SIMULATIONS FOR EVENT-CLOCK AUTOMATA

S. AKSHAY [a], PAUL GASTIN [b,d], R. GOVIND [a,e], AND B. SRIVATHSAN [c,d]

[a] Department of Computer Science and Engineering, Indian Institute of Technology Bombay, Mumbai, India
e-mail address: akshayss@cse.iitb.ac.in

[b] Université Paris-Saclay, ENS Paris-Saclay, CNRS, LMF, 91190, Gif-sur-Yvette, France
e-mail address: paul.gastin@ens-paris-saclay.fr

[c] Chennai Mathematical Institute, India
e-mail address: sri@cmi.ac.in

[d] CNRS, ReLaX, IRL 2000, Siruseri, India

[e] Uppsala University, Sweden
e-mail address: govind.rajanbabu@it.uu.se

ABSTRACT. Event-clock automata (ECA) are a well-known *semantic* subclass of timed automata (TA) which enjoy admirable theoretical properties, e.g., determinizability, and are practically useful to capture timed specifications. However, unlike for timed automata, there exist no implementations for checking non-emptiness of event-clock automata. As ECAs contain special *prophecy clocks* that guess and maintain the time to the next occurrence of specific events, they cannot be seen as a syntactic subclass of TA. Therefore, implementations for TA cannot be directly used for ECAs, and moreover the translation of an ECA to a semantically equivalent TA is expensive. Another reason for the lack of ECA implementations is the difficulty in adapting zone-based algorithms, critical in the timed automata setting, to the event-clock automata setting. This difficulty was studied by Geeraerts et al. in 2011, where the authors proposed a zone enumeration procedure that uses *zone extrapolations* for finiteness.

In this article, we propose a different zone-based algorithm to solve the reachability problem for event-clock automata, *using simulations* for finiteness. A surprising consequence of our result is that for event-predicting automata, the subclass of event-clock automata that only use prophecy clocks, we obtain finiteness even without any simulations. For general event-clock automata, our new algorithm exploits the $\mathcal{G}$-simulation framework, which is the coarsest known simulation relation in timed automata literature, and has been recently used for advances in other extensions of timed automata.

1. Introduction

Timed automata (TA) [AD94] are a well-established model for real-time systems and form the basis for employing model-checking techniques. The most popular property that has been considered in these systems is (control state) reachability. Reachability in timed automata is a well-studied problem and was shown to be decidable (and PSPACE-complete) using the so-called region construction [AD94]. This construction was primarily of theoretical interest, as the number of regions, which are collections of reachable configurations, explodes both in theory and in practice. On the other hand, timed automata have been implemented in several tools: UPPAAL [LPY97, BDL$^+$06], KRONOS [BDM$^+$98], PAT [SLDP09], RED [Wan06], TChecker [HP19], Theta [THV$^+$17], LTS-Min [KLM$^+$15], Symrob [RSM19], MCTA [KWNP08], etc. Most of these tools have a common underlying algorithm which is an explicit enumeration of reachable configurations stored as *zones* [BY03]. Since the late 90s, a substantial effort has been invested in improving zone enumeration techniques, the common challenge being how to get a sound and complete enumeration while exploring as few zones as possible.

The more general model-checking problem asks whether the system represented by a TA $\mathcal{A}$ satisfies the specification given by a TA $\mathcal{B}$. This problem reduces to checking language inclusion $\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{B})$. There are two challenges here: first, the inclusion problem is undecidable in its full generality, and second, having clocks, though excellent for timed implementations, is often less than ideal for modeling timed specifications. This has led to the introduction of event-clocks and the corresponding model of *event-clock automata (ECA)* [AFH99]. Event-clock automata make use of special clocks that track the time since the last occurrence of an event (history clocks) or the time until the next occurrence of an event (prophecy clocks). On one hand this makes writing timed specifications more natural. Indeed, the role of prophecy clocks is in the same spirit as future modalities in temporal logics. This has led to several extensions of temporal logics with event-clocks [DT04, ABG13, RS99], which are often used as specification languages and can be converted into ECA. On the other hand, ECAs can be determinized and hence complemented. So, model-checking event-clock specifications over TA models can be reduced to the emptiness problem on the product of the TA with an ECA (for the complement of the specification). This product contains usual clocks, history clocks and prophecy clocks. The usual clocks can be treated in the same way as history clocks for the zone analysis. Therefore, if we solve ECA reachability (with history and prophecy clocks) using zones, we can incorporate usual clocks into the procedure seamlessly.

Note that ECAs are a semantic (and not syntactic) subclass of TAs., i.e., while the class of languages recognized by ECA are a subclass of the class of languages recognized by TA, syntactically ECAs are not a subclass of TAs. Thus, even though there are several efficient zone-based algorithms based on time-abstract simulations for timed automata, these cannot be directly applied to event-clock automata. In the work that first introduced ECA [AFH99], a translation from an event-clock automaton to an equivalent timed automaton was also proposed. One could, in principle, first translate a given ECA into a timed automaton using the translation proposed in [AFH99], and then run the state-of-the-art reachability algorithm of [GMS18, GMS19, GMS20] on this timed automaton. However, the translation from ECA to TA is not efficient: in the worst case, this translation incurs a blowup that is linear in the number of clocks and exponential in the number of clocks [GRS14].

Thus, in this paper, we focus on the core problem of building efficient, zone-based algorithms for reachability in ECA. This problem turns out to be significantly different compared to zone-based reachability algorithms in usual TA, precisely due to prophecy clocks. Our goal is to align the zone-based reachability algorithms for ECAs with recent approaches for TAs that have shown significant gains.

As mentioned earlier, the core of an efficient TA reachability algorithm is an enumeration of zones, where the central challenge is that naïve enumeration does not terminate. One approach to guarantee termination is to make use of an *extrapolation* operation on zones: each new zone that is enumerated is extrapolated to a bigger zone. Any freshly enumerated zone that is contained in an existing zone is discarded. More recently, a new *simulation* approach to zone enumeration has been designed, where enumerated zones are left unchanged (i.e., no extrapolation). Instead, with each fresh zone it is checked whether the fresh zone is simulated by an already seen zone. If yes, the fresh zone is discarded. Otherwise, it is kept for further exploration. Different simulations have been considered: the LU-simulation [HSW12] which is based on maximum constants appearing in lower and upper bounded constraints in the TA, or the $\mathcal{G}$-simulation [GMS20], which is based on a carefully-chosen set of constraints from the TA. Coarser simulations lead to fewer zones being enumerated. The $\mathcal{G}$-simulation is currently the coarsest-known simulation that can be efficiently applied in the simulation approach. The simulation based approach offers several gains over the extrapolation approach: (1) since concrete zones are maintained, one could use dynamic simulation parameters and dynamic simulations, starting from a coarse simulation and refining whenever necessary [HSW13], (2) the simulation approach has been extended to richer models like timed automata with diagonal constraints [GMS19, GMS18], updatable timed automata [GMS20], weighted timed automata [BCM16] and pushdown timed automata [AGP21]. In these richer models, extrapolation has either been shown to be impossible [Bou04] or is unknown.

Surprisingly, for ECA, an arguably more basic and well-known model, it turns out that there are no existing simulation-based approaches. However, an extrapolation approach using maximal constants has been studied for ECA in [GRS11, GRS14]. In this work, the authors start by showing that prophecy clocks exhibit fundamental differences as compared to usual clocks. To begin with, it was shown that there is no finite time-abstract bisimulation for ECA in general. This is in stark contrast to TA where the region equivalence forms a finite time-abstract bisimulation. The correctness of extrapolation is strongly dependent on the region equivalence. Therefore, in order to get an algorithm, the authors define a weak semantics for ECA and a corresponding notion of weak regions which is a finite time-abstract bisimulation for the weak semantics and show that the weak semantics is sound for reachability. Building on this, they define an extrapolation operation for the zone enumeration.

Our contributions. Given the advantages of using simulations with respect to extrapolations in the TA setting described above, we extend the $\mathcal{G}$-simulation approach to ECA. Here are the technical contributions leading to the result.

- We start with a slightly modified presentation of zones in ECA and provide a clean algebra for manipulating weights in the graph representation for such event-zones. This simplifies the reasoning and allows us to adapt many ideas for simulation developed in the TA setting directly to the ECA setting.

- The $\mathcal{G}$-simulation is parameterized by a set of constraints at each state of the automaton. We adapt the constraint computation and the definition of the simulation to the context of ECA, the main challenge being the handling of prophecy clocks.
- We give a simulation test between two zones that runs in time quadratic in the number of clocks. This is an extension of the similar test that exists for timed automata, but now it incorporates new conditions that arise due to prophecy clocks.
- Finally, we show that the reachability algorithm using the $\mathcal{G}$-simulation terminates for ECA: for every sequence $Z_0, Z_1, \ldots$ of event-zones that are *reachable* during a zone enumeration of an ECA, there exist $i < j$ such that Z_j is simulated by Z_i. This is a notable difference to the existing methods in TA, where finiteness is guaranteed for all zones, not only the reachable zones. In the ECA case, this is not true: we can construct an infinite sequence of event-zones which are incomparable with respect to the new $\mathcal{G}$-simulation. However, we show that finiteness does hold when restricting to reachable zones, and this is sufficient to prove termination of the zone enumeration algorithm. Our argument involves identifying some crucial invariants in reachable zones, specially, involving the prophecy clocks.

The fundamental differences in the behaviour of prophecy clocks as compared to usual clocks constitute the major challenge in developing efficient procedures for the analysis of ECAs. In our work, we have developed methods to incorporate prophecy clocks alongside the usual clocks. We prove a surprising property: in all *reachable* event-zones, the constraints involving *prophecy* clocks come from a finite set. A direct consequence of this observation is that the event-zone graph of an ECA containing only prophecy clocks (known as Event-Predicting Automata (EPA)) is always finite. This observation is similar in spirit to an early work on automata with timers [Dil89], whose symbolic analysis was shown to terminate without additional constructions.

A preliminary version of this paper appeared in the conference proceedings [AGGS22]. The current version includes cleaner proofs and additional intuitions. Most importantly, we identify a mistake in the conference version and fix it here. Event-clocks can sometimes take *undefined values* – for instance, before seeing the first a, the history clock recording the time since the previous a is undefined, and similarly after seeing the last a, the prophecy clock predicting the next occurrence to a is undefined. One of the key contributions of this work is the novel mechanism to represent these undefined values using $+\infty$ and $-\infty$ and seamlessly integrating the manipulation of these quantities along with the finite (real) values. This required the definition of a new algebra to handle weights in the graph representation of event-zones. In the conference version, we had wrongly claimed that an event-zone is empty iff its graph representation has a negative cycle. We explain in Section 4 that this is not true as it is stated, due to the subtle interplay between finite and infinity weights. We fix this by introducing a *standard form* for the graph representation, and a *normal form* that is obtained from graphs in standard form. These changes have a significant impact on many of the proofs in the later sections and lead to a clearer presentation. As a result we are able to avoid certain cases, while in other places we use our new lemmas to get more succinct proofs.

As a follow-up to [AGGS22], an extended model of *Generalized Timed Automata* has been proposed, implemented in the open source tool TChecker, and is publicly available [AGG$^+$23]. This model subsumes event-clock automata, and also contains diagonal constraints. Reachability is undecidable for this model, and a decidable fragment has been identified. The decidable fragment strongly subsumes event-clock automata, and requires further sophisticated analysis of the objects that we develop in this document.

Organization of the paper. Section 2 recalls ECA and describes a slightly modified presentation of the ECA semantics. Section 3 introduces event-zones, event zone graph and the simulation-based reachability framework. Section 4 introduces the new algebra for representing event-zones and describes some operations needed to build the zone graph. Section 5 introduces the $\mathcal{G}$-simulation for event-clock automata and gives the simulation test. Section 6 proves finiteness of the simulation when restricted to reachable zones.

2. Event-Clock Automata and Valuations

Let X be a finite set of real-valued variables called *clocks*. Let $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, +\infty\}$ denote the set of all real numbers along with $-\infty$ and $+\infty$. The usual $<$ order on reals is extended to deal with $\{-\infty, +\infty\}$ as: $-\infty < c$, $c < +\infty$ for all $c \in \mathbb{R}$ and $-\infty < +\infty$. Similarly, $\overline{\mathbb{Z}} = \mathbb{Z} \cup \{-\infty, +\infty\}$ denotes the set of all integers along with $-\infty$ and $+\infty$. Let $\mathbb{R}_{\geq 0}$ (resp. $\mathbb{R}_{\leq 0}$) be the set of non-negative (resp. non-positive) reals. Let $\Phi(X)$ denote a set of clock constraints generated by the following grammar: $\varphi ::= x \lhd c \mid c \lhd x \mid \varphi \wedge \varphi$ where $x \in X$, $c \in \overline{\mathbb{Z}}$ and $\lhd \in \{<, \leq\}$. The base constraints of the form $x \lhd c$ and $c \lhd x$ will be called *atomic constraints*. Constraints $x < -\infty$ and $+\infty < x$ are equivalent to *false* and constraints $-\infty \leq x$ and $x \leq +\infty$ are equivalent to *true*.

Given a finite alphabet Σ, we define a set $X_H = \{\overleftarrow{a} \mid a \in \Sigma\}$ of *history clocks* and a set $X_P = \{\overrightarrow{a} \mid a \in \Sigma\}$ of *prophecy clocks*. Together, history and prophecy clocks are called *event-clocks*. In this paper, all clocks will be event-clocks, thus we set $X = X_H \cup X_P$.

Definition 2.1 (Valuation). A valuation of event-clocks is a function $v: X \mapsto \overline{\mathbb{R}}$ which maps history clocks to $\mathbb{R}_{\geq 0} \cup \{+\infty\}$ and prophecy clocks to $\mathbb{R}_{\leq 0} \cup \{-\infty\}$. We say a history clock $\overleftarrow{a}$, for some $a \in \Sigma$ is *undefined* (resp. *defined*) when $v(\overleftarrow{a}) = +\infty$ (resp. $v(\overleftarrow{a}) < +\infty$) and a prophecy clock $\overrightarrow{a}$ is undefined (resp. defined) when $v(\overrightarrow{a}) = -\infty$ (resp. $-\infty < v(\overrightarrow{a})$). We denote by $\mathbb{V}(X)$ or simply by $\mathbb{V}$ the set of valuations over X.

We remark that the history clock and the prophecy clock of an event a are symmetric notions, as illustrated in Figure 2. In the semantics that we introduce in this paper, history clock $\overleftarrow{a}$ stores the amount of time elapsed after seeing the last a, measuring how far ahead in the future we are w.r.t. the last occurrence of a. Before we

$$-\infty \longleftarrow \underset{\overrightarrow{a}}{\bullet} \quad \underset{0}{\bullet} \quad \underset{\overleftarrow{a}}{\bullet} \longrightarrow +\infty$$

Figure 1: Valuation of event-clocks.

see an a for the first time, $\overleftarrow{a}$ is set to $+\infty$. The prophecy clock $\overrightarrow{a}$ stores the negative of the amount of time that needs to be elapsed before seeing the next a. In other words, $-\overrightarrow{a}$ tells us how far behind in the past we are w.r.t. the next occurrence of a. If no more a's are going to be seen, then the prophecy clock of a is set to $-\infty$, i.e., $\overrightarrow{a} = -\infty$.

Notice that for history (resp. prophecy) clocks, *useful* constraints use non-negative (resp. non-positive) constants. Also, $\overleftarrow{a} < 0$ and $0 < \overrightarrow{a}$ are equivalent to *false* whereas $0 \leq \overleftarrow{a}$, $\overleftarrow{a} \leq +\infty$, $\overrightarrow{a} \leq 0$ and $-\infty \leq \overrightarrow{a}$ are equivalent to *true*. A constraint $c \lhd \overleftarrow{a}$ does not imply that the history clock $\overleftarrow{a}$ is defined, whereas a constraint $\overleftarrow{a} \lhd c$ with $(\lhd, c) \neq (\leq, +\infty)$ does. The same applies to prophecy clocks where a constraint $c \lhd \overrightarrow{a}$ with $(c, \lhd) \neq (-\infty, \leq)$ implies that $\overrightarrow{a}$ is defined, whereas $\overrightarrow{a} \lhd c$ does not; in fact, $\overrightarrow{a} \leq -\infty$ states that $\overrightarrow{a}$ is undefined.

Remark 2.2. In the earlier works on ECA [AFH99, GRS14], prophecy clocks assumed non-negative values and decreased along with time. This allowed to write guards on prophecy clocks with non-negative constants, e.g., $\overrightarrow{a} \leq 5$ means that the next a occurs in at most 5

time units. In our convention, this would be written as $-5 \leq \overrightarrow{a}$. Secondly, an undefined clock (history or prophecy) was assigned a special symbol $\perp$ in earlier works. We have changed this to use $-\infty$ and $+\infty$ for undefined prophecy and history clocks respectively. We adopt these new conventions as they allow to treat both history clocks and prophecy clocks in a symmetric fashion, and a clean integration of undefined values when we describe zones and simulations.

We say that a valuation v satisfies a constraint φ, denoted as $v \models \varphi$, if φ evaluates to *true*, when each variable x in φ is replaced by its value $v(x)$.

We write $[\overleftarrow{a}]v$ to denote the valuation v' obtained from v by resetting the history clock $\overleftarrow{a}$ to 0, keeping the value of other clocks unchanged. We denote by $[\overrightarrow{a}]v$ the *set* of valuations v' obtained from v by setting the prophecy clock $\overrightarrow{a}$ non-deterministically to some value in $[-\infty, 0]$, keeping the value of other clocks unchanged. We see $[\overrightarrow{a}]v$ as the result of an operation $[\overrightarrow{a}]$ on valuation v, which we call the *release* of a from v. The idea is that v maintains an exact value for the next occurrence of a. On releasing a from v, this value is forgotten and a fresh guess is made for the next a. This fresh guess has no constraints and can take any value between $-\infty$ and 0. As an example, consider a valuation v over two events a and b, with $v(\overrightarrow{a}) = 0, v(\overrightarrow{b}) = -2$, and some arbitrary values for the history clocks $v(\overleftarrow{a})$ and $v(\overleftarrow{b})$. This valuation v predicts that the next a will occur immediately, whereas b will occur in two time units from now. By releasing a from v, we get a set $[\overrightarrow{a}]v$ with valuations containing the different possible orders between the next occurrences of a and b: for instance, pick $v_1, v_2 \in [\overrightarrow{a}]v$ with $v_1(\overrightarrow{a}) = -3$, $v_2(\overrightarrow{a}) = -0.5$. In both the valuations, the value of $\overrightarrow{b}$ retains the value -2 from v. Therefore, valuation v_1 predicts that a will occur after b, whereas v_2 predicts that a will occur before b. There is also a valuation $v_3 \in [\overrightarrow{a}]v$ with $v_3(\overrightarrow{a}) = -\infty$. This valuation says that a will never occur from hereon.

The next operation on valuations is that of a *time elapse*. We denote by $v + \delta$ the valuation obtained by increasing the value of all clocks from the valuation v by $\delta \in \mathbb{R}_{\geq 0}$. The time elapse operation is illustrated in Figure 2. Not every time elapse may be possible from a valuation, since prophecy clocks need to stay at most 0. For example, if there are two events a, b, then a valuation with $v(\overrightarrow{a}) = -3$ and $v(\overrightarrow{b}) = -2$ can elapse at most 2 time units.

Definition 2.3 (Event-clock automata [AFH99])**.** An *event-clock automaton (ECA)* $\mathcal{A}$ is given by a tuple $(Q, \Sigma, X, T, q_0, F)$, where Q is a finite set of states, Σ is a finite alphabet of actions, $X = X_H \cup X_P$ is the set of event clocks for Σ, $q_0 \in Q$ is the initial state, $F \subseteq Q$ is the set of accepting states and $T \subseteq Q \times \Sigma \times \Phi(X) \times Q$ is a finite set of transitions.
The semantics of an ECA $\mathcal{A} = (Q, \Sigma, X, T, q_0, F)$ is given by a transition system $S_\mathcal{A}$ whose states are *configurations* (q, v) of $\mathcal{A}$, where $q \in Q$ and v is a valuation. A configuration (q, v) is initial if $q = q_0$, $v(x) = +\infty$ for all $x \in X_H$. A configuration (q, v) is accepting if $q \in F$, and $v(x) = -\infty$ for all $x \in X_P$. Transitions of $S_\mathcal{A}$ are of two forms:

- **Delay transition**: $(q, v) \xrightarrow{\delta} (q, v + \delta)$, if $(v + \delta)(x) \leq 0$ for all $x \in X_P$.
- **Action transition**: $(q, v) \xrightarrow{t} (q', [\overleftarrow{a}]v')$ if $t = (q, a, g, q')$ is a transition in $\mathcal{A}$, $v(\overrightarrow{a}) = 0$, $v' \in [\overrightarrow{a}]v$ and $v' \models g$.

A transition with action a can be taken when the value of the prophecy clock $\overrightarrow{a}$ is 0, then a new value in $[-\infty, 0]$ for $\overrightarrow{a}$ is non-deterministically guessed so that the resulting valuation v' satisfies the guard g, and finally, the history clock $\overleftarrow{a}$ is reset to 0.

Figure 2: Representation of valuations in event-clock automata. Here, $v' = v + \delta$.

Remark 2.4. In the semantics for ECA defined above, we say that a configuration is accepting only if $v(x) = -\infty$ for all prophecy clocks. Analogously, for initial configurations, we ask that all the history clocks are set to $+\infty$.

Here is an example to illustrate the transition semantics. A transition $t = (q, a, g, q')$ with guard $g := (\overleftarrow{a} \leq 5) \wedge (-3 \leq \overrightarrow{a}) \wedge (\overrightarrow{b} \leq -\infty)$ can be taken if the previous a was seen within 5 units, the next a will be seen within 3 units and there will be no b anymore. A valuation v can take this transition provided $v(\overleftarrow{a}) \leq 5$, $v(\overrightarrow{b}) = -\infty$ and $v(\overrightarrow{a}) = 0$. At this point, the guard $-3 \leq \overrightarrow{a}$ is not used. This constraint plays a role in the value to which $\overrightarrow{a}$ is released to: the only possible values of $\overrightarrow{a}$ after taking the transition lie in the interval $[-3, 0]$. More precisely, if $(q, v) \xrightarrow{t} (q', [\overleftarrow{a}]v')$, then $v'(\overrightarrow{a}) \in [-3, 0]$. Notice that in $[\overleftarrow{a}]v'$, the value of the history clock $\overleftarrow{a}$ is set to 0 to record the information that a was just seen. Figure 3 gives an example of an ECA. Intuitively, this ECA recognizes sequences of the form b^+a where there exists a b (not necessarily the last one) at time 1 before the final letter a.

An ECA is called an event-recording automaton (ERA) if it only contains history clocks and event-predicting automaton (EPA) if it only contains prophecy clocks. A *run* of an event-clock automaton is a finite sequence of transitions from an initial configuration of $S_\mathcal{A}$. A run is said to be *accepting* if its last configuration is accepting. We are interested in the *reachability problem* of an event-clock automaton. Formally,

Definition 2.5 (Reachability problem for ECA). The reachability problem for an event-clock automaton $\mathcal{A}$ is to decide whether $\mathcal{A}$ has an accepting run.

Different solutions based on regions and zones have been proposed in [AFH99, GRS11, GRS14]. For ERA, the standard region and zone-based algorithms for timed automata work directly. However, for EPA (and ECA), this is not the case. In fact, [GRS11] show that the standard region abstraction is not possible, as there exists no finite bisimulation due to the behavior of prophecy clocks. We provide an intuitive reasoning for this fact. Pick some $\alpha \geq 0$. From a valuation v_α with $v_\alpha(\overrightarrow{a}) = -\alpha$ and $v_\alpha(\overrightarrow{b}) = 0$, we can see $\lfloor \alpha \rfloor$ (the greatest integer smaller than or equal to α) occurrences of b, if we assume that the time between consecutive b's is 1 (see Figure 3 of [GRS14] or Figure 11 for a detailed example).

This creates a difference between v_α, and say $v_{\alpha+1}$, since there is an extra occurrence of b possible from $v_{\alpha+1}$. Therefore, there is no constant $M \geq 0$ such that we can put all valuations v_α with $\alpha < -M$ into one "equivalence class". In other words, there is no M for which we can forget the actual value of $\overrightarrow{a}$ as long as it is less than $-M$. This is in sharp contrast with history clocks (and classical timed automata) where once a clock goes beyond a maximum constant M, its actual value can be forgotten. In some sense, time elapse moves history clocks above some M only further away from M, whereas for prophecy clocks time elapse brings clocks which are $< -M$ closer to $-M$ and at some point they cross $-M$ and are no longer $< -M$. This creates a fundamental issue causing the impossibility of finite bisimulations. Also, the standard definition of zones used for timed automata is not sufficient to handle valuations with undefined clocks. The papers [GRS11, GRS14] make use of special symbols $\bot$ and ? for this purpose. In this work, we use a different formulation of zones by making use of $+\infty$ and $-\infty$. Instead of using $x = \bot$ (resp. $x \neq \bot$) to state that a clock is undefined (resp. defined) as in [GRS11, GRS14], we write $+\infty \leq x$ or $x \leq -\infty$ (resp. $x < +\infty$ or $-\infty < x$) depending on whether x is a history clock or a prophecy clock. This distinction between being undefined for history and prophecy clocks plays an important role.

3. Event-zones and simulation-based reachability

The most widely used approach for checking reachability in a timed automaton is based on reachability in a graph called the *zone graph* of a timed automaton [DT98]. Roughly, *zones* [BY03] are sets of valuations that can be represented efficiently using constraints between differences of clocks. In this section, we introduce an analogous notion for event-clock automata. We consider *event zones*, which are special sets of valuations of event-clock automata.

Definition 3.1 (Weights). Let $\mathcal{C} = \{(\lhd, c) \mid c \in \overline{\mathbb{R}} \text{ and } \lhd \in \{\leq, <\}\}$, called the set of weights.

We will use constraints of the form $y - x \lhd c$ with $(\lhd, c) \in \mathcal{C}$ and $x, y \in X \cup \{0\}$ (event-clocks are extended with the special constant clock 0, meaning that $v(0) = 0$ for all valuations $v \in \mathbb{V}$). Such constraints, between two clocks are sometimes called diagonal constraints. The introduction of the special constant clock allows us to treat constraints with just a single clock (sometimes referred to as non-diagonal constraints) as special cases. Indeed, for all weights $(\lhd, c) \in \mathcal{C}$, the constraint $x \lhd c$ is equivalent to $x - 0 \lhd c$ and the constraint $c \lhd x$ is equivalent to $0 - x \lhd -c$.

To evaluate such constraints, we extend addition on real numbers with the convention that $(+\infty) + \alpha = \alpha + (+\infty) = +\infty$ for all $\alpha \in \overline{\mathbb{R}}$ and $(-\infty) + \beta = \beta + (-\infty) = -\infty$, as long as $\beta \neq +\infty$. We also extend the unary minus operation from real numbers to $\overline{\mathbb{R}}$ by setting $-(+\infty) = -\infty$ and $-(-\infty) = +\infty$. Abusing notation, we write $\beta - \alpha$ for $\beta + (-\alpha)$. Notice that with this definition of extended addition, the minus operation does not distribute over addition.[1] We now highlight a few more important features of this definition.

Remark 3.2. This extended addition has the following properties that are easy to check:
(1) $(\overline{\mathbb{R}}, +, 0)$ is a monoid with 0 as neutral element. In particular, the extended addition is associative.

---

[1] Notice that $-(a + b) = (-a) + (-b)$ when a or b is finite or when $a = b$. But, when $a = +\infty$ and $b = -\infty$ then $-(a + b) = -\infty$ whereas $(-a) + (-b) = +\infty$.

(2) $(\overline{\mathbb{R}}, +, 0)$ is not a group, since $-\infty$ and $+\infty$ have no opposite values. Note that, $\alpha + (-\alpha) = 0$ when $\alpha \in \mathbb{R}$ is finite but $\alpha + (-\alpha) = +\infty$ when $\alpha \in \{-\infty, +\infty\}$. As a consequence, in an equation $\alpha + \beta = \alpha + \gamma$, we can cancel α and deduce $\beta = \gamma$ when α is finite, but not when α is infinite.

(3) The order $\leq$ is monotone on $\overline{\mathbb{R}}$: $b \leq c$ implies $a + b \leq a + c$, but the converse implication only holds when a is finite.

(4) The strict order $<$ is only monotone with respect to finite values: when a is finite, $b < c$ iff $a + b < a + c$.

(5) For all $a, b \in \overline{\mathbb{R}}$ and $(\lhd, c) \in \mathcal{C}$, we have $a \lhd b$ iff $-b \lhd -a$. Further, $a - b \lhd c$ implies $a \lhd b + c$. The converse of the latter statement holds when b is finite, but may be false when b is infinite.[2]

Definition 3.3. Let $x, y \in X \cup \{0\}$ be event-clocks (including 0) and $(\lhd, c) \in \mathcal{C}$ be a weight. For valuations $v \in \mathbb{V}$, define $v \models y - x \lhd c$ as $v(y) - v(x) \lhd c$.

Remark 3.4. From Definition 3.3, we easily check that the constraint $y - x \lhd c$ is equivalent to *true* (resp. *false*) when $(\lhd, c) = (\leq, +\infty)$ (resp. $(\lhd, c) = (<, -\infty)$): for all valuations $v \in \mathbb{V}$, $v \models y - x \leq +\infty$ and $v \not\models y - x < -\infty$. Constraints that are equivalent to *true* or *false* will be called trivial, whereas all others are non-trivial constraints.

If $(\lhd, c) \neq (\leq, +\infty)$ then $v \models y - x \lhd c$ never holds when $v(x) = -\infty$.

Also, if $v(x) = v(y) \in \{-\infty, +\infty\}$ then $v \models y - x \lhd c$ only holds for $(\lhd, c) = (\leq, +\infty)$.

Consider now a non-trivial constraint $y - x \lhd c$ with weight $(\lhd, c) \in \mathcal{C} \setminus \{(<, -\infty), (\leq, +\infty)\}$. We have $v \models y - x \lhd c$ iff $v(y) < +\infty = v(x)$ or $(v(x)$ is finite and $v(y) \lhd v(x) + c)$.

Let us consider special cases of Definition 3.3.

- $v \models y - x \leq -\infty$ iff $v(y) < +\infty = v(x)$ or $v(y) = -\infty < v(x)$.
- $v \models y - x < +\infty$ iff $v(x) \neq -\infty$ and $v(y) \neq +\infty$. $\square$

Definition 3.5 (**Event-zones**). An event-zone is a set of valuations satisfying a conjunction of constraints of the form $y - x \lhd c$, where $x, y \in X \cup \{0\}$, $c \in \overline{\mathbb{Z}} = \mathbb{Z} \cup \{-\infty, +\infty\}$ and $\lhd \in \{\leq, <\}$.

Let W be an arbitrary set of valuations (not necessarily an event-zone) and q be a state. For transition $t := (q, a, g, q_1)$, we write $(q, W) \xrightarrow{t} (q_1, W_t)$ if $W_t = \{v_1 \mid v_1$ is a valuation and $(q, v) \xrightarrow{t} \xrightarrow{\delta} (q_1, v_1)$ for some $v \in W$ and $\delta \in \mathbb{R}_{\geq 0}\}$. Notice that we have action $\xrightarrow{t}$ followed by delay $\xrightarrow{\delta}$ in this definition, a common convention adopted in the timed automata literature. This results in the set W_t being closed under time-successors. We will show in the next section that starting from an event-zone Z, the successors are also event-zones: $(q, Z) \xrightarrow{t} (q_1, Z_t)$ implies Z_t is an event-zone too. We use this feature to define an event-zone graph.

Definition 3.6 (**Event-zone graph**). Given an ECA $\mathcal{A}$, its *event-zone graph, denoted* EZG$(\mathcal{A})$, is defined as follows: Nodes are of the form (q, Z) where q is a state and Z is an event-zone. The initial node is (q_0, Z_0) where q_0 is the initial state and Z_0 is given by $\bigwedge_{a \in \Sigma} \left((+\infty \leq \overleftarrow{a}) \wedge (\overrightarrow{a} \leq 0) \right)$. This is the set of all initial valuations, which is already closed under time elapse[3]. For every node (q, Z) and every transition $t := (q, a, g, q_1)$ there

---

[2]For instance, if $a < +\infty = b$ then $a < b + (-\infty)$, but $a - b = -\infty \not< -\infty$.

If $a < +\infty$ and $b = -\infty$ then $a < b + \infty$, but $a - b = +\infty \not< +\infty$.

If $a = b \in \{-\infty, +\infty\}$ and c is finite then $a \leq b + c$, but $a - b = +\infty \not\leq c$.

[3]By definition, the time elapse operation only permits delays that keep prophecy clocks non-positive.

is a transition $(q, Z) \xrightarrow{t} (q_1, Z_t)$ in the event-zone graph. A node (q, Z) is accepting if $q \in F$ and $Z \cap Z_f$ is non-empty where the final zone Z_f is defined by $\bigwedge_{a \in \Sigma} \overrightarrow{a} \leq -\infty$.

An example of ECA with its event-zone graph is given in Figure 3. We use some shorthands when writing some constraints. For instance, we write $\overleftarrow{b} - \overrightarrow{a} = 1$ for the conjunction of $\overleftarrow{b} - \overrightarrow{a} \leq 1$ and $\overrightarrow{a} - \overleftarrow{b} \leq -1$.[4] Another example is $0 \leq \overleftarrow{b} - \overrightarrow{a} \leq 1$ which stands for the conjunction of $\overleftarrow{b} - \overrightarrow{a} \leq 1$ and $\overrightarrow{a} - \overleftarrow{b} \leq 0$.

Figure 3: The event-clock automaton $\mathcal{A}_1$ recognizes the language $\{b^n a \mid n \geq 1\}$ such that there exists some b which occurs exactly one time unit before a. Its event-zone graph is on the left.

Similar to the case of timed automata, the event-zone graph can be used to decide reachability. The next lemma follows by a straightforward adaptation of the corresponding proof [DT98] from timed automata.

Proposition 3.7. *The event-zone graph of an ECA is sound and complete for reachability.*

However, as in the case of zone graphs for timed automata, the event-zone graph for an ECA is also not guaranteed to be finite. In fact, in Geeraerts et al. [GRS11, GRS14], it was shown that there are ECAs for which there is no finite time-abstract bisimulation relation on valuations, e.g., the ECA $\mathcal{A}_2$ in Figure 11.

In the rest of this section, we will define what a simulation is and see how it can be used to get a finite truncation of the event-zone graph for ECAs, which is still sound and complete for reachability.

Definition 3.8 (Simulation). A simulation relation on the semantics of an ECA is a reflexive, transitive relation $(q, v) \preceq (q, v')$ relating configurations with the same control state

---

[4]Notice that for $a, b, c \in \overline{\mathbb{R}}$ and $\lhd \in \{\leq, <\}$, if $a - b \lhd c$ then $-c \lhd b - a$. The converse is true when a, b, c are finite, but not in general. For instance, when $a = b \in \{-\infty, +\infty\}$ and the weight $(\lhd, c) \in \mathcal{C} \setminus \{(<, -\infty), (\leq, +\infty)\}$ is non-trivial, then $-c \lhd b - a = +\infty$ but $a - b = +\infty \not\lhd c$.

and (1) for every $(q, v) \xrightarrow{\delta} (q, v + \delta)$, we have $(q, v') \xrightarrow{\delta} (q, v' + \delta)$ and $(q, v + \delta) \preceq (q, v' + \delta)$, (2) for every transition t, if $(q, v) \xrightarrow{t} (q_1, v_1)$ for some valuation v_1, then $(q, v') \xrightarrow{t} (q_1, v_1')$ for some valuation v_1' with $(q_1, v_1) \preceq (q_1, v_1')$.

For two event-zones Z, Z', we say $(q, Z) \preceq (q, Z')$ if for every $v \in Z$ there exists $v' \in Z'$ such that $(q, v) \preceq (q, v')$. Adopting the terminology from [GMS18, GMS19, GMS20], we say that the simulation $\preceq$ is *finite* if for every sequence $(q_1, Z_1), (q_2, Z_2), \ldots$ of *reachable* nodes (i.e., those having a path from initial node), there exists $j > i$ such that $(q_j, Z_j) \preceq (q_i, Z_i)$. That is, the converse or transpose of $\preceq$ is a well-quasi order over reachable nodes, and thus the set of downward closed sets is finite.

The reachability algorithm enumerates the nodes of the event-zone graph and uses $\preceq$ to truncate nodes that are smaller with respect to the simulation.

Definition 3.9 (**Reachability algorithm**). Let $\mathcal{A}$ be an ECA and $\preceq$ a finite simulation for $\mathcal{A}$. Add the initial node of the event-zone graph (q_0, Z_0) to a Waiting list. Repeat the following until Waiting list is empty:

- Pop a node (q, Z) from the Waiting list and add it to the Passed list.
- For every $(q, Z) \xrightarrow{t} (q_1, Z_1)$: if there exists a (q_1, Z_1') in the Passed or Waiting lists such that $(q_1, Z_1) \preceq (q_1, Z_1')$, discard (q_1, Z_1); else add (q_1, Z_1) to the Waiting list.

If some accepting node is reached, the algorithm terminates and returns a Yes. Else, it continues until there are no further nodes to be explored and returns a No answer.

The correctness of the reachability algorithm follows from the fact that $\preceq$ is a simulation relation. Moreover, termination is guaranteed when the simulation used is finite.

Theorem 3.10. *An ECA has an accepting run iff the reachability algorithm returns Yes.*

We have now presented the framework for the simulation approach in its entirety. However, to make it functional, we will need the following.

(1) An efficient representation for event-zones and algorithms to compute successors.
(2) A concrete simulation relation $\preceq$ for ECA with an efficient simulation test $(q, Z) \preceq (q, Z')$.
(3) A proof that $\preceq$ is finite, to guarantee termination of the reachability algorithm.

In the rest of the paper, we show how these can be achieved. To start with, for standard timed automata, zones are represented using Difference-Bound-Matrices (DBMs) [Dil89]. For such a representation to work on event-zones, we will need to incorporate the fact that valuations can now take $+\infty$ and $-\infty$. In Section 4, we propose a way to merge $+\infty$ and $-\infty$ seamlessly into the DBM technology. In the subsequent Section 5, we define a simulation for ECA based on $\mathcal{G}$-simulation, develop some technical machinery and present an efficient simulation test. Finally, in Section 6, we deal with the main problem of showing finiteness. For this, we prove some non-trivial invariants on the event-zones that are reachable in ECA and use them to show a surprising property regarding prophecy clocks. More precisely, we show that constraints involving prophecy clocks in reachable event-zones come from a finite set depending on the maximal constant of the ECA only.

4. Computing with event-zones and distance graphs

We now show that event-zones can be represented using Difference-Bound-Matrices (DBMs) and the operations required for the reachability algorithm can be implemented using DBMs.

Each entry in a DBM encodes a constraint of the form $x - y \triangleleft c$. For timed automata analysis, the entries could be $(<, +\infty)$ or $(\triangleleft, c)$ with $c \in \mathbb{R}$ and $\triangleleft \in \{<, \leq\}$. For ECA, we need to deal with valuations $+\infty$ or $-\infty$. For this purpose, we use more general weights as introduced in Definition 3.1 and we extend the algebra of weights to the new entries in a natural way. However, we need to rework the basic results on DBMs in our extended setting, since as we have seen in Remark 3.2, some seemingly obvious properties that hold for finite weights do not carry over to the extended weights. It turns out that even with the restricted versions of the properties that continue to hold, as listed in Remark 3.2, we can achieve the same results in the extended DBMs. This is the subject of this section.

4.1. Extending the algebra on weights.

Definition 4.1 (Order and sum of weights). Let $(\triangleleft, c), (\triangleleft', c') \in \mathcal{C}$ be weights.
Order. Define $(\triangleleft, c) < (\triangleleft', c')$ when either (1) $c < c'$, or (2) $c = c'$ and $\triangleleft$ is $<$ while $\triangleleft'$ is $\leq$. This is a total order with $(<, -\infty) < (\leq, -\infty) < (\triangleleft, c) < (<, +\infty) < (\leq, +\infty)$ for all $c \in \mathbb{R}$.
Sum. We define the *commutative* sum operation as follows.

$$
\begin{aligned}
(<, -\infty) + \alpha &= (<, -\infty) & &\text{if } \alpha \in \mathcal{C} \\
(\leq, +\infty) + \alpha &= (\leq, +\infty) & &\text{if } \alpha \in \mathcal{C} \setminus \{(<, -\infty)\} \\
(\leq, -\infty) + \alpha &= (\leq, -\infty) & &\text{if } \alpha \in \mathcal{C} \setminus \{(<, -\infty), (\leq, +\infty)\} \\
(<, +\infty) + \alpha &= (<, +\infty) & &\text{if } \alpha \in \mathcal{C} \setminus \{(<, -\infty), (\leq, -\infty), (\leq, +\infty)\} \\
(\triangleleft, c) + (\triangleleft', c') &= (\triangleleft'', c + c') & &\text{if } c, c' \in \mathbb{R} \text{ and } \triangleleft'' = \leq \text{ if } \triangleleft = \triangleleft' = \leq \text{ and } \triangleleft'' = < \text{ otherwise.}
\end{aligned}
$$

Notice that the sum of weights is an associative operation and $\alpha + (\leq, 0) = \alpha$ for all $\alpha \in \mathcal{C}$.

Remark 4.2. Note that our set of weights $\mathcal{C}$ has a sequence of partially absorbant elements with decreasing strength - $(<, -\infty)$ is an absorbant for $\mathcal{C}$, $(\leq, +\infty)$ is an absorbant for $\mathcal{C} \setminus \{(<, -\infty)\}$, $(\leq, -\infty)$ is an absorbant for $\mathcal{C} \setminus \{(<, -\infty), (\leq, +\infty)\}$, and $(<, +\infty)$ is an absorbant for $\mathcal{C} \setminus \{(<, -\infty), (\leq, +\infty), (\leq, -\infty)\}$. This implies that all the sums are well-defined.

The intuition behind the above definition of order is that when $(\triangleleft, c) < (\triangleleft', c')$, the set of valuations that satisfies a constraint $y - x \triangleleft c$ is contained in the solution set of $y - x \triangleleft' c'$. For the sum, the following lemma gives the idea behind our choice of definition.

Lemma 4.3. *Let* $x, y, z \in X \cup \{0\}$ *be event-clocks,* $(\triangleleft_1, c_1), (\triangleleft_2, c_2) \in \mathcal{C}$ *be weights and* $(\triangleleft, c) = (\triangleleft_1, c_1) + (\triangleleft_2, c_2)$. *For all valuations* $v \in \mathbb{V}$, *if* $v \models y - x \triangleleft_1 c_1$ *and* $v \models z - y \triangleleft_2 c_2$, *then* $v \models z - x \triangleleft c$.

Proof. When $(\triangleleft, c) \in \{(<, -\infty), (\leq, +\infty)\}$, this is clear. We assume for the rest of the proof that $(\triangleleft, c) \in \mathcal{C} \setminus \{(<, -\infty), (\leq, +\infty)\}$ is non-trivial. Consider a valuation $v \in \mathbb{V}$ such that $v \models y - x \triangleleft_1 c_1$ and $v \models z - y \triangleleft_2 c_2$.

From $v(y) - v(x) \triangleleft_1 c_1$ and $(\triangleleft_1, c_1) \neq (\leq, +\infty)$ we deduce that $v(y) \neq +\infty$ and $v(x) \neq -\infty$. Similarly, from $v(z) - v(y) \triangleleft_2 c_2$ and $(\triangleleft_2, c_2) \neq (\leq, +\infty)$ we deduce that $v(z) \neq +\infty$ and $v(y) \neq -\infty$. Hence, $v(y)$ is finite. If $v(x) = +\infty$, or $v(x) < +\infty$ and $v(z) = -\infty$ then $v(z) - v(x) = -\infty$, hence $v \models z - x \triangleleft c$.

The remaining case is when $v(x), v(y), v(z)$ are all finite. This rules out $(\triangleleft, c) = (\leq, -\infty)$. If $(\triangleleft, c) = (<, +\infty)$ then $v \models z - x \triangleleft c$. Otherwise, $c = c_1 + c_2$ is finite and $v(z) - v(x) = v(z) - v(y) + v(y) - v(x) \triangleleft c_2 + c_1$. This completes the proof. $\qquad\square$

The following technical lemma will be useful in many proofs.

Lemma 4.4.
(1) *Let $(\lhd, c)$ be a weight and $\alpha \in \overline{\mathbb{R}}$. Then,*
   - $\alpha \lhd c$ *iff* $(\leq, \alpha) \leq (\lhd, c)$ *iff* $(\leq, 0) \leq (\leq, -\alpha) + (\lhd, c)$,
   - $\alpha \not\lhd c$ *iff* $(\lhd, c) < (\leq, \alpha)$ *iff* $(\leq, -\alpha) + (\lhd, c) < (\leq, 0)$ *iff* $(\leq, -\alpha) + (\lhd, c) \leq (<, 0)$.
(2) *Let $(\lhd, c), (\lhd', c'), (\lhd'', c'')$ be weights with $(\leq, 0) \leq (\lhd, c) + (\lhd', c')$.*
   Then, there exists $\alpha \in \overline{\mathbb{R}}$ such that $\alpha \lhd c$ and $-\alpha \lhd' c'$.
   If in addition we have $(\lhd'', c'') < (\lhd, c)$ then there exists such an α with $\alpha \not\lhd'' c''$.

Proof.
(1) The first item is easy to prove, checking separately the cases where α or c is infinite. The second item follows from the first one using the fact that $\leq$ is a total order on weights.
(2) Assume first that $\lhd = \leq$. Take $\alpha = c$. Using 1, we easily check that α satisfies the desired properties.

  Next, assume that $\lhd = <$. From $(\leq, 0) \leq (\lhd, c) + (\lhd', c')$ we deduce that $c \neq -\infty \neq c'$. Next, we see that $-c' < c$ (consider separately the cases $c = +\infty$ or $c' = +\infty$). Take α such that $-c' < \alpha < c$. We obtain $\alpha \lhd c$ and $-\alpha < c'$. The latter inequality implies $-\alpha \lhd' c'$.

  If in addition we have $(\lhd'', c'') < (<, c)$, then $c'' < c$ and we take α such that $\max(-c', c'') < \alpha < c$. As above, we have $\alpha \lhd c$ and $-\alpha \lhd' c'$. We also get $\alpha \not\lhd'' c''$. $\square$

Equipped with the weights and the arithmetic over it, we will work with a graph representation of zones (called distance graphs), instead of matrices (i.e., DBMs), since this makes the analysis more convenient. We wish to highlight that our definition of weights, order and sum have been chosen to ensure that this notion of distance graphs remains identical to the one for usual TA. As a consequence, we are able to adapt many of the well-known properties about distance graphs for ECA.

4.2. Distance graphs over the extended algebra.

Definition 4.5 (Distance graphs). A distance graph $\mathbb{G}$ is a weighted directed graph without self-loops, with vertex set being $X \cup \{0\} = X_P \cup X_H \cup \{0\}$, edges being labeled with weights from $\mathcal{C} \setminus \{(<, -\infty)\}$.[5] We define $\llbracket \mathbb{G} \rrbracket := \{v \in \mathbb{V} \mid v \models y - x \lhd c \text{ for all edges } x \xrightarrow{\lhd c} y \text{ in } \mathbb{G}\}$. The weight of edge $x \to y$ is denoted $\mathbb{G}_{xy}$ and we set $\mathbb{G}_{xy} = (\leq, +\infty)$ if there is no edge $x \to y$.

We say that $\mathbb{G}$ is in standard form if it satisfies the following conditions;

(1) $\mathbb{G}_{0x} \leq (\leq, 0)$ for all $x \in X_P$ and $\mathbb{G}_{x0} \leq (\leq, 0)$ for all $x \in X_H$.
(2) For all $x, y \in X$, if $\mathbb{G}_{xy} \neq (\leq, +\infty)$ then $\mathbb{G}_{x0} \neq (\leq, +\infty)$ and $\mathbb{G}_{0y} \neq (\leq, +\infty)$.

We extend the order on weights to distance graphs pointwise: Let $\mathbb{G}, \mathbb{G}'$ be distance graphs, we write $\mathbb{G} \leq \mathbb{G}'$ when $\mathbb{G}_{xy} \leq \mathbb{G}'_{xy}$ for all edges $x \to y$. Notice that this implies $\llbracket \mathbb{G} \rrbracket \subseteq \llbracket \mathbb{G}' \rrbracket$.

Here is an example to illustrate the need for a standard form. Suppose that valuations are *finite* and that we have constraints: $y - x \leq 1$ and $-y \leq 2$. The distance graph representing these constraints is depicted with black edges in Figure 4. From these constraints, we can

---

[5]If we allowed an edge with weight $\mathbb{G}_{xy} = (<, -\infty)$ then we would get $\llbracket \mathbb{G} \rrbracket = \emptyset$ since the constraint $y - x < -\infty$ is equivalent to false.

Figure 4: An example of a distance graph for timed automata with clocks x, y. The dotted edge in black denotes that the edge is obtained by canonicalization, and the dotted edge in red depicts an edge introduced by a guard.

Figure 5: Distance graph with standardization using the extended algebra. Note that the green edges are the edges induced by standardization.

infer $-x \leq 3$ just by adding the inequalities - this corresponds to the edge depicted by dotted lines from x to 0. If there was another constraint $x \leq -4$ (denoted by the red dotted edge from 0 to x), we will get unsatisfiability, witnessed by a negative cycle $0 \xrightarrow{\leq -4} x \xrightarrow{\leq 3} 0$, as depicted by the dotted edges in Figure 4. Basically, adding the weights of $x \to y$ and $y \to 0$, we get the strongest possible constraint about $x \to 0$.

   This holds no more in the extended algebra, due to the fundamental difference while adding weight $(\leq, +\infty)$. Consider the constraint $y - x \leq 1$ corresponding to the edge $x \xrightarrow{\leq 1} y$ in the distance graph. Recall that missing edges correspond to the constraint $(\leq, +\infty)$ which is equivalent to *true*, hence we have implicit edges $x \xrightarrow{\leq \infty} 0$ and $0 \xrightarrow{\leq \infty} y$. These are *not* the strongest possible constraints on x and y induced by $y - x \leq 1$. Indeed, any valuation v satisfying $y - x \leq 1$ should have $v(y) \neq \infty$ and $v(x) \neq -\infty$. Now, if we also had a constraint $x - 0 \leq -\infty$, corresponding to edge $0 \xrightarrow{\leq -\infty} x$, we have no negative cycle in the corresponding distance graph, as can be seen by considering the part of the graph without green edges in Figure 5. But, as explained above, the set of constraints $y - x \leq 1$ and $x - 0 \leq -\infty$ is not satisfiable.

   In order to get a correspondence between negative cycles and empty solution sets, we propose the standard form. The standard form equips the graph with the additional information that when $y - x$ is bounded by a finite value, that is, edge $x \to y$ does not have weight $(\leq, +\infty)$, the constraints $0 - x$ and $y - 0$ are at most $(<, +\infty)$ - the edges induced

by standardization are depicted in green in Figure 5. With this information, we get negative cycles whenever there is a contradiction.

Now, each distance graph $\mathbb{G}$ can be transformed into an equivalent distance graph $\mathbb{G}'$ which is in standard form. By equivalent, we mean $[\![\mathbb{G}]\!] = [\![\mathbb{G}']\!]$. First, we set $\mathbb{G}'_{0x} = \min(\mathbb{G}_{0x}, (\leq, 0))$ for $x \in X_P$ and $\mathbb{G}'_{x0} = \min(\mathbb{G}_{x0}, (\leq, 0))$ for $x \in X_H$. Moreover, if $x \in X_P$ then we set $\mathbb{G}'_{x0} = \min(\mathbb{G}_{x0}, (<, +\infty))$ if $\mathbb{G}_{xy} \neq (\leq, +\infty)$ for some $y \neq x$, otherwise we keep $\mathbb{G}'_{x0} = \mathbb{G}_{x0}$. Similarly, if $y \in X_H$ then we set $\mathbb{G}'_{0y} = \min(\mathbb{G}_{0y}, (<, +\infty))$ if $\mathbb{G}_{xy} \neq (\leq, +\infty)$ for some $x \neq y$, otherwise we keep $\mathbb{G}'_{0y} = \mathbb{G}_{0y}$. Finally, for $x, y \in X$ with $x \neq y$ we set $\mathbb{G}'_{xy} = \mathbb{G}_{xy}$. The graph $\mathbb{G}'$ constructed above is called the standardization of $\mathbb{G}$.

Lemma 4.6. *The standardization $\mathbb{G}'$ of a distance graph $\mathbb{G}$ is in standard form and $[\![\mathbb{G}']\!] = [\![\mathbb{G}]\!]$.*

Proof. The graph $\mathbb{G}'$ is in standard form by construction. Clearly, $\mathbb{G}' \leq \mathbb{G}$, hence $[\![\mathbb{G}']\!] \subseteq [\![\mathbb{G}]\!]$. Now, let $v \in [\![\mathbb{G}]\!]$. We show that v satisfies all constraints in $\mathbb{G}'$. Let $x \in X_P$ be a prophecy clock. We have $v(x) \leq 0$. Hence v satisfies the constraint $\mathbb{G}'_{0x} = \min(\mathbb{G}_{0x}, (\leq, 0))$. Assume now that $\mathbb{G}_{xy} = (\lhd, c) < (\leq, +\infty)$ for some $y \neq x$. Since $v \in [\![\mathbb{G}]\!]$, we have $v(y) - v(x) \lhd c$, which implies $v(x) \neq -\infty$. Hence v satisfies the constraint $\mathbb{G}'_{x0} = \min(\mathbb{G}_{x0}, (<, +\infty))$. We proceed similarly for the new constraints when $x \in X_H$ is a history clock. □

Definition 4.7. The weight of a path in a distance graph $\mathbb{G}$ is the sum of the weights of its edges. A cycle in $\mathbb{G}$ is said to be negative if its weight is strictly less than $(\leq, 0)$.

A distance graph $\mathbb{G}$ is in *normal form* if it is in standard form, has no negative cycles, and the weight of each edge $x \to y$ is not greater than the weight of any path from x to y.

A standard distance graph with no negative cycles can be normalized. Let $\mathbb{G}$ be a standard distance graph with no negative cycles. Let $\mathbb{G}'$ be the graph where the weight of an edge $x \to y$ is set to the minimum of the weights of the paths from x to y in $\mathbb{G}$.[6] The graph $\mathbb{G}'$ constructed above is called the *normalization* of $\mathbb{G}$.

Lemma 4.8 (Normalization). *Let $\mathbb{G}$ be a standard distance graph with no negative cycles. The normalization $\mathbb{G}'$ of $\mathbb{G}$ is in normal form and $[\![\mathbb{G}]\!] = [\![\mathbb{G}']\!]$.*

Proof. First, we check that $\mathbb{G}'$ is in standard form. Let $x \in X_P$. We have $\mathbb{G}'_{0x} \leq \mathbb{G}_{0x} \leq (\leq, 0)$. Now assume that $\mathbb{G}'_{x0} = (\leq, +\infty)$. Then $\mathbb{G}'_{x0} \leq \mathbb{G}_{x0} = (\leq, +\infty)$ and $\mathbb{G}_{xy} = (\leq, +\infty)$ for all $y \neq x$. Since all paths in $\mathbb{G}$ from x to some $y \neq x$ start with some edge $x \to z$ of weight $(\leq, +\infty)$, we obtain $\mathbb{G}'_{xy} = (\leq, +\infty)$ for all $y \neq x$. The condition for history clocks is proved similarly.

It is easy to see that $\mathbb{G}'$ is in normal form. We have $\mathbb{G}' \leq \mathbb{G}$ by construction, hence $[\![\mathbb{G}']\!] \subseteq [\![\mathbb{G}]\!]$. Conversely, let $v \in [\![\mathbb{G}]\!]$ and let $x_1 \to x_2 \to \cdots \to x_n$ be an arbitrary path in $\mathbb{G}$ with weight $(\lhd, c)$. As $v \in [\![\mathbb{G}]\!]$, we have $v \models x_{i+1} - x_i \lhd_i c_i$ with $\mathbb{G}_{x_i x_{i+1}} = (\lhd_i, c_i)$. By Lemma 4.3, we get $v \models x_n - x_1 \lhd c$. We deduce that $v \in [\![\mathbb{G}']\!]$. □

Given two distance graphs $\mathbb{G}'$, $\mathbb{G}''$, we define $\mathbb{G} = \min(\mathbb{G}', \mathbb{G}'')$ as the distance graph obtained by setting $\mathbb{G}_{xy} = \min(\mathbb{G}'_{xy}, \mathbb{G}''_{xy})$ for all $x \neq y$. Notice that $[\![\mathbb{G}]\!] = [\![\mathbb{G}']\!] \cap [\![\mathbb{G}'']\!]$. Moreover, if both $\mathbb{G}'$ and $\mathbb{G}''$ are in standard form, then so is $\min(\mathbb{G}', \mathbb{G}'')$. On the other hand, even when $\mathbb{G}'$ and $\mathbb{G}''$ are in normal form, $\min(\mathbb{G}', \mathbb{G}'')$ need not be in normal form.

---

[6]When $\mathbb{G}$ has no negative cycles, the weight of a non-simple path $x \cdots z \cdots z \cdots y$ is at least the weight of the subpath $x \cdots z \cdots y$. Hence, we may restrict to simple paths when taking the minimum.

We will make use of important properties, which have been shown when weights come from $\mathcal{C} \setminus \{(<, -\infty), (\leq, -\infty), (\leq, +\infty)\}$, and continue to hold even with the new weights. First, we show that a classical and crucial property of distance graphs for timed automata extends to the *standard* distance graphs over the extended algebra that we have defined above for event-clock automata.

Lemma 4.9. *Let $\mathbb{G}$ be a* standard *distance graph. Then, $[\![\mathbb{G}]\!] \neq \emptyset$ iff $\mathbb{G}$ has no negative cycles.*

Proof. Suppose $[\![\mathbb{G}]\!]$ is non-empty. Let $v \in [\![\mathbb{G}]\!]$. Let $x_1 \to x_2 \to \cdots \to x_n \to x_{n+1} = x_1$ be an arbitrary cycle in $\mathbb{G}$ and let $(\lhd, c)$ be the weight of the cycle. As $v \in [\![\mathbb{G}]\!]$, we have $v \models x_{i+1} - x_i \lhd_i c_i$ with $(\lhd_i, c_i)$ the weight of $x_i \to x_{i+1}$ in $\mathbb{G}$. By Lemma 4.3, we deduce that $v \models x_{n+1} - x_1 \lhd c$. Since $x_{n+1} = x_1$, we get $v(x_{n+1}) - v(x_1) \in \{0, +\infty\}$. Hence, the cycle is not negative.

Conversely, let $\mathbb{G}$ be a standard distance graph with no negative cycles. We will tighten the distance graph, progressively fixing the value of each clock, while preserving a standard graph with no negative cycles. By Lemma 4.8, we assume w.l.o.g. that $\mathbb{G}$ is in normal form.

Let $x \in X$ be a clock. Considering the cycle $0 \to x \to 0$, we get $(\leq, 0) \leq \mathbb{G}_{0x} + \mathbb{G}_{x0}$. By Lemma 4.4, we find $\alpha \in \overline{\mathbb{R}}$ with $(\leq, \alpha) \leq \mathbb{G}_{0x}$ and $(\leq, -\alpha) \leq \mathbb{G}_{x0}$. Notice that $0 \leq \alpha$ if x is a history clock and $\alpha \leq 0$ if x is a prophecy clock.

Define the graph $\mathbb{G}'$ obtained from $\mathbb{G}$ by setting $\mathbb{G}'_{0x} = (\leq, \alpha)$, $\mathbb{G}'_{x0} = (\leq, -\alpha)$ and keeping the other weights unchanged. We can easily check that $\mathbb{G}'$ is in standard form. We claim that $\mathbb{G}'$ has no negative cycles. Cycles using only weights from $\mathbb{G}$ are non-negative since $\mathbb{G}$ has no negative cycles. Consider now a cycle $0 \to x \cdots \to 0$. If $x \to \cdots \to 0$ consists of a single edge, then the weight of the cycle in $\mathbb{G}'$ is $(\leq, \alpha) + (\leq, -\alpha)$ which is either $(\leq, 0)$ or $(\leq, +\infty)$ and hence non-negative. Otherwise, the path $x \cdots 0$ only uses weights from $\mathbb{G}$. Let $(\lhd, c)$ be its weight. We have $\mathbb{G}_{x0} \leq (\lhd, c)$ since $\mathbb{G}$ is in normal form. Now, using $(\leq, -\alpha) \leq \mathbb{G}_{x0}$ and Lemma 4.4, we get $(\leq, 0) \leq (\leq, \alpha) + (\lhd, c)$ showing that the cycle is not negative. We proceed similarly for cycles of the form $0 \to \cdots x \to 0$.

The graph $\mathbb{G}'$, denoted $\mathbb{G}[x = \alpha]$, is obtained from $\mathbb{G}$ by fixing the value of x to α. By construction we have $\mathbb{G}' \leq \mathbb{G}$, hence $[\![\mathbb{G}']\!] \subseteq [\![\mathbb{G}]\!]$. Moreover, $v(x) = \alpha$ for all $v \in [\![\mathbb{G}']\!]$.

Since $\mathbb{G}'$ is standard and has no negative cycles, we can repeat the process, fixing the value of each clock in X one after the other. At the end of the process, we obtain a standard distance graph $\mathbb{G}''$ with no negative cycles and where the value of each clock $x \in X$ has been fixed to some $\alpha_x \in \overline{\mathbb{R}}$. We still have $[\![\mathbb{G}'']\!] \subseteq [\![\mathbb{G}]\!]$.

Let $v \in \mathbb{V}$ be defined by $v(x) = \alpha_x$ for all $x \in X$. We show that $v \in [\![\mathbb{G}'']\!]$. Clearly, v satisfies all the constraints $\mathbb{G}''_{0x} = (\leq, \alpha_x)$ and $\mathbb{G}''_{x0} = (\leq, -\alpha_x)$. Now, consider an edge $x \to y$ with weight $\mathbb{G}''_{xy} = (\lhd, c)$. If $\alpha_y = +\infty$ then y is a history clock and $\mathbb{G}''_{0y} = (\leq, +\infty)$. Since $\mathbb{G}''$ is standard, we deduce that $\mathbb{G}''_{xy} = (\leq, +\infty)$. Similarly, if $\alpha_x = -\infty$ then x is a prophecy clock, $\mathbb{G}''_{x0} = (\leq, +\infty)$ and $\mathbb{G}''_{xy} = (\leq, +\infty)$. In both cases we get $v \models y - x \lhd c$. We assume below that $\alpha_y \neq +\infty$ and $\alpha_x \neq -\infty$. The cycle $x \to y \to 0 \to x$ is non-negative and has weight $(\lhd, c) + (\leq, -\alpha_y) + (\leq, \alpha_x) = (\lhd, c) + (\leq, \alpha_x - \alpha_y)$. We deduce that $(\leq, -(\alpha_x - \alpha_y)) \leq (\lhd, c)$. Recall that $-(b - a) = a - b$ unless $a = b \in \{-\infty, +\infty\}$. Since $\alpha_y \neq +\infty$ and $\alpha_x \neq -\infty$ we obtain $(\leq, \alpha_y - \alpha_x) \leq (\lhd, c)$, which implies $v \models y - x \lhd c$. This concludes the proof. $\qquad\square$

We will now establish a stronger result, which will be used in several proofs, which will also show that a distance graph in normal form is *canonical*, in a sense that we explain below. The result (which generalizes the one from timed automata) intuitively says that if a distance

graph is in normal form, then for every pair of clocks $x, y \in X \cup \{0\}$, there is a valuation v in its semantics which assigns time-stamps to x and y such that their difference (in the extended algebra) is a value arbitrarily close to the weight of the edge $x \to y$. Formally,

Lemma 4.10. *Let $\mathbb{G}$ be a distance graph in normal form, $x, y \in X \cup \{0\}$ be a pair of distinct clocks and $\alpha \in \overline{\mathbb{R}}$. There is a valuation $v \in \llbracket \mathbb{G} \rrbracket$ with $v(y) - v(x) = \alpha$ if and only if*

(1) $(\leq, \alpha) \leq \mathbb{G}_{xy}$ *and* $(\leq, -\alpha) \leq \mathbb{G}_{yx}$, *and*
(2) *if $x, y \in X$ and $\alpha \in \mathbb{R}$ is finite then the weights $\mathbb{G}_{x0}, \mathbb{G}_{0x}, \mathbb{G}_{y0}, \mathbb{G}_{0y}$ are all different from $(\leq, -\infty)$, and*
(3) *if $x, y \in X$ and $\alpha = -\infty$ then $\mathbb{G}_{0x} \neq (\leq, -\infty) \neq \mathbb{G}_{y0}$.*

Proof. We first show that the conditions are necessary. Let $v \in \llbracket \mathbb{G} \rrbracket$ be a valuation with $v(y) - v(x) = \alpha$.

(1) Let $\mathbb{G}_{xy} = (\triangleleft, c)$. We know that $v \models y - x \triangleleft c$ and we deduce that $\alpha \triangleleft c$, which is equivalent to $(\leq, \alpha) \leq \mathbb{G}_{xy}$. Let $\mathbb{G}_{yx} = (\triangleleft', c')$. We know that $v \models x - y \triangleleft' c'$ and we deduce that $v(x) - v(y) \triangleleft' c'$. Recall that $-(b - a) = a - b$ unless $a = b \in \{-\infty, +\infty\}$, in which case $a - b = b - a = +\infty$. We deduce that $-\alpha \leq v(x) - v(y) \triangleleft' c'$, which implies $(\leq, -\alpha) \leq \mathbb{G}_{yx}$.
(2) Assume that $\alpha \in \mathbb{R}$ is finite, then $v(x), v(y) \in \mathbb{R}$ are both finite. Assume that $\mathbb{G}_{x0} = (\leq, -\infty)$ (resp. $\mathbb{G}_{0x} = (\leq, -\infty)$). Then $v \in \llbracket \mathbb{G} \rrbracket$ implies $v(x) = +\infty$ (resp. $v(x) = -\infty$), a contradiction. Similarly, $\mathbb{G}_{y0}, \mathbb{G}_{0y}$ are not $(\leq, -\infty)$.
(3) Assume that $\alpha = -\infty$. Then, $v(y) = -\infty \neq v(x)$ or $v(x) = +\infty \neq v(y)$. Now, if $\mathbb{G}_{0x} = (\leq, -\infty)$ then $v(x) = -\infty$, a contradiction. Similarly, we must have $\mathbb{G}_{y0} \neq (\leq, -\infty)$.

Before proving the other direction, we give a short comparison with the timed automata setting. For distance graphs arising in classical timed automata, Condition 1 is already sufficient to get the right-to-left direction: if we replace the edges $x \to y$ and $y \to x$ with $(\leq, \alpha)$ and $(\leq, -\alpha)$ respectively, we can deduce that the resulting graph has no negative cycles, and hence the semantics of this graph is non-empty. Any valuation v in the semantics of the resulting graph has $v(y) - v(x) = \alpha$, which gives us the converse direction of the lemma. On the other hand, in the current setting with infinity weights, doing this same replacement might not result in a graph in standard form. As we have seen earlier, for graphs not in standard form, the absence of negative cycles does not imply a non-empty solution set. This is why we need the second and third conditions which indirectly allow to reason on a standard form for the resulting graph, depending on the α that is chosen. Let us now prove this direction.

Assume that the three conditions are satisfied. We consider different cases.

Assume first that $x = 0$. We fix the value of y to α by considering the graph $\mathbb{G}' = \mathbb{G}[y = \alpha]$ defined in the proof of Lemma 4.9. We have seen that $\mathbb{G}'$ is standard with no negative cycles. By Lemma 4.9 we find $v \in \llbracket \mathbb{G}' \rrbracket \subseteq \llbracket \mathbb{G} \rrbracket$ and we have $v(y) = \alpha$. The case $y = 0$ is handled similarly by considering $\mathbb{G}' = \mathbb{G}[x = -\alpha]$ which fixes the value of x to $-\alpha$. For the rest of the proof, we assume that $x, y \in X$ and we distinguish three cases depending on α.

• The first case is when $\alpha = +\infty$. Since $\mathbb{G}$ is in normal form, we have $(\leq, +\infty) \leq \mathbb{G}_{xy} \leq \mathbb{G}_{x0} + \mathbb{G}_{0y}$. Hence, $\mathbb{G}_{x0} = (\leq, +\infty)$ or $\mathbb{G}_{0y} = (\leq, +\infty)$. Assume that $\mathbb{G}_{x0} = (\leq, +\infty)$. We have $(\leq, -\infty) \leq \mathbb{G}_{0x}$. Hence we may fix the value of x to $-\infty$ by taking $\mathbb{G}' = \mathbb{G}[x = -\infty]$. Applying Lemma 4.9 we find a valuation $v \in \llbracket \mathbb{G}' \rrbracket \subseteq \llbracket \mathbb{G} \rrbracket$. We have $v(x) = -\infty$. We deduce that $v(y) - v(x) = +\infty = \alpha$. We proceed similarly when $\mathbb{G}_{0y} = (\leq, +\infty)$, fixing the value of y to $+\infty$.

- The next case is when $\alpha \in \mathbb{R}$ is finite. To fix the value of $y - x$ to α we define the graph $\mathbb{G}'$ by setting $\mathbb{G}'_{xy} = (\leq, \alpha)$, $\mathbb{G}'_{yx} = (\leq, -\alpha)$, $\mathbb{G}'_{x0} = \min(\mathbb{G}_{x0}, (<, +\infty))$, $\mathbb{G}'_{0x} = \min(\mathbb{G}_{0x}, (<, +\infty))$, $\mathbb{G}'_{y0} = \min(\mathbb{G}_{y0}, (<, +\infty))$, $\mathbb{G}'_{0y} = \min(\mathbb{G}_{0y}, (<, +\infty))$, and the weight of other edges $z \to z'$ are kept unchanged: $\mathbb{G}'_{zz'} = \mathbb{G}_{zz'}$. This graph $\mathbb{G}'$ is denoted $\mathbb{G}[y - x = \alpha]$. It is easy to see that $\mathbb{G}'$ is in standard form. Since $\mathbb{G}' \leq \mathbb{G}$ we have $[\![\mathbb{G}']\!] \subseteq [\![\mathbb{G}]\!]$. Moreover, $v(y) - v(x) = \alpha$ for all $v \in [\![\mathbb{G}']\!]$. It remains to show that $[\![\mathbb{G}']\!] \neq \emptyset$, i.e., that $\mathbb{G}'$ has no negative cycles. Since $\mathbb{G}$ is in normal form, we may restrict to cycles not using two consecutive weights from $\mathbb{G}$. In particular, cycles using only weights from $\mathbb{G}$ are non negative. It remains to consider the following cycles.
- Cycle $x \to y \to x$ has weight $(\leq, \alpha) + (\leq, -\alpha) = (\leq, 0)$ which is non-negative.
- Cycle $0 \to x \to 0$ with $\mathbb{G}'_{0x} = (<, +\infty)$ or $\mathbb{G}'_{x0} = (<, +\infty)$ is not negative since we have $\mathbb{G}_{x0} \neq (\leq, -\infty) \neq \mathbb{G}_{0x}$. We argue similarly for cycle $0 \to y \to 0$.
- Cycle $0 \to x \to y \to 0$ with $\mathbb{G}'_{0x} = (<, +\infty)$ or $\mathbb{G}'_{y0} = (<, +\infty)$ is not negative since $\mathbb{G}_{y0} \neq (\leq, -\infty) \neq \mathbb{G}_{0x}$. We argue similarly for cycle $0 \to y \to x \to 0$.

Therefore, $\mathbb{G}'$ has no negative cycles and we are done with the case $\alpha \in \mathbb{R}$ finite.

- The last case is $\alpha = -\infty$. From $(\leq, -\alpha) \leq \mathbb{G}_{yx}$ we get $\mathbb{G}_{yx} = (\leq, +\infty)$. Since $\mathbb{G}$ is in normal form, $\mathbb{G}_{yx} \leq \mathbb{G}_{y0} + \mathbb{G}_{0x}$ and we deduce that $\mathbb{G}_{y0} = (\leq, +\infty)$ or $\mathbb{G}_{0x} = (\leq, +\infty)$.

  Assume that $\mathbb{G}_{0x} = (\leq, +\infty)$. We define a graph $\mathbb{G}'$ which fixes x to $+\infty$ and forces y to be different from $+\infty$. We set $\mathbb{G}'_{x0} = (\leq, -\infty)$, $\mathbb{G}'_{0y} = \min(\mathbb{G}_{0y}, (<, +\infty))$ and $\mathbb{G}'_{zz'} = \mathbb{G}_{zz'}$ otherwise. The graph $\mathbb{G}'$ is denoted $\mathbb{G}[x = +\infty, y < +\infty]$. It is easy to check that $\mathbb{G}'$ is in standard form. Moreover, $\mathbb{G}' \leq \mathbb{G}$ and for all $v \in [\![\mathbb{G}']\!]$ we have $v(x) = +\infty$ and $v(y) < +\infty$, so we get $v(y) - v(x) = -\infty$ as desired. It remains to show that $[\![\mathbb{G}']\!] \neq \emptyset$, i.e., that $\mathbb{G}'$ has no negative cycles. Since $\mathbb{G}$ is in normal form, we may restrict to cycles not using two consecutive weights from $\mathbb{G}$. In particular, cycles using only weights from $\mathbb{G}$ are non negative. It remains to consider the following cycles.
- Cycle $0 \to x \to 0$ has weight $(\leq, +\infty)$ since $\mathbb{G}_{0x} = (\leq, +\infty)$.
- Cycle $0 \to y \to 0$ with $\mathbb{G}'_{0y} = (<, +\infty)$ is not negative since $\mathbb{G}_{y0} \neq (\leq, -\infty)$.
- Cycle $0 \to y \to x \to 0$ with $\mathbb{G}'_{0y} = (<, +\infty)$ has weight $(\leq, +\infty)$ since $\mathbb{G}_{yx} = (\leq, +\infty)$.

Therefore, $\mathbb{G}'$ has no negative cycles.

  We proceed similarly when $\mathbb{G}_{y0} = (\leq, +\infty)$ by defining a graph $\mathbb{G}' = \mathbb{G}[x \neq -\infty, y = -\infty]$ fixing y to $-\infty$ and forcing x to be different from $-\infty$. This concludes the proof. $\square$

To conclude, we show that a distance graph in *normal form* is *canonical*, in the sense that if two distance graphs in normal form have the same semantics, then they must be identical.

Lemma 4.11. *Let $\mathbb{G}, \mathbb{G}'$ be two distance graphs with $\mathbb{G}$ in normal form. If $[\![\mathbb{G}]\!] \subseteq [\![\mathbb{G}']\!]$ then $\mathbb{G} \leq \mathbb{G}'$. In particular, if both $\mathbb{G}, \mathbb{G}'$ are in normal form and if $[\![\mathbb{G}]\!] = [\![\mathbb{G}']\!]$ then $\mathbb{G} = \mathbb{G}'$.*

Proof. Assume that $\mathbb{G} \nleq \mathbb{G}'$. Let $x, y \in X \cup \{0\}$ such that $\mathbb{G}'_{xy} < \mathbb{G}_{xy}$. Since the cycle $x \to y \to x$ is not negative in $\mathbb{G}$, by Lemma 4.4(2) we find $\alpha \in \overline{\mathbb{R}}$ such that $(\leq, \alpha) \leq \mathbb{G}_{xy}$, $(\leq, -\alpha) \leq \mathbb{G}_{yx}$, and $\mathbb{G}'_{xy} < (\leq, \alpha)$.

We will apply Lemma 4.10. We have chosen α so that Condition 1 is satisfied. Notice that $\alpha \neq -\infty$ since $\mathbb{G}'_{xy} < (\leq, \alpha)$ (recall that weight $(<, -\infty)$ is not allowed in distance graphs). Assume that $x, y \in X$ and α finite. From the choice of α in the proof of Lemma 4.4, we see that if $\mathbb{G}_{xy} = (\leq, +\infty)$ then $\alpha = +\infty$. Hence, $\mathbb{G}_{xy} \neq (\leq, +\infty)$. Since $\mathbb{G}$ is standard, this implies $\mathbb{G}_{x0} \neq (\leq, +\infty) \neq \mathbb{G}_{0y}$. Since $\mathbb{G}$ has no negative cycles, we deduce that

$\mathbb{G}_{0x} \neq (\leq, -\infty) \neq \mathbb{G}_{y0}$. Using $(\leq, -\infty) \neq \mathbb{G}_{xy} \leq \mathbb{G}_{x0} + \mathbb{G}_{0y}$ and $\mathbb{G}_{x0} \neq (\leq, +\infty) \neq \mathbb{G}_{0y}$, we deduce that $\mathbb{G}_{x0} \neq (\leq, -\infty) \neq \mathbb{G}_{0y}$. Therefore, Condition 2 is also satisfied. We apply Lemma 4.10 and get a valuation $v \in \mathbb{G}$ with $v(y) - v(x) = \alpha$. From $\mathbb{G}'_{xy} < (\leq, \alpha)$, we deduce that $v \notin [\![\mathbb{G}']\!]$. Therefore, $[\![\mathbb{G}]\!] \nsubseteq [\![\mathbb{G}']\!]$. □

This lemma finally allows us to define *the* canonical distance graph of a non-empty event-zone Z, that can be computed from a distance graph defining the zone Z (extending the result from usual zones in timed automata).

Definition 4.12. For a non-empty event-zone Z, we denote by $\mathbb{G}(Z)$ the unique distance graph in normal form satisfying $[\![\mathbb{G}(Z)]\!] = Z$. $\mathbb{G}(Z)$ is called the canonical distance graph of Z. We denote by Z_{xy} the weight of the edge $x \to y$ in $\mathbb{G}(Z)$.

4.3. **Successor computation.** To implement the computation of transitions $(q, Z) \xrightarrow{t} (q_1, Z_1)$ in an event-zone graph, we will make use of some operations on event-zones that we define below. Using distance graphs, we will show in Lemmas 4.16, 4.18, 4.17 and 4.19 that these operations preserve event-zones, that is, starting from an event-zone and applying any of the operations leads to an event-zone again. Thanks to the algebra over the new weights that we have defined, the arguments are very similar to the case of standard timed automata.

Definition 4.13 (Operations on event-zones). Let g be a guard and Z an event-zone.
- Guard intersection: $Z \wedge g := \{v \mid v \in Z \text{ and } v \models g\}$
- Release: $[\overrightarrow{a}]Z = \bigcup_{v \in Z}[\overrightarrow{a}]v$
- Reset: $[\overleftarrow{a}]Z = \{[\overleftarrow{a}]v \mid v \in Z\}$
- Time elapse: $\overrightarrow{Z} = \{v + \delta \mid v \in Z, \delta \in \mathbb{R}_{\geq 0} \text{ s.t. } v + \delta \models \bigwedge_{a \in \Sigma} \overrightarrow{a} \leq 0\}$

Let $M \in \mathbb{N}$. We say that an event-zone Z is *M-reachable* if it can be obtained starting from the initial zone Z_0 and applying the above operations, where guards g use M-bounded finite constants, i.e., the constants allowed in a guard are from $\{-\infty, -M, \ldots, -1, 0, 1, \ldots, M, +\infty\}$. Recall that the initial zone Z_0 is given by $\bigwedge_{a \in \Sigma}\left((+\infty \leq \overleftarrow{a}) \wedge (\overrightarrow{a} \leq 0)\right)$.

When M is clear from context, we will sometimes abuse notation and just say reachable zone instead of M-reachable zone. A guard g can be seen as yet another event-zone and hence guard intersection is just an intersection operation between two event-zones. By definition, for a transition $t := (q, a, g, q')$ and a node (q, Z) the successor $(q, Z) \xrightarrow{t} (q', Z')$ can be computed in the following sequence:

$$Z_1 := Z \cap (0 \leq \overrightarrow{a}) \qquad Z_2 := [\overrightarrow{a}]Z_1 \qquad Z_3 := Z_2 \cap g \qquad Z_4 := [\overleftarrow{a}]Z_3 \qquad Z' := \overrightarrow{Z_4}$$

As an example, in Figure 6, suppose an action b with guard $(\overrightarrow{a} = -1) = (\overrightarrow{a} \leq -1) \wedge (-1 \leq \overrightarrow{a})$ is fired from Zone Z as depicted, applying the above sequence in order gives $Z_1 := Z \cap (0 \leq \overrightarrow{b})$, $Z_2 := [\overrightarrow{b}]Z_1$, $Z_3 := Z_2 \cap (\overrightarrow{a} = -1)$, $Z_4 := [\overleftarrow{b}]Z_3$ resulting in the successor zone $Z' := \overrightarrow{Z_4}$, as depicted in the figure.

We will now translate the operations from event-zones to distance graphs.

Definition 4.14 (Operations on distance graphs). Let $\mathbb{G}$ be a distance graph in normal form. Let g be a guard.
- Guard intersection: a distance graph $\mathbb{G}_g$ is obtained from $\mathbb{G}$ as follows,

Figure 6: Successor computation from event-zone Z on an action b with guard $\overrightarrow{a} = -1$

- for each constraint $x \lhd c$ in g, replace weight of edge $0 \rightarrow x$ with $\min(\mathbb{G}_{0x}, (\lhd, c))$,
  - for each constraint $d \lhd y$ in g, replace weight of edge $y \rightarrow 0$ with $\min(\mathbb{G}_{y0}, (\lhd, -d))$,
  - normalize the resulting graph if it has no negative cycles.
- Release: a distance graph $[\overrightarrow{a}]\mathbb{G}$ is obtained from $\mathbb{G}$ by
  - removing all edges involving $\overrightarrow{a}$ and then
  - adding the edges $0 \xrightarrow{(\leq,0)} \overrightarrow{a}$ and $\overrightarrow{a} \xrightarrow{(\leq,+\infty)} 0$, and then
  - normalizing the resulting graph.
- Reset: a distance graph $[\overleftarrow{a}]\mathbb{G}$ is obtained from $\mathbb{G}$ by
  - removing all edges involving $\overleftarrow{a}$ and then
  - adding the edges $0 \xrightarrow{(\leq,0)} \overleftarrow{a}$ and $\overleftarrow{a} \xrightarrow{(\leq,0)} 0$, and then
  - normalizing the resulting graph.
- Time elapse: the distance graph $\overrightarrow{\mathbb{G}}$ is obtained by the following transformation:
  - if $\mathbb{G}_{0\overleftarrow{x}} \neq (\leq, +\infty)$ then replace it with $(<, +\infty)$,
  - if $\mathbb{G}_{0\overrightarrow{x}} \neq (\leq, -\infty)$ then replace it with $(\leq, 0)$,
  - normalize the resulting graph.

The theorem below says that the operations on event-zones translate easily to operations on distance graphs and that the successor of an event-zone is an event-zone. Note that, except for the release operation $[\overrightarrow{a}]$, the rest of the operations are standard in timed automata, but they do not use $(\leq, +\infty), (\leq, -\infty)$. We show that we can perform all these operations in the new algebra with quadratic complexity, as in timed automata without diagonal constraints [ZLZ05].

Theorem 4.15. *Let $\mathbb{G}$ be a distance graph in normal form, $a \in \Sigma$, and g be a guard. We can compute, in $\mathcal{O}(|X_P \cup X_H|^2)$ time, distance graphs $\mathbb{G}_g$, $[\overrightarrow{a}]\mathbb{G}$, $[\overleftarrow{a}]\mathbb{G}$ and $\overrightarrow{\mathbb{G}}$ in normal form, such that $\llbracket \mathbb{G} \rrbracket \wedge g = \llbracket \mathbb{G}_g \rrbracket$, $[\overrightarrow{a}]\llbracket \mathbb{G} \rrbracket = \llbracket [\overrightarrow{a}]\mathbb{G} \rrbracket$, $[\overleftarrow{a}]\llbracket \mathbb{G} \rrbracket = \llbracket [\overleftarrow{a}]\mathbb{G} \rrbracket$, and $\overrightarrow{\llbracket \mathbb{G} \rrbracket} = \llbracket \overrightarrow{\mathbb{G}} \rrbracket$.*

This theorem follows from Lemmas 4.16, 4.17, 4.18 and 4.19 which handle respectively the operations on distance graphs given in Definition 4.14. Other than normalization, it can be easily checked that these operations can be computed in quadratic time. We discuss the normalization procedure in the stated lemmas.

In general, the normal form of a distance graph cannot always be computed in quadratic time. However, starting from an event zone in normal form, and applying the operations of Definition 4.14 gives us special event-zones whose normal forms can be computed in quadratic time. This is explained in Lemmas 4.16, 4.17, 4.18 and 4.19.

Lemma 4.16. *Let $\mathbb{G}$ be a distance graph in normal form and let g be a guard. Then $\llbracket \mathbb{G}_g \rrbracket = \llbracket \mathbb{G} \rrbracket \wedge g$, and $\mathbb{G}_g$ can be computed in time $\mathcal{O}(|X|^2)$.*

Proof. According to the definition, we first construct an intermediate graph $\mathbb{G}'$ by replacing weights of edges of the form $0 \to x$ and $y \to 0$ depending on the guards present. It is easy to see that $[\![\mathbb{G}']\!] = [\![\mathbb{G}]\!] \wedge g$. The normalization process does not change the solution set.

Since g has only non-diagonal guards (as g is from an ECA, and our ECAs do not have diagonal guards) and $\mathbb{G}$ is in normal form, we can check if there is a negative cycle in $\mathbb{G}'$ in quadratic time (relevant cycles use one or two modified edges and are of the form $x \to 0 \to x$ or $x \to 0 \to y \to x$). If not, one first computes in quadratic time all shortest paths $x \to 0$ and $0 \to x$. The shortest path $x \to 0$ is obtained as $\mathbb{G}''_{x0} = \min(\mathbb{G}'_{x0}, \mathbb{G}_{xz} + \mathbb{G}'_{z0})$ over all $z \to 0$ that comes from guard g. Similarly the shortest path $0 \to x$ is $\mathbb{G}''_{0x} = \min(\mathbb{G}'_{0x}, \mathbb{G}'_{0z} + \mathbb{G}_{zx})$ over all $0 \to z$ coming from guard g. Finally, the shortest path $x \to y$ has weight $\min(\mathbb{G}'_{xy}, \mathbb{G}''_{x0} + \mathbb{G}''_{0y})$. $\hfill\square$

Lemma 4.17. *Let $\mathbb{G}$ be a distance graph in normal form and $a \in \Sigma$. Then, $[\![[\overrightarrow{a}]\mathbb{G}]\!] = [\overrightarrow{a}][\![\mathbb{G}]\!]$, and $[\overrightarrow{a}]\mathbb{G}$ can be computed in time $\mathcal{O}(|X|^2)$.*

Moreover, the weights of edges in $[\overrightarrow{a}]\mathbb{G}$ are given by

- $x \to y$ *has weight* $\mathbb{G}_{xy}$*, if* $x, y \neq \overrightarrow{a}$*,*
- $\overrightarrow{a} \to x$ *has weight* $(\leq, +\infty)$ *and* $x \to \overrightarrow{a}$ *has weight* $\mathbb{G}_{x0}$ *if* $x \neq \overrightarrow{a}$*.*

Proof. The release of a prophecy clock $\overrightarrow{a}$ corresponds to removing all edges involving the node $\overrightarrow{a}$, and then adding the edges $0 \xrightarrow{(\leq,0)} \overrightarrow{a}$ and $\overrightarrow{a} \xrightarrow{(\leq,+\infty)} 0$. Let $\mathbb{G}'$ be the distance graph thus obtained from $\mathbb{G}$. It is easy to see $[\overrightarrow{a}][\![\mathbb{G}]\!] \subseteq [\![\mathbb{G}']\!]$. For the converse inclusion, we pick $v \in [\![\mathbb{G}']\!]$ and show that there exists some $u \in [\![\mathbb{G}]\!]$ such that u coincides with v in all variables except $\overrightarrow{a}$. To see this, we construct a distance graph $\mathbb{G}''$ by setting $\mathbb{G}''_{0x} = (\leq, v(x))$ and $\mathbb{G}''_{x0} = (\leq, -v(x))$ for all $x \in X \setminus \{\overrightarrow{a}\}$; and $\mathbb{G}''_{xy} = \mathbb{G}_{xy}$ for all other edges $x \to y$. We show below that (a) $\mathbb{G}'' \leq \mathbb{G}$, (b) $\mathbb{G}''$ is in standard form, and (c) $\mathbb{G}''$ has no negative cycles. We deduce that $\emptyset \neq [\![\mathbb{G}'']\!] \subseteq [\![\mathbb{G}]\!]$ and $v \in [\overrightarrow{a}]u$ for all $u \in [\![\mathbb{G}'']\!]$.

(a) Let $x \in X \setminus \{\overrightarrow{a}\}$. Since $v \in [\![\mathbb{G}']\!]$, we have $\mathbb{G}''_{0x} = (\leq, v(x)) \leq \mathbb{G}'_{0x} = \mathbb{G}_{0x}$ and similarly $\mathbb{G}''_{x0} = (\leq, -v(x)) \leq \mathbb{G}'_{x0} = \mathbb{G}_{x0}$.
(b) Let $x \in X_H$. We have $\mathbb{G}''_{x0} = (\leq, -v(x)) \leq (\leq, 0)$. Assume that $\mathbb{G}''_{0x} = (\leq, +\infty)$. From (a) we get $\mathbb{G}_{0x} = (\leq, +\infty)$. Since $\mathbb{G}$ is standard, this implies $\mathbb{G}''_{yx} = \mathbb{G}_{yx} = (\leq, +\infty)$ for all $y \neq x$. We proceed similarly when $x \in X_P \setminus \{\overrightarrow{a}\}$.
(c) Since $\mathbb{G}$ is in normal form, cycles of $\mathbb{G}$ that are possibly negative are of the form $0 \to x \to 0$ or $0 \to x \to y \to 0$ with $x, y \neq \overrightarrow{a}$. The weight of $0 \to x \to 0$ in $\mathbb{G}''$ is $(\leq, 0)$ or $(\leq, +\infty)$, hence not negative. For the other cycle, notice first that since $v \in [\![\mathbb{G}']\!]$ we have $(\leq, v(y) - v(x)) \leq \mathbb{G}'_{xy} = \mathbb{G}_{xy}$, which is equivalent to $(\leq, 0) \leq \mathbb{G}_{xy} + (\leq, -(v(y) - v(x)))$. Recall that $-(b - a) \leq a - b$. Therefore, $(\leq, 0) \leq \mathbb{G}_{xy} + (\leq, v(x) - v(y))$, which is the weight of the cycle $0 \to x \to y \to 0$.

Then, $[\overrightarrow{a}]\mathbb{G}$ is obtained by normalization of $\mathbb{G}'$. Observe that the weight of no edge was decreased: $\mathbb{G}_{0\overrightarrow{a}} \leq (\leq, 0)$ and $\mathbb{G}_{\overrightarrow{a}0} \leq (\leq, +\infty)$. Therefore, this transformation does not lead to shorter paths: the edge $x \to y$ in $[\overrightarrow{a}]\mathbb{G}$ has weight $\mathbb{G}'_{xy} = \mathbb{G}_{xy}$ if $x \neq \overrightarrow{a} \neq y$. Now, non-trivial paths in $\mathbb{G}'$ starting from $\overrightarrow{a}$ start with weight $(\leq, +\infty)$, hence the weight of edge $\overrightarrow{a} \to x$ in $[\overrightarrow{a}]\mathbb{G}$ is $(\leq, +\infty)$. Finally, the shortest path in $\mathbb{G}'$ from x to $\overrightarrow{a}$ is $x \to 0 \to \overrightarrow{a}$, which is of weight $\mathbb{G}_{x0} + (\leq, 0) = \mathbb{G}_{x0}$. Hence, the weight of $x \to \overrightarrow{a}$ in $[\overrightarrow{a}]\mathbb{G}$ is $\mathbb{G}_{x0}$. $\hfill\square$

Lemma 4.18. *Let $\mathbb{G}$ be a distance graph in normal form and $a \in \Sigma$. Then, $[\![[\overleftarrow{a}]\mathbb{G}]\!] = [\overleftarrow{a}][\![\mathbb{G}]\!]$, and $[\overleftarrow{a}]\mathbb{G}$ can be computed in time $\mathcal{O}(|X|^2)$.*

Moreover, the weights of edges in $[\overleftarrow{a}]\mathbb{G}$ are given by

- $x \to y$ has weight $\mathbb{G}_{xy}$, if $x, y \neq \overleftarrow{a}$,
- $x \to \overleftarrow{a}$ has weight $\mathbb{G}_{x0}$ and $\overleftarrow{a} \to x$ has weight $\mathbb{G}_{0x}$ if $x \neq \overleftarrow{a}$ (including $x = 0$ assuming $\mathbb{G}_{00} = (\leq, 0)$).

Proof. The reset of a history clock $\overleftarrow{a}$ corresponds to removing all edges involving node $\overleftarrow{a}$, and then setting its value to 0 by adding the edges $0 \xrightarrow{(\leq,0)} \overleftarrow{a}$ and $\overleftarrow{a} \xrightarrow{(\leq,0)} 0$. Let $\mathbb{G}'$ be the distance graph thus obtained from $\mathbb{G}$. Similar to timed automata it can be shown that $[\![\mathbb{G}']\!] = [\overleftarrow{a}][\![\mathbb{G}]\!]$.

Then, $[\overleftarrow{a}]\mathbb{G}$ is obtained by normalization of $\mathbb{G}'$. Normalization does not affect the weight of the edges $x \to y$ if $x, y \neq \overleftarrow{a}$ (as any path from x to y in $\mathbb{G}'$ using the new edges would involve a cycle $\overleftarrow{a} \to 0 \to \overleftarrow{a}$ or $0 \to \overleftarrow{a} \to 0$ of weight $(\leq, 0)$). Thus, the weight of edge $x \to y$ in $[\overleftarrow{a}]\mathbb{G}$ is equal to $\mathbb{G}_{xy}$ if $x, y \neq \overleftarrow{a}$. Now, paths in $\mathbb{G}'$ from $x \neq \overleftarrow{a}$ to $\overleftarrow{a}$ ends with the edge $0 \to \overleftarrow{a}$ of weight $(\leq, 0)$. Since $\mathbb{G}$ is in normal form, we deduce that the weight of edge $x \to \overleftarrow{a}$ in $[\overleftarrow{a}]\mathbb{G}$ is $\mathbb{G}'_{x0} = \mathbb{G}_{x0}$. Similarly, the weight of edge $\overleftarrow{a} \to x$ in $[\overleftarrow{a}]\mathbb{G}$ is $\mathbb{G}'_{0x} = \mathbb{G}_{0x}$. $\qquad\square$

Lemma 4.19. *Let $\mathbb{G}$ be a distance graph in normal form. Then, $[\![\overrightarrow{\mathbb{G}}]\!] = \overrightarrow{[\![\mathbb{G}]\!]}$, and $\overrightarrow{\mathbb{G}}$ can be computed in time $\mathcal{O}(|X|^2)$.*

Proof. The time elapse operation corresponds to (1) replacing the weight of $0 \to \overleftarrow{a}$ with $(<, +\infty)$ if it is not $(\leq, +\infty)$, and (2) replacing the weight of $0 \to \overrightarrow{a}$ with $(\leq, 0)$ if it is not $(\leq, -\infty)$. Let $\mathbb{G}'$ be the distance graph thus obtained from $\mathbb{G}$. Similar to timed automata it can be shown that $[\![\mathbb{G}']\!] = \overrightarrow{[\![\mathbb{G}]\!]}$.

Then, $\overrightarrow{\mathbb{G}}$ is obtained by normalization of $\mathbb{G}'$. The transformation from $\mathbb{G}$ to $\mathbb{G}'$ does not decrease the weights of edges, it may only increase the weights of edges from 0 to $x \in X$. Therefore, no shorter paths may be obtained by this transformation. We deduce that, for all edges $0 \neq x \to y$ we have $\overrightarrow{\mathbb{G}}_{xy} = \mathbb{G}'_{xy} = \mathbb{G}_{xy}$, and $\mathbb{G}_{0y} \leq \overrightarrow{\mathbb{G}}_{0y} \leq \mathbb{G}'_{0y}$ for all $y \in X$.

Since $\mathbb{G}$ is in normal form, the shortest path in $\mathbb{G}'$ from 0 to $y \in X$ is either the edge $0 \to y$ with weight $\mathbb{G}'_{0y}$ or of the form $0 \to x \to y$ which is of weight $\mathbb{G}'_{0x} + \mathbb{G}_{xy}$. We show below that, if $\mathbb{G}'_{0x} + \mathbb{G}_{xy} < \mathbb{G}'_{0y}$ then $x \in X_P$ is a prophecy clock and $\mathbb{G}'_{0x} = (\leq, 0)$. We deduce that $\overrightarrow{\mathbb{G}}_{0y} = \min(\{\mathbb{G}'_{0y}\} \cup \{\mathbb{G}_{xy} \mid x \in X_P\}) = \min(\{\mathbb{G}'_{xy} \mid x \in X_P \cup \{0\}\})$.

Assume that $\mathbb{G}'_{0x} + \mathbb{G}_{xy} < \mathbb{G}'_{0y}$. Then, $\mathbb{G}'_{0x} \neq (\leq, +\infty) \neq \mathbb{G}_{xy}$ and $\mathbb{G}'_{0y} \neq (\leq, -\infty)$. Hence, $\mathbb{G}_{0x} \neq (\leq, +\infty)$ and $\mathbb{G}_{0y} \neq (\leq, -\infty)$. Since $\mathbb{G}$ is in normal form, we have $\mathbb{G}_{0y} \leq \mathbb{G}_{0x} + \mathbb{G}_{xy}$ and we deduce that $\mathbb{G}_{0x} \neq (\leq, -\infty) \neq \mathbb{G}_{xy}$. If $x \in X_P$, we get $\mathbb{G}'_{0x} = (<, +\infty) = \mathbb{G}'_{0x} + \mathbb{G}_{xy} < \mathbb{G}'_{0y}$. This implies $\mathbb{G}_{0y} = \mathbb{G}'_{0y} = (\leq, +\infty)$, a contradiction with $\mathbb{G}_{xy} \neq (\leq, +\infty)$ since $\mathbb{G}$ is in standard form. Hence, $x \in X_H$ is a history clock and $\mathbb{G}_{0x} \neq (\leq, -\infty)$ implies $\mathbb{G}'_{0x} = (\leq, 0)$.

Finally, from $\overrightarrow{\mathbb{G}}_{0y} = \min(\{\mathbb{G}'_{0y}\} \cup \{\mathbb{G}_{xy} \mid x \in X_P\})$ and the definition of $\mathbb{G}'_{0y}$, we get

$$\overrightarrow{\mathbb{G}}_{0y} = \begin{cases} (\leq, +\infty) & \text{if } \mathbb{G}_{0y} = (\leq, +\infty) \\ (\leq, -\infty) & \text{if } \mathbb{G}_{0y} = (\leq, -\infty) \\ \min((<, +\infty), (\triangleleft, c)) & \text{if } y \text{ is a history clock and } G_{0y} \neq (\leq, +\infty) \\ \min((\leq, 0), (\triangleleft, c)) & y \text{ is a prophecy clock and } G_{0y} \neq (\leq, -\infty) \end{cases}$$

where $(\triangleleft, c) = \min\{\mathbb{G}_{xy} \mid x \in X_P\}$. $\qquad\square$

We conclude this section by showing some properties of reachable zones that will be useful in the sequel.

Lemma 4.20. *Let Z be a non-empty reachable zone and let $\mathbb{G}$ be its canonical distance graph.*

(1) *For all $x \in X_H$, we have $\mathbb{G}_{x0} = (\leq, -\infty)$ or $\mathbb{G}_{0x} \leq (<, +\infty)$.*

(2) *For all $x, y \in X$, if $\mathbb{G}_{xy} = (\leq, -\infty)$ then $\mathbb{G}_{x0} = (\leq, -\infty)$ or $\mathbb{G}_{0y} = (\leq, -\infty)$.*

Proof. Let $\overleftarrow{a} \in X_H$ be a history clock. The weight of $\overleftarrow{a} \to 0$ in the initial distance graph Z_0 is $(\leq, -\infty)$, which is the least possible weight. It stays unchanged until we first apply the reset operation on $\overleftarrow{a}$, resulting in the weighted edges $0 \xrightarrow{(\leq, 0)} \overleftarrow{a}$ and $\overleftarrow{a} \xrightarrow{(\leq, 0)} 0$. Then, the weight of edge $0 \to \overleftarrow{a}$ may only be increased by the time elapse operation, which sets it to $(<, +\infty)$. This proves the first property.

For the second property, let $x, y \in X$ with $\mathbb{G}_{xy} = (\leq, -\infty)$ and $\mathbb{G}_{x0} \neq (\leq, -\infty)$. We have to show that $\mathbb{G}_{0y} = (\leq, -\infty)$. If $x \in X_H$ then we get $\mathbb{G}_{0x} \leq (<, +\infty)$ by the first property. If $x \in X_P$ then we have $\mathbb{G}_{0x} \leq (\leq, 0)$. In both cases, since $\mathbb{G}$ is normal, we obtain $\mathbb{G}_{0y} \leq \mathbb{G}_{0x} + \mathbb{G}_{xy} = (\leq, -\infty)$ and we are done. $\qquad\square$

We have so far seen the computation of the event-zone graph. As discussed in Section 3, we require a simulation to truncate the event-zone graph to a finite prefix that is sound and complete for reachability. We discuss a new simulation relation for ECAs in the next section.

5. A concrete simulation relation for ECAs

We fix an event-clock automaton $\mathcal{A} = (Q, \Sigma, X, T, q_0, F)$ for this section. We will define a simulation relation $\preceq_{\mathcal{A}}$ on the configurations of the ECA. We first define a map $\mathcal{G}$ from Q to sets of atomic constraints. The map $\mathcal{G}$ is obtained as the least fixpoint of the set of equations:

$$\mathcal{G}(q) = \{\overrightarrow{b} \leq 0, 0 \leq \overrightarrow{b} \mid b \in \Sigma\} \cup \bigcup_{(q,a,g,q') \in T} \mathsf{split}(g) \cup \mathsf{pre}(a, \mathcal{G}(q'))$$

where $\mathsf{split}(g)$ is the set of atomic constraints on *history* clocks occurring in g and, for a set of atomic constraints G, $\mathsf{pre}(a, G)$ is defined as the set of constraints on history clocks in G except those on $\overleftarrow{a}$. Notice that constraints in $\mathcal{G}(q)$ use the constant 0 and constants used in constraints of $\mathcal{A}$. As a consequence, it is easy to see that the least fixpoint computation terminates and results in a finite set. Intuitively, $\mathcal{G}(q)$ contains an atomic constraint φ of the form $\overleftarrow{a} \lhd c$ or $c \lhd \overleftarrow{a}$ if φ appears in a transition out of some state q', and there is a path $q \to \cdots \to q'$ from q to q' over transitions that do not read a. This makes the value of $\overleftarrow{a}$ at q important for the verification of the constraint φ at q'. Transitions on letter a reset the history clock $\overleftarrow{a}$ to 0, and so, the actual value of $\overleftarrow{a}$ at q is irrelevant for constraints on $\overleftarrow{a}$ that are witnessed after a "reset". This is why such constraints are ignored in $\mathcal{G}(q)$. Figure 7 illustrates the computation of the $\mathcal{G}$-sets associated with each state of a given event-clock automaton.

Let G be a set of atomic constraints. The preorder $\preceq_G$ is defined on valuations by

$$v \preceq_G v' \qquad \text{if } \forall \varphi \in G, \ \forall \delta \geq 0, \qquad v + \delta \models \varphi \implies v' + \delta \models \varphi.$$

Notice that in the condition above, we *do not* restrict δ to those such that $v + \delta$ is a valuation: we may have $v(\overrightarrow{a}) + \delta > 0$ for some $a \in \Sigma$[7]. In usual timed automata, this question does not arise, as elapsing any δ from any given valuation always results in a

---

[7] We highlight the distinction between time elapse operation in general, and the time elapse operation allowed by the semantics of ECA. In a general time elapse, there is no restriction on the amount of time

Figure 7: An event-clock automaton $\mathcal{A}$ and illustration of the computation of the $\mathcal{G}$-map. Note that the set P is used to denote the set of constraints involving prophecy clocks, i.e., $P = \{\overrightarrow{a} \leq 0, 0 \leq \overrightarrow{a}, \overrightarrow{b} \leq 0, 0 \leq \overrightarrow{b}\}$ - this set is part of the $\mathcal{G}$-set of each state. Further, observe that the constraints involving $\overleftarrow{a}$ do not get propagated to state q_0 as the transition $q_0 \to q_1$ is on the action a, and therefore resets $\overleftarrow{a}$.

valuation. But this is crucial for the proof of Theorem 5.4 below. We illustrate this using Example 5.1. It also allows us to get a clean characterization of the simulation (Lemma 5.5) which in turn is useful for deriving the simulation test and in showing finiteness.

Example 5.1. Consider two valuations v and v' defined as follows: $v(\overrightarrow{a}) = -4$, $v(\overrightarrow{b}) = -1$ and $v'(\overrightarrow{a}) = -3$ and $v'(\overrightarrow{b}) = -1$. Let $G = \{\overrightarrow{a} \leq 0, 0 \leq \overrightarrow{a}, \overrightarrow{b} \leq 0, 0 \leq \overrightarrow{b}\}$. Firstly, notice that $v \npreceq_G v'$: if we take $\delta = 3.5$, we have $v + \delta \models \overrightarrow{a} \leq 0$, whereas $v' + \delta \not\models \overrightarrow{a} \leq 0$. Notice that in $v + \delta$ and $v' + \delta$, the value of $\overrightarrow{b}$ is $+2.5$, which is not a legal valuation reachable in an ECA. This means that if v, v' were valuations at a certain state of an ECA, they cannot elapse δ units staying at that state.

Now, suppose in the definition of $\preceq_G$, we restrict to δ such that $v + \delta$ of all prophecy clocks is at most 0. Then, $v \preceq_G v'$: firstly $\delta \leq 1$ due to the value of $\overrightarrow{b}$, and further, for all $\delta \leq 1$, whenever $v + \delta$ satisfies a constraint in G, the valuation $v' + \delta$ satisfies the same constraint. However, this way of relating v and v' does not guarantee a simulation, especially after a transition that updates the value of $\overrightarrow{b}$ to a fresh value. In particular, we can come up with a sequence of transitions that is feasible from v, but not from v'. This is illustrated in Figure 8.

Remark 5.2. Let v, v' be valuations and $a \in \Sigma$. If $v \preceq_{\{0 \leq \overrightarrow{a}, \overrightarrow{a} \leq 0\}} v'$ then $v(\overrightarrow{a}) = v'(\overrightarrow{a})$.

The proof is easy. First, if $v(\overrightarrow{a}) = -\infty$ then $v + \delta \models \overrightarrow{a} \leq 0$ for all $\delta \geq 0$. Since v' simulates v we deduce that $v' + \delta \models \overrightarrow{a} \leq 0$ for all $\delta \geq 0$. This implies $v'(\overrightarrow{a}) = -\infty$. Next, if $-\infty < v(\overrightarrow{a}) \neq v'(\overrightarrow{a}) \leq 0$. Then, for $\delta = -v(\overrightarrow{a}) \geq 0$ we have $v(\overrightarrow{a}) + \delta = 0 \neq v'(\overrightarrow{a}) + \delta$, a contradiction with $v \preceq_{\{0 \leq \overrightarrow{a}, \overrightarrow{a} \leq 0\}} v'$. $\square$

Based on $\preceq_G$ and the $\mathcal{G}(q)$ computation, we can define a preorder $\preceq_{\mathcal{A}}$ between configurations of ECA $\mathcal{A}$.

---

that can be elapsed from a valuation. But for the semantics of ECA, we restrict to time elapses such that prophecy clocks stay at most 0.

Figure 8: An example to explain why it is not sufficient to only look at time-elapse restricted to ECA valuations in the definition of $\mathcal{G}$-simulation. The $*$ symbol for the value of $\overrightarrow{b}$ denotes that $\overrightarrow{b}$ may take any legal value in this valuation.

Definition 5.3. $(q, v) \preceq_{\mathcal{A}} (q', v')$ if $q = q'$ and $v \preceq_{\mathcal{G}(q)} v'$.

With this definition in place, we can now show the following theorem.

Theorem 5.4. *The relation $\preceq_{\mathcal{A}}$ is a simulation on the transition system $S_{\mathcal{A}}$ of ECA $\mathcal{A}$.*

Proof. Assume that $(q, v_1) \preceq_{\mathcal{A}} (q, v_2)$, i.e., $v_1 \preceq_{\mathcal{G}(q)} v_2$.

Delay transition: Assume that $(q, v_1) \xrightarrow{\delta} (q, v_1 + \delta)$ is a transition of $S_{\mathcal{A}}$. Then, $v_1 + \delta \models \bigwedge_{x \in \Sigma} \overrightarrow{x} \leq 0$. Since $\mathcal{G}(q)$ contains $\overrightarrow{x} \leq 0$ for all $x \in \Sigma$ and $v_1 \preceq_{\mathcal{G}(q)} v_2$, we deduce that $v_2 + \delta \models \bigwedge_{x \in \Sigma} \overrightarrow{x} \leq 0$. Therefore, $(q, v_2) \xrightarrow{\delta} (q, v_2 + \delta)$ is a transition in $S_{\mathcal{A}}$. It is easy to see that $v_1 + \delta \preceq_{\mathcal{G}(q)} v_2 + \delta$.

Action transition: Let $t = (q, g, a, q')$ be a transition in $\mathcal{A}$ and assume that $(q, v_1) \xrightarrow{t} (q', v'_1)$ is a transition in $S_{\mathcal{A}}$, i.e., $v_1 \models 0 \leq \overrightarrow{a}$ and for some $v''_1 \in [\overrightarrow{a}]v_1$ we have $v''_1 \models g$ and $v'_1 = [\overleftarrow{a}]v''_1$. Since $\mathcal{G}(q)$ contains $0 \leq \overrightarrow{a}$ and $v_1 \preceq_{\mathcal{G}(q)} v_2$, we deduce that $v_2 \models 0 \leq \overrightarrow{a}$.

We have $v''_1 = v_1[\overrightarrow{a} \mapsto \alpha]$ for some $\alpha \in [-\infty, 0]$ and $v'_1 = v''_1[\overleftarrow{a} \mapsto 0] = v_1[\overleftarrow{a} \mapsto 0, \overrightarrow{a} \mapsto \alpha]$. Define $v''_2 = v_2[\overrightarrow{a} \mapsto \alpha] \in [\overrightarrow{a}]v_2$ and $v'_2 = v''_2[\overleftarrow{a} \mapsto 0] = v_2[\overleftarrow{a} \mapsto 0, \overrightarrow{a} \mapsto \alpha]$. Notice that, $v'_1(x) = v''_1(x) = v''_2(x) = v'_2(x)$ for all prophecy clocks $x \in X_P$ (follows from $v_1 \preceq_{\mathcal{G}(q)} v_2$ and $\{x \leq 0, 0 \leq x\} \subseteq \mathcal{G}(q)$). From $v_1 \preceq_{\mathcal{G}(q)} v_2$ and the definition of v''_2, we deduce that $v''_1 \preceq_{\mathcal{G}(q)} v''_2$. Since $\mathcal{G}(q)$ contains $\mathsf{split}(g)$, and $v''_1 \preceq_{\mathcal{G}(q)} v''_2$, we deduce that $v''_2 \models g$.

Therefore, $(q, v_2) \xrightarrow{t} (q', v'_2)$ is a transition in $S_{\mathcal{A}}$. Since $\mathcal{G}(q)$ contains $\mathsf{pre}(a, \mathcal{G}(q'))$, we easily get from $v_1 \preceq_{\mathcal{G}(q)} v_2$ and the definitions of v'_1, v'_2 that $v'_1 \preceq_{\mathcal{G}(q')} v'_2$. Note that, to get this, we crucially use the fact that in the definition of the simulation $\preceq_G$ we consider *all* $\delta \geq 0$ and not only those such that $v + \delta$ is a valuation. To see why, notice that the α that is picked to make $v'_1(\overrightarrow{a}) = \alpha$ could be any value in $[-\infty, 0]$, in particular, it could be smaller than all values in v_1. This potentially allows δ such that $v'_1 + \delta$ is a valuation whereas $v_1 + \delta$ is not. □

When $G = \{\varphi\}$ is a singleton, we simply write $\preceq_{\varphi}$ for $\preceq_{\{\varphi\}}$. The definition of the $\preceq_{\mathcal{A}}$ simulation above in some sense declares what is expected out of the simulation. Below, we give a constructive characterization of the simulation in terms of the constants used and the valuations. For example, if $v(\overleftarrow{a}) = 3$ and $\overleftarrow{a} \leq 5$ is a constraint in G, point 2 below

says that all v' with $v'(\overleftarrow{a}) \leq 3$ simulate v. The next lemma is a generalization of [GMS20, Lemma 8] to our setting containing prophecy clocks and the undefined values $+\infty$ and $-\infty$.

Lemma 5.5. *Let v, v' be valuations and G a set of atomic constraints. We have*

(1) $v \preceq_G v'$ *iff* $v \preceq_\varphi v'$ *for all* $\varphi \in G$.
(2) $v \preceq_{x \triangleleft c} v'$ *iff* $v(x) \ntriangleleft c$ *or* $v'(x) \leq v(x)$ *or* $(\triangleleft, c) = (\leq, +\infty)$ *or* $(\triangleleft, c) = (<, +\infty) \wedge v'(x) < +\infty$.
(3) $v \preceq_{c \triangleleft x} v'$ *iff* $c \triangleleft v'(x)$ *or* $v(x) \leq v'(x)$ *or* $(\triangleleft, c) = (<, +\infty)$ *or* $(\triangleleft, c) = (\leq, +\infty) \wedge v(x) < +\infty$.

Proof.

(1) is clear.
(2) The right to left implication is easy. Notice for instance that if $(\triangleleft, c) = (<, +\infty)$ and $v'(x) < +\infty$ then for all $\delta \geq 0$ we have $v' + \delta \models x < +\infty$. Conversely, assume that $v \preceq_{x \triangleleft c} v'$ and $v(x) \triangleleft c$ and $(\triangleleft, c) \neq (\leq, +\infty)$. If $(\triangleleft, c) = (<, +\infty)$ then, using $\delta = 0$ and $v(x) \triangleleft c$, we get $v'(x) < +\infty$. If $(\triangleleft, c) = (\leq, -\infty)$ then, using $\delta = 0$ and $v(x) \triangleleft c$, we get $v'(x) \leq -\infty = v(x)$. Otherwise, $c \in \mathbb{Z}$ and we have to show that $v'(x) \leq v(x)$. Assume that $v(x) < v'(x)$. Then, we find $\delta \geq 0$ such that $v(x) + \delta \triangleleft c < v'(x) + \delta$, a contradiction.
(3) Again, the implication from right to left is easy. Notice that in the last two cases, $(\triangleleft, c) = (<, +\infty)$ or $(\triangleleft, c) = (\leq, +\infty) \wedge v(x) < +\infty$, then for all $\delta \geq 0$ we have $v + \delta \not\models c \triangleleft x$. Conversely, assume that $v \preceq_{c \triangleleft x} v'$ and $c \ntriangleleft v'(x)$ and $(\triangleleft, c) \neq (<, +\infty)$. If $(\triangleleft, c) = (\leq, +\infty)$ then, using $\delta = 0$ and $c \ntriangleleft v'(x)$, we get $c \ntriangleleft v(x)$, i.e., $v(x) < +\infty$. If $(\triangleleft, c) = (<, -\infty)$ then, using $\delta = 0$ and $c \ntriangleleft v'(x)$, we get $c \ntriangleleft v(x)$, i.e., $v(x) = -\infty = v'(x)$. Otherwise, $c \in \mathbb{Z}$ and we have to show that $v(x) \leq v'(x)$. Assume that $v'(x) < v(x)$. Then, we find $\delta \geq 0$ such that $c \triangleleft v(x) + \delta$ but $c \ntriangleleft v'(x) + \delta$, a contradiction. □

We now state some useful properties that get derived from Lemma 5.5.

Remark 5.6. Let v, v' be valuations and G a set of atomic constraints.

(1) Let $x \triangleleft_1 c_1$ and $x \triangleleft_2 c_2$ be constraints with $(\triangleleft_1, c_1) \leq (\triangleleft_2, c_2) < (<, +\infty)$ (we say that $x \triangleleft_1 c_1$ is subsumed by $x \triangleleft_2 c_2$). If $v \preceq_{x \triangleleft_2 c_2} v'$ then $v \preceq_{x \triangleleft_1 c_1} v'$. Indeed, from $(\triangleleft_2, c_2) < (<, +\infty)$ and $v \preceq_{x \triangleleft_2 c_2} v'$ we get either (1) $v'(x) \leq v(x)$ or (2) $v(x) \ntriangleleft_2 c_2$ which implies $v(x) \ntriangleleft_1 c_1$ since $(\triangleleft_1, c_1) \leq (\triangleleft_2, c_2)$.
(2) Let $c_1 \triangleleft_1 x$ and $c_2 \triangleleft_2 x$ be constraints with $(c_1, \triangleleft_1) \leq (c_2, \triangleleft_2) < (+\infty, \leq)$ (we say that $c_1 \triangleleft_1 x$ is subsumed by $c_2 \triangleleft_2 x$). If $v \preceq_{c_2 \triangleleft_2 x} v'$ then $v \preceq_{c_1 \triangleleft_1 x} v'$. Indeed, from $(c_2, \triangleleft_2) < (+\infty, \leq)$ and $v \preceq_{c_2 \triangleleft_2 x} v'$ we get either (1) $v(x) \leq v'(x)$ or (2) $c_2 \triangleleft_2 v'(x)$ which implies $c_1 \triangleleft_1 v'(x)$ since $(c_1, \triangleleft_1) \leq (c_2, \triangleleft_2)$.

     The ordering between *lower weights* is defined by $(c_1, \triangleleft_1) < (c_2, \triangleleft_2)$ if $c_1 < c_2$ or $c_1 = c_2$, $\triangleleft_1 = \leq$ and $\triangleleft_2 = <$. We have $(c_1, \triangleleft_1) < (c_2, \triangleleft_2)$ iff $(\triangleleft_2, -c_2) < (\triangleleft_1, -c_1)$.

Before lifting the simulation to event-zones, we present a central technical object that will be used from time to time in the next set of results.

5.1. Distance graph for valuations that simulate a given valuation.
     For a valuation v and a set of constraints G, we let $\uparrow_G v = \{v' \in \mathbb{V} \mid v \preceq_G v'\}$, i.e., the set of valuations v' which simulate v. We will define a distance graph, denoted $\mathbb{G}_G(v)$, such that $[\![\mathbb{G}_G(v)]\!] = \uparrow_G v$. We remark that $[\![\mathbb{G}_G(v)]\!]$ is not really a zone since it may use constants that are not integers.

Figure 9: The distance graph $\uparrow_G v$ when v is such that $v(\overleftarrow{a}) = 2, v(\overrightarrow{a}) = -7, v(\overleftarrow{b}) = 3, v(\overrightarrow{b}) = -5$ and $\mathcal{G} = P \cup \{\overleftarrow{a} \leq 3, \overleftarrow{a} > 1, \overleftarrow{b} \leq 2, \overleftarrow{b} \geq 5\}$. Here, the set $P = \{\overrightarrow{a} \leq 0, \overrightarrow{a} \geq 0, \overrightarrow{b} \leq 0, \overrightarrow{b} \geq 0\}$ denotes the set of constraints involving prophecy clocks.

We assume that G contains $\{0 \leq \overrightarrow{a}, \overrightarrow{a} \leq 0 \mid a \in \Sigma\}$ so that $v \preceq_G v'$ implies $v(\overrightarrow{a}) = v'(\overrightarrow{a})$ for all prophecy clocks $\overrightarrow{a}$ with $a \in \Sigma$ (see Remark 5.2). We remove from G constraints equivalent to true, such as $x \leq +\infty$, $-3 < \overleftarrow{a}$ or $0 \leq \overleftarrow{a}$, or equivalent to false, such as $\overleftarrow{a} < 0$ or $+\infty < x$. Also, by Remark 5.6, we may remove from G constraints that are subsumed by other constraints in G, while not changing the simulation relation. Hence, for history clocks, we have at most one upper-bound constraint $\overleftarrow{a} \triangleleft c$ with $(\leq, 0) \leq (\triangleleft, c) < (<, +\infty)$, and at most one lower-bound constraint $c \triangleleft \overleftarrow{a}$ with $(0, \leq) < (c, \triangleleft) < (+\infty, \leq)$. From now on, we always assume that the sets G of atomic constraints that we consider satisfy the above conditions.

The definition of the distance graph $\mathbb{G}_G(v)$ which defines $\uparrow_G v$ is based on Lemma 5.5.

- For each prophecy clock $\overrightarrow{a}$, we have the edges $\overrightarrow{a} \xrightarrow{(\leq, -v(\overrightarrow{a}))} 0$ and $0 \xrightarrow{(\leq, v(\overrightarrow{a}))} \overrightarrow{a}$.
- For each history clock $\overleftarrow{a}$, we have the edge $0 \to \overleftarrow{a}$ with weight
  - $(\leq, v(\overleftarrow{a}))$ if $\overleftarrow{a} \triangleleft c \in G$ with $(\triangleleft, c) < (<, +\infty)$ and $v(\overleftarrow{a}) \triangleleft c$,
  - $(<, +\infty)$ if we are not in the case above and $\overleftarrow{a} < +\infty \in G$, $v(\overleftarrow{a}) < +\infty$,
  - $(\leq, +\infty)$ otherwise.
- For each history clock $\overleftarrow{a}$, we have the edge $\overleftarrow{a} \to 0$ with weight
  - $(\leq, -\infty)$ if $+\infty \leq \overleftarrow{a} \in G$ and $v(\overleftarrow{a}) = +\infty$, and if we are not in this case:
  - $(\triangleleft, -c)$ if $c \triangleleft \overleftarrow{a} \in G$ with $(c, \triangleleft) < (+\infty, \leq)$ and $c \triangleleft v(\overleftarrow{a})$,
  - $(\leq, -v(\overleftarrow{a}))$ if $c \triangleleft \overleftarrow{a} \in G$ with $(c, \triangleleft) < (+\infty, \leq)$ and $c \ntriangleleft v(\overleftarrow{a})$,
  - $(\leq, 0)$ otherwise.

An example of $\uparrow_G v$ is given in Figure 9.

With this definition, the distance graph $\mathbb{G}_G(v)$ is in standard form, but not in normal form. Using Lemma 5.5, we easily see that it has the desired property:

Lemma 5.7. *We have $v \preceq_G v'$ iff v' satisfies all the constraints of $\mathbb{G}_G(v)$.*

5.2. **Simulation for event-zones and an effective algorithmic check.** Let Z, Z' be two event-zones and G be a set of atomic constraints. We say that Z is G-simulated by Z', denoted $Z \preceq_G Z'$, if for all $v \in Z$ there exists $v' \in Z'$ such that $v \preceq_G v'$. Finally, we define $(q, Z) \preceq_{\mathcal{A}} (q', Z')$ if $q = q'$ and $Z \preceq_{\mathcal{G}(q)} Z'$. In the rest of this section, we show how to

check this relation efficiently. We let $\downarrow_G Z = \{v \in \mathbb{V} \mid v \preceq_G v' \text{ for some } v' \in Z\}$. Notice that $Z \preceq_G Z'$ iff $Z \subseteq \downarrow_G Z'$ iff $\downarrow_G Z \subseteq \downarrow_G Z'$.

To check $Z \npreceq_G Z'$, we require a valuation $v \in Z$ with a witness that $\uparrow_G v \cap Z'$ is empty. Let $\mathbb{G}'$ be a distance graph in standard form with $Z' = [\![\mathbb{G}']\!]$. Since $\mathbb{G}_G(v)$ is also in standard form, so is the distance graph $\min(\mathbb{G}_G(v), \mathbb{G}')$ which defines $\uparrow_G v \cap Z'$. So we may apply Lemma 4.9 and the witness will be a negative cycle in $\min(\mathbb{G}_G(v), \mathbb{G}')$. We show that if $\uparrow_G v \cap Z'$ is empty and $\mathbb{G}'$ is in normal form, then there is a small witness, i.e., a negative cycle in $\min(\mathbb{G}_G(v), \mathbb{G}')$ containing at most three edges, and belonging to one of three specific forms.

Lemma 5.8. *Let v be a valuation, Z' a non-empty* reachable *event-zone with canonical distance graph $\mathbb{G}'$ and G a set of atomic constraints. Then, $\uparrow_G v \cap Z'$ is empty iff there is a negative cycle in one of the following forms:*

(1) $0 \to x \to 0$ *with* $0 \to x$ *from* $\mathbb{G}_G(v)$ *and* $x \to 0$ *from* $\mathbb{G}'$,
(2) $0 \to y \to 0$ *with* $0 \to y$ *from* $\mathbb{G}'$ *and* $y \to 0$ *from* $\mathbb{G}_G(v)$*, and*
(3) $0 \to x \to y \to 0$*, with weight of* $x \to y$ *from* $\mathbb{G}'$ *and the others from* $\mathbb{G}_G(v)$*. Moreover, this negative cycle has finite weight.*

Proof. Since the distance graph $\mathbb{G}'$ is in normal form, it has no negative cycle. Similarly, $\mathbb{G}_G(v)$ has no negative cycle since $v \in \uparrow_G v \neq \emptyset$. We know that $\uparrow_G v \cap Z' = \emptyset$ iff there is a (simple) negative cycle in $\min(\mathbb{G}_G(v), \mathbb{G}')$. Since $\mathbb{G}'$ is in normal form, we may restrict to negative cycles which do not use two consecutive edges from $\mathbb{G}'$. Now all edges of $\mathbb{G}_G(v)$ are adjacent to node 0. Hence, if a simple cycle uses an edge from $\mathbb{G}'$ which is adjacent to 0, it consists of only two edges $0 \to x \to 0$, one from $\mathbb{G}'$ and one from $\mathbb{G}_G(v)$. Otherwise, the simple cycle is of the form $0 \to x \to y \to 0$ where the edge $x \to y$ is from $\mathbb{G}'$ and the other two edges are from $\mathbb{G}_G(v)$. It remains to show that the two clock negative cycle $0 \to x \to y \to 0$ can be considered to have finite weight, i.e., weight is not $(\leq, -\infty)$.

For the cycle to have weight $(\leq, -\infty)$, one of the edges should have weight $(\leq, -\infty)$ and the others should have a weight different from $(\leq, +\infty)$. We will show that for every such combination, there is a smaller negative cycle with a single clock and 0. Hence we can ignore negative cycles of the form $0 \to x \to y \to 0$ with weight $(\leq, -\infty)$.

Suppose $\mathbb{G}'_{xy} = (\leq, -\infty)$. Since Z' (the zone corresponding to the distance graph $\mathbb{G}'$) is a reachable zone, using Lemma 4.20 we get $\mathbb{G}'_{x0} = (\leq, -\infty)$ or $\mathbb{G}'_{0y} = (\leq, -\infty)$. This gives a smaller negative cycle $0 \to x \xrightarrow{(\leq, -\infty)} 0$ or $0 \xrightarrow{(\leq, -\infty)} y \to 0$ with the other edge $0 \to x$ or $y \to 0$ coming from $\mathbb{G}_G(v)$, since by our hypothesis of a negative cycle, these edges have weight different from $(\leq, +\infty)$.

Suppose the weight of $0 \to x$ is $(\leq, -\infty)$ in $\mathbb{G}_G(v)$. This can happen only when x is a prophecy clock and $v(x) = -\infty$. Since $\mathbb{G}'_{xy} \neq (\leq, +\infty)$ and $\mathbb{G}'$ is in standard form, we infer $\mathbb{G}'_{x0} \neq (\leq, +\infty)$. Hence $0 \xrightarrow{(\leq, v(x))} x \xrightarrow{\mathbb{G}'_{x0}} 0$ is also a negative cycle.

Suppose $y \to 0$ has weight $(\leq, -\infty)$ in $\mathbb{G}_G(v)$. This can happen only when y is a history clock and $v(y) = +\infty$. Since $\mathbb{G}'_{xy} \neq (\leq, +\infty)$ and $\mathbb{G}'$ is in standard form, we obtain $\mathbb{G}'_{0y} \neq (\leq, +\infty)$. Hence, $0 \xrightarrow{\mathbb{G}'_{0y}} y \xrightarrow{(\leq, -v(y))} 0$ is a negative cycle. $\qquad\square$

We now have all the results required to state our inclusion test. Using the above lemma, and relying on a careful analysis we obtain the following theorem.

Theorem 5.9. *Let Z, Z' be non-empty* reachable *zones, and G a set of atomic constraints containing $\overrightarrow{a} \leq 0$ and $0 \leq \overrightarrow{a}$ for every prophecy clock $\overrightarrow{a}$. Then, $Z \npreceq_G Z'$ iff one of the following conditions holds:*[8]

(1) $Z'_{x0} < Z_{x0}$ *for some prophecy clock x, or for some history clock x with*
   - $(x < +\infty) \in G$ *and* $Z'_{x0} = (\leq, -\infty)$, *or*
   - $(x \lhd_1 c) \in G$ *for* $c \in \mathbb{N}$ *and* $(\leq, 0) \leq Z_{x0} + (\lhd_1, c)$.
(2) $Z'_{0y} < Z_{0y}$ *for some prophecy clock y, or for some history clock y with*
   - $(+\infty \leq y) \in G$ *and* $Z_{0y} = (\leq, +\infty)$, *or*
   - $(d \lhd_2 y) \in G$ *for* $d \in \mathbb{N}$ *and* $Z'_{0y} + (\lhd_2, -d) < (\leq, 0)$.
(3) $Z'_{xy} < Z_{xy}$ *and* Z'_{xy} *is finite for two distinct (prophecy or history) clocks x, y with* $(x \lhd_1 c), (d \lhd_2 y) \in G$ *for* $c, d \in \mathbb{N}$ *and* $(\leq, 0) \leq Z_{x0} + (\lhd_1, c)$ *and* $Z'_{xy} + (\lhd_2, -d) < Z_{x0}$.

Proof. By definition, $Z \npreceq_G Z'$ iff there is a $v \in Z$ such that $\uparrow_G v \cap Z' = \emptyset$. Lemma 5.8 gives three kinds of negative cycles that witness $\uparrow_G v \cap Z' = \emptyset$. We show that the three conditions in the theorem respectively characterize the presence of the three kinds of negative cycles.

Case 1. There is a negative cycle $0 \to x \to 0$ with $0 \to x$ from $\mathbb{G}_G(v)$ and $x \to 0$ from $\mathbb{G}(Z')$ iff Item 1 above is true.

($\Rightarrow$). Suppose there is such a negative cycle $0 \to x \to 0$. Weight of $0 \to x$ is either $(\leq, v(x))$ or $(<, +\infty)$.

When weight of $0 \to x$ is $(\leq, v(x))$: We have $(\leq, v(x)) + Z'_{x0} < (\leq, 0)$. Lemma 4.4 implies $Z'_{x0} < (\leq, -v(x))$. Now, since $v \in Z$, it satisfies all constraints of $\mathbb{G}(Z)$. In particular, $(\leq, -v(x)) \leq Z_{x0}$. We obtain $Z'_{x0} < Z_{x0}$. Notice that the weight of $0 \to x$ is $(\leq, v(x))$ when either x is a prophecy clock or x is a history clock with $(x \lhd_1 c) \in G$ for $c \in \mathbb{N}$ and $v(x) \lhd_1 c$. We can rewrite $v(x) \lhd_1 c$ as $(\leq, v(x)) \leq (\lhd_1, c)$. Hence, we get $(\leq, 0) \leq (\lhd_1, c) + (\leq, -v(x)) \leq (\lhd_1, c) + Z_{x0}$.

When weight of $0 \to x$ is $(<, +\infty)$: This happens when x is a history clock, $(x < +\infty) \in G$ and $v(x) < +\infty$ (and we are not in the case above). For the cycle to be negative, we must have $Z'_{x0} = (\leq, -\infty)$. From $v \in Z$ and $v(x) < +\infty$, we get $Z_{x0} \neq (\leq, -\infty)$. Hence $Z'_{x0} < Z_{x0}$.

($\Leftarrow$). Assume Item 1 is true.

Suppose first that x is a prophecy clock. Since $Z'_{x0} < Z_{x0}$, using Lemma 4.4 we find α satisfying both $Z'_{x0} < (\leq, -\alpha) \leq Z_{x0}$ and $(\leq, \alpha) \leq Z_{0x}$. By Lemma 4.10 we find $v \in Z$ with $v(x) = \alpha$. This gives the negative cycle $0 \xrightarrow{(\leq, v(x))} x \xrightarrow{Z'_{x0}} 0$.

Next, suppose that x is a history clock with $x \lhd_1 c$ in G for $c \in \mathbb{N}$ and $(\leq, 0) \leq Z_{x0} + (\lhd_1, c)$. Since $Z'_{x0} < Z_{x0}$, using Lemma 4.4, we can find α with $Z'_{x0} < (\leq, -\alpha) \leq Z_{x0}$ and $(\leq, \alpha) \leq \min(Z_{0x}, (\lhd_1, c))$. As above, by Lemma 4.10 we find $v \in Z$ with $v(x) = \alpha$, resulting in the negative cycle $0 \xrightarrow{(\leq, v(x))} x \xrightarrow{Z'_{x0}} 0$.

Now, suppose there is a history clock x with $(x < +\infty) \in G$ and $(\leq, -\infty) = Z'_{x0} < Z_{x0}$. This implies $Z_{x0} \neq (\leq, -\infty)$ and by Lemma 4.20 we obtain $Z_{0x} \leq (< +\infty)$. Let $v \in Z$. We have $v(x) < +\infty$. For this v, the weight of $0 \to x$ in $\uparrow_G v$ will be at most $(<, +\infty)$. As $Z'_{x0} = (\leq, -\infty)$, we get a negative cycle $0 \to x \to 0$ with $0 \to x$ from $\mathbb{G}_G(v)$ and $x \to 0$ from $\mathbb{G}(Z')$.

---

[8]Recall that for a non-empty zone Z, we simply write Z_{xy} for the weight of the edge $x \to y$ in the canonical distance graph of Z.

Case 2. There is a negative cycle $0 \to y \to 0$ with $0 \to y$ from $\mathbb{G}(Z')$ and $y \to 0$ from $\mathbb{G}_G(v)$ iff Item 2 in the theorem statement is true.

($\Rightarrow$). Suppose there is such a negative cycle.

When weight of $y \to 0$ is $(\leq, -v(y))$. This happens when y is a prophecy clock or y is a history clock with $(d \vartriangleleft_2 y) \in G$ with $d \in \mathbb{N}$ and $d \not\vartriangleleft_2 v(y)$. From the negative cycle, we have $Z'_{0y} + (\leq, -v(y)) < (\leq, 0)$, i.e., $Z'_{0y} < (\leq, v(y))$. Since $v \in Z$, we have $(\leq, v(y)) \leq Z_{0y}$. We deduce that $Z'_{0y} < Z_{0y}$. In the case when y is a history clock, we have $d \not\vartriangleleft_2 v(y)$. Therefore, either $v(y) < d$ or $v(y) = d$ and $\vartriangleleft_2 = <$. In both cases, $(\vartriangleleft_2, -d) \leq (\leq, -v(y))$. Hence $Z'_{0y} + (\vartriangleleft_2, -d) < (\leq, 0)$.

When weight of $y \to 0$ is $(\leq, -\infty)$. This occurs when y is a history clock, $+\infty \leq y$ is in G and $v(y) = +\infty$. As the cycle is negative we get $Z'_{0y} \neq (\leq, +\infty)$. Since $v \in Z$, we get $Z_{0y} = (\leq, +\infty)$. This gives $Z'_{0y} < Z_{0y}$.

When weight of $y \to 0$ is $(\vartriangleleft_2, -d)$. This is when y is a history clock, we are not in the subcase above and there is $d \vartriangleleft_2 y$ in G with $d \in \mathbb{N}$ and $d \vartriangleleft_2 v(y)$. The negative cycle gives $Z'_{0y} + (\vartriangleleft_2, -d) < (\leq, 0)$. From $d \vartriangleleft_2 v(y)$, we can infer that $(\leq, -v(y)) \leq (\vartriangleleft_2, -d)$. Therefore we also have $Z'_{0y} + (\leq, -v(y)) < (\leq, 0)$. As in the first subcase above, we get $Z'_{0y} < Z_{0y}$.

The remaining case is when y is a history clock and the weight of $y \to 0$ is $(\leq, 0)$. Since $(\leq, 0) \leq Z'_{0y}$, the cycle $0 \to y \to 0$ cannot be negative and we can ignore this case.

($\Leftarrow$). Assume Item 2 is true.

Let us start with the case when y is a prophecy clock. We have $Z'_{0y} < Z_{0y}$. By Lemma 4.4, we can find α with both $Z'_{0y} < (\leq, \alpha) \leq Z_{0y}$ and $(\leq, -\alpha) \leq Z_{y0}$. By Lemma 4.10 we find $v \in Z$ with $v(y) = \alpha$. This gives the negative cycle $0 \xrightarrow{Z'_{0y}} y \xrightarrow{(\leq, -v(y))} 0$.

Suppose y is a history clock with $+\infty \leq y$ in G, and $Z_{0y} = (\leq, +\infty)$. For reachable zones, this implies that $Z_{y0} = (\leq, -\infty)$ (Lemma 4.20). Hence every valuation in Z has y-value to be $+\infty$. Pick an arbitrary $v \in Z$. We have $v(y) = +\infty$. For this v, the value of $y \to 0$ in $\mathbb{G}_G(v)$ is $(\leq, -\infty)$. Now, since $Z'_{0y} < Z_{0y}$, the cycle $Z'_{0y} + (\leq, -\infty)$ is negative.

Finally, let y be a history clock with $d \vartriangleleft_2 y$ in G for $d \in \mathbb{N}$ and $Z'_{0y} + (\vartriangleleft_2, -d) < (\leq, 0)$. Since $Z'_{0y} < Z_{0y}$, using again Lemma 4.4, we find α with both $Z'_{0y} < (\leq, \alpha) \leq Z_{0y}$ and $(\leq, -\alpha) \leq Z_{y0}$. By Lemma 4.10 we find $v \in Z$ with $v(y) = \alpha$. If $v(y) \not\vartriangleleft_2 d$, then the weight of $y \to 0$ is at most $(\leq, -v(y))$. Using $Z'_{0y} < (\leq, \alpha)$ gives $0 \to y \to 0$ to be negative cycle. Suppose $d \vartriangleleft_2 v(y)$. Then weight of $y \to 0$ is at most $(\vartriangleleft_2, -d)$. But we already have $Z'_{0y} + (\vartriangleleft_2, -d) < (\leq, 0)$ in our hypothesis, which gives the required negative cycle.

Case 3. There is a *finite weight* negative cycle $0 \to x \to y \to 0$ with $0 \to x$ and $y \to 0$ from $\mathbb{G}_G(v)$ and $x \to y$ from $\mathbb{G}(Z')$ iff the third condition of the theorem is true.

($\Rightarrow$). Suppose there is such a negative cycle. Since the weight of the cycle is finite, the weight of each edge is also finite. Hence, the weight of the edge $0 \to x$ is $(\leq, v(x))$. We find $x \vartriangleleft_1 c$ in G with $c \in \mathbb{N}$ and $v(x) \vartriangleleft_1 c$ (if x is a prophecy clock then $(\vartriangleleft_1, c) = (\leq, 0)$). As in case 1 above, from $v \in Z$ we deduce that $(\leq, 0) \leq Z_{x0} + (\leq, v(x))$ and from $v(x) \vartriangleleft_1 c$ we get $(\leq, v(x)) \leq (\vartriangleleft_1, c)$. We obtain $(\leq, 0) \leq Z_{x0} + (\vartriangleleft_1, c)$.

Now, since the weight of $y \to 0$ is finite, we find $d \vartriangleleft_2 y$ in G (if y is a prophecy clock we take $(d, \vartriangleleft_2) = (0, \leq)$). The weight of $y \to 0$ is $\max((\leq, -v(y)), (\vartriangleleft_2, -d))$. Therefore: $Z'_{xy} + (\leq, v(x) - v(y)) < (\leq, 0)$ and $(\leq, v(x)) + Z'_{xy} + (\vartriangleleft_2, -d) < (\leq, 0)$. Since $v \in Z$, we have $(\leq, v(y) - v(x)) \leq Z_{xy}$. Using $Z'_{xy} < (\leq, -(v(x) - v(y))) = (\leq, v(y) - v(x))$ (recall that $v(x)$ and $v(y)$ are finite in the present case), we get $Z'_{xy} < Z_{xy}$. Again, as $v \in Z$, we have $(\leq, -v(x)) \leq Z_{x0}$. Together with $Z'_{xy} + (\vartriangleleft_2, -d) < (\leq, -v(x))$, we get $Z'_{xy} + (\vartriangleleft_2, -d) < Z_{x0}$.

($\Leftarrow$). Suppose the third condition is true. Instead of explicitly constructing a v as in the previous two cases, we will simply prove that there exists a valuation that forms the required negative cycle. We start by defining some new weights and observe some properties that will be used later. Define $w_1 = (<, -e')$ if $Z'_{xy} = (\leq, e')$ and $w_1 = (\leq, -e')$ if $Z'_{xy} = (<, e')$, and $w_2 = (\leq, -e' + d)$ if either Z'_{xy} or $(\lhd_2, -d)$ has a strict inequality, and $w_2 = (<, -e' + d)$ otherwise. Notice that $w_1 + Z'_{xy} = (<, 0)$ and $w_2 + Z'_{xy} + (\lhd_2, -d) = (<, 0)$ are negative. Using $Z'_{xy} + (\lhd_2, -d) < Z_{x0}$, we get $(<, 0) = w_2 + Z'_{xy} + (\lhd_2, -d) < w_2 + Z_{x0}$ and we obtain $(\leq, 0) \leq w_2 + Z_{x0}$. Similarly, using $Z'_{xy} < Z_{xy}$ we get $(<, 0) = w_1 + Z'_{xy} < w_1 + Z_{xy}$ and $(\leq, 0) \leq w_1 + Z_{xy}$.

Consider a distance graph $\mathbb{G}$ formed by taking $\mathbb{G}(Z)$ and modifying two of its edges as follows: change $0 \to x$ to $\min(Z_{0x}, (\lhd_1, c), w_2)$; change $y \to x$ to $\min(Z_{yx}, w_1)$. Notice first that $\mathbb{G}$ is in standard form. Next, we claim that $\mathbb{G}$ has only non-negative cycles. It is sufficient to show that $0 \to x \to 0$ and $y \to x \to y$ are non-negative. From $(\leq, 0) \leq Z_{x0} + (\lhd_1, c)$, $(\leq, 0) \leq w_2 + Z_{x0}$ and the fact that $\mathbb{G}(Z)$ does not have negative cycles, we infer that $0 \to x \to 0$ is non-negative in $\mathbb{G}$. The weight of $y \to x \to y$ is either $Z_{yx} + Z_{xy}$ or $w_1 + Z_{xy}$. The former is non-negative as Z is non-empty. We have shown above that $w_1 + Z_{xy}$ is non-negative. Hence $\mathbb{G}$ is standard with no negative cycles, and its solution set is non-empty.

Now, pick a v that satisfies constraints of $\mathbb{G}$. Valuation v is in Z, and additionally satisfies $v(x) \lhd_1 c$, $(\leq, v(x)) \leq w_2$ and $(\leq, v(x) - v(y)) \leq w_1$. Recall that $w_1 + Z'_{xy} = (<, 0)$. Using $(\leq, v(x) - v(y)) \leq w_1$, we obtain $(\leq, v(x) - v(y)) + Z'_{xy} < (\leq, 0)$. If $d \not\lhd_2 v(y)$, then this corresponds to the weight of the cycle $0 \to x \to y \to 0$ which we have now shown to be negative. Otherwise, we have $d \lhd_2 v(y)$ and the weight of edge $y \to 0$ is $(\lhd_2, -d)$. The weight of the cycle would be: $(\leq, v(x)) + Z'_{xy} + (\lhd_2, -d)$. But, $(\leq, v(x)) \leq w_2$ and $w_2 + Z'_{xy} + (\lhd_2, -d) = (<, 0)$. Hence the required cycle is negative. $\square$

From Theorem 5.9, we can see that the inclusion test requires iteration over clocks x, y and checking if the conditions are satisfied by the respective weights.

Corollary 5.10. *Checking if $(q, Z) \preceq_{\mathcal{A}} (q', Z')$ can be done in time $\mathcal{O}(|X|^2) = \mathcal{O}(|\Sigma|^2)$.*

6. Finiteness of the simulation relation

In this section, we will show that the simulation relation $\preceq_{\mathcal{A}}$ defined in Section 5 is finite, which implies that the reachability algorithm of Definition 3.9 terminates. Recall that given an event-clock automaton $\mathcal{A}$, we have an associated map $\mathcal{G}$ from states of $\mathcal{A}$ to sets of atomic constraints. Let $M = \max\{|c| \mid c \in \mathbb{Z}$ is used in some constraint of $\mathcal{A}\}$, the maximal constant of $\mathcal{A}$. We have $M \in \mathbb{N}$ and constraints in the sets $\mathcal{G}(q)$ use constants in $\{-\infty, +\infty\} \cup \{c \in \mathbb{Z} \mid |c| \leq M\}$.

Recall that the simulation relation $\preceq_{\mathcal{A}}$ was defined on nodes of the event-zone graph $\mathsf{EZG}(\mathcal{A})$ by $(q, Z) \preceq_{\mathcal{A}} (q', Z')$ if $q = q'$ and $Z \preceq_{\mathcal{G}(q)} Z'$. This simulation relation $\preceq_{\mathcal{A}}$ is *finite* if for any infinite sequence $(q, Z_0), (q, Z_1), (q, Z_2), \ldots$ of *reachable* nodes in $\mathsf{EZG}(\mathcal{A})$ we find $i < j$ with $(q, Z_j) \preceq_{\mathcal{A}} (q, Z_i)$, i.e., $Z_j \preceq_{\mathcal{G}(q)} Z_i$. Notice that we restrict to *reachable* zones in the definition above. Our goal now is to prove that the relation $\preceq_{\mathcal{A}}$ is finite. The structure of the proof is as follows.

(1) We prove in Lemma 6.3 of Section 6.1 that for any *reachable* node (q, Z) of $\mathsf{EZG}(\mathcal{A})$, the canonical distance graph $\mathbb{G}(Z)$ satisfies a set of conditions, that we call ($\dagger$) conditions below, which depend only on the maximal constant M of $\mathcal{A}$.

(2) We introduce an equivalence relation $\sim_M$ of *finite index* on valuations (depending on M only) and show in Lemma 6.6 of Section 6.2 that, if G is a set of atomic constraints using constants in $\{c \in \mathbb{Z} \mid |c| \le M\} \cup \{-\infty, +\infty\}$ and if Z is a zone such that its canonical distance graph $\mathbb{G}(Z)$ satisfies (†) conditions, then $\downarrow_G Z$ is a union of $\sim_M$ equivalence classes.

6.1. Prophecy clocks and distance graphs in reachable event-zones.

We start with a lemma which highlights an important and surprising property of *prophecy* clocks in reachable event-zones. Namely, we show that if an M-reachable event-zone contains a valuation v in which a prophecy clock has a finite value that is below $-M$, then it must contain all valuations obtained from v by setting that prophecy clock to any finite value smaller than $-M$. The implication of this property is illustrated in Figure 10. This property is essential for the proof of the (†) conditions. The proof follows from the observation that the property is true in the initial zone, and is invariant under the zone operations, namely, guard intersection, reset, release and time elapse.

Lemma 6.1. *Let Z be an M-reachable event-zone. For every valuation $v \in Z$, and for every prophecy clock $\overrightarrow{x}$, if $-\infty < v(\overrightarrow{x}) < -M$, then $v[\overrightarrow{x} \mapsto \alpha] \in Z$ for every $-\infty < \alpha < -M$.*

Proof. The initial zone is $(\bigwedge_{\overleftarrow{x}} \overleftarrow{x} = +\infty) \wedge (\bigwedge_{\overrightarrow{x}} -\infty \le \overrightarrow{x} \le 0)$. The property is true in this zone. We now show that the property is invariant under guard intersection (with maximal constant M), reset, release and time elapse. Assume Z is a zone that satisfies the property. Let v be an arbitrary valuation (not necessarily in Z) with $-\infty < v(\overrightarrow{x}) < -M$. For $-\infty < \alpha < -M$, we simply write $v_\alpha = v[\overrightarrow{x} \mapsto \alpha]$.

Guard intersection: Let g be a guard, which is in general a conjunction of atomic constraints. Suppose g contains an atomic constraint on $\overrightarrow{x}$, either $c \lhd \overrightarrow{x}$ or $\overrightarrow{x} \lhd c$. The constant c is either $-\infty$, or $-M \le c \le 0$. Therefore if v satisfies the constraint, every v_α will satisfy the atomic constraint. Now, suppose g contains $d \lhd y$ or $y \lhd d$ for some $y \ne \overrightarrow{x}$. Once again, notice that if v satisfies this guard, every v_α will satisfy the guard since $v(y) = v_\alpha(y)$. Hence, if v satisfies the guard, v_α satisfies the guard. The property is true in the zone $Z \wedge g$.

Release: The release operation on $\overrightarrow{y}$ takes a valuation u and adds a valuation $u[\overrightarrow{y} \mapsto \beta]$ for each $-\infty \le \beta \le 0$. Let $v \in [\overrightarrow{y}]Z$. Then, there is some valuation $u \in Z$ such that $v = u[\overrightarrow{y} \mapsto \beta]$ for some $-\infty \le \beta \le 0$. When $\overrightarrow{y} = \overrightarrow{x}$, associating α to $\overrightarrow{x}$ in the release operation starting from u gives v_α. When $\overrightarrow{y} \ne \overrightarrow{x}$, we use the induction hypothesis: valuation $u_\alpha \in Z$. Setting $\overrightarrow{y}$ to β in the release operation starting from u_α gives v_α.

Reset: The reset operation takes every valuation in Z and makes some history clock to 0. This does not perturb the required property.

Time elapse: Suppose $v + \delta \in \overrightarrow{Z}$ for some $v \in Z$ and $\delta \ge 0$. This means $(v + \delta)(\overrightarrow{y}) \le 0$ for every prophecy clock $\overrightarrow{y}$. We have to show that if $-\infty < (v + \delta)(\overrightarrow{x}) < -M$, then $(v + \delta)_\alpha \in \overrightarrow{Z}$ for all $-\infty < \alpha < -M$. Since $-\infty < (v + \delta)(\overrightarrow{x}) < -M$, we also have $-\infty < v(\overrightarrow{x}) < -M$. For some given α, let $\beta = \alpha - \delta$. We have $v_\beta \in Z$ by hypothesis. Notice that $(v + \delta)_\alpha = v_\beta + \delta$. Moreover, $v_\beta + \delta \in \overrightarrow{Z}$: for each clock $\overrightarrow{y} \ne \overrightarrow{x}$, $(v_\beta + \delta)(\overrightarrow{y}) = (v + \delta)(\overrightarrow{y}) \le 0$ and $(v_\beta + \delta)(\overrightarrow{x}) = \alpha < -M \le 0$. $\square$

Remark 6.2. We now make three significant observations about the above lemma.

Figure 10: A pictorial depiction of Lemma 6.1, where the shaded region depicts the zone Z.

(1) The lemma holds for M-reachable event-zones but not for all event-zones.
(2) There is no similar version of the above lemma for history clocks. A reset of a history clock makes its value exactly equal to 0 in every valuation and creates non-trivial diagonal constraints with other clocks. Moreover, repeated resets can generate arbitrarily large diagonal constraints, for e.g., a loop with guard $x = 1$ and reset x. This is why simulations are particularly needed to control history clocks.
(3) In our simulation $v \preceq_G v'$, we have $v(\overrightarrow{a}) = v'(\overrightarrow{a})$ (Remark 5.2): there is no abstraction of the value of prophecy clocks and the simulation relation by itself does not have any means to show finiteness. However, as we show below, due to the above property, the reachable zones themselves take care of finiteness with respect to prophecy clocks.

The challenge is then to combine this observation on prophecy clocks along with the non-trivial simulation happening for history clocks to prove that we still get a finite simulation. This is what we do in Section 6.2.

Now, we give the (†) conditions and prove that they are satisfied by canonical distance graphs of reachable zones.

Lemma 6.3. *Let $Z \neq \emptyset$ be an M-reachable zone. Then, the canonical distance graph $\mathbb{G}(Z)$ satisfies the (†) conditions:*

†$_1$ *If $Z_{\overrightarrow{x}0} = (<, +\infty)$ then for all $y \neq \overrightarrow{x}$, either y is a prophecy clock which is undefined in Z and $Z_{\overrightarrow{x}y} = Z_{0y} = (\leq, -\infty)$ or $Z_{\overrightarrow{x}y} \in \{(<, +\infty), (\leq, +\infty)\}$.*

†$_2$ *If $Z_{\overrightarrow{x}0} < (<, +\infty)$ then $(\leq, 0) \leq Z_{\overrightarrow{x}0} \leq (\leq, M)$.*

†$_3$ *If $Z_{\overrightarrow{x}\overleftarrow{y}} < (<, +\infty)$ then $(\leq, 0) \leq Z_{\overrightarrow{x}0} \leq (\leq, M)$.*

†$_4$ *Either $Z_{0\overrightarrow{x}} = (\leq, -\infty)$ or $(<, -M) \leq Z_{0\overrightarrow{x}} \leq (\leq, 0)$.*

†$_5$ *Either $Z_{0\overrightarrow{y}} = (\leq, -\infty)$ or $Z_{x0} + (<, -M) \leq Z_{x\overrightarrow{y}}$ for all $x \in X$ with $x \neq \overrightarrow{y}$.*

†$_6$ *Either $Z_{\overrightarrow{x}\overrightarrow{y}} \in \{(\leq, -\infty), (<, +\infty), (\leq, +\infty)\}$ or $(<, -M) \leq Z_{\overrightarrow{x}\overrightarrow{y}} \leq (\leq, M)$.*

Proof.

†$_1$ Assume that $Z_{\overrightarrow{x}0} = (<, +\infty)$ and let $y \neq \overrightarrow{x}$. Consider first the case $Z_{0y} = (\leq, -\infty)$, i.e., y is a prophecy clock which is undefined in Z. Then, since $\mathbb{G}(Z)$ is in normal form, we have $Z_{\overrightarrow{x}y} \leq Z_{\overrightarrow{x}0} + Z_{0y} = (<, +\infty) + (\leq, -\infty) = (\leq, -\infty)$.

The second case is when $Z_{0y} \neq (\leq, -\infty)$. This implies $Z_{\overrightarrow{x}y} \neq (\leq, -\infty)$ since otherwise we would get $Z_{0y} \leq Z_{0\overrightarrow{x}} + Z_{\overrightarrow{x}y} = (\leq, -\infty)$. We claim that there is a valuation $v \in Z$ with $-\infty < v(y)$ and $-\infty < v(\overrightarrow{x}) < -M$.

Consider the distance graph $\mathbb{G}'$ obtained from $\mathbb{G}(Z)$ by setting the weight of edge $y \to 0$ to $\min(Z_{y0}, (<, +\infty))$ and of edge $0 \to \overrightarrow{x}$ to $\min(Z_{0\overrightarrow{x}}, (<, -M))$. Notice that $\mathbb{G}'$ is in standard form. We show that $\mathbb{G}'$ has no negative cycles. Since $\mathbb{G}(Z)$ is in normal form, the candidates for being negative must use the new weight $(<, -M)$ of $0 \to \overrightarrow{x}$ or the new weight $(<, +\infty)$ of $y \to 0$ or both. This gives the cycle $0 \to \overrightarrow{x} \to 0$ with weight $(<, -M) + Z_{\overrightarrow{x}0} = (<, +\infty)$, the cycle $0 \to y \to 0$ with weight $Z_{0y} + (<, +\infty)$

which is not negative since $Z_{0y} \neq (\leq, -\infty)$, and the cycle $y \to 0 \to \overrightarrow{x} \to y$ with weight $(<, +\infty) + (<, -M) + Z_{\overrightarrow{x}y}$ which is not negative since $Z_{\overrightarrow{x}y} \neq (\leq, -\infty)$. Since $\mathbb{G}'$ is standard and has no negative cycle, Lemma 4.9 implies $[\![\mathbb{G}']\!] \neq \emptyset$. Note that $[\![\mathbb{G}']\!] \subseteq [\![\mathbb{G}(Z)]\!] = Z$. Finally, $-\infty < v(y)$ and $-\infty < v(\overrightarrow{x}) < -M$ for all $v \in \mathbb{G}'$, which proves the claim.

By Lemma 6.1, $v_\alpha = v[\overrightarrow{x} \mapsto \alpha] \in Z$ for all $-\infty < \alpha < -M$. Now, $v_\alpha(y) - v_\alpha(\overrightarrow{x}) = v(y) - \alpha$ satisfies the constraint $Z_{\overrightarrow{x}y}$. We deduce that $Z_{\overrightarrow{x}y}$ is either $(<, +\infty)$ or $(\leq, +\infty)$.

$\dagger_2$ Suppose $(\leq, M) < Z_{\overrightarrow{x}0} < (<, +\infty)$. By Lemma 4.4, we find α such that $(\leq, \alpha) \leq Z_{\overrightarrow{x}0}$, $(\leq, -\alpha) \leq Z_{0\overrightarrow{x}}$ and $\alpha \not\leq M$. Now, by Lemma 4.10, we find $v \in Z$ with $-\infty < v(\overrightarrow{x}) = -\alpha < -M$. By Lemma 6.1, $v[\overrightarrow{x} \mapsto \alpha] \in Z$ for all $-\infty < \alpha < -M$, a contradiction with $Z_{\overrightarrow{x}0}$ finite.

$\dagger_3$ Assume $Z_{\overrightarrow{x}\overleftarrow{y}} < (<, +\infty)$ is finite. Since $\mathbb{G}(Z)$ is standard, we get $Z_{\overrightarrow{x}0} \neq (\leq, +\infty)$. Now, $\dagger_1$ implies $Z_{\overrightarrow{x}0} < (<, +\infty)$. We conclude with $\dagger_2$.

$\dagger_4$ The proof is similar to $\dagger_2$. Suppose $(\leq, -\infty) < Z_{0\overrightarrow{x}} < (<, -M)$. By Lemma 4.4, we find α such that $(\leq, \alpha) \leq Z_{0\overrightarrow{x}}$, $(\leq, -\alpha) \leq Z_{\overrightarrow{x}0}$ and $\alpha \neq -\infty$. Now, by Lemma 4.10, we find $v \in Z$ with $-\infty < v(\overrightarrow{x}) = -\alpha < -M$. Since $Z_{0\overrightarrow{x}} < (<, -M)$, we can find $-\infty \neq \beta < -M$ such that $Z_{0\overrightarrow{x}} < (\leq, \beta)$. Now, Lemma 6.1 implies that $v[\overrightarrow{x} \mapsto \beta] \in Z$, which is a contradiction with $Z_{0\overrightarrow{x}} < (\leq, \beta)$.

$\dagger_5$ Suppose $Z_{0\overrightarrow{y}} \neq (\leq, -\infty)$. If $Z_{x0} = (\leq, -\infty)$ or $Z_{x\overrightarrow{y}} = (\leq, +\infty)$, the condition is trivially true. So we also assume in the following that $Z_{x0} \neq (\leq, -\infty)$ and $Z_{x\overrightarrow{y}} \neq (\leq, +\infty)$. Since $\mathbb{G}(Z)$ is in standard form, we get $Z_{x0} \neq (\leq, +\infty)$. By Lemma 4.20, we also have $Z_{x\overrightarrow{y}} \neq (\leq, -\infty)$.

Consider the distance graph $\mathbb{G}'$ obtained from the canonical graph $\mathbb{G}(Z)$ by setting $\mathbb{G}'_{0x} = \min(Z_{0x}, (<, +\infty))$, $\mathbb{G}'_{\overrightarrow{y}0} = \min(Z_{\overrightarrow{y}0}, (<, +\infty))$, and keeping other weights unchanged. The graph $\mathbb{G}'$ is in standard form. Moreover, it has no negative cycles. Indeed, since $\mathbb{G}(Z)$ is in normal form, the new cycles that we have to check in $\mathbb{G}'$ are $0 \to x \to 0$, $0 \to \overrightarrow{y} \to 0$ and $\overrightarrow{y} \to 0 \to x \to \overrightarrow{y}$. Since $Z_{x0} \neq (\leq, -\infty)$, $Z_{0\overrightarrow{y}} \neq (\leq, -\infty)$ and $Z_{x\overrightarrow{y}} \neq (\leq, -\infty)$, these cycles are non-negative. Let $\mathbb{G}''$ be the normalization of $\mathbb{G}'$. Since $Z_{x0} \neq (\leq, +\infty)$, we have $\mathbb{G}''_{x0} = \mathbb{G}'_{x0} = Z_{x0}$. Notice that $\mathbb{G}'' \leq \mathbb{G}' \leq \mathbb{G}(Z)$, hence $[\![\mathbb{G}'']\!] \subseteq [\![\mathbb{G}']\!] \subseteq Z$.

We claim that for all $\varepsilon > 0$ and all $c \in \mathbb{R}$ with $(\leq, c) < Z_{x0}$ we have $(\leq, c - M - \varepsilon) < Z_{x\overrightarrow{y}}$. Indeed, if $(\leq, c) < Z_{x0} = \mathbb{G}''_{x0}$ then by Lemma 4.4 we find α such that $(\leq, \alpha) \leq \mathbb{G}''_{x0}$, $(\leq, -\alpha) \leq \mathbb{G}''_{0x}$ and $c < \alpha$. Next, by Lemma 4.10, we find $v \in [\![\mathbb{G}'']\!] \subseteq Z$ such that $v(x) = -\alpha$. If $v(\overrightarrow{y}) < -M$ then by Lemma 6.1 we get $v' = v[\overrightarrow{y} \mapsto -M - \varepsilon] \in Z$. Otherwise, we let $v' = v$. We have $c - M - \varepsilon < \alpha - M - \varepsilon \leq v'(\overrightarrow{y}) - v'(x)$. We obtain $(\leq, c - M - \varepsilon) < (\leq, v'(\overrightarrow{y}) - v'(x)) \leq Z_{x\overrightarrow{y}}$ and the claim is proved.

Finally, if $Z_{x0} = (<, +\infty)$ then we may take c arbitrarily large and the claim implies $(<, +\infty) \leq Z_{x\overrightarrow{y}}$. Otherwise, $Z_{x0} = (\lhd, d)$ is finite and taking $c = d - \varepsilon$ we obtain $(\leq, d - M - 2\varepsilon) < Z_{x\overrightarrow{y}}$ for all $\varepsilon > 0$. This implies $Z_{x0} + (<, -M) = (<, d - M) \leq Z_{x\overrightarrow{y}}$.

$\dagger_6$ Assume that $Z_{\overrightarrow{x}\overrightarrow{y}} \notin \{(\leq, -\infty), (<, +\infty), (\leq, +\infty)\}$. Since $\mathbb{G}(Z)$ is in standard form, we have $Z_{\overrightarrow{x}0} \neq (\leq, +\infty)$. Now, $(\leq, -\infty) \neq Z_{\overrightarrow{x}\overrightarrow{y}} \leq Z_{\overrightarrow{x}0} + Z_{0\overrightarrow{y}}$ implies that $Z_{0\overrightarrow{y}} \neq (\leq, -\infty)$. Next, since $Z_{0\overrightarrow{y}} \neq (\leq, -\infty)$ and $Z_{\overrightarrow{x}\overrightarrow{y}} \notin \{(<, +\infty), (\leq, +\infty)\}$, by $\dagger_1$ we deduce that $Z_{\overrightarrow{x}0} \neq (<, +\infty)$.

Applying $\dagger_5$, we deduce that $(<, -M) \leq Z_{\overrightarrow{x}0} + (<, -M) \leq Z_{\overrightarrow{x}\overrightarrow{y}}$.

Finally, $Z_{\overrightarrow{x}\overrightarrow{y}} \leq Z_{\overrightarrow{x}0} + Z_{0\overrightarrow{y}} \leq Z_{\overrightarrow{x}0} \leq (\leq, M)$ by $\dagger_2$. $\qquad\square$

We now see that the (†) conditions imply that the weights of edges of the form $0 \to \overrightarrow{x}$, $\overrightarrow{x} \to 0$ and $\overrightarrow{x} \to \overrightarrow{y}$ belong to the finite set

$$\{(\leq, -\infty), (<, +\infty), (\leq, +\infty)\} \cup \{(\lhd, c) \mid c \in \mathbb{Z} \wedge -M \leq c \leq M\}.$$

For an example, see Figure 11. Note that the event-zone graph is sound and complete for reachability. Thus, we obtain as a corollary that, for event-predicting automata (EPA), we do not even need simulation to obtain finiteness.

Corollary 6.4. *Let $\mathcal{A}$ be an EPA. The zone graph $\mathsf{EZG}(\mathcal{A})$ is finite.*

Figure 11: Event-*predicting* automaton $\mathcal{A}_2$, for which there exists no finite time-abstract bisimulation on valuations, and its (finite) event-zone graph $\mathsf{EZG}(\mathcal{A}_2)$.

Thus, for EPA, one could simply construct the event-zone graph without simulations to solve the reachability problem: more precisely, the reachability algorithm of Definition 3.9 can be modified by taking equality $=$ instead of simulation $\preceq$. At the same time, notice that in the definition of $v \preceq_G v'$, we have $v(\overrightarrow{a}) = v'(\overrightarrow{a})$ for prophecy clocks. Therefore, for EPA, the check $v \preceq_G v'$ boils down to $v = v'$ and the check $Z \preceq_G Z'$ amounts to checking $Z \subseteq Z'$, an inclusion between zones.

We can also see that Theorem 5.9 when restricted to prophecy clocks boils down to mere inclusion. We provide a short explanation for this fact. The first two conditions of Theorem 5.9 simply check $Z'_{x0} < Z_{x0}$ and $Z'_{0y} < Z_{0y}$ when there are only prophecy clocks. In the third condition, it remains to show that the extra tests (1) $(\leq, 0) \leq Z_{x0} + (\lhd_1, c)$ and (2) $Z'_{xy} + (\lhd_2, -d) < Z_{x0}$ are true whenever $Z'_{xy} < Z_{xy}$, and x, y are prophecy clocks. So there is no need to check the extra conditions separately.

When x and y are prophecy clocks, we have $x \leq 0$ and $0 \leq y$ in G. By definition of standard form, we have $(\leq, 0) \leq Z_{x0}$. This automatically gives (1). For (2), since $Z'_{xy} < Z_{xy}$ and $Z_{xy} \leq Z_{x0} + Z_{0y}$, we have $Z'_{xy} < Z_{x0} + Z_{0y}$. But $Z_{0y} \leq (\leq, 0)$ since $\mathbb{G}(Z)$ is in standard form. Therefore, we deduce $Z_{xy} \leq Z'_{x0}$.

For general ECA, i.e., when we have both history and prophecy clocks, we need to make sure that the finiteness of the simulation that we do for handling history clocks does not get affected in the presence of prophecy clocks. We turn to this issue in the next section.

6.2. **An equivalence relation of finite index on valuations.** We define an equivalence relation of finite index $\sim_M$ on valuations. First, we define $\sim_M$ on $\alpha, \beta \in \overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, +\infty\}$ by $\alpha \sim_M \beta$ if $(\alpha \lhd c \Longleftrightarrow \beta \lhd c)$ for all $(\lhd, c)$ with $\lhd \in \{<, \leq\}$ and $c \in \{-\infty, +\infty\} \cup \{d \in \mathbb{Z} \mid |d| \leq M\}$. In particular, if $\alpha \sim_M \beta$ then $(\alpha = -\infty \Longleftrightarrow \beta = -\infty)$ and $(\alpha = +\infty \Longleftrightarrow \beta = +\infty)$.

Next, for valuations $v_1, v_2 \in \mathbb{V}$, we define $v_1 \sim_M v_2$ by two conditions: $v_1(x) \sim_M v_2(x)$ and $v_1(x) - v_1(y) \sim_{2M} v_2(x) - v_2(y)$ for all clocks $x, y \in X$. Notice that we use $2M$ for differences of values. Clearly, $\sim_M$ is an equivalence relation of finite index on valuations.

The next result relates the equivalence relation $\sim_M$ and the simulation relation $\preceq_G$ when the finite constants used in the constraints are bounded by M. Recall from Section 5.1 the definition of the distance graph $\mathbb{G}_G(v)$ for the set of valuations $\uparrow_G v$.

Lemma 6.5. *Let $v_1, v_2 \in \mathbb{V}$ be valuations with $v_1 \sim_M v_2$ and let G be a set of atomic constraints using constants in $\{-\infty, +\infty\} \cup \{c \in \mathbb{Z} \mid |c| \leq M\}$. By replacing the weights $(\leq, v_1(x))$ (resp. $(\leq, -v_1(x))$) by $(\leq, v_2(x))$ (resp. $(\leq, -v_2(x))$) in the graph $\mathbb{G}_G(v_1)$ we obtain the graph $\mathbb{G}_G(v_2)$.*

Proof. This is clear by definition for edges $\overrightarrow{x} \to 0$ and $0 \to \overrightarrow{x}$ adjacent to a prophecy clock. We consider now edges adjacent to history clocks $\overleftarrow{x}$.

- Consider the edge $0 \to \overleftarrow{x}$.
  - If its weight is $(\leq, v_1(\overleftarrow{x}))$ in $\mathbb{G}_G(v_1)$ then there is some $\overleftarrow{x} \lhd c \in G$ with $(\lhd, c) < (<, +\infty)$ and $v_1(\overleftarrow{x}) \lhd c$. Since $v_1 \sim_M v_2$ we deduce that $v_2(\overleftarrow{x}) \lhd c$ and the edge $0 \to \overleftarrow{x}$ has weight $(\leq, v_2(\overleftarrow{x}))$ in $\mathbb{G}_G(v_2)$.
  - If its weight is $(<, +\infty)$ in $\mathbb{G}_G(v_1)$ then we are not in the case above and $\overleftarrow{x} < +\infty \in G$, $v_1(\overleftarrow{x}) < +\infty$. Since $v_1 \sim_M v_2$ we deduce that $v_2(\overleftarrow{x}) < +\infty$ and the edge $0 \to \overleftarrow{x}$ has weight $(<, +\infty)$ in $\mathbb{G}_G(v_2)$.
  - Otherwise, the weight is $(\leq, +\infty)$ in both $\mathbb{G}_G(v_1)$ and $\mathbb{G}_G(v_2)$.
- Consider the edge $\overleftarrow{x} \to 0$.
  - If its weight is $(\leq, -\infty)$ in $\mathbb{G}_G(v_1)$ then $+\infty \leq \overleftarrow{x} \in G$ and $v_1(\overleftarrow{x}) = +\infty$. Since $v_1 \sim_M v_2$ we deduce that $v_2(\overleftarrow{x}) = +\infty$ and the edge $\overleftarrow{x} \to 0$ has weight $(\leq, -\infty)$ in $\mathbb{G}_G(v_2)$.
  - If its weight is $(\lhd, -c)$ in $\mathbb{G}_G(v_1)$ then we are not in the case above and there is some $c \lhd \overleftarrow{x} \in G$ with $(c, \lhd) < (+\infty, \leq)$ and $c \lhd v_1(\overleftarrow{x})$. Since $v_1 \sim_M v_2$ we deduce that $c \lhd v_2(\overleftarrow{x})$ and the edge $\overleftarrow{x} \to 0$ has weight $(\lhd, -c)$ in $\mathbb{G}_G(v_2)$.
  - If its weight is $(\leq, -v_1(\overleftarrow{x}))$ in $\mathbb{G}_G(v_1)$ then we are not in the cases above and there is some $c \lhd \overleftarrow{x} \in G$ with $(c, \lhd) < (+\infty, \leq)$ and $c \not\lhd v(\overleftarrow{x})$, Since $v_1 \sim_M v_2$ we deduce that $c \not\lhd v_2(\overleftarrow{x})$ and the edge $\overleftarrow{x} \to 0$ has weight $(\leq, -v_2(\overleftarrow{x}))$ in $\mathbb{G}_G(v_2)$.
  - Otherwise, the weight is $(\leq, 0)$ in both $\mathbb{G}_G(v_1)$ and $\mathbb{G}_G(v_2)$. $\square$

Next we state the central lemma that says that $\downarrow_G Z$ is a union of $\sim_M$ equivalence classes.

Lemma 6.6. *Let $v_1, v_2 \in \mathbb{V}$ be valuations with $v_1 \sim_M v_2$ and let G be a set of atomic constraints using constants in $\{-\infty, +\infty\} \cup \{c \in \mathbb{Z} \mid |c| \leq M\}$. Let Z be a zone with a canonical distance graph $\mathbb{G}(Z)$ satisfying $(\dagger)$. Then, $v_1 \in \downarrow_G Z$ iff $v_2 \in \downarrow_G Z$.*

Proof. Notice that $v \in \downarrow_G Z$ iff $\uparrow_G v \cap Z \neq \emptyset$. The proof is by contradiction. We assume that $\uparrow_G v_1 \cap Z \neq \emptyset$ and $\uparrow_G v_2 \cap Z = \emptyset$. By Lemma 5.8 we find a negative cycle C_2 using one edge from $\mathbb{G}(Z)$ and one or two edges from $\mathbb{G}_G(v_2)$. By Lemma 6.5, we have a corresponding cycle C_1 using the same edge from $\mathbb{G}(Z)$ and the same one or two edges from $\mathbb{G}_G(v_1)$. The

cycle C_1 is not negative since $\uparrow_G v_1 \cap Z \neq \emptyset$. Therefore, the negative cycle C_2 should use at least one edge labelled $(\leq, v_2(x))$ or $(\leq, -v_2(x))$. We consider the different cases.

(1) Cycle $C_2 = 0 \xrightarrow{(\leq, v_2(\overleftarrow{y}))} \overleftarrow{y} \xrightarrow{Z_{\overleftarrow{y}0}} 0$. We have $C_1 = 0 \xrightarrow{(\leq, v_1(\overleftarrow{y}))} \overleftarrow{y} \xrightarrow{Z_{\overleftarrow{y}0}} 0$.

Since we have the edge $0 \xrightarrow{(\leq, v_1(\overleftarrow{y}))} \overleftarrow{y}$ in $\mathbb{G}_G(v_1)$, there is a constraint $\overleftarrow{y} \triangleleft' c'$ in G with $(\triangleleft', c') < (<, +\infty)$ and $v_1(\overleftarrow{y}) \triangleleft' c'$. We deduce that $0 \leq v_1(\overleftarrow{y}) \leq M$.

Let $Z_{\overleftarrow{y}0} = (\triangleleft, c)$. Since C_1 is not a negative cycle, we get $(\leq, 0) \leq (\triangleleft, c + v_1(\overleftarrow{y}))$, which is equivalent to $-c \leq v_1(\overleftarrow{y})$. Using $0 \leq v_1(\overleftarrow{y}) \leq M$ and $v_1 \sim_M v_2$ we deduce that $-c \leq v_2(\overleftarrow{y})$. This is equivalent to $(\leq, 0) \leq (\triangleleft, c + v_2(\overleftarrow{y}))$, a contradiction with C_2 being a negative cycle.

(2) Cycle $C_2 = 0 \xrightarrow{Z_{0\overleftarrow{y}}} \overleftarrow{y} \xrightarrow{(\leq, -v_2(\overleftarrow{y}))} 0$. We have $C_1 = 0 \xrightarrow{Z_{0\overleftarrow{y}}} \overleftarrow{y} \xrightarrow{(\leq, -v_1(\overleftarrow{y}))} 0$.

Since we have the edge $\overleftarrow{y} \xrightarrow{(\leq, -v_1(\overleftarrow{y}))} 0$ in $\mathbb{G}_G(v_1)$, there is a constraint $c' \triangleleft' \overleftarrow{y}$ in G with $(c', \triangleleft') < (+\infty, \leq)$ and $c' \triangleleft' v_1(\overleftarrow{y})$. We deduce that $0 \leq v_1(\overleftarrow{y}) \leq M$.

Let $Z_{0\overleftarrow{y}} = (\triangleleft, c)$. Since C_1 is not a negative cycle, we get $(\leq, 0) \leq (\triangleleft, c - v_1(\overleftarrow{y}))$, which is equivalent to $v_1(\overleftarrow{y}) \leq c$. Using $v_1 \sim_M v_2$ and $0 \leq v_1(\overleftarrow{y}) \leq M$, we deduce that $v_2(\overleftarrow{y}) \leq c$. This is equivalent to $(\leq, 0) \leq (\triangleleft, c - v_2(\overleftarrow{y}))$, a contradiction with C_2 being a negative cycle.

(3) Cycle $C_2 = 0 \xrightarrow{(\leq, v_2(\overrightarrow{x}))} \overrightarrow{x} \xrightarrow{Z_{\overrightarrow{x}0}} 0$. We have $C_1 = 0 \xrightarrow{(\leq, v_1(\overrightarrow{x}))} \overrightarrow{x} \xrightarrow{Z_{\overrightarrow{x}0}} 0$.

Since C_2 is negative, we have $Z_{\overrightarrow{x}0} \neq (\leq, +\infty)$. Also, if $Z_{\overrightarrow{x}0} = (<, +\infty)$ then we must have $v_2(\overrightarrow{x}) = -\infty$, which implies $v_1(\overrightarrow{x}) = -\infty$ since $v_1 \sim_M v_2$, a contradiction with C_1 being non-negative. Hence, $Z_{\overrightarrow{x}0} = (\triangleleft, c) < (<, +\infty)$ and by $(\dagger_2)$, we infer $0 \leq c \leq M$.

Since C_1 is not negative, we get $(\leq, 0) \leq (\triangleleft, c + v_1(\overrightarrow{x}))$, which is equivalent to $-c \leq v_1(\overrightarrow{x})$. Using $v_1 \sim_M v_2$ and $0 \leq c \leq M$ we deduce that $-c \leq v_2(\overrightarrow{x})$. This is equivalent to $(\leq, 0) \leq (\triangleleft, c + v_2(\overrightarrow{x}))$, a contradiction with C_2 being a negative cycle.

(4) Cycle $C_2 = 0 \xrightarrow{Z_{0\overrightarrow{x}}} \overrightarrow{x} \xrightarrow{(\leq, -v_2(\overrightarrow{x}))} 0$. We have $C_1 = 0 \xrightarrow{Z_{0\overrightarrow{x}}} \overrightarrow{x} \xrightarrow{(\leq, -v_1(\overrightarrow{x}))} 0$.

Let $Z_{0\overrightarrow{x}} = (\triangleleft, c)$. Since C_2 is negative, we deduce that $v_2(\overrightarrow{x}) \neq -\infty$. Using $v_1 \sim_M v_2$, we infer $v_1(\overrightarrow{x}) \neq -\infty$. Since C_1 is not negative, we get $Z_{0\overrightarrow{x}} \neq (\leq, -\infty)$. From $(\dagger_4)$, we infer $(<, -M) \leq Z_{0\overrightarrow{x}}$ and $-M \leq c \leq 0$.

Since C_1 is not a negative cycle, we get $(\leq, 0) \leq (\triangleleft, c - v_1(\overrightarrow{x}))$, which is equivalent to $v_1(\overrightarrow{x}) \leq c$. Using $v_1 \sim_M v_2$ and $-M \leq c \leq 0$, we deduce that $v_2(\overrightarrow{x}) \leq c$. This is equivalent to $(\leq, 0) \leq (\triangleleft, c - v_2(\overrightarrow{x}))$, a contradiction with C_2 being a negative cycle.

(5) Cycle $C_2 = 0 \xrightarrow{(\leq, v_2(\overleftarrow{y}))} \overleftarrow{y} \xrightarrow{Z_{\overleftarrow{y}\overrightarrow{x}}} \overrightarrow{x} \xrightarrow{(\leq, -v_2(\overrightarrow{x}))} 0$.

We have $C_1 = 0 \xrightarrow{(\leq, v_1(\overleftarrow{y}))} \overleftarrow{y} \xrightarrow{Z_{\overleftarrow{y}\overrightarrow{x}}} \overrightarrow{x} \xrightarrow{(\leq, -v_1(\overrightarrow{x}))} 0$.

Let $Z_{\overleftarrow{y}\overrightarrow{x}} = (\triangleleft, c)$. As in case 1 above, we get $0 \leq v_1(\overleftarrow{y}) \leq M$. From the fact that the cycle $0 \xrightarrow{(\leq, v_1(\overleftarrow{y}))} \overleftarrow{y} \xrightarrow{Z_{\overleftarrow{y}0}} 0$ is not negative, we get $(\leq, -M) \leq Z_{\overleftarrow{y}0}$. Since C_2 is negative, we get $v_2(\overrightarrow{x}) \neq -\infty$. Using $v_1 \sim_M v_2$, we infer $v_1(\overrightarrow{x}) \neq -\infty$. From the fact that the cycle $0 \xrightarrow{Z_{0\overrightarrow{x}}} \overrightarrow{x} \xrightarrow{(\leq, -v_1(\overrightarrow{x}))} 0$ is not negative, we deduce $Z_{0\overrightarrow{x}} \neq (\leq, -\infty)$. Using $(\dagger_5)$ we obtain

$$(\leq, -M) + (<, -M) \leq Z_{\overleftarrow{y}0} + (<, -M) \leq Z_{\overleftarrow{y}\overrightarrow{x}} = (\triangleleft, c)$$

and we deduce that $-2M \leq c \leq 0$.

Since C_1 is not a negative cycle, we get $(\leq, 0) \leq (\lhd, c + v_1(\overleftarrow{y}) - v_1(\overrightarrow{x}))$, which is equivalent to $v_1(\overrightarrow{x}) - v_1(\overleftarrow{y}) \leq c$. Using $v_1 \sim_M v_2$ and $-2M \leq c \leq 0$ we deduce that $v_2(\overrightarrow{x}) - v_2(\overleftarrow{y}) \leq c$. We conclude as in the previous cases.

(6) Cycle $C_2 = 0 \xrightarrow{(\leq, v_2(\overrightarrow{x}))} \overrightarrow{x} \xrightarrow{Z_{\overrightarrow{x}\,\overleftarrow{y}}} \overleftarrow{y} \xrightarrow{(\leq, -v_2(\overleftarrow{y}))} 0$.

We have $C_1 = 0 \xrightarrow{(\leq, v_1(\overrightarrow{x}))} \overrightarrow{x} \xrightarrow{Z_{\overrightarrow{x}\,\overleftarrow{y}}} \overleftarrow{y} \xrightarrow{(\leq, -v_1(\overleftarrow{y}))} 0$.

Since C_2 is negative but not C_1, we get first $Z_{\overrightarrow{x}\,\overleftarrow{y}} \neq (\leq, +\infty)$ and then $v_1(\overrightarrow{x}) \neq -\infty$. As in case 2 above, we get $0 \leq v_1(\overleftarrow{y}) \leq M$. We deduce that $Z_{\overrightarrow{x}\,\overleftarrow{y}} = (\lhd, c) < (<, +\infty)$ and $c \neq +\infty$. From $(\dagger_3)$ we obtain $Z_{\overrightarrow{x}0} \leq (\leq, M)$. Since $0 \xrightarrow{(\leq, v_1(\overrightarrow{x}))} \overrightarrow{x} \xrightarrow{Z_{\overrightarrow{x}0}} 0$ is not a negative cycle, we get $-M \leq v_1(\overrightarrow{x}) \leq 0$. Finally, we obtain $0 \leq v_1(\overleftarrow{y}) - v_1(\overrightarrow{x}) \leq 2M$.

Since C_1 is not a negative cycle, we get $(\leq, 0) \leq (\lhd, c + v_1(\overrightarrow{x}) - v_1(\overleftarrow{y}))$, which is equivalent to $v_1(\overleftarrow{y}) - v_1(\overrightarrow{x}) \leq c$. Using $v_1 \sim_M v_2$ and $0 \leq v_1(\overleftarrow{y}) - v_1(\overrightarrow{x}) \leq 2M$, we deduce that $v_2(\overleftarrow{y}) - v_2(\overrightarrow{x}) \leq c$. We conclude as in the previous cases.

(7) Cycle $C_2 = 0 \xrightarrow{(\leq, v_2(\overrightarrow{x}))} \overrightarrow{x} \xrightarrow{Z_{\overrightarrow{x}\,\overleftarrow{y}}} \overleftarrow{y} \xrightarrow{(\lhd', -c')} 0$.

We have $C_1 = 0 \xrightarrow{(\leq, v_1(\overrightarrow{x}))} \overrightarrow{x} \xrightarrow{Z_{\overrightarrow{x}\,\overleftarrow{y}}} \overleftarrow{y} \xrightarrow{(\lhd', -c')} 0$.

Let $Z_{\overrightarrow{x}\,\overleftarrow{y}} = (\lhd, c)$. As in case 6 above, we show that $c \neq +\infty$ and $-M \leq v_1(\overrightarrow{x}) \leq 0$. Since C_1 is not a negative cycle, we get $0 \leq c + v_1(\overrightarrow{x}) - c'$, which is equivalent to $c' - c \leq v_1(\overrightarrow{x})$. Using $v_1 \sim_M v_2$ and $-M \leq v_1(x) \leq 0$, we deduce that $c' - c \leq v_2(\overrightarrow{x})$, which is equivalent to $0 \leq c + v_2(\overrightarrow{x}) - c'$, a contradiction with C_2 being a negative cycle.

(8) Cycle $C_2 = 0 \xrightarrow{(\leq, v_2(\overrightarrow{x}))} \overrightarrow{x} \xrightarrow{Z_{\overrightarrow{x}\,\overrightarrow{y}}} \overrightarrow{y} \xrightarrow{(\leq, -v_2(\overrightarrow{y}))} 0$ with $x \neq y$.

We have $C_1 = 0 \xrightarrow{(\leq, v_1(\overrightarrow{x}))} \overrightarrow{x} \xrightarrow{Z_{\overrightarrow{x}\,\overrightarrow{y}}} \overrightarrow{y} \xrightarrow{(\leq, -v_1(\overrightarrow{y}))} 0$.

Since C_2 is negative but not C_1, using $v_1 \sim_M v_2$ we get successively $Z_{\overrightarrow{x}\,\overrightarrow{y}} \neq (\leq, +\infty)$, $v_2(\overrightarrow{y}) \neq -\infty \neq v_1(\overrightarrow{y})$, $v_1(\overrightarrow{x}) \neq -\infty \neq v_2(\overrightarrow{x})$, and finally $(\leq, -\infty) < Z_{\overrightarrow{x}\,\overrightarrow{y}} < (<, +\infty)$.

Let $Z_{\overrightarrow{x}\,\overrightarrow{y}} = (\lhd, c)$. From $(\dagger_6)$, we deduce that $-M \leq c \leq M$.

Since C_1 is not a negative cycle, we get $(\leq, 0) \leq (\lhd, c + v_1(\overrightarrow{x}) - v_1(\overrightarrow{y}))$, which is equivalent to $v_1(\overrightarrow{y}) - v_1(\overrightarrow{x}) \leq c$. Using $v_1 \sim_M v_2$ and $-M \leq c \leq M$, we deduce that $v_2(\overrightarrow{y}) - v_2(\overrightarrow{x}) \leq c$. We conclude as in the previous cases.

(9) Cycle $C_2 = 0 \xrightarrow{(\leq, v_2(\overleftarrow{x}))} \overleftarrow{x} \xrightarrow{Z_{\overleftarrow{x}\,\overleftarrow{y}}} \overleftarrow{y} \xrightarrow{(\leq, -v_2(\overleftarrow{y}))} 0$ with $x \neq y$.

We have $C_1 = 0 \xrightarrow{(\leq, v_1(\overleftarrow{x}))} \overleftarrow{x} \xrightarrow{Z_{\overleftarrow{x}\,\overleftarrow{y}}} \overleftarrow{y} \xrightarrow{(\leq, -v_1(\overleftarrow{y}))} 0$.

As in case 1 above, we get $0 \leq v_1(\overleftarrow{x}) \leq M$. As in case 2 above, we get $0 \leq v_1(\overleftarrow{y}) \leq M$. We obtain $-M \leq v_1(\overleftarrow{y}) - v_1(\overleftarrow{x}) \leq M$.

Let $Z_{\overleftarrow{x}\,\overleftarrow{y}} = (\lhd, c)$. Since C_1 is not negative, we get $(\leq, 0) \leq (\lhd, c + v_1(\overleftarrow{x}) - v_1(\overleftarrow{y}))$, which is equivalent to $v_1(\overleftarrow{y}) - v_1(\overleftarrow{x}) \leq c$. Using $v_1 \sim_M v_2$ and $-M \leq v_1(\overleftarrow{y}) - v_1(\overleftarrow{x}) \leq M$, we deduce that $v_2(\overleftarrow{y}) - v_2(\overleftarrow{x}) \leq c$. We conclude as in the previous cases.

(10) Cycle $C_2 = 0 \xrightarrow{(\leq, v_2(\overleftarrow{x}))} \overleftarrow{x} \xrightarrow{Z_{\overleftarrow{x}\,\overleftarrow{y}}} \overleftarrow{y} \xrightarrow{(\lhd', -c')} 0$ with $x \neq y$.

We have $C_1 = 0 \xrightarrow{(\leq, v_1(\overleftarrow{x}))} \overleftarrow{x} \xrightarrow{Z_{\overleftarrow{x}\,\overleftarrow{y}}} \overleftarrow{y} \xrightarrow{(\lhd', -c')} 0$.

As in case 1 above, we get $0 \leq v_1(\overleftarrow{x}) \leq M$.

Let $Z_{\overleftarrow{x}\,\overleftarrow{y}} = (\lhd, c)$. Since C_1 is not negative, we get $0 \leq c + v_1(\overleftarrow{x}) - c'$, which is equivalent to $c' - c \leq v_1(\overleftarrow{x})$. Using $v_1 \sim_M v_2$ and $0 \leq v_1(\overleftarrow{x}) \leq M$, we deduce that $c' - c \leq v_2(\overleftarrow{x})$, which is equivalent to $0 \leq c + v_2(\overleftarrow{x}) - c'$, a contradiction with C_2 being a negative cycle.

Notice that we have crucially used the "$2M$" occurring in the definition of $v_1 \sim_M v_2$ (as $v_1(x) - v_1(y) \sim_{2M} v_2(x) - v_2(y)$) in the cases where we deal with cycles containing one prophecy clock and one history clock (Cases 5 and 6). In the rest of the cases, it was sufficient to use $v_1(x) \sim_M v_2(x)$. $\qquad\square$

Finally, from Lemmas 6.3 and 6.6, we obtain our main theorem of the section.

Theorem 6.7. *The simulation relation $\preceq_{\mathcal{A}}$ is finite.*

Proof. Let $(q, Z_0), (q, Z_1), (q, Z_2), \ldots$ be an infinite sequence of *reachable* nodes in $\mathsf{EZG}(\mathcal{A})$. By Lemma 6.3, for all i, the distance graph $\mathbb{G}(Z_i)$ in canonical form satisfies conditions (†).

The atomic constraints in $G = \mathcal{G}(q)$ use constants in $\{-\infty, +\infty\} \cup \{c \in \mathbb{Z} \mid |c| \leq M\}$. From Lemma 6.6 we deduce that for all i, $\downarrow_G Z_i$ is a union of $\sim_M$-classes. Since $\sim_M$ is of finite index, there are only finitely many unions of $\sim_M$-classes. Therefore, we find $i < j$ with $\downarrow_G Z_i = \downarrow_G Z_j$, which implies $Z_j \preceq_G Z_i$. $\qquad\square$

Note that the number of enumerated zones is bounded by 2^r where r is the number of equivalence classes of $\sim_M$. This is similar to the known upper bound in classical timed automata, where instead of r, we can use the number of regions as defined in [AD94]. Indeed, despite this blow up the interest in zone algorithms is that, at least in the timed setting, they work significantly better in practice. We hope the above zone-based approach for ECA will also pave the way for fast implementations for ECA.

7. Conclusion

In this paper, we propose a simulation-based approach for reachability in ECAs. The main difficulty and difference from timed automata is the use of prophecy clocks and undefined values. We believe that the crux of our work has been in identifying the new representation for prophecy clocks and undefined values. With this as the starting point, we have been able to adapt the zone graph computation and the $\mathcal{G}$-simulation technique to the ECA setting. This process required us to closely study the mechanics of prophecy clocks in the zone computations and we discovered this surprising property that prophecy clocks by themselves do not create a problem for finiteness.

The final reachability algorithm looks almost identical to the timed automata counterpart and hence provides a mechanism to transfer timed automata technology to the ECA setting. As a follow-up to this work, we have extended our reachability algorithm to a more general model, called *Generalized timed automaton (GTA)* [AGG$^+$23] (which subsumes event-clock automata). In [AGG$^+$23], we also developed a prototype implementation of our algorithm in TCHECKER, an open-source platform for timed automata analysis. To the best of our knowledge, this is the first tool that can handle event-clock automata, and it provides a way to efficiently check event-clock specifications on timed automata models.

References

[ABG13]   S. Akshay, Benedikt Bollig, and Paul Gastin. Event clock message passing automata: a logical characterization and an emptiness checking algorithm. *Formal Methods Syst. Des.*, 42(3):262–300, 2013.

[AD94]   Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126:183–235, 1994.

[AFH99] Rajeev Alur, Limor Fix, and Thomas A. Henzinger. Event-clock automata: A determinizable class of timed automata. *Theor. Comput. Sci.*, 211(1-2):253–273, 1999.

[AGG⁺23] S. Akshay, Paul Gastin, R. Govind, Aniruddha R. Joshi, and B. Srivathsan. A unified model for real-time systems: Symbolic techniques and implementation. In *CAV (1)*, volume 13964 of *Lecture Notes in Computer Science*, pages 266–288. Springer, 2023.

[AGGS22] S. Akshay, Paul Gastin, R. Govind, and B. Srivathsan. Simulations for event-clock automata. In *CONCUR*, volume 243 of *LIPIcs*, pages 13:1–13:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[AGP21] S. Akshay, Paul Gastin, and Karthik R. Prakash. Fast zone-based algorithms for reachability in pushdown timed automata. In *CAV (1)*, volume 12759 of *Lecture Notes in Computer Science*, pages 619–642. Springer, 2021.

[BCM16] Patricia Bouyer, Maximilien Colange, and Nicolas Markey. Symbolic optimal reachability in weighted timed automata. In *CAV (1)*, volume 9779 of *Lecture Notes in Computer Science*, pages 513–530. Springer, 2016.

[BDL⁺06] Gerd Behrmann, Alexandre David, Kim Guldstrand Larsen, John Hakansson, Paul Pettersson, Wang Yi, and Martijn Hendriks. UPPAAL 4.0. In *QEST*, pages 125–126. IEEE Computer Society, 2006.

[BDM⁺98] Marius Bozga, Conrado Daws, Oded Maler, Alfredo Olivero, Stavros Tripakis, and Sergio Yovine. Kronos: A model-checking tool for real-time systems. In *CAV*, volume 1427 of *Lecture Notes in Computer Science*, pages 546–550. Springer, 1998.

[Bou04] Patricia Bouyer. Forward analysis of updatable timed automata. *Formal Methods Syst. Des.*, 24(3):281–320, 2004.

[BY03] Johan Bengtsson and Wang Yi. Timed automata: Semantics, algorithms and tools. In *ACPN 2003*, volume 3098 of *Lecture Notes in Computer Science*, pages 87–124. Springer, 2003.

[Dil89] David L. Dill. Timing assumptions and verification of finite-state concurrent systems. In *Automatic Verification Methods for Finite State Systems*, volume 407 of *Lecture Notes in Computer Science*, pages 197–212. Springer, 1989.

[DT98] Conrado Daws and Stavros Tripakis. Model checking of real-time reachability properties using abstractions. In *TACAS*, volume 1384 of *Lecture Notes in Computer Science*, pages 313–329. Springer, 1998.

[DT04] Deepak D'Souza and Nicolas Tabareau. On timed automata with input-determined guards. In *FORMATS/FTRTFT*, volume 3253 of *Lecture Notes in Computer Science*, pages 68–83. Springer, 2004.

[GMS18] Paul Gastin, Sayan Mukherjee, and B. Srivathsan. Reachability in timed automata with diagonal constraints. In *CONCUR*, volume 118 of *LIPIcs*, pages 28:1–28:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[GMS19] Paul Gastin, Sayan Mukherjee, and B. Srivathsan. Fast algorithms for handling diagonal constraints in timed automata. In *CAV (1)*, volume 11561 of *Lecture Notes in Computer Science*, pages 41–59. Springer, 2019.

[GMS20] Paul Gastin, Sayan Mukherjee, and B. Srivathsan. Reachability for updatable timed automata made faster and more effective. In *FSTTCS*, volume 182 of *LIPIcs*, pages 47:1–47:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[GRS11] Gilles Geeraerts, Jean-François Raskin, and Nathalie Sznajder. Event clock automata: From theory to practice. In *FORMATS*, volume 6919 of *Lecture Notes in Computer Science*, pages 209–224. Springer, 2011.

[GRS14] Gilles Geeraerts, Jean-François Raskin, and Nathalie Sznajder. On regions and zones for event-clock automata. *Formal Methods Syst. Des.*, 45(3):330–380, 2014.

[HP19] Frédéric Herbreteau and Gerald Point. TChecker. https://github.com/fredher/tchecker, v0.2 - April 2019.

[HSW12] Frédéric Herbreteau, B. Srivathsan, and Igor Walukiewicz. Better abstractions for timed automata. In *LICS*, pages 375–384. IEEE Computer Society, 2012.

[HSW13] Frédéric Herbreteau, B. Srivathsan, and Igor Walukiewicz. Lazy abstractions for timed automata. In *CAV*, volume 8044 of *Lecture Notes in Computer Science*, pages 990–1005. Springer, 2013.

[KLM⁺15] Gijs Kant, Alfons Laarman, Jeroen Meijer, Jaco van de Pol, Stefan Blom, and Tom van Dijk. LTSmin: High-performance language-independent model checking. In *TACAS*, volume 9035 of *Lecture Notes in Computer Science*, pages 692–707. Springer, 2015.

[KWNP08] Sebastian Kupferschmid, Martin Wehrle, Bernhard Nebel, and Andreas Podelski. Faster than UPPAAL? In *CAV*, volume 5123 of *Lecture Notes in Computer Science*, pages 552–555. Springer, 2008.

[LPY97] Kim Guldstrand Larsen, Paul Pettersson, and Wang Yi. UPPAAL in a nutshell. *STTT*, 1(1-2):134–152, 1997.

[RS99] Jean-François Raskin and Pierre-Yves Schobbens. The logic of event clocks - decidability, complexity and expressiveness. *J. Autom. Lang. Comb.*, 4(3):247–282, 1999.

[RSM19] Victor Roussanaly, Ocan Sankur, and Nicolas Markey. Abstraction refinement algorithms for timed automata. In *CAV (1)*, volume 11561 of *Lecture Notes in Computer Science*, pages 22–40. Springer, 2019.

[SLDP09] Jun Sun, Yang Liu, Jin Song Dong, and Jun Pang. PAT: towards flexible verification under fairness. In *CAV*, volume 5643 of *Lecture Notes in Computer Science*, pages 709–714. Springer, 2009.

[THV⁺17] Tamás Tóth, Ákos Hajdu, András Vörös, Zoltán Micskei, and István Majzik. Theta: a framework for abstraction refinement-based model checking. In Daryl Stewart and Georg Weissenbacher, editors, *Proceedings of the 17th Conference on Formal Methods in Computer-Aided Design*, pages 176–179, 2017.

[Wan06] Farn Wang. REDLIB for the formal verification of embedded systems. In *ISoLA*, pages 341–346. IEEE Computer Society, 2006.

[ZLZ05] Jianhua Zhao, Xuandong Li, and Guoliang Zheng. A quadratic-time DBM-based successor algorithm for checking timed automata. *Inf. Process. Lett.*, 96(3):101–105, 2005.