The Surprising Robustness of (Closed) Timed Automata against Clock-Drift

Mani Swaminathan[1,2], Martin Fränzle[1], and Joost-Pieter Katoen[3]

[1] University of Oldenburg {mani.swaminathan,fraenzle}@informatik.uni-oldenburg.de
[2] ASKON Consulting Group GmbH
[3] RWTH Aachen University katoen@cs.rwth-aachen.de

Abstract. We investigate reachability (or equivalently, safety) for timed systems modelled as Timed Automata (TA) under notions of "robustness", i.e., when the clocks of the TA may drift by small amounts. Our contributions are two-fold: (1) We first consider the model of clock-drift introduced by Puri [1] and subsequently studied in other works [2, 3, 4]. We show that the standard zone-based forward reachability analysis performed by tools such as UPPAAL is in fact exact for TA with closed guards, invariants, and targets, when testing robust safety of timed systems having an arbitrary, but finite lifetime. (2) Next, we consider a more realistic model of drifting clocks that takes into account the regular resynchronization performed in most practical systems. We then show that the standard reachability analysis of tools like UPPAAL again suffices to test for robust safety in this model of clock-drift, for TA with closed guards, invariants, and targets, but now without any restrictions on system life-time.

1 Introduction

Real-time systems, which have strict timing requirements, have emerged as an enabling technology for several important application domains such as air traffic control, telecommunications, and medicine, to name a few. Such systems are becoming increasingly pervasive, and hence rigorous methods and techniques to ensure their correct functioning are of utmost importance. Timed Automata (TA) [5] have been extensively studied as a formalism for modelling real-time systems. TA extend ω-automata by augmenting them with "clock" variables based on a dense-time model, which quantitatively capture the behaviour of the system with time. TA model checkers such as UPPAAL [10] and KRONOS [7] are now available and have been successfully used in several industrial case studies, such as [8].

A key result for the decidability properties of TA is the region-automaton construction [5], which partitions the inherently infinite state space of the TA into finitely many equivalence classes or "regions". The number of such regions is, however, exponential in the number of clocks, and the region construction is therefore not suited in practice for model checking TA when the number of clocks is large. Most available tools for model checking TA (such as UPPAAL) instead use on-the-fly algorithms that dynamically search through the

Please use the following format when citing this chapter:

Swaminathan, M., Fränzle, M. and Katoen, J.-P., 2008, in IFIP International Federation for Information Processing, Volume 273; Fifth IFIP International Conference on Theoretical Computer Science; Giorgio Ausiello, Juhani Karhumäki, Giancarlo Mauri, Luke Ong; (Boston: Springer), pp. 537–553.

state space of the TA, which is partitioned into "zones" [10]. Associated data structures such as Difference Bound Matrices (DBMs) [10] are used to represent zones in TA-based verification. Reachability analysis forms the core of such verification tools [9] and is implemented by a Forward Reachability Analysis (FRA) algorithm that computes the set of successors of a zone, with termination being enforced by zone-widening using k-normalization [10].

However, such analyses, whether region- or zone-based, assume that the clocks of the TA are perfectly synchronous, which is not the case in practice, where the clocks could drift by small amounts. It is shown in [1] that the usual region-based analysis is not correct w.r.t. reachability when considering perturbations in the clocks, in the sense that an unsafe state, reported as unreachable for perfect clocks, might well be reachable by iterating often enough through a cycle in the TA, even when the clocks drift by infinitesimally small amounts, and such a TA is therefore not "robustly safe". This insight leads to the definition of robust reachability, where a reachability property is considered to be "robustly (in-)valid" iff it does not change its validity for some small relative drift between clocks.

"Robust" reachability analysis [1, 2] therefore computes the set of states that are reachable for *every* (i.e., even the slightest) drift, reporting the TA as not being robustly safe iff that enlarged reach-set contains an unsafe state (where the guards and invariants of the TA, and the unsafe target state, are all assumed to be closed). Robust reach-set computation in [1, 2] is based on searching the strongly connected components of the region-graph, thus suffering from the exponential size of the region-graph in the number of clocks. Zone-based algorithms that compute this reach-set more efficiently are presented in [3, 4]. For a given TA with maximum clock-drift parameterized by $\varepsilon > 0$, with the corresponding reachable state-space being $Reach^{\varepsilon}$, the algorithms in [1, 2, 3] compute the set $\cap_{\varepsilon>0} Reach^{\varepsilon}$ and test it for empty intersection with the (closed) target. It is shown in [1] that $\cap_{\varepsilon>0} Reach^{\varepsilon}$ has an empty intersection with the closed target state iff there exists some $\varepsilon > 0$ such that the intersection of $Reach^{\varepsilon}$ with the (closed) target state is again empty. The algorithms in all these works however alternate between forward and backward analysis, and thus induce a performance overhead compared to the standard FRA algorithm used within tools like UPPAAL. All the above works (except [4]) assume that the guards, invariants, and targets of the TA are closed. Furthermore, all of them assume that each cycle of the TA is a *progress cycle*, wherein every clock is reset at-least once per cycle. The unsafe states that become reachable with drifting clocks (but which are unreachable with perfect clocks) are added to such robust reach-sets only by iterating an *unbounded* number of times through the (progress) cycles of the TA, thereby requiring that the *life-time of the systems be infinite*. Moreover, the model of clock-drift considered in these works is one of *unbounded* relative drift between the clocks, which does not take into account the *regular resynchronization* of clocks that is performed in practical real-time systems. This paper addresses these two issues, with two main contributions:

1. We first consider the model of clock-drift introduced by Puri [1] and studied subsequently by others [2, 3, 4]. We show that, under the assumption of closed guards, invariants, and targets, the standard zone-based FRA of TA performed by tools such as UPPAAL is indeed exact when testing for robust safety of timed systems having an *arbitrary, but finite* life-time. We test here whether the TA can robustly avoid the target arbitrarily long, in the following sense: for any given number i of iterations of the transition relation, there is $\varepsilon_i > 0$ such that $Reach_i^{\varepsilon_i}$ has an empty intersection with the target state, where $Reach_i^{\varepsilon_i}$ is the reachable state space after i iterations of the transition relation under maximum perturbation ε_i of the clocks. Note that ε_i may depend on the number i of executed iterations, with ε_i decreasing (not necessarily strictly) with i, and potentially tending to 0 as i tends to ∞. Thus, robust safety under our notion does not imply the existence of a homogeneous $\varepsilon > 0$ that is independent of the number of iterations and such that $Reach^{\varepsilon}$ has an empty intersection with the target state (which is the notion considered in previous works [1, 2, 3, 4]). However, our notion of robust safety implies avoidance of the target state by some strictly positive value of the perturbation for any *arbitrary, but finite* number of iterations. This is applicable to all systems having a finite life-time.
2. Next, we introduce a more realistic model of clock-drift that takes into account the *regular resynchronization* performed in practical real-time systems (such as bit-stuffing in communication protocols), which results in a *bounded* relative clock-drift. Under the assumption of closed guards, invariants, and targets, we show that the standard zone-based FRA of TA is again exact when testing for robust safety of such timed systems with clock resynchronization. In this case, a certification of robust safety imposes no restriction on the life-time of the system — it implies avoidance of the (closed) target by all $0 < \varepsilon < 1$ (where the ε now parameterizes the maximum relative bounded clock-drift subject to periodic resynchronization) independent of the number of iterations.

The rest of the paper is organized as follows: Section 2 briefly reviews TA definitions and semantics, along with our assumptions. It also presents the standard algorithm for zone-based FRA. Section 3 describes the robustness problem for TA in the context of the model of clock-drift considered by Puri and others, and shows the exactness of the standard zone-based FRA algorithm w.r.t robust safety for systems having a finite life-time. Section 4 then introduces our model of bounded clock-drift that accounts for regular clock resynchronization, and shows the exactness of the standard zone-based FRA algorithm w.r.t robust safety, but now without any restrictions on the life-time of the system. Section 5 concludes the paper and sketches future research directions.

2 Timed Automata (TA)

Given a finite set C of *clocks*, a *clock valuation* over C is a map $v : C \to \mathbb{R}_{\geq 0}$ that assigns a non-negative real value to each clock in C. If n is the number of clocks, a clock valuation is basically a point in $\mathbb{R}^n_{\geq 0}$, which we henceforth denote by $\mathbf{u}, \mathbf{v}$ etc.

Definition 1. A *zone* over a set of clocks C is a system of constraints defined by the grammar $g ::= x \triangleright d \mid x - y \triangleright d \mid g \wedge g$, where $x, y \in C$, $d \in \mathbb{N}$, and $\triangleright \in \{<, \leq, >, \geq\}$. The set of zones over C is denoted $Z(C)$.

A *closed zone* is one in which $\triangleright \in \{\leq, \geq\}$, and we denote the set of closed zones over C by $Z_c(C)$. A zone with no bounds on clock differences (i.e., with no constraint of the form $x - y \triangleright d$) is said to be *diagonal-free*, and we denote the corresponding set of zones by $Z_d(C)$. The set $Z_{cd}(C)$ denotes zones that are *both closed and diagonal-free*. The set $Z_{cdU}(C)$ denotes the set of *closed, diagonal-free zones having no lower bounds on the clocks*.

Definition 2. A TA is a tuple $A = (L, C, (l_0, \mathbf{0}), T, Inv)$, with

- a finite set L of *locations* and a finite set C of *clocks*, with $|C| = n$.
- An *initial* location $l_0 \in L$ together with the initial clock-valuation $\mathbf{0}$ where all clocks are set to 0 [1]
- a set $T \subseteq L \times Z_{cd}(C) \times 2^C \times L$ of possible *transitions* between locations. A transition t between two locations (l, l') is denoted $l \xrightarrow{t} l'$, and involves a *guard* $G(t) \in Z_{cd}(C)$ and a *reset set* $Res_t \subseteq C$.
- $Inv : L \to Z_{cdU}(C)$ assigns *invariants* to locations

In the sequel, we will denote by k the *clock ceiling* of the TA A under investigation, which is the largest constant among the constraints of A (including the predicate defining the unsafe state). Note that we assume that the guards of the automaton are *closed and diagonal-free zones*. Invariants in addition have only *upper-bounds* on clocks. Diagonal constraints of the form $x - y \triangleright d$ thus are not part of the TA syntax, but are of relevance, since they occur during the course of forward reachability analysis as a result of the *time-passage* operation defined as follows:

Definition 3. For a clock valuation $\mathbf{x}$, its *time-passage* is $timepass(\mathbf{x}) = \{\mathbf{x} + d \mid d > 0\}$, where $\mathbf{x} + d$ denotes the addition of a strictly positive scalar d to each component of $\mathbf{x}$. This is canonically lifted to clock-zones Z as $timepass(Z) = \bigcup_{\mathbf{x} \in Z} timepass(\mathbf{x})$.

Definition 4. $\lfloor \mathbf{x} \rfloor_k$ denotes the k-region containing $\mathbf{x}$, which is the equivalence class induced by the *k-region-equivalence relation* $\approx_k$. For two clock valuations $\mathbf{x}$ and $\mathbf{y}$, $\mathbf{x} \approx_k \mathbf{y}$ iff

[1] We assume without loss of generality that all clocks are initially set to 0.

$$\forall i \leq n : \left(\begin{array}{l} (x_i > k) \wedge (y_i > k) \\ \vee\ ((int(x_i) = int(y_i)) \wedge (fr(x_i) = 0 \Leftrightarrow fr(y_i) = 0) \wedge \\ \quad \forall j \leq n :\ \ (fr(x_i) \leq fr(x_j) \Leftrightarrow fr(y_i) \leq fr(y_j))) \end{array} \right)$$

Here, for a clock valuation $\mathbf{x} \in \mathbb{R}^n_{\geq 0}$, x_i denotes its i-th component, i.e., the value of the i-th clock, and $int(x_i)$ and $fr(x_i)$ respectively denote the integer and fractional parts of x_i.

Definition 5. [11] A *k-bounded zone (k-zone)* has no constant exceeding k among its constraints. For a zone Z, its *k-normalization*, denoted $norm_k(Z)$, is the smallest k-bounded zone containing Z.

If Z is a k-zone, $norm_k(Z) = Z$. k is taken to be the largest constant appearing in the constraints (including the unsafe state) of the TA.

Definition 6. $Reach \subseteq L \times (C \to \mathbb{R}_{\geq 0})$ is the reach-set of the TA A, consisting of an infinite set of (concrete) states of the TA of the form $(l, \mathbf{x})$, where $l \in L$ and $\mathbf{x} \in \mathbb{R}^n_{\geq 0}$. It is defined inductively as follows, with $Reach_i$ denoting the reach-set under $i \in \mathbb{N}$ steps, starting from the initial state $(l_0, \mathbf{0})$ and alternating between time-passage and discrete-location transitions:[2]

- $Reach_0 = \{(l_0, \mathbf{0})\}$.
  - if i even $Succ(Reach_i) = \left\{ (l, \mathbf{x}) \left| \begin{array}{l} \exists \mathbf{u} \in Inv(l) : (l, \mathbf{u}) \in Reach_i \\ \wedge\ \ \mathbf{x} \in timepass(\mathbf{u}) \cap Inv(l) \end{array} \right. \right\}$
  - if i odd $Succ(Reach_i) = \left\{ (l, \mathbf{x}) \left| \begin{array}{l} \exists t \in T, l' \in L, \mathbf{u} \in Inv(l') \cap G(t) : \\ \quad l' \xrightarrow{t} l \wedge (l', \mathbf{u}) \in Reach_i \\ \wedge\ \ \mathbf{x} \in Inv(l) \cap Res_t(\mathbf{u}) \end{array} \right. \right\}$,
    where $Res_t(\mathbf{u})(c) = \mathbf{u}(c)$ iff $c \notin Res_t$, else $Res_t(\mathbf{u})(c) = 0$.
- $\forall i \geq 0, Reach_{i+1} = Reach_i \ \cup \ Succ(Reach_i)$.
- $Reach = \bigcup_{i \in \mathbb{N}} Reach_i$.

Reach is computed in tools like UPPAAL by the following zone-based forward reachability algorithm. Given a timed automaton A with the target (l, B), it decides whether $Reach \cap (l, B) \neq \emptyset$. Reachable state sets are represented by lists $\langle (l_1, Z_1), \ldots, (l_m, Z_m) \rangle$ of location-zone pairs. Let R_i denote the (symbolic) reachable state-space at the i-th ($i \geq 0$) iteration.

1. Start with the state-set $R_0 = \{(l_0, \mathbf{0})\}$, or equivalently, in DBM form, $R_0 = l_0 \times \{\bigwedge_{x \in C} x - x_0 \leq 0)\}$, where $x_0 \notin C$ is a pseudo-clock used to represent the constant 0.
2. For $i \geq 0$, compute the symbolic successors of R_i, denoted $Post(R_i)$, separately for even and odd values of i, as follows:
   - If i even, $Post(R_i) = \{(l, Z) \mid \exists (l, Z') \in R_i$ $: Z = norm_k(timepass(Z')) \wedge Inv(l)\}$

[2] To simplify the proofs, we use even- and odd-numbered steps to distinguish between time-passage (of possibly zero duration) and transitions between discrete locations.

– If i odd, $Post(R_i) = \{(l, Z) \mid \exists (l', Z') \in R_i, t \in T : l' \xrightarrow{t} l \wedge Z = Res_t(Z' \wedge G(t)) \wedge Inv(l)\}$

3. Build $R_{i+1} = R_i \cdot Post(R_i)$, where $\cdot$ denotes conditional concatenation that suppresses subsumed zones, i.e., removes (l, Z) if there is another (l, Z') with Z implying Z'.
4. Repeat steps (2) and (3) until $R_{i+1} = R_i$. Denote the last set R_i thus computed as R. Termination is guaranteed by the use of k-normalization, as there are only finitely many different k-zones such that only subsumed zones arise eventually.
5. Test whether $Z \wedge B$ is satisfiable for some $(l, Z) \in R$. If so then report "(l, B) is reachable", otherwise report "(l, B) is un-reachable".

It has been shown that this algorithm is sound and complete w.r.t. reachability [10] in the sense that $Reach \cap (l, B) = \emptyset$ iff $R \cap (l, B) = \emptyset$.

3 Robustness w.r.t. Clock-Drift

We have hitherto considered perfectly synchronous clocks. We now consider drifting clocks that could occur in practice, as introduced in [1] and studied subsequently by others [2, 3, 4]. This phenomenon is modelled by introducing a parameter $\varepsilon > 0$ that characterizes the relative drift between the clocks. The slopes of the clocks are assumed to be within the range $\left[\frac{1}{1+\varepsilon}, 1+\varepsilon\right]$. This is equivalent to a relative drift in the range $\left[(\frac{1}{1+\varepsilon})^2, (1+\varepsilon)^2\right]$ between the clocks. We could alternatively consider the slopes to be in the range $[1-\varepsilon, 1+\varepsilon]$. The behaviour of both models w.r.t. infinitesimally small values of ε is identical, only that in our case, the slope of a clock never becomes negative no matter how large ε is. We then have a modification of the time-passage operation as follows:

Definition 7. For a clock valuation $\mathbf{x}$, its *time-passage under perturbation of* ε is: $timepass^{\varepsilon}(\mathbf{x}) = \left\{\mathbf{x} + d \cdot \mathbf{e} \,\middle|\, d > 0,\ \mathbf{e} \in \left[\frac{1}{1+\varepsilon}, 1+\varepsilon\right]^n\right\}$.
For a Zone Z, $timepass^{\varepsilon}(Z) = \bigcup_{\mathbf{x} \in Z}\ timepass^{\varepsilon}(\mathbf{x})$

While this model restricts the slopes of the clocks based on the value of parameter ε, the actual relative drift between the clocks increases without bound with increasing delay $d > 0$. The reachable state space also gets enlarged. For a given perturbation of ε, the corresponding *perturbed reach-set* $Reach^{\varepsilon}$ is defined inductively, similar to the non-perturbed case, by accounting for drifting clocks through the replacement of the deterministic $timepass()$ by an appropriate non-deterministic $timepass^{\varepsilon}()$ for steps corresponding to time-passage.

We now consider the effect of clock-drift on deciding whether some location-zone pair (l, B) is reachable. As an example (cf. Fig. 1), consider a timed automaton A, consisting of a single location l_0, two clocks x, y, the invariant of l_0

being $x \leq 2$, and a self-looping transition t consisting of a guard $x = 2?$, with the associated resets $x := 0, y := 0$. Let the unsafe state of A be characterized by $(l_0, B) = (l_0, y > 2)$. Assuming perfect clocks, the state-space of A is given by $Reach = (l_0, Z)$, where $Z \equiv (x \leq 2 \wedge y = x)$, and A is clearly safe, as $Reach \cap (l_0, B) = \emptyset$. For drift characterized by a given $\varepsilon > 0$, the corresponding state space is $Reach^{\varepsilon} = (l_0, Z^{\varepsilon})$, where $Z^{\varepsilon} \equiv x \leq 2 \wedge \frac{x}{(1+\varepsilon)^2} \leq y \leq x(1+\varepsilon)^2$. Thus, $\forall \varepsilon > 0 : Reach^{\varepsilon} \cap (l_0, B) \neq \emptyset$ and A is therefore not "robustly" safe. The automaton along with the associated state-space for each case is illustrated in Fig. 1.

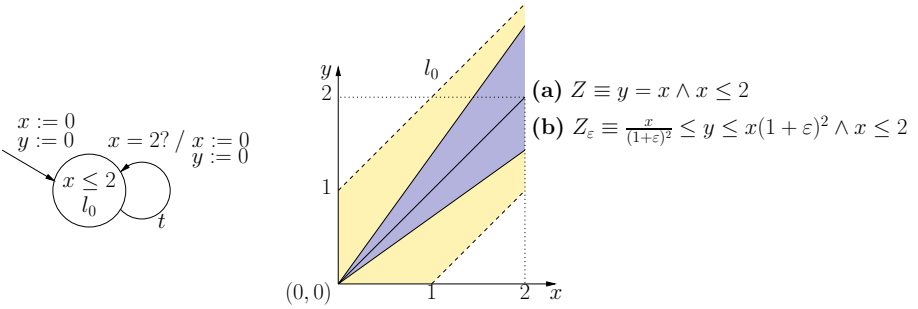

Fig. 1 A timed automaton A along with its state-spaces (a) without drift: (l_0, Z), (b) for a drift of ε : (l_0, Z^{ε}).

Related work on robust reachability of TA [1, 2, 3] compute the set $\cap_{\varepsilon>0} Reach^{\varepsilon}$. For this example, $\cap_{\varepsilon>0} Reach^{\varepsilon} = Reach$. This is because, for a zone Z, $\cap_{\varepsilon>0}\ timepass^{\varepsilon}(Z) = timepass(cl(Z))$, where $cl(Z)$ is the closure of Z, obtained by relaxing each strict inequality of Z to the corresponding non-strict one. In the present case, $Z \equiv \mathbf{0}$ is closed, as is $(Z \cup timepass(Z)) \cap Inv(l_0) \equiv y = x \wedge x \leq 2$, so $\cap_{\varepsilon>0} Reach^{\varepsilon} \ \cap \ (l_0, B) = \emptyset$. Hence, if open target states were allowed, the algorithms in [1, 2, 3] would all report this automaton as being robustly safe, while even the slightest perturbation would actually make the unsafe state reachable. However, if $B \equiv y \geq 2$, we see that the automaton of Fig. 1 is unsafe even with perfect clocks, while for $B \equiv y \geq 3$, the automaton is now safe even for drifting clocks, for all $0 < \varepsilon < \sqrt{1.5} - 1$.

We thus observe that *closed constraints* give consistent results while testing the automaton of Fig. 1 for safety, both with perfect clocks and under drift. Note also that this automaton has a single *progress cycle*, which additionally *resets all clocks simultaneously in a single transition.* The remit of this paper is to formulate conditions under which tests on TA for robust safety give identical results for both perfect and drifting clocks. We define for this purpose a *grid-point* and its associated *neighbourhood* as follows:

Definition 8. *Grid* denotes the set-of all *grid-points* in $\mathbb{R}^n_{\geq 0}$, i.e., $Grid = \{\mathbf{x_g} \in \mathbb{R}^n_{\geq 0} \mid \forall 1 \leq i \leq n : fract(x_{gi}) = 0\}$. For $\mathbf{x} \in \mathbb{R}^n_{\geq 0}$,

$grid(\mathbf{x}) = \{\mathbf{x_g} \in Grid \mid dist(\mathbf{x}, \mathbf{x_g}) < 1\}$, where
$dist(\mathbf{x}, \mathbf{x_g}) = max_{1 \leq i \leq n} |x_i - x_{gi}|$. The subset of $Grid$ that contains only those grid-points bounded by k is denoted $k - Grid$.

Thus, $\forall \mathbf{x_{kg}} \in k - Grid$: $\lfloor \mathbf{x_{kg}} \rfloor_k = \mathbf{x_{kg}}$. We will henceforth denote points in $Grid$ by the suffix g ($\mathbf{x_g}$ etc.) and points in $k - Grid$ by the suffix kg ($\mathbf{x_{kg}}$ etc.).

Definition 9. For $\mathbf{u_g} \in Grid$, we define its *neighbourhood*
$N_k(\mathbf{u_g}) = \bigcap_{\varepsilon > 0} \lfloor timepass^{\varepsilon}(\mathbf{u_g}) \rfloor_k$. For a zone Z, its *neighbourhood* is defined as: $N_k(Z) = \bigcup_{\mathbf{u_g} \in Z \cap Grid} N_k(\mathbf{u_g})$

$N_k(\mathbf{u_g})$ is the union of all *neighbouring* k-regions of $\mathbf{u_g}$, where a k-region r is said to *neighbour* $\mathbf{u_g}$ iff a point in r is reachable by time-passage from $\mathbf{u_g}$ for *every* drift, i.e., $\forall \varepsilon > 0 : timepass^{\varepsilon}(\mathbf{u_g}) \cap r \neq \emptyset$. Thus $N_k(\mathbf{u_g})$ is the result of adding to $\mathbf{u_g}$ all k-regions of Hausdorff distance 0 in temporally non-backward directions.

It must be understood here that the neighbourhood is defined only for grid-points[3]. It then follows that for any zone Z, $N_k(Z)$ is *idempotent*, i.e., $N_k(N_k(Z)) = N_k(Z)$, and that for a zone Z that has *no closed diagonal borders*, $N_k(Z)$ contains exactly the same grid-points as $norm_k(timepass(Z))$, and thus $N_k(Z) = norm_k(timepass(Z))$ for such a zone. Also, $\forall \mathbf{u_g} \notin k - Grid$: $N_k(\mathbf{u_g}) = norm_k(timepass(\mathbf{u_g}))$. The following lemmas establish some useful properties of the neighbourhood operator.

Lemma 1. $\forall \mathbf{x} \in \mathbb{R}^n_{\geq 0}, \forall \mathbf{u_g} \in Grid$:
$\mathbf{x} \in N_k(\mathbf{u_g}) \Leftrightarrow \forall \varepsilon > 0 \;\; \exists \mathbf{y} \in \lfloor \mathbf{x} \rfloor_k : \mathbf{y} \in timepass^{\varepsilon}(\mathbf{u_g})$

The proof is immediate from the definition of $N_k(\mathbf{u_g})$.

Lemma 2. *For any* $\mathbf{u_g} \in Grid$, $N_k(\mathbf{u_g})$ *is given by:*

$$N_k(\mathbf{u_g}) = norm_k \left\{ \mathbf{u_g} + d + \sum_{i=1}^{n} a_i \cdot \mathbf{e_i} \;\mid\; d > 0, \;\; a_i \in [0, 1) \right\},$$

where $\mathbf{e_i}$ *is the i-th unit vector.*

Here $\mathbf{u_g} + d$ denotes the addition of d to each component of $\mathbf{u_g}$. The proof follows from Lemma 1 and the definition of $\lfloor \mathbf{x} \rfloor_k$ for $\mathbf{x} \in \mathbb{R}^n_{\geq 0}$. This means that for any zone Z, $N_k(Z)$ is obtained as follows: First apply the standard unperturbed time-passage operator on Z, and then widen the *diagonal constraints which are non-strict inequalities* of the resulting conjunctive system by 1, to the *next higher strict inequalities*, i.e., $x - y \leq c$ is widened to $x - y < c + 1$, followed by standard k-normalization.

[3] By considering only closed guards and invariants for the automaton, we ensure that all the zones we encounter during FRA contain at least one grid-point.

Lemma 3. *Given any (diagonal-free) k-zone* Z, $\mathbf{x} \in \mathbb{R}^n_{\geq 0}$, $\mathbf{u_g} \in Grid$,
$\mathbf{x} \in N_k(\mathbf{u_g}) \cap Z \Leftrightarrow \forall \varepsilon > 0 \ \exists \mathbf{y} \in \lfloor \mathbf{x} \rfloor_k : \mathbf{y} \in timepass^{\varepsilon}(\mathbf{u_g}) \cap Z$

The proof follows from Lemma (1) and the following property of any diagonal-free k-zone B [11]:

Property 1. $\forall \mathbf{x}, \forall \mathbf{y} \in \lfloor \mathbf{x} \rfloor_k : \mathbf{x} \in B \Leftrightarrow \mathbf{y} \in B$.

Lemma 4. *For any two* closed zones Z_1 *and* Z_2,

$$Z_1 \cap N_k(Z_2) = \emptyset \Leftrightarrow Z_1 \cap norm_k(timepass(Z_2)) = \emptyset$$

The proof of "$\Rightarrow$" is immediate, as $N_k(Z_2) \supseteq norm_k(timepass(Z_2))$. The proof of "$\Leftarrow$" is also obvious if $N_k(Z_2) = norm_k(timepass(Z_2))$.
When $N_k(Z_2) \supset norm_k(timepass(Z_2))$, we prove "$\Leftarrow$" as follows:
Z_1, Z_2 (and thus $timepass(Z_2)$, except for its "bottom") are closed. For $N_k(Z_2) \supset norm_k(timepass(Z_2))$, it must be the case that Z_2 is a k-zone and so $norm_k(timepass(Z_2)) = timepass(Z_2)$ is also closed (except for its "bottom"). Thus, in order for Z_1 to have an empty intersection with $norm_k(timepass(Z_2))$, the two must be separated by a (max. norm) distance of at least 1. It also follows that the only additions to $norm_k(timepass(Z_2))$ to form $N_k(Z_2)$ are the open diagonal borders (obtained by relaxing the diagonal constraints of Z_2 by 1). These borders thus added being open, can at most *touch, but not intersect* Z_1, which entails our result.

Lemma 5. *For any* closed k-zone Z, *for any* $\mathbf{u_g} \in Grid$, *any* $\mathbf{v} \in \mathbb{R}^n_{\geq 0}$

$$\mathbf{v} \in Z \cap N_k(\mathbf{u_g}) \Rightarrow \exists \mathbf{v_g} \in (grid(\mathbf{v}) \cap Z \cap norm_k(timepass(\mathbf{u_g})))$$

The proof follows as a consequence of Lemma 4 and the definition of $grid(\mathbf{v})$. This means that any closed guard (Z, referring to Lemma 5) that is enabled by a point ($\mathbf{v}$) obtained by time-passage from a grid-point ($\mathbf{u_g}$) under the smallest of drifts (and thus included into that point's ($\mathbf{u_g}$'s) neighbourhood) is also enabled by a different grid-point ($\mathbf{v_g}$) obtained by time-passage (without drift) from that grid-point ($\mathbf{u_g}$).

Lemma 6. *For any* closed, diagonal-free k-zone Z, *any* $\mathbf{x}, \mathbf{u} \in \mathbb{R}^n_{\geq 0}$,

$$\begin{aligned} &\mathbf{u} \in Z \ \wedge \ \forall \varepsilon > 0 : (\lfloor \mathbf{x} \rfloor_k \cap timepass^{\varepsilon}(\mathbf{u}) \cap Z) \neq \emptyset \\ \Rightarrow &\exists \mathbf{u_g} \in grid(\mathbf{u}) \cap Z \ \wedge \ \mathbf{x} \in N_k(\mathbf{u_g}) \cap Z \end{aligned}$$

The proof is immediate from Lemmas 3 and 5, and the definition of $grid(\mathbf{u})$.

Definition 10. Let R^*_i be the reach-set at the i-th iteration, computed by modifying the time-passage steps of the standard FRA algorithm as follows: the $norm_k(timepass())$ operator is replaced by its neighbourhood $N_k()$. R^*_i is termed the corresponding *robust reach-set*.

Let R^* be the robust reach-set that is ultimately computed by the FRA algorithm by using $N_k()$ instead of $norm_k(timepass())$, while computing the time-passage successors of zones [4], and R be the reach-set that is computed by the standard zone-based FRA (cf. Definition 6).
From Lemma 4, we get $R^* = \{(l, Z \cup (N_k(Z) \wedge Inv(l))) \mid (l, Z) \in R\}$, thereby resulting in the following corollary:

Corollary 1. *For any closed zone B and any $l \in L$,*
$R^* \cap (l, B) = \emptyset \Leftrightarrow R \cap (l, B) = \emptyset$.

We now establish useful properties of the sets R_i^* through the following lemmas.

Lemma 7. *Given any $i \in \mathbb{N}$, any $l \in L$, and any $\mathbf{x} \in \mathbb{R}^n_{\geq 0}$,*

$$(l, \mathbf{x}) \models R_i^* \Rightarrow \forall \varepsilon > 0 \;\; \exists \mathbf{y} \in \lfloor \mathbf{x} \rfloor_k : (l, \mathbf{y}) \in Reach_i^{\varepsilon}$$

Here, by $(l, \mathbf{x}) \models R_i^*$, we mean that there exists a zone $Z \in R_i^*$ such that $\mathbf{x} \in Z$. This lemma shows that the set R_i^* collects the regions that can be "touched" in the sense of some (but not necessarily all) points within being reachable for *every* perturbation. The proof is by induction over the number i of iterations, separately for even and odd values of i, using Lemma 3, Property 1, and the definitions of R_i^* and $Reach_i^{\varepsilon}$.

Lemma 8. *For any $l \in L$, any diagonal-free k-zone B, any $i \in \mathbb{N}$,*
$R_i^* \cap (l, B) \neq \emptyset \Rightarrow \forall \varepsilon > 0 : \;\; Reach_i^{\varepsilon} \cap (l, B) \neq \emptyset$. [5]

This lemma implies that at any iteration depth i, if the set R_i^* intersects with a target state, then the corresponding perturbed reach-set under even the smallest of perturbations likewise intersects with the target state. The proof follows from Lemma 7 and Property 1.

Lemma 9. *For any* even *i, $l \in L$, $\mathbf{u_g} \in Grid \cap Inv(l)$, $\mathbf{v} \in \mathbb{R}^n_{\geq 0}$,*

$$\begin{aligned} &(l, \mathbf{u_g}) \models R_i^* \wedge \exists l' \in L \;\; \exists t \in T : l \xrightarrow{t} l' \wedge \;\; \mathbf{v} \in N_k(\mathbf{u_g}) \cap Inv(l) \cap G(t) \\ \Rightarrow\; &\exists \mathbf{v_g} \in norm_k(timepass(\mathbf{u_g})) \cap grid(\mathbf{v}) \cap Inv(l) \cap G(t) : \\ &\quad \exists \mathbf{w_g} \in (Inv(l') \cap Res_t(\mathbf{v_g}) : (l', \mathbf{w_g}) \models R_{i+2}^* \end{aligned}$$

The proof is immediate from Lemma 5 and the definition of R_i^*. Here we assume, in addition to the guards and invariants being closed and diagonal-free, the following condition of *admissible target locations*, which ensures consistency between the invariants of a location and the guards of the transitions entering and leaving that location:
For any locations l and l', and any transition t with $l \xrightarrow{t} l'$:
$Inv(l) \cap G(t) \neq \emptyset \;\; \wedge \;\; Inv(l') \cap G(t) \neq \emptyset$.

[4] Termination is guaranteed for such an algorithm by the use of k-normalization in the computation of the neighbourhood $N_k()$ of zones encountered during the FRA.

[5] Here $R_i^* \cap (l, B) \neq \emptyset$ denotes $Z \wedge B$ being satisfiable for some $(l, Z) \in R_i^*$.

Lemma 10. *For any* even i*, any* $l \in L$*, any* $\mathbf{x} \in \mathbb{R}^n_{\geq 0}$*,*
$\forall \mathbf{u_g} \in Grid \cap Inv(l) : (l, \mathbf{u_g}) \models R_i^*,\ \mathbf{x} \notin N_k(\mathbf{u_g}) \cap \bar{I}nv(l)$
$\Rightarrow \exists \varepsilon_i > 0\ \ \forall \mathbf{y} \in \lfloor \mathbf{x} \rfloor_k, \forall \mathbf{u} \in Inv(l) : (l, \mathbf{u}) \in Reach_i^{\varepsilon_i} :$
$\mathbf{y} \notin timepass^{\varepsilon_i}(\mathbf{u}) \cap Inv(l)$

The proof follows from Lemma 6 and Lemma 9, by induction over even i. A consequence is that the following converse of Lemma 7 also holds:

Lemma 11. *Given any* $i \in \mathbb{N}$*, any* $l \in L$*, and any* $\mathbf{x} \in \mathbb{R}^n_{\geq 0}$*,*

$$(l, \mathbf{x}) \not\models R_i^* \Rightarrow \exists \varepsilon_i > 0\ \ \forall \mathbf{y} \in \lfloor \mathbf{x} \rfloor_k :\ \ (l, \mathbf{y}) \notin Reach_i^{\varepsilon_i}$$

The proof is by induction over the number i of iterations, separately for even and odd values of i, using Lemmas 3 and 10, Property 1, and the definitions of R_i^* and $Reach_i^{\varepsilon_i}$.

Lemma 12. *For any* $l \in L$*,* $i \in \mathbb{N}$*, any diagonal-free* k*-zone* B*,*

$$R_i^* \cap (l, B) = \emptyset \Rightarrow \exists \varepsilon_i > 0 : Reach_i^{\varepsilon_i} \cap (l, B) = \emptyset.$$

The above lemma implies that at any iteration depth i, the set R_i^* does not intersect with a target state iff there exists a strictly positive value of the perturbation, such that the corresponding perturbed reach-set at that iteration depth likewise avoids the target state. The proof follows from Lemma 11 and Property 1. The following corollary is then a direct consequence of Lemmas 8 and 12.

Corollary 2. *Given any* $l \in L$*, any diagonal-free* k*-zone* B*,*
$R^* \cap (l, B) = \emptyset \Leftrightarrow \forall i \in \mathbb{N}\ \ \exists \varepsilon_i > 0 : Reach_i^{\varepsilon_i} \cap (l, B) = \emptyset$

Corollaries 1 and 2 lead us to the following theorem, which is a main result of this paper.

Theorem 1. *Let* R *be the final reach-set computed by the standard zone-based FRA, for a TA with closed and diagonal-free guards and invariants. Then for any closed and diagonal-free* k*-zone* B *and any* $l \in L$*,* $R \cap (l, B) = \emptyset \Leftrightarrow \forall i \in \mathbb{N}\ \ \exists \varepsilon_i > 0 : Reach_i^{\varepsilon_i} \cap (l, B) = \emptyset$

It follows from this theorem that the standard zone-based FRA used in tools like UPPAAL is exact (sound and complete) while testing TA with closed guards and invariants for robust safety against closed targets.

The "$\Leftarrow$" part of Theorem 1 states that a closed target is reported as reachable by standard zone-based FRA only if it is also reachable in a finite number of iterations of the transition relation of the TA, under even the slightest of perturbations. This result is intuitively obvious, because even the smallest perturbed reach-set is a strict superset of its non-perturbed version. The "$\Rightarrow$" part of Theorem 1 states that a closed target is reported as unreachable by zone-based FRA only if for any given number of iterations i of the transition relation,

there exists a strictly positive value of the perturbation ε_i that the automaton can tolerate and yet remains safe, in the sense that the corresponding perturbed reach-set $Reach_i^{\varepsilon_i}$ has an empty intersection with the (closed) target state. It must be noted here that this does not mean the existence of a homogeneous $\varepsilon > 0$ independent of the number of iterations, for which the unsafe state can be avoided, which is the notion considered in related works [1, 2, 3, 4]. Rather, as mentioned in the introduction, the magnitude of the tolerated perturbation ε_i could (but not necessarily) decrease with the number i of iterations, with ε_i potentially tending to 0 as i tends to ∞[6]. However, so long as we execute an *arbitrary, but finite number* of iterations, we are guaranteed a positive value of the tolerable perturbation for robust safety.

The analyses in [1, 2, 3, 4], on the other hand, add states that can be reached in any (unbounded) number of iterations through the (progress) cycles of the automaton[7], for even the slightest perturbation. Therefore, a state $(l, \mathbf{x})$ is considered to be *robustly unreachable* in our sense (i.e., not included in R^*), but reachable in the sense of the works in [1, 2, 3, 4] iff $\lim_{\varepsilon \to 0} \min\{i \in \mathbb{N} \mid (l, \mathbf{x}) \in Reach_i^{\varepsilon} \} = \infty$.

4 Robustness w.r.t. Imperfect Synchronization

In the previous section, we considered a model of drifting clocks where the relative drift between the clocks increases without bound with the passage of time, although the clock-slopes are themselves bounded according to the parameter ε. This is, however, rarely the case in practice, where the clocks, though subject to drift, are *regularly resynchronized* by diverse means, ranging from bit-stuffing in communication protocols to high-level clock synchronisation schemes. A parameter Δ characterizes the *post-synchronization-gap* and a parameter μ the longest possible gap between synchronizations. If the slopes of the clocks (w.r.t absolute time) are in the range $\left[\frac{1}{1+\theta}, 1+\theta\right]$ between synchronizations, such a resynchronization enforces a *uniform bound* given by

$$\varepsilon = \max\left(\Delta + \mu\left(\frac{1}{1+\theta}\right)^2, \Delta + \mu(1+\theta)^2\right) = \Delta + \mu(1+\theta)^2$$

[6] For closed TA in which each cycle has at-least one transition that resets all clocks simultaneously, the robust reach-sets computed by the algorithms in [2, 3, 4] coincide with the standard reach-set computed by UPPAAL, as seen in automaton of Fig. 1. Thus, a certificate of safety by standard UPPAAL for such TA w.r.t. closed targets implies a robust safety margin independent of iteration depth.

[7] We make no assumption on the cycles of the automaton.

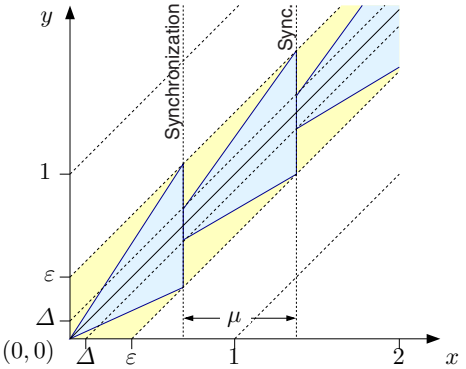

Fig. 2 Periodic resynchronization resulting in a bound ε on relative drift between clocks

on the relative drift between the clocks, irrespective of the extent of time-passage. The phenomenon is illustrated for two clocks x and y in Fig. 2. Throughout this section, we assume $0 < \varepsilon < 1$.

We incorporate such a resynchronization into TA by associating a *drift-offset* $\boldsymbol{\delta} \in [-\varepsilon, \varepsilon]^n$ for each clock valuation $\mathbf{x} \in \mathbb{R}^n_{\geq 0}$. This drift-offset keeps track of the extent to which the individual clocks in $\mathbf{x}$ have deviated from an implicit reference clock maintained by the synchronization scheme. The states of a TA in this semantics are thus tuples $(l, \mathbf{x}, \boldsymbol{\delta}) \in L \times \mathbb{R}^n_{\geq 0} \times [-\varepsilon, \varepsilon]^n$. As the deviation $\boldsymbol{\delta}$ is controlled by the synchronization scheme such that it always remains below ε, the (perturbed) time-passage under synchronization is as follows:

Definition 11. Given any $\mathbf{x} \in \mathbb{R}^n_{\geq 0}$, any $\boldsymbol{\delta} \in [-\varepsilon, \varepsilon]^n$,

$$timepass^{\varepsilon}_{sync}(\mathbf{x}, \boldsymbol{\delta}) = \{(\mathbf{x'}, \boldsymbol{\delta'}) \mid \boldsymbol{\delta'} \in [-\varepsilon, \varepsilon]^n \wedge \exists d > 0 : \mathbf{x'} = \mathbf{x} - \boldsymbol{\delta} + d + \boldsymbol{\delta'}\}$$

A *run* of a perturbed TA subject to clock synchronization with accuracy ε is a sequence $\langle (l_0, \mathbf{x}_0, \boldsymbol{\delta}_0), (l_1, \mathbf{x}_1, \boldsymbol{\delta}_1), \ldots \rangle$ of states such that

1. l_0 is the initial location and $\mathbf{x}_0 = \boldsymbol{\delta}_0 = \mathbf{0}$,
2. For even i, $l_{i+1} = l_i$, $\mathbf{x}_{i+1} \in Inv(l_i)$
   $\wedge\ (\mathbf{x}_{i+1}, \boldsymbol{\delta}_{i+1}) \in \{(\mathbf{x_i}, \boldsymbol{\delta_i})\} \cup timepass^{\varepsilon}_{sync}(\mathbf{x}_i, \boldsymbol{\delta}_i)$ [8]
3. For odd i, $\exists t_i \in T : l_i \xrightarrow{t_i} l_{i+1} : \mathbf{x}_i \in Inv(l_i) \cap G(t_i)$,
   $\mathbf{x}_{i+1} \in Inv(l_{i+1}) \cap Res_{t_i}(\mathbf{x}_i)$, $\boldsymbol{\delta}_{i+1} = Res_{t_i}(\boldsymbol{\delta}_i)$.

Due to memorizing the current deviation $\boldsymbol{\delta}$ and adjusting it consistently to the constraint that the overall accuracy is better than ε, this semantics is subtly more constrained than the —superficially similar— semantics permitting an arbitrarily directed ε-deviation upon every time passage.

$SReach^{\varepsilon}$ is the corresponding perturbed reach-set, defined inductively as follows, with $SReach^{\varepsilon}_i$ denoting the perturbed reach-set in $i \in \mathbb{N}$ steps, starting

[8] By abuse of notation, the subscripts i here denote the sequence of tuples in a run, and not individual vector components.

from the initial state $(l_0, \mathbf{0}, \mathbf{0})$ and alternating between (perturbed) time-passage and (exact) discrete-location transitions:

- $SReach_0^\varepsilon \equiv \{(l_0, \mathbf{0}, \mathbf{0})\}$
- For i even, $Succ(SReach_i^\varepsilon) = \{(l, \mathbf{x}, \boldsymbol{\delta}) \mid \mathbf{x} \in Inv(l)$
  $\wedge \ \exists \mathbf{x'} \in Inv(l), \exists \boldsymbol{\delta'} \in [-\varepsilon, \varepsilon]^n : (l', \mathbf{x'}, \boldsymbol{\delta'}) \in SReach_i^\varepsilon$
  $\wedge \ (\mathbf{x}, \boldsymbol{\delta}) \in timepass_{sync}^\varepsilon(\mathbf{x'}, \boldsymbol{\delta'})\}$
- For i odd, $Succ(SReach_i^\varepsilon) = \{(l, \mathbf{x}, \boldsymbol{\delta}) \mid \exists t \in T, l' \in L : l' \xrightarrow{t} l,$
  $\exists \mathbf{x'} \in Inv(l') \cap G(t) \ \exists \boldsymbol{\delta'} \in [-\varepsilon, \varepsilon]^n : (l', \mathbf{x'}, \boldsymbol{\delta'}) \in SReach_i^\varepsilon :$
  $\wedge \ \mathbf{x} \in Inv(l) \cap Res_t(\mathbf{x'}) \ \wedge \ \boldsymbol{\delta} = Res_t(\boldsymbol{\delta'})\}$
- $\forall i \geq 0, SReach_{i+1}^\varepsilon = SReach_i^\varepsilon \ \cup \ Succ(SReach_i^\varepsilon)$
- $SReach^\varepsilon = \bigcup_{i \in \mathbb{N}} SReach_i^\varepsilon$

As before, we assume that all guards and invariants are closed and diagonal-free. Let *Reach* denote the reach-set obtained by considering perfectly synchronous clocks ($\varepsilon = 0$), where $Reach_i$ denotes the reach-set at step i, as defined previously (cf. Definition 6). We establish the relationship between the sets $SReach^\varepsilon$ and *Reach* through the following lemmas.

Lemma 13. *For any* $i \in \mathbb{N}$, $l \in L$, $\mathbf{x} \in \mathbb{R}^n_{\geq 0}$, $\boldsymbol{\delta} \in [-\varepsilon, \varepsilon]^n$,

$$(l, \mathbf{x}, \boldsymbol{\delta}) \in SReach_i^\varepsilon \ \Rightarrow \ \exists \mathbf{x_g} \in grid(\mathbf{x}) : (l, \mathbf{x_g}) \in Reach_i$$

The proof is by induction over i, from the definitions of $SReach_i^\varepsilon$ and $Reach_i$. The following corollary is an immediate consequence.

Corollary 3. *For any* $i \in \mathbb{N}$, *it holds that:*

$$sup_{s \in SReach_i^\varepsilon} \ dist(s, Reach_i) < 1 \ ,$$

where for $s = (l, \mathbf{x}, \boldsymbol{\delta}) \in SReach_i^\varepsilon$, $dist(s, Reach_i) = \inf_{(l, \mathbf{x'}) \in Reach_i} dist(\mathbf{x}, \mathbf{x'})$.

Corollary 3 intuitively means that irrespective of the iteration depth i, the perturbed reach-set $SReach_i^\varepsilon$ stays "close-enough" to the standard reach-set $Reach_i$, in the sense that even the "farthest" point in the perturbed reach-set is less than unit distance away from the standard reach-set.

Lemma 14. *For a TA with only closed and diagonal-free guards and invariants, and any closed target location-zone pair of the form* (l, B)*:*

$$Reach \cap (l, B) = \emptyset \Leftrightarrow \forall 0 < \varepsilon < 1 : SReach^\varepsilon \cap (l, B) = \emptyset \ ,$$

where $SReach^\varepsilon \cap (l, B) = \emptyset$ *denotes* $\forall (l, \mathbf{x}, \boldsymbol{\delta}) \in SReach^\varepsilon : \mathbf{x} \notin B$.

The proof of "$\Leftarrow$" is obvious as $\forall \varepsilon > 0 : SReach^\varepsilon \supset Reach$, in the following sense: $\forall (l, \mathbf{x}) \in Reach : (l, \mathbf{x}, \mathbf{0}) \in SReach^\varepsilon$. The proof of "$\Rightarrow$" follows from Corollary 3, in conjuction with the fact that B is a closed zone, as are all the guards and invariants of the TA, and $0 < \varepsilon < 1$. This lemma, together with the soundness and completeness result for standard zone-based FRA [10], leads us to the following theorem, which is the second main result of this paper:

Theorem 2. *For a TA with only closed and diagonal-free guards and invariants, any location l, and any closed, diagonal-free k-zone B :*

$$R \cap (l, B) = \emptyset \Leftrightarrow \forall 0 < \varepsilon < 1 : SReach^{\varepsilon} \cap (l, B) = \emptyset \ ,$$

where R is the symbolic reachable state-space that is ultimately computed by the standard zone-based FRA.

Theorem 2 thus establishes the exactness of standard zone-based forward analysis using a tool like UPPAAL for TA with closed guards and invariants, when testing for robust safety against closed targets, with drifting clocks subject to periodic resynchronizations that enforce accuracy better than 1. A certification of robust safety in this case implies that the target state could be avoided by all values of the perturbation ε that are strictly less than 1, independent of the depth of iteration, unlike the case for unbounded relative clock-drift that was considered in the previous section.

Theorem 2 may also be proven using the neighbourhood construction for grid-points, as was previously done for Theorem 1.

5 Conclusion

We have investigated reachability (and thus, safety) of TA subject to drifting clocks – a phenomenon that occurs in practical implementations of timed systems. We first considered the model of clock-drift introduced in [1] and studied in [2, 3, 4], and analyzed the reachability for TA with closed guards, invariants, and targets, but without the assumption of progress cycles, as was made in [1, 2, 3, 4]. We showed the exactness of the standard zone-based FRA of UPPAAL for such TA, under a notion of robustness weaker than that in [1, 2, 3, 4], in the sense that we do not add states that require an unbounded number of iterations in order to be reached, under infinitesimally small clock-drift (cf. Theorem 1). Our notion is applicable to all systems having a finite life-time, where for any particular projected life-time, an appropriate worst-case clock drift enforcing behavior indistinguishable from the ideal can be chosen. For long life-times, the permissible clock drift may become extremely small. As technical realizations in many systems (like, e.g., bit-stuffing in communication protocols or the central-master synchronization incorporated in GPS-controlled systems) address this problem by regular clock resynchronization, thus bounding the relative drift within an set of clocks even over arbitrarily long life-times, we have also modelled and analyzed such synchronization schemes. We have shown that the standard zone-based analysis of UPPAAL is again exact while testing such models for robust safety, but now with the assertion of a uniform strictly positive robustness margin of 1, independent of system life-time.
Note that our definition of TA admits only *diagonal-free constraints* for the guards, invariants, and targets. This is because TA with diagonal constraints

of the form $x - y \triangleright c$ have been shown to be incompatible with forward reachability analysis that employs standard k-normalization for termination, and a modified normalization that takes into account the diagonal constraints of the TA is in fact necessary for dealing with such cases [11, 10]. However, the techniques of this paper extend quite naturally to TA with diagonal constraints and a suitably modified normalization operation. An extension of these techniques to Probabilistic TA [12] (TA with discrete probability distributions annotating transitions between locations) also appears straight-forward.

We finally wish to mention the following alternate notions of robustness for TA: [13] imposes a topological closure on timed traces, which has been shown in [14] to affect digitization of TA. [15] considers robust model-checking of LTL properties, while [16] considers robustness analysis via channel machines.

Acknowledgements This work was supported mainly by the Deutsche Forschungsgemeinschaft (DFG) under grant GRK 1076/1 for the Graduate School Trustsoft `http://www.uni-oldenburg.de/trustsoft/en/`. We also acknowledge support of the DFG under the AVACS project `www.avacs.org`, of the EU under the Quasimodo Project
`http://www.quasimodo.aau.dk/`, and of ASKON Consulting Group GmBH `www.askon.de`.

The comments of two of the reviewers were most helpful in refining this paper, as were discussions with Alexandre David, Kim Larsen, Laurent Doyen, Jean-Francois Raskin, Olaf Owe, Wang Yi, Stefan Leue, Michael Adelaide, Piotr Kordy, Christian Herde, and Abhishek Dhama.

References

1. A. Puri, "Dynamical Properties of Timed Automata", *Discrete Event Dynamic Systems* **10**, Kluwer (2000), 87–113
2. M. de Wulf, L. Doyen, N. Markey, and J.-F. Raskin, "Robustness and Implementability of Timed Automata", *Proc. of FORMATS-FTRTFT'04*, LNCS **3253**, Springer (2004), 118-133
3. C. Daws and P. Kordy, "Symbolic Robustness Analysis of Timed Automata", *Proc. of FORMATS'06*, LNCS **4202**, Springer (2006), 143-155
4. C. Dima, "Dynamical Properties of Timed Automata Revisited", *Proc. of FORMATS'07*, LNCS **4763**, Springer (2007), 130–146
5. R. Alur and D. Dill, "A Theory of Timed Automata", *Theoretical Computer Science* **126**, Elsevier (1994), 183–235
6. G. Behrmann, A. David, and K. Larsen, "A Tutorial on Uppaal", *Formal Methods for the Design of Real-Time Systems*, LNCS **3185**, Springer-Verlag (2004), 200–236
7. M. Bozga, C. Daws, O. Maler, A. Olivero, S. Tripakis, S. Yovine, "KRONOS: A Model-Checking Tool for Real-Time Systems", *Proc. of FTRTFT'98*, LNCS **1486**, Springer (1998), 298–302
8. M. Lindahl, P. Pettersson, and W. Yi, "Formal Design and Analysis of a Gear Controller", *International Journal on Software Tools for Technology Transfer* **3**, Springer-Verlag (2001), 353–368
9. L. Aceto, P. Bouyer, A. Burgueno, and K. Larsen, "The Power of Reachability Testing for Timed Automata", *Theoretical Computer Science* **300**, Elsevier (2003), 411-475

10. J. Bengtsson and W. Yi, "Timed Automata: Semantics, Algorithms, and Tools", *Lectures on Concurrency and Petri Nets*, LNCS **3098**, Springer (2004), 87–124
11. P. Bouyer, "Forward Analysis of Updatable Timed Automata", *Formal Methods in System Design* **24**, Kluwer (2004), 281–320
12. M. Kwiatkowska, G. Norman, R. Segala, and J. Sproston, "Automatic Verification of Real-time Systems with Discrete Probability Distributions", *Theoretical Computer Science* **282**, Elsevier (2002), 101-150
13. V. Gupta, T. A. Henzinger, and R. Jagadeesan, "Robust Timed Automata", *Proc. of HART'97*, LNCS **1201**, Springer (1997), 331–345
14. J. Ouaknine and J. Worrell, "Revisiting Digitization, Robustness, and Decidability for Timed Automata", *Proc. of LICS'03*, IEEE CS Press (2003), 198-207
15. P. Bouyer, N. Markey, and P.-A. Reynier, "Robust Model-Checking of Linear-Time Properties in Timed Automata", *Proc. of LATIN'06*, LNCS **3887**, Springer (2006), 238–249.
16. P. Bouyer, N. Markey, and P.-A. Reynier, "Robust Analysis of Timed Automata via Channel Machines", *Proc. of FoSSaCS'08*, LNCS **4962**, Springer (2008), 157-171