma_1, \boldsymbol{D}_1[\dot{\mathtt{C}} := 0] \rangle \langle \gamma_2, \boldsymbol{D}_2 \rangle \rangle \end{gathered}$$

for some collapsed valuations Λ_1' and Λ_2.

We define $\Lambda_1 : (\dot{\mathcal{X}}_{\mathtt{C}} \cup \underset{\bullet}{\mathcal{X}}_{\mathtt{C}}) \to \mathbb{C}$ so that it satisfies both the following condition:

1. $\Lambda_1(x) = \Lambda_1'(x)$ for all $x \in \dot{\mathcal{X}} \cup \underset{\bullet}{\mathcal{X}}_{\mathtt{C}}$.

2. $\Lambda_1(\dot{\mathtt{C}}) = r$ where $r \in \mathbb{R}_+$ is a non-negative real number that satisfies $\Lambda_1 \models \boldsymbol{D}_1$.

We show the following:

(A) $\boldsymbol{w}\langle\gamma_1, \Lambda_1\rangle \models \boldsymbol{W}\langle\gamma_1, \boldsymbol{D}_1\rangle$.

(B) $\mathsf{WF}(\boldsymbol{w}\langle\gamma_1, \Lambda_1\rangle)$.

(C) $\langle q, \boldsymbol{w}\langle\gamma_1, \Lambda_1\rangle\rangle \xrightarrow{\mathsf{push}(\gamma_2)} \langle q', \boldsymbol{w}\langle\gamma_1, \Lambda_1'\rangle\langle\gamma_2, \Lambda_2\rangle\rangle$.

(A) is immediate from the definition of Λ_1. (B) is shown by the assumption $\mathsf{WF}(\boldsymbol{w}\langle\gamma_1, \Lambda_1'\rangle\langle\gamma_2, \Lambda_2\rangle)$ and the fact that $\Lambda_1 \restriction \underset{\cdot}{\mathcal{X}}_{\mathtt{C}} = \Lambda_1' \restriction \underset{\cdot}{\mathcal{X}}_{\mathtt{C}}$.

For (C), we show the following:

- $\Lambda_1'(\dot{x}) = \Lambda_2(\underset{\cdot}{x})$ for any $x \in \mathcal{X}_{\mathtt{C}}$.

We split this into the two parts: showing $\lfloor\Lambda_1'(\dot{x})\rfloor = \lfloor\Lambda_2(\underset{\cdot}{x})\rfloor$ and $\{\Lambda_1'(\dot{x})\} = \{\Lambda_2(\underset{\cdot}{x})\}$.

The former $\lfloor\Lambda_1'(\dot{x})\rfloor = \lfloor\Lambda_2(\underset{\cdot}{x})\rfloor$ comes from $\Lambda_1' \models \boldsymbol{D}_1[\dot{\mathtt{C}} := 0]$, $\Lambda_2 \models \boldsymbol{D}_2$, and $\boldsymbol{D}_1[\dot{\mathtt{C}} := 0] \mathbin{/\!/} \boldsymbol{D}_2$.

The latter $\{\Lambda_1'(\dot{x})\} = \{\Lambda_2(\underset{\cdot}{x})\}$ is somewhat involved and shown by the following steps:

1. $\Lambda_1' \precapprox \Lambda_2$ holds from $\mathsf{WF}(\boldsymbol{w}\langle\gamma_1, \Lambda_1'\rangle\langle\gamma_2, \Lambda_2\rangle)$.
2. $\{\Lambda_1'(\dot{\mathtt{C}})\} = 0.0$ holds from $\Lambda_1' \models \boldsymbol{D}_1[\dot{\mathtt{C}} := 0]$.
3. $\{\Lambda_2(\underset{\cdot}{\mathtt{C}})\} = 0.0$ holds from $\boldsymbol{D}_1[\dot{\mathtt{C}} := 0] \mathbin{/\!/} \boldsymbol{D}_2$ and $\Lambda_2 \models \boldsymbol{D}_2$.

Finally, combining $\Lambda_1' \precapprox \Lambda_2$ and $\{\Lambda_1'(\dot{\mathtt{C}})\} = \{\Lambda_2'(\underset{\cdot}{\mathtt{C}})\} = 0.0$, we have $\{\Lambda_1'(\dot{x})\} = \{\Lambda_2(\underset{\cdot}{x})\}$ for all clock $x \in \underset{\cdot}{\mathcal{X}}_{\mathtt{C}} \cup \dot{\mathcal{X}}_{\mathtt{C}}$. □

We remark that the above proof needs reference clocks; indeed, without reference clocks, the case **push** fails. Let us consider the following diagram:

$$\begin{array}{ccc} & & \dfrac{\{\dot{x} \mapsto 0;\ \underset{\cdot}{x} \mapsto 0.5\}}{\{\underset{\cdot}{x} \mapsto 0;\ \dot{x} \mapsto 0.4\}} \\ & & \top\!\!\top \\ \{(\underset{\cdot}{x}, 0)\}_0 \{(\dot{x}, 0)\} & \xrightarrow[\textsc{Digi}]{\mathsf{push}} & \dfrac{\{(\dot{x}, 0)\}_0 \{(\underset{\cdot}{x}, 0)\}}{\{(\underset{\cdot}{x}, 0)\}_0 \{(\dot{x}, 0)\}} \end{array}$$

Although the above diagram satisfies all the conditions of Lemma 8, the following transition is not allowed on the collapsed semantics without reference clocks:

$$\{\underset{\cdot}{x} \mapsto 0;\ \dot{x} \mapsto 0.4\} \xrightarrow[\textsc{Coll}]{\mathsf{push}}\!\!\!\!\!\!\!/\ \ \dfrac{\{\dot{x} \mapsto 0;\ \underset{\cdot}{x} \mapsto 0.5\}}{\{\underset{\cdot}{x} \mapsto 0;\ \dot{x} \mapsto 0.4\}}$$

Compared to this, the following stack is not a well-formed stack and we do not need to consider it:

$$\dfrac{\{\ldots;\ \underset{\cdot}{\mathtt{C}} \mapsto 0.0;\ \underset{\cdot}{x} \mapsto 0.5\}}{\{\ldots;\ \dot{\mathtt{C}} \mapsto 0.0;\ \dot{x} \mapsto 0.4\}}$$

Proof of Case time

Before giving the proof for the case **time**, we state a technical lemma that is key to the case **time**.

Lemma 9 (♣ Key Lemma for Backward Simulation). *Let λ_1, λ_2, and λ_3 be collapsed valuations on a finite clock set $\mathcal{X}$. Also, let $\boldsymbol{d}_1$, $\boldsymbol{d}_2$, and $\boldsymbol{d}_3$ be digital valuations on the set $\mathcal{X}$.*

$$\begin{array}{lccccc} & \lambda_1 & & \lambda_2 & & \lambda_3 \\ \textit{If} & \top\!\!\top & & \top\!\!\top & & \top\!\!\top \\ & \boldsymbol{d}_1 & \vdash^* & \boldsymbol{d}_2 & \vdash^* & \boldsymbol{d}_3 \end{array} \wedge \left\{ \begin{array}{c} \lambda_1 \preccurlyeq \lambda_3, \\ \lambda_2 \preccurlyeq \lambda_3, \\ \exists x. x \in_0 \boldsymbol{d}_1 \end{array} \right\},$$

then $\lambda_1 \preccurlyeq \lambda_2$.

For $\boldsymbol{d} = d_0 d_1 \ldots d_n$, we write $x \in_0 \boldsymbol{d}$ if $x \in d_0$, i.e., a clock x belongs to d_0 of $\boldsymbol{d}$.

We put the proof of this lemma on the appendix for the readability and continue to give a proof for the case **time**. First, we consider the following case:

$$\langle q, \langle\gamma, \boldsymbol{D}\rangle\rangle \to \langle q, \langle\gamma, \boldsymbol{D'}\rangle\rangle$$

where $\boldsymbol{D} \vdash^* \boldsymbol{D'}$. This case is immediate from Proposition 12.

Next, we consider the following general case:

$$\langle q, \boldsymbol{W}\langle\gamma_1, \boldsymbol{D}_1\rangle\langle\gamma_2, \boldsymbol{D}_2\rangle\rangle \to \langle q, \boldsymbol{W}\langle\gamma_1, \boldsymbol{D}_1\rangle\langle\gamma_2, \boldsymbol{D}_2'\rangle\rangle$$

where $\boldsymbol{D}_2 \vdash^* \boldsymbol{D}_2'$. The assumption can be rewritten as follows for some valuations:

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

Since $\boldsymbol{D}_2 \vdash^* \boldsymbol{D}_2'$, by Proposition 12, there is a collapsed valuation Λ_2 such that

$$\begin{array}{ccc} \Lambda_2 & \preccurlyeq & \Lambda_2' \\ \top\!\!\top & & \top\!\!\top \\ \boldsymbol{D}_2 & \vdash^* & \boldsymbol{D}_2'. \end{array}$$

For this, $\boldsymbol{w}\,\Lambda_1\Lambda_2 \models \boldsymbol{W}\boldsymbol{D}_1\boldsymbol{D}_2$ is clear.

Next, we show that $\boldsymbol{w}\,\Lambda_1\,\Lambda_2$ is a well-formed stack. From the well-formedness $\mathsf{WF}(\boldsymbol{w}\Lambda_1\Lambda_2')$, it is clear that $\Lambda_1(\dot{\mathtt{C}}) = 0.0$ and $\mathsf{WF}(\boldsymbol{w}\Lambda_1)$. Hence, it suffices to show $\Lambda_1 \precapprox \Lambda_2$. To show this, we use the above technical lemma (Lemma 9).

We extract some valuations from Λ_1, Λ_2, and Λ_2' as follows:

- $\dot{\lambda}_1 : \dot{\mathcal{X}}_{\mathtt{C}} \to \mathbb{C}$ is the restricted function of Λ_1 to $\dot{\mathcal{X}}_{\mathtt{C}}$. Furthermore, we unmark the clocks of $\dot{\lambda}_1$ and obtain the valuation $\lambda_1 : \mathcal{X}_{\mathtt{C}} \to \mathbb{C}$.
- $\underset{\cdot}{\lambda}_2 : \underset{\cdot}{\mathcal{X}}_{\mathtt{C}} \to \mathbb{C}$ is the restricted function of Λ_2 to $\underset{\cdot}{\mathcal{X}}_{\mathtt{C}}$. We also unmark the clocks of $\underset{\cdot}{\lambda}_2$ and obtain the valuation $\lambda_2 : \mathcal{X}_{\mathtt{C}} \to \mathbb{C}$. $\underset{\cdot}{\lambda}_2' : \underset{\cdot}{\mathcal{X}}_{\mathtt{C}} \to \mathbb{C}$ and $\lambda_2' : \mathcal{X}_{\mathtt{C}} \to \mathbb{C}$ are defined in the same way from Λ_2'.

We also extract some valuations from $\boldsymbol{D}_1$, $\boldsymbol{D}_2$, and $\boldsymbol{D}_2'$ as follows:

- $\boldsymbol{d}_1$ is defined by restricting $\boldsymbol{D}_1$ to $\dot{\mathcal{X}}_{\mathtt{C}}$ and unmarking the clocks.

- $\boldsymbol{d}_2$ is defined by restricting $\boldsymbol{D}_2$ to $\underset{\bullet}{\mathcal{X}}_{\mathtt{C}}$ and unmarking the clocks. $\boldsymbol{d}_2'$ is defined in the same manner from $\boldsymbol{D}_2'$.

We show $\lambda_1 \preccurlyeq \lambda_2$ since it immediately implies $\Lambda_1 \precapprox \Lambda_2$. The above definition leads to the following diagram:

$$\begin{array}{ccccc} \lambda_1 & & \lambda_2 & & \lambda_2' \\ \rotatebox{90}{$\models$} & & \rotatebox{90}{$\models$} & & \rotatebox{90}{$\models$} \\ \boldsymbol{d}_1 & \vdash^* & \boldsymbol{d}_2 & \vdash^* & \boldsymbol{d}_2' \end{array}$$

where $\lambda_1 \models \boldsymbol{d}_1$, $\lambda_2 \models \boldsymbol{d}_2$, and $\lambda_2' \models \boldsymbol{d}_2'$ are derived from $\Lambda_1 \models \boldsymbol{D}_1$, $\Lambda_2 \models \boldsymbol{D}_2$, and $\Lambda_2' \models \boldsymbol{D}_2'$, respectively. The relation $\Lambda_1 \precapprox \Lambda_2'$ implies $\lambda_1 \preccurlyeq \lambda_2$ and $\Lambda_2 \preccurlyeq \Lambda_2'$ implies $\lambda_2 \preccurlyeq \lambda_2'$. The well-formedness $\mathsf{WF}(\boldsymbol{W}\boldsymbol{D}_1\boldsymbol{D}_2)$ implies $\dot{\mathtt{C}} \in_0 \boldsymbol{D}_1$ and $\mathtt{C} \in_0 \boldsymbol{d}_1$.

Now we can use Lemma 9 and obtain $\lambda_1 \preccurlyeq \lambda_2$ and it implies $\Lambda_1 \precapprox \Lambda_2$. □

7.6 Decidability of Configuration Reachability Problem

Combining the forward simulation between the standard and collapsed semantics (Lemma 6) and the forward/backward simulation between the collapsed and digitized semantics (Lemma 7 and 8), we can show our main theorem, Theorem 1. We introduce a notation for the stack correspondence between the standard and digitized semantics $w \models \boldsymbol{W}$ to formalize our main theorem.

Definition 30. Let w and $\boldsymbol{W}$ be stacks of the standard and digitized semantics, respectively. If there is a stack $\boldsymbol{w}$ of the collapsed semantics such that $w \models \boldsymbol{w}$ and $\boldsymbol{w} \models \boldsymbol{W}$, we write $w \models \boldsymbol{W}$. ♢

Theorem 1. *Let $\langle q, w\rangle$ be a configuration of the standard semantics. The following are equivalent:*
(1) $\langle q_{init}, \langle \bot, \mathbf{0}_{\mathcal{X}}\rangle\rangle \xrightarrow[\textsc{Stnd}]{}{}^* \langle q, w\rangle$.
(2) $\langle q_{init}, \langle \bot, \boldsymbol{D}_0\rangle\rangle \xrightarrow[\textsc{Digi}]{}{}^* \langle q, \boldsymbol{W}\rangle$ *for some stack $\boldsymbol{W}$ such that $\mathit{WF}(\boldsymbol{W})$ and $w \models \boldsymbol{W}$ where*

$$\boldsymbol{D}_0 = \{(\underset{\bullet}{x}_1, 0), (\dot{x}_1, 0), \ldots, (\underset{\bullet}{x}_n, 0), (\dot{x}_n, 0), (\underset{\bullet}{\mathtt{C}}, 0), (\dot{\mathtt{C}}, 0)\}_0.$$

Proof. The direction (1) ⇒ (2) is shown by using Lemma 6 and 7 sequentially.

We consider the other direction (2) ⇒ (1). From the assumption $w \models \boldsymbol{W}$, there is a stack $\boldsymbol{w}$ of the collapsed semantics such that $w \models \boldsymbol{w} \models \boldsymbol{W}$. The backward simulation lemma (Lemma 8) implies the following:

$$\exists(\lambda_0 : \text{collapsed valuation}).$$
$$\lambda_0 \models \boldsymbol{D}_0 \wedge \langle q_{\text{init}}, \langle \bot, \lambda_0\rangle\rangle \xrightarrow[\textsc{Coll}]{}{}^* \langle q, \boldsymbol{w}\rangle.$$

Since $\mathbf{0}_{\underset{\bullet}{\mathcal{X}}_{\mathtt{C}} \cup \dot{\mathcal{X}}_{\mathtt{C}}}$ is the only collapsed valuation that satisfies $\mathbf{0}_{\underset{\bullet}{\mathcal{X}}_{\mathtt{C}} \cup \dot{\mathcal{X}}_{\mathtt{C}}} \models \boldsymbol{D}_0$, we also have the following:

$$\langle q_{\text{init}}, \langle \bot, \mathbf{0}_{\underset{\bullet}{\mathcal{X}}_{\mathtt{C}} \cup \dot{\mathcal{X}}_{\mathtt{C}}}\rangle\rangle \xrightarrow[\textsc{Coll}]{}{}^* \langle q, \boldsymbol{w}\rangle.$$

The forward simulation of the collapsed semantics by the standard semantics (Lemma 6) implies the following transition:

$$\langle q_{\text{init}}, \langle \bot, \mathbf{0}_{\mathcal{X}}\rangle\rangle \xrightarrow[\textsc{Stnd}]{}{}^* \langle q, w\rangle.$$

□

The above theorem allows to reduce the configuration reachability problem of the standard semantics to that of the digitized semantics. For a given configuration $\langle q, w\rangle$, there are finitely many well-formed stack $\boldsymbol{W}$ such that $w \models \boldsymbol{W}$ and we can enumerate all such stacks $\boldsymbol{W}$. This computability and the decidability of the configuration reachability problem of pushdown systems imply the decidability of the configuration reachability problem of SRTA.

Corollary 1. *The configuration reachability problem of SRTA is decidable.*

8 Regions vs Digital Valuations

We compare the two kinds of regions:

- the conventional region given by Alur and Dill in [3], which does not keep the fractional parts of the clocks that exceed a given upper bound M.
- the region given by Abdulla et al. in [1], which keeps the fractional parts of clocks even if their values are beyond M. To the authors' best knowledge, such a nonstandard region first appeared at the work of Ouaknine and Worrell [17] to show the decidability of the language inclusion problem on one-clock timed automata.

The difference between keeping and not keeping the fractional parts of the clocks that exceed M is significant: on the region of Alur and Dill, the key lemma for the backward simulation of Digi by Coll (Lemma 9) does not hold.

For the ease of comparison, we consider a slightly modified version of the definition given by Alur and Dill. The region of Alur and Dill can be understood through introducing the following collapsed domain:

$$\mathbb{C}' \triangleq \big([0..(\mathsf{M}-1)] \times [0,1)\big) \cup \{\infty\}.$$

For this collapsed domain, the new collapsing function $\mathcal{C}' : \mathbb{R}_+ \to \mathbb{C}'$ is defined as follows:

$$\mathcal{C}'(r) = \begin{cases} (\lfloor r \rfloor, \{r\}) & \text{if } r < \mathsf{M} \\ \infty & \text{if } r \geq \mathsf{M} \end{cases}$$

We forget the fractional parts of the clocks that are equal to or greater than M.

Instead of giving the precise definition of the region for $\mathbb{C}'$, let us consider the following example under $\mathsf{M} = 1$:

1. The region $R = (\{\}_0 \{(x,0),(y,0)\}, \emptyset)$ means that the values of the two clocks x and y are the same. For example, the collapsed valuation $\rho = \{x \mapsto 0.4;\ y \mapsto 0.4\}$ realizes this region ($\rho \models R$).
2. The successor region R' of the region R ($R \vdash R'$) is $R' = (\{\}_0, \{x, y\})$ and R' means that the values of the two clocks x and y exceed the upper bound $\mathsf{M} = 1$. On the region R', there is no information that the fractional parts of two clocks are the same. The collapsed valuation $\rho' = \{x \mapsto \infty;\ y \mapsto \infty\}$ of the collapsed domain $\mathbb{C}'$ realizes this region ($\rho' \models R'$).

8.1 Key Lemma fails on Region of Alur and Dill

As we have seen in the previous section, the following property (Lemma 9) is key to establishing the backward simulation of Digi by Coll:

$$\begin{array}{ccccc} \lambda_1 & & \lambda_2 & & \lambda_3 \\ \top\!\!\top & & \top\!\!\top & & \top\!\!\top \\ \boldsymbol{d}_1 & \vdash^* & \boldsymbol{d}_2 & \vdash^* & \boldsymbol{d}_3 \end{array} \wedge \left\{ \begin{array}{c} \lambda_1 \preccurlyeq \lambda_3, \\ \lambda_2 \preccurlyeq \lambda_3, \\ \exists x. x \in_0 \boldsymbol{d}_1 \end{array} \right\}$$
$$\implies \lambda_1 \preccurlyeq \lambda_2.$$

This property does not hold on the region of Alur and Dill. Let us consider the following collapsed valuations and regions on $\mathbb{C}'$ under $\mathsf{M} = 1$:

$$(1): \begin{array}{c} \rho_1 = \{x \mapsto 0.0;\ y \mapsto 0.5\} \\ \top\!\!\top \\ R_1 = \big(\{(x,0)\}_0 \{(y,0)\}, \emptyset\big), \end{array}$$

$$(2): \begin{array}{c} \rho_2 = \{x \mapsto 0.5;\ y \mapsto 0.9\} \\ \top\!\!\top \\ R_2 = \big(\{(x,0)\}_0 \{(y,0)\}, \emptyset\big), \end{array}$$

$$(3): \begin{array}{c} \rho_3 = \{x \mapsto \infty;\ y \mapsto \infty\} \\ \top\!\!\top \\ R_3 = \big(\{\}_0, \{x, y\}\big). \end{array}$$

Although these valuations and regions satisfy the assumption of the above property, $\rho_1 \preccurlyeq \rho_2$ does not hold. Since we forget the fractional parts of the clocks that exceed an upper-bound constant, the assumptions $\rho_1 \preccurlyeq \rho_3$ and $\rho_2 \preccurlyeq \rho_3$ do not help to relate the fractional parts of ρ_1 and ρ_2.

On the other hand, on the collapsed domain $\mathbb{C} = \Big(\big([0..(\mathsf{M}-1)] \cup \{\infty\}\big) \times [0,1)\Big)$ considered by Abdulla et al, there is no collapsed valuation λ_3 that satisfies both $\{x \mapsto 0.0;\ y \mapsto 0.5\} \preccurlyeq \lambda_3$ and $\{x \mapsto 0.5;\ y \mapsto 0.9\} \preccurlyeq \lambda_3$. This enables us to avoid the above problem and establish the key lemma for the backward simulation.

9 Related Work

Dense-Timed Pushdown Automata.

Abdulla, Atig, and Stenman introduced dense-timed pushdown automata (TPDA) in [1]. A TPDA is a 5-tuple $\mathcal{T} = (Q, q_{\text{init}}, \Gamma, \mathcal{X}, \hookrightarrow)$ where Q is a finite set of locations, $q_{\text{init}} \in Q$ is the initial location, Γ is a finite set of stack symbols, $\mathcal{X}$ is a finite set of (global) clocks, and $\hookrightarrow$ is a finite set of transition rules. A configuration $\langle p, \mathtt{X}, w \rangle$ is a triple of a location $p \in Q$, valuation $\mathtt{X} : \mathcal{X} \to \mathbb{R}_+$, and aged stack $w \in (\Gamma \times \mathbb{R}_+)^*$. Each transition rule is a triple $p \overset{\tau}{\hookrightarrow} q$ of two locations and an action τ. TPDA have the following four kinds of discrete operations:

$\mathsf{push}(\gamma, I)$: $\langle p, \mathtt{X}, w \rangle \to \langle q, \mathtt{X}, w \langle \gamma, k \rangle \rangle$ where $k \in I$,
$\mathsf{pop}(\gamma, I)$: $\langle p, \mathtt{X}, w \langle \gamma, k \rangle \rangle \to \langle q, \mathtt{X}, w \rangle$ if $k \in I$,
$x \leftarrow I$: $\langle p, \mathtt{X}, w \rangle \to \langle q, \mathtt{X}[x := k], w \rangle$ where $k \in I$,
$x \in_? I$: $\langle p, \mathtt{X}, w \rangle \to \langle q, \mathtt{X}, w \rangle$ if $\mathtt{X}(x) \in I$.

Note that there are no fractional constraints ($\{x\} = 0$ or $\{x\} \bowtie \{y\}$).

They showed the decidability of the location reachability problem of TPDA. For a given control location q, the *location reachability problem* decides if there is a run from the initial configuration $\langle q_{\text{init}}, \mathbf{0}, \epsilon \rangle$ to $\langle q, \mathtt{X}, w \rangle$ for some valuation $\mathtt{X}$ and stack w.

Main Result of [1]. *The location reachability* *problem of TPDA is* ExpTime*-complete with respect to* $|Q| + |\mathcal{X}| + |\Gamma| + |\hookrightarrow|$.

In our previous work, we showed that we can convert a TPDA to the corresponding SRTA with linear size increase [20]. Furthermore, we showed that SRTA are strictly more expressive than TPDA:

The SRTA language L_{ex} *in Section 3 cannot be recognized by any TPDA.*

Recall that we used fractional constraints $\{x\} = 0$ to recognize the language L_{ex} in Section 3. On the other hand, since there are no fractional constraints in TPDA, we cannot accept L_{ex} by any TPDA. The interested reader is referred to the long version of our previous work [21] for the detailed proof.

Pushdown Timed Systems.

In advance of TPDA, Bouajjani, Echahed, and Robbana introduced pushdown timed systems (PTS) in [6]. PTS can be seen as timed automata with an untimed stack or TPDA without local clocks: a configuration $\langle p, \mathtt{X}, w\rangle$ of PTS is a triple of a location $p \in Q$, valuation $\mathtt{X} : \mathcal{X} \to \mathbb{R}_+$, and (untimed) stack $w \in \Gamma^*$. They showed the decidability of the location reachability problem of PTS (and it immediately follows from the decidability of that of TPDA). Interestingly, Clemente and Lasota showed that the language classes of TPDA and PTS are equivalent [9]. In other words, we can remove all the local clocks from stacks of TPDA.

TPDA with Diagonal Constraints.

Clemente and Lasota [9] equipped TPDA with diagonal constraints and showed that the language classes of TPDA and TPDA with diagonal constraints are equivalent. The diagonal constraints $\psi \in \Psi_{\mathcal{X}}$ on a clock set $\mathcal{X}$ in their model are given as follows:

$$\psi ::= x \bowtie k \mid x - y \bowtie k \mid \psi \wedge \psi$$

where $x, y \in \mathcal{X}$, $k \in \mathbb{Z}$, and $\bowtie \in \{<, \leq, \geq, >\}$. Note again that there are no fractional constraints ($\{x\} = 0$ or $\{x\} \bowtie \{y\}$). On TPDA, the rules push and pop are extended with a diagonal constraint as follows:

$$\begin{aligned}
&\mathsf{push}(\gamma, \psi) : \langle p, \mathtt{X}, w\rangle \to \langle q, \mathtt{X}, w\langle\gamma, k\rangle\rangle,\\
&\qquad \text{if } \mathtt{X} \cup \{z \mapsto k\} \models \psi \text{ where } \psi \in \Psi_{\mathcal{X}\cup\{z\}},\\
&\mathsf{pop}(\gamma, \psi) : \langle p, \mathtt{X}, w\langle\gamma, k\rangle\rangle \to \langle q, \mathtt{X}, w\rangle,\\
&\qquad \text{if } \mathtt{X} \cup \{z \mapsto k\} \models \psi \text{ where } \psi \in \Psi_{\mathcal{X}\cup\{z\}}.
\end{aligned}$$

They showed that adding local clocks to PTS and diagonal constraints to TPDA does not increase the expressiveness of PTS.

Main Result of [9]. *For a given TPDA with diagonal constraints* $\mathcal{T}$*, we can construct a PTS* $\mathcal{P}$ *such that the language* $L(\mathcal{P})$ *equals to* $L(\mathcal{T})$*.*

In our previous work, we consider SRTA with diagonal constraints and showed the following two results on SRTA with diagonal constraints [20]:

1. We can translate a TPDA with diagonal constraints to the corresponding SRTA with diagonal constraints with linear size increase.
2. We can translate an SRTA with diagonal constraints to the corresponding SRTA without diagonal constraints with exponential size increase.

It should be noted that adding fractional constraints to TPDA allows to accept the SRTA language L_{ex}. Therefore, we cannot remove local clocks from TPDA with fractional constraints.

Recursive Timed Automata.

Trivedi and Wojtczak introduced recursive timed automata (RTA) [19] and Benerecetti, Minopoli, and Peron introduced timed recursive state machines [4][5] independently. The two models are essentially equivalent. Compared to SRTA, timed transitions of RTA increase only the top frame of a stack as follows:

$$\begin{aligned}
&\langle q, \langle\gamma_1, \nu_1\rangle \ldots \langle\gamma_n, \nu_n\rangle\rangle\\
&\quad \overset{\delta}{\rightsquigarrow} \langle q, \langle\gamma_1, \nu_1\rangle \ldots \langle\gamma_n, \nu_n + \delta\rangle\rangle
\end{aligned}$$

where $\delta \in \mathbb{R}_+$. The difference of timed transitions between SRTA and RTA is crucial because RTA can simulate two-counter machines by using the timed transitions effectively and thus the reachability problem of RTA is undecidable [19][4].

Li, Cai, Ogawa, and Yuen introduced nested timed automata (NeTA) in [15]. A configuration of NeTA is a stack of timed automata that are synchronously evolved by timed transitions. NeTA can be seen as SRTA without fractional constraints and value passing mechanisms between frames like the rule **dig**. Due to the absence of value passing mechanisms, we cannot keep values of clocks in the top frame when removing the top frame; indeed, the following transition that mixes the top and next to the top frames is not allowed:

$$\begin{aligned}
&\langle\gamma_1, \{x \mapsto 1.0; y \mapsto 2.0\}\rangle\langle\gamma_2, \{x \mapsto 3.5; y \mapsto 4.5\}\rangle\\
&\xrightarrow{\mathsf{pop}(\gamma_2, \{x\})} \langle\gamma_1, \{x \mapsto 1.0;\ y \mapsto 4.5\}\rangle.
\end{aligned}$$

NeTA only removes the top frame as follows:

$$\langle \gamma_1, \{x \mapsto 1.0; y \mapsto 2.0\}\rangle\langle \gamma_2, \{x \mapsto 3.5; y \mapsto 4.5\}\rangle \xrightarrow{\mathsf{pop}'(\gamma_2)} \langle \gamma_1, \{x \mapsto 1.0;\ y \mapsto 2.0\}\rangle.$$

Krishna, Manasa, and Trivedi introduced the subset of RTA called $\mathrm{RTA}_{\mathrm{RN}}$ (RTA only with pass-by-reference and non-passing) in [14] and showed that the location reachability problem of $\mathrm{RTA}_{\mathrm{RN}}$ is decidable. $\mathrm{RTA}_{\mathrm{RN}}$ can be seen as SRTA without fractional constraints. Although their proof closely followed that of Abdulla et al. [1] and depended on the details of the construction of Abdulla et al., we adapt the construction of Abdulla et al. to simplify our proof.

Configuration Reachability Problem

We have considered the configuration reachability problem of SRTA rather than the location reachability problem of SRTA. The former problem decides if there is a run from the initial configuration to a given configuration $\langle q, w\rangle$, $\langle q_{\mathrm{init}}, \mathbf{0}\rangle \to^* \langle q, w\rangle$, and the latter problem decides if $\langle q_{\mathrm{init}}, \mathbf{0}\rangle \to^* \langle q, {}^{\exists}w\rangle$ for a given location q. Although we have shown the decidability of the former problem, we can show the decidability of the latter problem in the same argument of Theorem 1 because the latter problem is reduced to the location reachability problem of pushdown systems and it is decidable.

In the context of timed automata (without the timed stack), it has been shown that the decidability of the configuration reachability problem in general form [10] [12] [18]. They showed that the binary reachability relation of two locations q and q', $\{(\nu, \nu') : (q, \nu) \to^* (q', \nu')\}$, is effectively definable in the additive theory of real numbers [10] [18] or representable by a class of automata called $2n$-automata [12]. On the basis of those characterizations, they showed the decidability of the general configuration reachability problem of timed automata: we can decide if $\langle q, \nu\rangle \to^* \langle q', \nu'\rangle$ for given configurations $\langle q, \nu\rangle$ and $\langle q', \nu'\rangle$.

Our proof for SRTA is also applicable to timed automata and then we obtain the decidability of the configuration reachability problem of the form $\langle q_{\mathrm{init}}, \mathbf{0}\rangle \to^* \langle q, w\rangle$ for a given configuration $\langle q, w\rangle$. Although this decidability result is weaker than that of the general configuration reachability problem, our proof based on the backward simulation is simpler than their involved proofs.

The decidability of the general configuration reachability problem was extended to pushdown timed systems (timed automata with untimed stacks) [11]. However, it seems hard to use the same technique of [11] for timed automata with timed stacks such as TPDA and SRTA because his argument highly depends on the untimed stack rather than the timed stack.

10 Conclusion

We have studied synchronized recursive timed automata (SRTA) and shown the decidability of the configuration reachability problem of SRTA. Our SRTA are described from the two perspectives:

- SRTA are a variant of recursive timed automata (RTA) of Trivedi and Wojtczak, and Benerecetti et al. [19] [4] [5] because SRTA are obtained by synchronizing timed transitions of RTA;
- SRTA are an extension of dense timed pushdown automata of Abdulla et al. [1] with fractional constraints.

We have introduced the three semantics—the lazy semantics Lazy, the collapsed semantics Coll, and the digitized semantics Digi—to show the decidability of the configuration reachability problem. These intermediate semantics separated the involved backward-forward simulation of Abdulla et al. into simple simulations, and this made our proof easy to follow.

As we have seen in Section 4. 2, the backward simulation of the digitized semantics by the collapsed semantics is crucial to the configuration reachability problem. We think the backward simulation is discovered by explicitly introducing the collapsed semantics. Indeed, the backward simulation of digitized semantics by the standard semantics does not hold (See the remark after Proposition 12).

Through our translations from the standard semantics to the digitized semantics, reference clocks play important roles. We have seen that reference clocks are necessary to establish several lemmas by presenting several concrete examples. Although we have given explanations about "why reference clocks are needed", we have not given a description about "what reference clocks are". To answer this, we need a deeper analysis of SRTA. We believe such analysis extends pushdown-extensions of timed automata to more interesting theoretical model.

Acknowledgment This work was supported by JSPS KAKENHI Grant Numbers 15J01843 and 15K00087.

References

[1] Abdulla, P., Atig, M., and Stenman, J.: Dense-Timed Pushdown Automata, in *LICS'12*, IEEE, 2012, pp. 35–44.

[2] Alur, R., Courcoubetis, C., and Dill, D.: Model-checking for real-time systems, in *LICS'90*, 1990, pp. 414–425.

[3] Alur, R. and Dill, D.: A theory of timed automata, *TCS*, Vol. 126, No. 2(1994), pp. 183–235.

[4] Benerecetti, M., Minopoli, S. and Peron, A.: Analysis of Timed Recursive State Machines, in *TIME'10*, IEEE, 2010, pp. 61–68.

[5] Benerecetti, M. and Peron, A.: Timed recursive state machines: Expressiveness and complexity, *TCS*, Vol. 625(2016), pp. 85–124.

[6] Bouajjani, A., Echahed, R. and Robbana, R.: On the automatic verification of systems with continuous variables and unbounded discrete data structures, *Hybrid Systems II*, Springer, 1995, pp. 64–85.

[7] Bouajjani, A., Esparza, J. and Maler, O.: Reachability analysis of pushdown automata: Application to model-checking, in *CONCUR'97*, Springer, 1997, pp. 135–150.

[8] Büchi, J.: Regular canonical systems, *Arch. Math. Logik Grundlag*, Vol. 6(1964), pp. 91–111.

[9] Clemente, L. and Lasota, S.: Timed pushdown automata revisited, in *LICS'15*, IEEE, 2015, pp. 738–749.

[10] Comon, H. and Jurski, Y.: Timed Automata and the Theory of Real Numbers, in *CONCUR'99*, Springer, 1999, pp. 242–257.

[11] Dang, Z.: Pushdown timed automata: a binary reachability characterization and safety verification, *TCS*, Vol. 302, No. 1(2003), pp. 93–121.

[12] Dima, C.: Computing reachability relations in timed automata, in *LICS'02*, 2002, pp. 177–186.

[13] Finkel, A., Willems, B. and Wolper, P.: A direct symbolic approach to model checking pushdown systems, in *INFINITY'97*, Elsevier, 1997, pp. 27–37.

[14] Krishna, S., Manasa, L. and Trivedi, A.: What's Decidable About Recursive Hybrid Automata?, in *HSCC'15*, ACM, 2015, pp. 31–40.

[15] Li, G., Cai, X., Ogawa, M. and Yuen, S.: Nested Timed Automata, in *FORMATS'13*, Springer, 2013, pp. 168–182.

[16] Lynch, N. and Vaandrager, F.: Forward and Backward Simulations, *Inf. & Comp.*, Vol. 121, No. 2(1995), pp. 214–233.

[17] Ouaknine, J. and Worrell, J.: On the language inclusion problem for timed automata: closing a decidability gap, in *LICS'04*, IEEE, 2004, pp. 54–63.

[18] Quaas, K., Shirmohammadi, M. and Worrell, J.: Revisiting Reachability in Timed Automata, *CoRR*, Vol. abs/1702.03450(2017).

[19] Trivedi, A. and Wojtczak, D.: Recursive Timed Automata, in *ATVA'10*, Springer, 2010, pp. 306–324.

[20] Uezato, Y. and Minamide, Y.: Synchronized Recursive Timed Automata, in *LPAR-20*, Springer, 2015, pp. 249–265.

[21] Uezato, Y. and Minamide, Y.: Synchronized Recursive Timed Automata (Long version), http://www.logic.cs.tsukuba.ac.jp/~uezato/LPAR20_uezato.pdf, 2015.

[22] Uezato, Y. and Minamide, Y.: Coq Proof Script for SRTA, http://www.logic.cs.tsukuba.ac.jp/~uezato/srta, 2016.

A Proof of Lemma 9

We show the following technical lemma:

$$\begin{array}{ccccc} \lambda_1 & & \lambda_2 & & \lambda_3 \\ \sqcap\!\!\sqcap & & \sqcap\!\!\sqcap & & \sqcap\!\!\sqcap \\ \boldsymbol{d}_1 & \vdash^* & \boldsymbol{d}_2 & \vdash^* & \boldsymbol{d}_3 \end{array} \land \left\{ \begin{array}{c} \lambda_1 \preccurlyeq \lambda_3, \\ \lambda_2 \preccurlyeq \lambda_3, \\ \exists x. x \in_0 \boldsymbol{d}_1 \end{array} \right\}$$

$\Longrightarrow \lambda_1 \preccurlyeq \lambda_2.$

Before giving a proof, let us see that we cannot drop any condition.

If we drop the condition $\lambda_1 \preccurlyeq \lambda_3$ ...?

Let us consider the following diagram:

$$\begin{array}{ccccc} \left\{ \begin{array}{c} x \mapsto 0.0 \\ y \mapsto 0.5 \end{array} \right\} & & \left\{ \begin{array}{c} x \mapsto 0.2 \\ y \mapsto 0.3 \end{array} \right\} & & \left\{ \begin{array}{c} x \mapsto 0.4 \\ y \mapsto 0.5 \end{array} \right\} \\ \sqcap\!\!\sqcap & & \sqcap\!\!\sqcap & & \sqcap\!\!\sqcap \\ \{(x,0)\}_0 \{(y,0)\} & \vdash^* & \boldsymbol{d}_2 & \vdash^* & \boldsymbol{d}_3 \end{array}$$

where $\boldsymbol{d}_2 = \boldsymbol{d}_3 = \{\}_0 \{(x,0)\} \{(y,0)\}$. Although this case satisfies the assumptions except $\lambda_1 \preccurlyeq \lambda_3$, $\lambda_1 \preccurlyeq \lambda_2$ does not hold. For the same reason, we cannot remove the condition $\lambda_2 \preccurlyeq \lambda_3$.

If we drop the condition $\exists x. x \in_0 \boldsymbol{d}_1$ ...?

Let us consider the following diagram:

$$\begin{array}{ccccc} \{x \mapsto 0.5\} & & \{x \mapsto 0.3\} & & \{x \mapsto 0.5\} \\ \sqcap\!\!\sqcap & & \sqcap\!\!\sqcap & & \sqcap\!\!\sqcap \\ \{\}_0 \{(x,0)\} & \vdash^* & \{\}_0 \{(x,0)\} & \vdash^* & \{\}_0 \{(x,0)\} \end{array}$$

Although this case satisfies the assumptions except $\exists x. x \in_0 \boldsymbol{d}_1$, $\lambda_1 \preccurlyeq \lambda_2$ does not hold. Therefore, we cannot drop the condition $\exists x. x \in_0 \boldsymbol{d}_1$.

Proof of Lemma 9

We prepare several notations to prove this lemma. Let λ and λ' be collapsed valuations on a finite clock set $\mathcal{X}$.

$$\begin{array}{lcl} \{\lambda\}(x) & \triangleq & \{\lambda(x)\}, \\ \lfloor\lambda\rfloor(x) & \triangleq & \lfloor\lambda(x)\rfloor, \\ \lambda \doteqdot \lambda' & \Longleftrightarrow & \exists\delta \in \mathbb{R}_+.\ \{\lambda+\delta\} = \{\lambda'\}. \end{array}$$

It is clear that the relation $\doteqdot$ is an equivalence relation.

Proposition 14. *Let λ and λ' be collapsed valuations on a finite clock set $\mathcal{X}$.*

If $\lambda \doteqdot \lambda'$ and $\exists x \in \mathcal{X}. \{\lambda\}(x) = \{\lambda'\}(x)$, then $\{\lambda\} = \{\lambda'\}$.

We show that related valuations λ and λ' with $\lambda \doteqdot \lambda'$ imply $\lambda = \lambda'$ under some constraints.

Proposition 15.

$$\begin{array}{ccc} \lambda & \doteqdot & \lambda' \\ \rotatebox[origin=c]{270}{$\models$} & & \rotatebox[origin=c]{270}{$\models$} \\ \boldsymbol{d} & = & \boldsymbol{d} \end{array} \wedge (\exists x. x \in_0 \boldsymbol{d}) \implies \lambda = \lambda'.$$

Proof. It is clear that $\lfloor\lambda\rfloor = \lfloor\lambda'\rfloor$ from $\lambda \models \boldsymbol{d}$ and $\lambda' \models \boldsymbol{d}$. Since there exists a clock x such that $x \in_0 \boldsymbol{d}$, we have $\{\lambda(x)\} = \{\lambda'(x)\} = 0.0$ from the assumption $\lambda \doteqdot \lambda'$. Proposition 14 implies $\{\lambda\} = \{\lambda'\}$. This and $\lfloor\lambda\rfloor = \lfloor\lambda'\rfloor$ imply $\lambda = \lambda'$. □

Furthermore, since $\lambda \preccurlyeq \lambda'$ implies $\lambda \doteqdot \lambda'$, we have the following property.

Proposition 16.

$$\begin{array}{ccc} \lambda & \preccurlyeq & \lambda' \\ \rotatebox[origin=c]{270}{$\models$} & & \rotatebox[origin=c]{270}{$\models$} \\ \boldsymbol{d} & = & \boldsymbol{d} \end{array} \wedge\ \exists x. x \in_0 \boldsymbol{d} \implies \lambda = \lambda'.$$

Although $\lambda \doteqdot \lambda'$ does not imply $\lambda \preccurlyeq \lambda'$ in general, if there are digital valuations $\boldsymbol{d}$ and $\boldsymbol{d'}$ such that $\lambda \models \boldsymbol{d}$, $\lambda' \models \boldsymbol{d'}$, and $\boldsymbol{d} \vdash \boldsymbol{d'}$, then $\lambda \doteqdot \lambda'$ implies $\lambda \preccurlyeq \lambda'$.

Proposition 17.

$$\begin{array}{ccc} \lambda & \doteqdot & \lambda' \\ \rotatebox[origin=c]{270}{$\models$} & & \rotatebox[origin=c]{270}{$\models$} \\ \boldsymbol{d} & \vdash & \boldsymbol{d'} \end{array} \implies \lambda \preccurlyeq \lambda'.$$

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

Case $\emptyset\, d_1 \cdots d_{n-1}\, d_n \vdash d'_n\, d_1 \cdots d_{n-1}$**:** Note there exists a clock x that belongs to d'_n and thus $x \in_0 \boldsymbol{d'}$. By the forward simulation of digital valuations by collapsed valuations (Proposition 11), we can find a collapsed valuation λ'' that satisfies the following diagram:

$$\begin{array}{ccc} \lambda & \preccurlyeq & \lambda'' \\ \rotatebox[origin=c]{270}{$\models$} & & \rotatebox[origin=c]{270}{$\models$} \\ \boldsymbol{d} & \vdash & \boldsymbol{d'} \end{array}$$

Since $\lambda \preccurlyeq \lambda''$ implies $\lambda \doteqdot \lambda''$, we have $\lambda' \doteqdot \lambda''$. Using Proposition 15 along with $\lambda' \models \boldsymbol{d'}$, $\lambda'' \models \boldsymbol{d'}$, and $x \in_0 \boldsymbol{d'}$, we have $\lambda' = \lambda''$ and thus $\lambda \preccurlyeq \lambda'$.

Case $d_0\, d_1 \cdots d_n \vdash \emptyset\, d'_0\, d_1 \cdots d_n$**:** We can use the same argument as above. There exists a clock x that belongs to d_0 and thus $x \in_0 \boldsymbol{d}$. By the backward simulation of digital valuations by collapsed valuations (Proposition 12), we can find a collapsed valuation λ'' that satisfies the following diagram:

$$\begin{array}{ccc} \lambda'' & \preccurlyeq & \lambda' \\ \rotatebox[origin=c]{270}{$\models$} & & \rotatebox[origin=c]{270}{$\models$} \\ \boldsymbol{d} & \vdash & \boldsymbol{d'} \end{array}$$

Since $\lambda'' \preccurlyeq \lambda'$ implies $\lambda' \doteqdot \lambda''$, we have $\lambda \doteqdot \lambda''$. Using Proposition 15 along with $\lambda \models \boldsymbol{d}$, $\lambda'' \models \boldsymbol{d}$, and $x \in_0 \boldsymbol{d}$, we have $\lambda = \lambda''$ and thus $\lambda \preccurlyeq \lambda'$.

□

We can easily generalize the above proposition to the following one.

Proposition 18.

$$\begin{array}{ccc} \lambda & \doteqdot & \lambda' \\ \rotatebox[origin=c]{270}{$\models$} & & \rotatebox[origin=c]{270}{$\models$} \\ \boldsymbol{d} & \vdash^+ & \boldsymbol{d'} \end{array} \implies \lambda \preccurlyeq \lambda'.$$

Finally, Proposition 16 and 18 imply our key technical lemma.

Lemma 9.

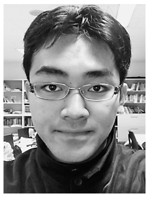
$$\begin{array}{ccccc} \lambda_1 & & \lambda_2 & & \lambda_3 \\ \rotatebox[origin=c]{270}{$\models$} & & \rotatebox[origin=c]{270}{$\models$} & & \rotatebox[origin=c]{270}{$\models$} \\ \boldsymbol{d}_1 & \vdash^* & \boldsymbol{d}_2 & \vdash^* & \boldsymbol{d}_3 \end{array} \wedge \left\{ \begin{array}{c} \lambda_1 \preccurlyeq \lambda_3, \\ \lambda_2 \preccurlyeq \lambda_3, \\ \exists x. x \in_0 \boldsymbol{d}_1 \end{array} \right\}$$

$$\implies \lambda_1 \preccurlyeq \lambda_2.$$

Proof. From the assumption $\lambda_1 \preccurlyeq \lambda_3$ and $\lambda_2 \preccurlyeq \lambda_3$, $\lambda_1 \doteqdot \lambda_2 \doteqdot \lambda_3$ holds.

First, we consider the case $\boldsymbol{d}_1 = \boldsymbol{d}_2$. Since $\lambda_1 \doteqdot \lambda_2$, this case is immediate from Proposition 16.

Next, we consider the other case $\boldsymbol{d}_1 \neq \boldsymbol{d}_2$ (i.e., $\boldsymbol{d}_1 \vdash^+ \boldsymbol{d}_2$). This case is immediate from Proposition 18. □



Uezato Yuya

Yuya Uezato is a doctoral student at the Department of Computer Science, University of Tsukuba. He received his M.Eng. degree from University of Tsukuba in 2015. He is interested in applying the theory of formal languages to software and program verification.

Minamide Yasuhiko

Yasuhiko Minamide is a professor at the Department of Mathematical and Computing Science, Tokyo Institute of Technology. His research interests focus on software verification and programming languages. He is also interested in the theory and applications of automata and formal languages. He received his M.Sc. and Ph.D degrees from Kyoto University in 1993 and 1997, respectively.