Robust Reachability in Timed Automata and Games: A Game-based Approach ☆

Patricia Bouyer[a], Nicolas Markey[a], Ocan Sankur[b]

[a] *LSV, CNRS & ENS Cachan, Cachan, France.*
[b] *Université Libre de Bruxelles, Brussels, Belgique*

Abstract

Reachability checking is one of the most basic problems in verification. By solving this problem in a game, one can synthesize a strategy that dictates the actions to be performed for ensuring that the target location is reached. In this work, we are interested in synthesizing "robust" strategies for ensuring reachability of a location in timed automata. By robust, we mean that it must still ensure reachability even when the delays are perturbed by the environment. We model this perturbed semantics as a game between the controller and its environment, and solve the parameterized robust reachability problem: we show that the existence of an upper bound on the perturbations under which there is a strategy reaching a target location is EXPTIME-complete. We also extend our algorithm, with the same complexity, to turn-based timed games, where the successor state is entirely determined by the environment in some locations.

1. Introduction

Timed automata [4] are a timed extension of finite-state automata. They come with an automata-theoretic framework to design, model, verify and synthesize systems with timing constraints. One of the most basic problems in timed automata is the reachability problem: given a timed automaton and a target location, is there a path that leads to that location? This can be rephrased in the context of control as follows: is there a *strategy* that dictates how to choose time delays and edges to be taken so that a target location is reached? This problem has been solved long ago [4], generalized to timed games [6], and efficient algorithms have then been developed and implemented [30, 38].

However, the abstract model of timed automata is an idealization of real timed systems. For instance, we assume in timed automata that strategies can choose the delays with arbitrary precision. In particular, the delays can be arbitrarily close to zero (the system is arbitrarily fast), and clock constraints can enforce exact delays (time can be measured exactly). Although these assumptions are natural in abstract models, they need to be justified after the design phase. Indeed the situation is different in real-world systems: digital systems have response times that may not be negligible, and control software cannot ensure timing constraints exactly, but only up to some error, caused by

☆ Partly supported by the French ANR project ImpRo (ANR-10-BLAN-0317), by ERC Starting grants EQualIS (FP7-308087) and inVEST (FP7-279499), and by European FET project Cassting (FP7-601148)

Email addresses: bouyer@lsv.ens-cachan.fr (Patricia Bouyer), markey@lsv.ens-cachan.fr (Nicolas Markey), ocan.sankur@normalesup.org (Ocan Sankur)

clock imprecisions, measurement errors, and communication delays. A good control software must be *robust*, *i.e.*, it must ensure good behavior in spite of small imprecisions [23, 28].

Consequently, there has been a recent effort to consider imprecisions inherent to real systems in the theory of timed systems. In particular, there has been several attempts to define convenient notions of robustness for timed automata. The approach initiated in [32, 18, 17] is the closest to our framework. It consists in modeling imprecisions by *enlarging* all clock constraints of the automaton by some parameter δ, that is transforming each constraint of the form $x \in [a, b]$ into $x \in [a-\delta, b+\delta]$, and in synthesizing $\delta > 0$ such that all runs of the enlarged automaton satisfy a given property. Several model-checking algorithms for timed automata were then re-visited and extended to this setting in [9, 10, 11, 34] and symbolic algorithms were studied in [26, 16]. This notion of robustness also corresponds to a concrete implementation semantics; in fact robustness implies implementability in a simplied model of a micro-processor, see [18].

In all these works, the robustness condition is satisfied if there exists an enlargement parameter for which *all* runs of the enlarged automaton satisfy a given property. In other terms, for any choice of the delays and edges, and any possible perturbation, the given property must hold. Although this is a convenient notion for *e.g.* safety properties, this does not capture the system's ability to adapt to perturbations that were observed earlier in a given run. In fact, if perturbations have accumulated and deviated the system from its course, a smart control software should be able to adapt its action to correct this. Thus, we believe a good notion of robustness can be defined by a game played between two players: *Controller* with a reachability objective, and *Perturbator* with the complementary objective.

Following this idea, we are interested in the synthesis of robust strategies in timed automata and games for reachability objectives, taking into account response times and imprecisions. In our semantics, which is parameterized by δ_P and δ_R with $0 < \delta_P \leq \delta_R$, Controller chooses to delay an amount $d \geq \delta_R$ after which the guard of a chosen edge is satisfied, and the system delays d' and takes the edge, where d' is chosen by Perturbator satisfying $|d - d'| \leq \delta_P$. Observe that the guard may not be satisfied after the delay d', but the chosen edge is taken whatever the perturbations. We say that a given location is *robustly reachable* if there exist parameters $0 < \delta_P \leq \delta_R$ such that Controller has a winning strategy ensuring that the location is reached against any strategy of Perturbator. To simplify the presentation, but w.l.o.g., we assume in this paper that $\delta = \delta_P = \delta_R$; our algorithm can easily be adapted to the general case (see Section 4).

The main result of this paper is the following: We show that deciding the existence of $\delta > 0$, and of a strategy for the controller so as to ensure reachability of a given location in a turn-based timed game (whatever the imprecision, up to δ), is EXPTIME-complete. Moreover, if there is a strategy, we can compute a *uniform* one, which is parameterized by δ, using *shrunk difference bound matrices* (shrunk DBMs) that we introduced in [36]. In this case, our algorithm provides a bound $\delta_0 > 0$ such that the strategy is correct for all $\delta \in [0, \delta_0]$. Our strategies also give quantitative information on how perturbations accumulate or can compensate. Technically, our work extends shrunk DBMs by *constraints*, and establishes non-trivial algebraic properties of this data structure (Section 3). The main result is then obtained by transforming the infinite-state game into a finite abstraction, which we prove can be used to symbolically compute a winning strategy, if any (Section 4).

A variant of our semantics was studied in [15] for timed games with *fixed* parameters. In this variant, Controller can only suggest delays and edges whose guards are satisfied after any perturbation; thus, this is a *conservative* variant of our semantics. When δ is fixed, the semantics can be encoded by a usual timed game, and standard algorithms can be applied. Whether one can synthesize $\delta > 0$ for which the controller has a winning strategy was left as a challenging open

problem. We solve this problem for reachability objectives in turn-based timed games. Parameterized reachability and safety in the conservative semantics of [15] are studied in [37]. See also Section 2.3 for notes on related work.

2. Robust Reachability in Timed Automata

2.1. Timed Automata and Games, and Robust Reachability

Given a finite set of clocks $\mathcal{C}$, we call *valuations* the elements of $\mathbb{R}^{\mathcal{C}}_{\geq 0}$, which are nonnegative real vectors of dimension $|\mathcal{C}|$. For a subset $R \subseteq \mathcal{C}$ and a valuation v, $v[R \leftarrow 0]$ is the valuation defined by $v[R \leftarrow 0](x) = v(x)$ for $x \in \mathcal{C} \setminus R$ and $v[R \leftarrow 0](x) = 0$ for $x \in R$. Given $d \in \mathbb{R}_{\geq 0}$ and a valuation v, the valuation $v + d$ is defined by $(v + d)(x) = v(x) + d$ for all $x \in \mathcal{C}$. We extend these operations to sets of valuations in the obvious way. We write $\vec{0}$ for the valuation that assigns 0 to every clock.

An atomic clock constraint is a formula of the form $k \preceq x \preceq' l$ or $k \preceq x - y \preceq' l$ where $x, y \in \mathcal{C}$, $k, l \in \mathbb{Z} \cup \{-\infty, \infty\}$ and $\preceq, \preceq' \in \{<, \leq\}$. A *guard* is a conjunction of atomic clock constraints. A valuation v satisfies a guard g, denoted $v \models g$, if all constraints are satisfied when each $x \in \mathcal{C}$ is replaced with $v(x)$. We write $\Phi_{\mathcal{C}}$ for the set of guards built on $\mathcal{C}$.

Definition 2.1. *A* (two-player) turn-based timed game $\mathcal{A}$ *is a tuple* $(\mathcal{L}_C \cup \mathcal{L}_P, \mathcal{C}, \ell_0, E)$*, where* $\mathcal{L} = \mathcal{L}_C \cup \mathcal{L}_P$ *is a finite set of locations satisfying* $\mathcal{L}_C \cap \mathcal{L}_P = \emptyset$*,* $\mathcal{C}$ *is a finite set of clocks,* $E \subseteq \mathcal{L} \times \Phi_{\mathcal{C}} \times 2^{\mathcal{C}} \times \mathcal{L}$ *is a set of edges, and* $\ell_0 \in \mathcal{L}_C$ *is the initial location. An edge* $e = (\ell, g, R, \ell')$ *is also written as* $\ell \xrightarrow{g,R} \ell'$*. A* timed automaton *is a turn-based timed game with* $\mathcal{L}_P = \emptyset$*.*

Standard semantics of timed automata is usually given as a timed transition system [4]. To capture robustness, we define the semantics as a game where perturbations in delays are uncontrollable. The semantics extends naturally to turn-based timed games, where our game semantics simply gives control over perturbations to one of the players.

Given a turn-based timed game $\mathcal{A} = (\mathcal{L}_C \cup \mathcal{L}_P, \mathcal{C}, \ell_0, E)$ and $\delta > 0$, we define the *perturbation game* of $\mathcal{A}$ w.r.t. δ as a two-player turn-based game $\mathcal{G}_\delta(\mathcal{A})$ between players *Controller* and *Perturbator*. Intuitively the semantics is the following: At locations $\mathcal{L}_C$, Controller first suggests a delay and an edge at Perturbator perturbs the delay by some amount in the range $[-\delta, \delta]$, and the designated edge is taken with the perturbed delay; at locations $\mathcal{L}_P$, the edge and the delay are entirely determined by Perturbator. Observe that the semantics is not symmetric; Perturbator has the role of the environment. Formally, the state space of $\mathcal{G}_\delta(\mathcal{A})$ is partitioned into $V_C \cup V_P$ where $V_C = \mathcal{L}_C \times \mathbb{R}^{\mathcal{C}}_{\geq 0}$ belong to Controller, and $V_P = \mathcal{L}_P \times \mathbb{R}^{\mathcal{C}}_{\geq 0} \cup \mathcal{L}_C \times \mathbb{R}^{\mathcal{C}}_{\geq 0} \times \mathbb{R}_{\geq 0} \times E$ belong to Perturbator. The initial state is $(\ell_0, \vec{0})$ and belongs to Controller. The transitions are defined as follows: from any state $(\ell, v) \in V_C$, there is a transition to $(\ell, v, d, e) \in V_P$ whenever $d \geq \delta$, $e = (\ell, g, R, \ell')$ is an edge such that $v + d \models g$. Then, from any such state $(\ell, v, d, e) \in V_P$, there is a transition to $(\ell', (v + d + \epsilon)[R \leftarrow 0]) \in V_C$, for any $\epsilon \in [-\delta, \delta]$. Further, for any $(\ell, \nu) \in V_P$, there is an edge from (ℓ, ν) to $(\ell', (\nu + d)[R \leftarrow 0])$ for any $d \geq 0$ and edge $e = (\ell, g, R, \ell')$ such that $\nu + d \models g$.

We assume familiarity with basic notions in game theory, and quickly survey the main definitions. A run in $\mathcal{G}_\delta(\mathcal{A})$ is a finite or infinite sequence of consecutive states starting at $(\ell_0, \vec{0})$. It is said maximal if it is infinite or cannot be extended. A *strategy for Controller* is a function that assigns to every non-maximal run ending in some $(\ell, v) \in V_C$, a pair (d, e) where $d \geq \delta$ and e is an edge enabled at $v + d$, i.e., there is a transition from (ℓ, v) to $(\ell, v, d, e) \in V_P$. A *strategy for Perturbator* assigns to all states $(\ell, v, d, e) \in V_P$ a perturbation $\epsilon \in [-\delta, \delta]$, and to all states $(\ell, v) \in V_P$, a delay and an

edge whose guard is satisfied at v. A *run* ρ is compatible with a strategy f if for every prefix ρ' of ρ ending in V_C, the next transition along ρ after ρ' is given by f. Given a target location ℓ, a strategy f is winning for the reachability objective defined by ℓ whenever all maximal runs that are compatible with f visit ℓ.

Observe that we require at any state (ℓ, v), that Controller should choose a delay $d \geq \delta$ and an edge e that is enabled after the chosen delay d. The edge chosen by Controller is always taken but there is no guarantee that the guard will be satisfied exactly when the transition takes place. In fact, Perturbator can perturb the delay d chosen by Controller by any amount $\epsilon \in [-\delta, \delta]$, including those that do not satisfy the guard. Notice that $\mathcal{G}_0(\mathcal{A})$ corresponds to the standard (non-robust) semantics of $\mathcal{A}$. We are interested in the following problem.

Problem 2.2 (Parameterized Robust Reachability)**.** *Given a turn-based timed game $\mathcal{A}$ and a target location ℓ, decide whether there exists $\delta > 0$ such that Controller has a winning strategy in $\mathcal{G}_\delta(\mathcal{A})$ for the reachability objective ℓ.*

Our main result is the decidability of this problem. Moreover, if there is a solution, we compute a strategy represented by parameterized difference-bound matrices where δ is the parameter; the strategy is thus *uniform* with respect to δ. In fact, we provide a bound $\delta_0 > 0$ such that the strategy is winning for Controller for *any* $\delta \in [0, \delta_0]$. These strategies also provide a quantitative information on how much the perturbation accumulates (See Fig. 3). The main result of this paper is the following:

Theorem 2.3. *Parameterized robust reachability is* EXPTIME*-complete both for timed automata and turn-based timed games.*

Notice that we are interested in the parameterized problem: δ is not fixed in advance. When δ is fixed, the problem can be formulated and solved as a usual timed game (see [15] for such an encoding). The parameterized version is important since one does not necessarily have a precise estimation for perturbations at the design phase. It is also interesting to directly synthesize δ and a *uniform* strategy, as described above, rather than finding one by trial-and-error.

Checking parameterized robust reachability is different from usual reachability checking mainly for two reasons. First, in order to reach a given location, Controller has to choose the delays along a run, in such a way that these perturbations do not accumulate and block the run. In particular, it shouldn't play too close to the borders of the guards (see Fig. 3). Second, due to these uncontrollable perturbations, some regions that are not reachable in the absence of perturbation can become reachable (see Fig. 5). So, Controller must also be able to win from these newly reachable regions. The regions that become reachable in our semantics are those *neighboring* reachable regions. The characterization of these neighboring regions is one of the main difficulties in this paper (see Section 3.5).

The paper is organized as follows. In Section 3, we study shrunk DBMs, a data structure used to represent the parameterized state space of timed automata under perturbations [36]. We extend this data structure to include *shrinking constraints*, and study *neighborhoods* of shrunk DBMs. For the sake of readability, we first present and prove the algorithm for timed automata, in Section 4, then explain how to extend the algorithm to turn-based timed games in Section 5. The EXPTIME-hardness result is given in Section 6.

2.2. Motivating example: robust real-time scheduling

An application of timed automata is the synthesis of schedulers in various contexts [1]. We show that robust reachability can help providing a better schedulability analysis: we show that schedulers

synthesized by standard reachability analysis may not be robust: even the slightest *decrease* in task execution times can result in a large *increase* in the total time. This is a phenomenon known as *timing anomalies*, first identified in [20].

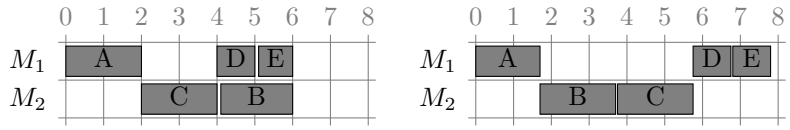
Consider the scheduling problem described in Fig. 1, inspired by [33]. Assume that we look for a *greedy* (*i.e.*, work-conserving) scheduler, that will immediately start executing a task if a machine is free for execution on an available task. What execution time can a greedy scheduling policy guarantee on this instance? One can model this problem as a timed automaton, and prove, by classical reachability analysis, that these tasks can be scheduled using a greedy policy within six time units. However the scheduler obtained this way may not be robust, as illustrated in Fig. 1(b). If the duration of task A unexpectedly drops by a small amount $\delta > 0$, then any greedy scheduler will schedule task B before task C, since the latter is not ready for execution at time $2 - \delta$. This yields a scheduling of tasks in $8 - \delta$ time units.

Our robust reachability algorithm is able to capture such phenomena, and can provide correct and robust schedulers. In fact, it would answer that the tasks are not schedulable in six time units (with a greedy policy), but only in eight time units. See Appendix A for a detailed model and discussion of this example.

2.3. Related work: robustness in timed automata and games

Robustness in timed automata and hybrid automata models have been studied in the last two decades. The objective is mainly twofold: First, robustness is a desired property of real-time systems, so developing a formal framework for robustness analysis and robust synthesis is a natural goal. Second, taking into consideration perturbations can yield a more realistic semantics, which does not have the excessive expressive power of abstract timed automata. In fact, giving up on infinite precision may render some undecidable problems decidable, such as checking safety in linear hybrid automata under some conditions [19].

Several works focused particularly on defining a good notion of robustness in timed automata. [31] surveys some of these. A first such attempt is [21], where a topological notion is introduced, with the hope that the language inclusion problem could become decidable under this semantics, but the undecidability was later shown to hold even in the robust setting [22]. This approach differs from our work since we rather focus on a more concrete perturbation model with modelling and synthesis purposes. Also, topological robustness consists in discarding isolated behaviours, whereas our semantics adds behaviours.



(a) A has duration 2. (b) A has duration $2 - \delta$.

Figure 1: Consider tasks A, B, C of duration 2 and D, E of duration 1, to be scheduled on two machines. As usual, we assume that a machine can execute a single task at a time. Dependences between tasks are as follows: $A \to B$ and $C \to D, E$, meaning *e.g.* that A must be completed before B can start. Task A must be executed on machine M_1 and tasks B, C on machine M_2. Moreover, the release time of task C means that it cannot be scheduled before 2 time units. Fig. 1(a) shows the optimal greedy schedule for these tasks under these constraints, while Fig. 1(b) shows the outcome of any greedy scheduler when the duration of task A is less than 2.

The semantics under an unknown regular sampling of time was considered in [14, 2, 3]. Here, the goal is to synthesize a sampling parameter under which a reachability or safety objective holds, or the untimed language is preserved. [8] shows that one can "compile" any timed automaton to an "approximate" one whose semantics is preserved under sampling. These approaches are orthogonal to ours since they privilege discrete semantics rather than considering continous delays and perturbations, and study loss of behaviour. We believe continous semantics provide more precision in timed automaton models since perturbations and environment's delays cannot always be assumed to be discrete.

Let us state more precisely the relation of our semantics with that of [32, 18, 17]. Robust model-checking, studied in these works, can be reformulated as follows: does there exists some $\delta > 0$ such that whatever Controller and Perturbator do in $\mathcal{G}_\delta(\mathcal{A})$, a given property is satisfied. This is therefore the universal counterpart of our formulation of the parameterized robustness problem. It has been shown that this universal parameterized robust model-checking is no more difficult (in terms of complexity) than standard model-checking. This has to be compared with our result, where complexity goes up from PSPACE to EXPTIME.

In [36], we studied robustness of timed automata to guard shrinkings, which is the dual notion of the robustness against guard enlargement. We introduced shrunk DBMs, which we also use and extend in this paper, in order to study the state space of timed automata with shrunk guards. We also give in [36] a methodology to implement timed automata, while systematically avoiding the effect of imprecisions, modelled by guard enlargement. As in robust model-checking, this approach studies worst-case behaviour under a fixed (though parameterized) imprecision bound, and does not take into account systems' ability to react to actual perturbations.

As mentioned in the introduction, a similar but unparameterized notion of robustness for timed games was studied in [15]. This unparameterized setting was also considered in the context of timed interface theories in [29] but the parameter synthesis problem was not addressed. The semantics of [15] is different since it requires Controller to suggest delays and edges whose guards are satisfied after the perturbed delay, for any perturbation. It turns out that parameterized reachability and safety are easier in this semantics; both are PSPACE-complete [37].

A related line of work is that of parameter synthesis in timed automata and games. The parametric reachability problem is undecidable in general for timed automata [5]. Subclasses with decidable reachability problems include the case of timed automata with a single parametric clock [5], and the L/U timed automata [25]. The extension of this latter class to timed games were considered recently in [27] where it is shown that the problem is decidable on L/U timed games. This latter algorithm could be applied to a restriction of our problem with only nonnegative perturbations. In fact, each edge could be redirected to a fresh state controlled by the second player, who is required to choose a perturbation upto δ. However, the general case, or the case of negative perturbations cannot be modelled this way.

The present paper is an extended version of [12] with full proofs, and the extension to turn-based timed games.

3. Shrinking DBMs

3.1. Regions, zones and DBMs

In all timed automata and games we consider, we assume that all clocks are bounded above by a constant. We do not lose generality with this hypothesis since reachability objectives are preserved by choosing a constant large enough, and requiring that all clocks stay below it at any transition.

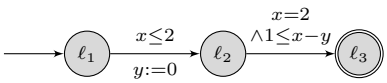


Figure 2: Timed automaton $\mathcal{A}$.

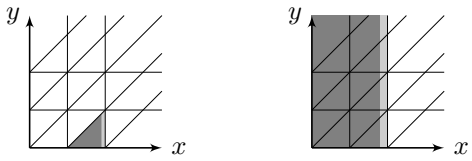


Figure 3: Winning states in ℓ_2 (left) and in ℓ_1 (right).

We assume familiarity with the notions of regions and zones, and refer to [4, 7] for more background. For two regions r and r', we write $r \lessdot r'$ if r' is the immediate (strict) time-successor of r. A *zone* is a set of clock valuations satisfying a guard.

We write $\mathcal{C}_0$ for the set $\mathcal{C} \cup \{0\}$. A *difference-bound matrix (DBM)* is a $|\mathcal{C}_0| \times |\mathcal{C}_0|$-matrix over $(\mathbb{R} \times \{<, \leq\}) \cup \{(\infty, <)\}$. We adopt the following notation: for any DBM M, we write $M = (\mathsf{M}, \prec^M)$, where M is the matrix made of the first components, with elements in $\mathbb{R} \cup \{\infty\}$, while $\prec^M$ is the matrix of the second components, with elements in $\{<, \leq\}$. A DBM M naturally represents a zone (which we abusively write M as well), defined as the set of valuations v such that, for all $x, y \in \mathcal{C}_0$, it holds $v(x) - v(y) \prec^M_{x,y} \mathsf{M}_{x,y}$ (where $v(0) = 0$). Conversely, any zone can be represented by a DBM. We will write $M \subseteq M'$ to mean that the inclusion holds between the zones defined by the respective DBMs.

We define a total order on $(\mathbb{R} \times \{<, \leq\}) \cup \{(\infty, <)\}$ as follows. We let

$$(a, \prec) \leq (b, \prec') \Leftrightarrow \begin{cases} a < b \\ \text{ or} \\ a = b \text{ and either } \prec = \prec' \text{ or } \prec' = \leq. \end{cases}$$

The addition is defined by $(a, \prec) + (b, \prec') = (a + b, \prec'')$, where $\prec'' = \leq$ if $\prec = \prec' = \leq$, and $<$ otherwise. We also let $(a, \prec) + b = (a + b, \prec)$.

For any DBM M, let $\mathsf{G}(M)$ denote the graph over nodes $\mathcal{C}_0$, where there is an edge $(x, y) \in \mathcal{C}_0^2$ of weight $M_{x,y}$ if $M_{x,y} < (\infty, <)$. The normalization of M corresponds to assigning to each edge (x, y) the weight of the shortest path in $\mathsf{G}(M)$ from x to y. We say that M is *normalized* when it is stable under normalization [7].

3.2. Shrinking

Consider the automaton $\mathcal{A}$ of Fig. 2, where the goal is to reach ℓ_3. If there is no perturbation or lower bound on the delays between transitions (*i.e.*, $\delta = 0$), then the states from which Controller can reach location ℓ_3 can be computed backwards. One can reach ℓ_3 from location ℓ_2 and any state in the zone $X = (x \leq 2) \wedge (y \leq 1) \wedge (1 \leq x - y)$, shown by (the union of the light and dark) gray areas on Fig. 3 (left); this is the set of time-predecessors of the corresponding guard. The set of winning states from location ℓ_1 is the zone $Y = (x \leq 2)$, shown in Fig. 3 (right), which is simply the set of predecessors of X at ℓ_2. When $\delta > 0$ however, the set of winning states at ℓ_2 is a "shrinking" of X, shown by the dark gray area. If the value of the clock x is too close to 2 upon arrival in ℓ_2, Controller will fail to satisfy the guard $x = 2$ due to the lower bound δ on the delays. Thus, the winning states from ℓ_2 are described by $X \wedge (x \leq 2 - \delta)$. Then, this shrinking is backward propagated to ℓ_1: the winning states are $Y \wedge (x \leq 2 - 2\delta)$, where we "shrink" Y by 2δ in order to compensate for a possible perturbation.

An important observation here is that the sets $X \wedge (x \leq 2 - \delta)$ and $(x \leq 2 - 2\delta)$ precisely describe the winning states at respective locations for any $\delta \geq 0$. Thus, we have a *uniform* description of the winning states parameterized by δ. In general, we will be able to uniformly describe such sets

of winning states "for small enough $\delta > 0$". For instance, if the winning states are described by $x \leq 2 - 2\delta \wedge x \leq 1$ then the set is non-empty for all $\delta \in [0, 1]$. However, we will simplify this set as $x \leq 1$, which is equivalent to the above formula for all $\delta \in [0, \frac{1}{2}]$. Thus, we give up on computing the winning states for *all* values of δ, and concentrate on an infinitesimal analysis. These simplifications allow us to compute the state space by a simpler representation, and still provide safe bounds on δ.

We now define *shrunk DBMs*, a data structure we introduced in [36], in order to manipulate "shrinkings" of zones.

3.3. Shrunk DBMs

For any interval $[a, b]$, we define the *shrinking operator* as $\mathsf{shrink}_{[a,b]}(Z) = \{v \mid v + [a, b] \subseteq Z\}$ for any zone Z. We only use operators $\mathsf{shrink}_{[0,\delta]}$ and $\mathsf{shrink}_{[-\delta,\delta]}$ in the sequel. For a zone Z represented as a DBM, $\mathsf{shrink}_{[0,\delta]}(Z)$ is the DBM $Z - \delta \cdot \mathbb{1}_{\mathcal{C}\times\{0\}}$ and $\mathsf{shrink}_{[-\delta,\delta]}(Z)$ is the DBM $Z - \delta \cdot \mathbb{1}_{\mathcal{C}\times\{0\}\cup\{0\}\times\mathcal{C}}$, for any $\delta > 0$.

Our aim is to handle these DBMs symbolically. For this, we define *shrinking matrices (SM)*, which are nonnegative integer square matrices with zeroes on their diagonals. A *shrunk DBM* is then a pair (M, P) where M is a DBM, P is a shrinking matrix [36]. The meaning of this pair is that we consider DBMs $M - \delta P$ where $\delta \in [0, \delta_0]$ for some $\delta_0 > 0$. In the rest of this paper, we abusively say "for all small enough $\delta > 0$" meaning "there exists $\delta_0 > 0$ such that for all $\delta \in [0, \delta_0]$". We also adopt the following notation: when we write a statement involving shrunk DBMs (M, P), we mean that the statement holds for $(M - \delta P)$ for all small enough $\delta > 0$. For instance, $(M, P) = \mathsf{Pre}_{\mathsf{time}}((N, Q))$ means that $M - \delta P = \mathsf{Pre}_{\mathsf{time}}(N - \delta Q)$ for all small enough $\delta > 0$. In the same vein, shrunk DBMs can be re-shrunk, and we write $\mathsf{shrink}((M, P))$ (resp. $\mathsf{shrink}^+((M, P))$ for the shrunk DBM (N, Q) such that $N - \delta Q = \mathsf{shrink}_{[-\delta,\delta]}(M - \delta P)$ (resp. $N - \delta Q = \mathsf{shrink}_{[0,\delta]}(M - \delta P)$) for all small enough $\delta > 0$.

As usual DBMs, shrunk DBMs also have *normal forms*. Similarly, the normalization of shrunk DBMs can be defined using finite shortest paths in the graph $\mathsf{G}(M)$ of a given DBM M, computed by the Floyd-Warshall algorithm. In order to adapt this algorithm to our data structure, we need to define the basic operations of summation and minimization on shrunk DBMs. We define, $((\alpha, \prec), k) + ((\beta, \prec'), l) = ((\alpha + \beta, \prec''), k + l)$, where $\prec'' = \leq$ if $\prec = \prec' = \leq$ and $<$ otherwise. The comparison of elements is defined as follows.

$$((\alpha, \prec), k) \preceq ((\beta, \prec'), l) \quad \Leftrightarrow \quad \begin{cases} \alpha < \beta \text{ or} \\ \alpha = \beta \text{ and } k > l, \text{ or} \\ \alpha = \beta \text{ and } k = l \text{ and } \prec' = < \ \Rightarrow \ \prec = <. \end{cases} \tag{1}$$

Intuitively, the smaller constraint is the most restrictive. Note that if $((\alpha, \prec), k) \preceq ((\beta, \prec'), l)$, then one can compute δ_0 such that $\alpha - k\delta \leq \beta - l\delta$ for all $\delta \in [0, \delta_0]$. The minimum is then defined as follows.

$$\min(((\alpha, \prec), k), ((\beta, \prec'), l)) = \begin{cases} ((\alpha, \prec), k) & \text{if } ((\alpha, \prec), k) \preceq ((\beta, \prec'), l), \\ ((\beta, \prec'), l) & \text{otherwise.} \end{cases} \tag{2}$$

Now, a shrunk DBM (M, P) is *normalized* if for all $x, y, z \in \mathcal{C}_0$, $(M_{x,y}, P_{x,y}) \leq (M_{x,z}, P_{x,z}) + (M_{z,y}, P_{z,y})$. As one might expect, a shrunk DBM means that the DBMs $M - \delta P$ are normalized for small enough $\delta > 0$ as shown in the next lemma.

Lemma 3.1. *For any shrunk DBM (M, P) that is normalized, there exists $\delta_0 > 0$ such that $M - \delta P$ is a normalized DBM for all $\delta \in [0, \delta_0]$.*

Proof. This follows from the fact that there is a finite number of inequalities to satisfy for the normalization of (M, P), so one can choose δ small enough to satisfy all of them. It follows that for such $\delta > 0$, for all $x, y, z \in \mathcal{C}_0$, $\mathsf{M}_{x,y} - \delta P_{x,y} \prec^M_{x,y} \mathsf{M}_{x,z} - \delta P_{x,z} + \mathsf{M}_{z,y} - \delta P_{z,y}$, which means that the DBM $M - \delta P$ is normalized. □

Any shrunk DBM can be made normalized by the Floyd-Warshall shortest paths algorithm applied with the operations defined on shrunk DBMs. Algorithm 1 recalls this procedure, and the next lemma shows it correctness. Let us define $\mathsf{norm}((M, P))$ the normalized shrunk DBM obtained by this procedure.

```
Data: Shrunk DBM (M, P)
for x ∈ C_0 do
    for y ∈ C_0 do
        for z ∈ C_0 do
            (M, P)_{x,y} := min((M, P)_{x,y}, (M, P)_{x,z} + (M, P)_{z,y});
        end
    end
end
```

Algorithm 1: Normalization algorithm for shrunk DBMs.

Lemma 3.2. *For any shrunk DBM (M, P), let (M', P') denote the result of the computation of Algorithm 1. Then, (M', P') is normalized, and there exists $\delta_0 > 0$ such that for all $\delta \in [0, \delta_0]$, $M - \delta P$ describes the same set as $M' - \delta P'$.*

Proof. The fact that (M', P') is normalized as a shrunk DBM follows by the correctness of the shortest paths algorithm. Furthermore, since the algorithm performs a finite number of operations, all minima computed during the procedure hold for all small enough $\delta > 0$. Therefore, for all $\delta \in [0, \delta_0]$, where δ_0 can be computed by updating it at each iteration, $M' - \delta P'$ and $M - \delta P$ are equivalent. □

We will now consider shrunk DBMs (M, P) where M is already normalized as a DBM. In this case, the normalization of the shrunk DBM can be described considering the shortest paths in the constraint graph. We consider paths $\pi = \pi_1 \dots \pi_n$ of $\mathsf{G}(M)$ *weighted by* a given DBM or any matrix, as follows. For a matrix P, the *P-weight* of π, written $P(\pi)$, (or the weight of π in P) is the sum of the weights $P_{\pi_j, \pi_{j+1}}$ for $1 \leq j \leq n-1$. Recall that M denotes the matrix of the constants of a DBM M (without the inequalities).

Definition 3.3. *Let M be a normalized DBM. For any $x, y \in \mathcal{C}_0$, we define $\Pi_{x,y}(\mathsf{G}(M))$ as the set of paths with least and finite M-weight from x to y in $\mathsf{G}(M)$.*

Notice that the shortest paths are defined with respect to weights in M and not M. The M-sign of a path π, written $\mathsf{sign}_M(\pi)$ is $<$ if $\prec^M_{\pi_i, \pi_{i+1}} = <$ for some i, and $\leq$ otherwise. Finite-weighted shortest paths can be used to characterize the non-emptiness and the normalization of shrunk DBMs.

Lemma 3.4. *Let M be a normalized non-empty DBM and P be a shrinking matrix. Then, (M, P) is non-empty if, and only if, for all $x \in \mathcal{C}_0$, there is no path in $\Pi_{x,x}(\mathsf{G}(M))$ with positive P-weight. Moreover, if (M, P) is not empty, then, define (M', P') by letting $P'_{x,y}$ be the largest P-weight of the paths in $\Pi_{x,y}(\mathsf{G}(M))$, and define $M'_{x,y} = (\mathsf{M}_{x,y}, \prec^{M'}_{x,y})$ where $\prec^{M'}_{x,y} = <$ if some path of $\Pi_{x,y}(\mathsf{G}(M))$ of P-weight $P'_{x,y}$ has M-sign $<$, and $\leq$ otherwise. Then $(M', P') = \mathsf{norm}((M, P))$.*

Proof. Observe that (M', P') satisfies the following, for all $x, y, z \in \mathcal{C}_0$,

$$\mathsf{M'}_{x,y} = \mathsf{M'}_{x,z} + \mathsf{M'}_{z,y} \quad \Rightarrow \quad (\mathsf{M'}_{x,y}, P'_{x,y}) \prec^{M'}_{x,y} (\mathsf{M'}_{x,z}, P'_{x,z}) + (\mathsf{M'}_{z,y}, P'_{z,y}). \tag{3}$$

In fact, the left hand side condition means that $(x, z, y) \in \Pi_{x,y}(\mathsf{G}(M))$, so, by definition of P', $P'_{x,y} \geq P'_{x,z} + P'_{z,y}$, where in case of equality, $\mathsf{sign}_{M'}(x, z, y) =<$ implies $\prec^{M'}_{x,y}=<$ by construction. It follows that (M', P') is normalized.

Now, a normalized DBM is non-empty if, and only if all its diagonal entries are $(0, \leq)$. Therefore, (M, P) is empty whenever $\Pi_{x,x}(\mathsf{G}(M))$ has a path with positive P-weight, for some $x \in \mathcal{C}_0$. If any such path has P-weight equal to 0, then $\prec^{M}_{x,y}=\leq$ since M is non-empty. The first statement follows. □

Remark 3.5. *Consider a normalized DBM M and any SM P. If $(M', P') = \mathsf{norm}((M, P))$, then $\mathsf{M'} = \mathsf{M}$, $M \subseteq M'$ and $P \leq P'$. This follows from the definition of normalization. In fact, normalization does not change the values M, and if $\prec^{M'}_{x,y}= <$, then $\Pi_{x,y}(\mathsf{G}(M))$ has a path with sign $<$, so $\prec^{M}_{x,y}= <$. That $P \leq P'$ follows from the fact that for all x, y, $P'_{x,y}$ is the greatest P-weight of the shortest paths from x to y, so it is at least $P_{x,y}$.*

Here is an example where the signs of the inequalities change due to normalization: Consider $x \leq 1 \wedge 1 \leq y \wedge x - y < 1$, which is normalized (seen as a DBM over two clocks). However the shrinking $x \leq 1 - \delta \wedge 1 \leq y \wedge x - y < 1$, yields, after normalization $x \leq 1 - \delta \wedge 1 \leq y \wedge x - y \leq 1 - \delta$, where the last inequality becomes non-strict.

It was shown in [36] that when usual operations are applied on shrunk DBMs, one always obtains shrunk DBMs, whose shrinking matrices can be computed. The proof was only given for shrunk DBMs with non-strict inequalities. We recall the constructions, and extend it to shrunk DBMs with both strict and non-strict inequalities.

Lemma 3.6 ([36])**.** *Let M, M', N be normalized non-empty DBMs.*

1. *If $N = \mathsf{Pre}_{\mathsf{time}}(M)$, then for all SMs P, there exists a shrunk DBM (N', Q) such that $\mathsf{N} = \mathsf{N'}$, $N \subseteq N'$ and $(N', Q) = \mathsf{Pre}_{\mathsf{time}}((M, P))$. Moreover, Q is obtained from P, by setting $Q_{0,x} = 0$ for all $x \in \mathcal{C}$ and applying normalization.*

2. *If $N = M \cap O$, then for all SMs P and P', there exists a shrunk DBM (N', Q) such that $\mathsf{N'} = \mathsf{N}$, $N \subseteq N'$ and $(N', Q) = (M, P) \cap (O, P')$. Moreover, (N', Q) is given by setting $(N'_{x,y}, Q_{x,y}) = \min((M_{x,y}, P_{x,y}), (O_{x,y}, P'_{x,y}))$ and applying normalization.*

3. *If $N = \mathsf{Unreset}_R(M)$ for some $R \subseteq \mathcal{C}$, then for all SMs P, there exists a shrunk DBM (N', Q) such that $\mathsf{N'} = \mathsf{N}$, $N \subseteq N'$ and $(N', Q) = \mathsf{Unreset}_R((M, P))$. To obtain (N', Q), first compute (N'', Q') such that $(N'', Q') = (M, P) \cap (R = 0)$ by previous case, then set all components (x, y) with $x \in R$ or $y \in R$, to $(N'', Q')_{x,y} = ((\infty, <), 0)$ and apply normalization.*

4. *If $N = \mathsf{Post}_{\mathsf{time}}(M)$, then for all SMs P, there exists a shrunk DBM (N, Q) such that $(N, Q) = \mathsf{Post}_{\mathsf{time}}((M, P))$. Q can be obtained from P by setting $Q_{x,0} = 0$ for all $x \in \mathcal{C}$.*

5. *If $(N, Q) = \mathsf{shrink}((M, P))$, then $\mathsf{N} = \mathsf{M}$ and Q is obtained from P by incrementing all $P_{x,0}$ and $P_{0,x}$ for $x \in \mathcal{C}$. For shrink^+, one only increments $P_{x,0}$, for all $x \in \mathcal{C}$.*

Proof. The proofs for shrunk DBMs with only non-strict inequalities were given in [36], except for the $\mathsf{Post}_{\mathsf{time}}$ operator which we prove below.

Let us recall here the idea and explain the extension to strict inequalities.

Assume (M, P) is normalized. All operations are defined following corresponding ones for usual DBMs. For instance, to compute $\mathsf{Pre}_{\mathsf{time}}((M, P))$, one sets the first row to $((0, \leq), 0)$, then applies normalization. In fact, for small enough $\delta > 0$, this is precisely the computation of $\mathsf{Pre}_{\mathsf{time}}(M - \delta P)$, seen as a classical DBM. Note that the bound on δ and the types of the inequalities can (only) change during normalization. The rest of the statement follows from Remark 3.5.

However (M, P) cannot always be assumed to be normalized since P is chosen arbitrarily. If $N = f(M)$ denotes one of the equations in this lemma, because all operations are non-decreasing, if $(M', P') = \mathsf{norm}((M, P))$, then $N'' = f(M')$ satisfies $N \subseteq N''$, and moreover $\mathsf{N} = \mathsf{N}''$. The latter equality follows from $N \subseteq N''$ and the fact that any shortest path of N is also a shortest path of N'', though possibly with a different sign. Thus, by Remark 3.5, the lemma yields a shrunk DBM $(N', Q) = f((M', P)) = f((M, P))$ with $N \subseteq N'$ and $\mathsf{N} = \mathsf{N}'$, as desired.

We now prove the case of $\mathsf{Post}_{\mathsf{time}}$. The $\mathsf{Post}_{\mathsf{time}}$ operation on DBMs simply sets the first column to $(\infty, <)$. The resulting DBM is already normalized. Accordingly, we define $Q'_{x,y} = P_{x,y}$ for all $(x, y) \in \mathcal{C}_0 \times \mathcal{C}$ and $Q'_{x,0} = 0$ for $x \in \mathcal{C}_0$, and let $(N', Q) = \mathsf{norm}((N, Q'))$. In fact, for small enough $\delta > 0$, this corresponds to setting the first column of $N - \delta Q$ to $((\infty, <), 0)$ and applying normalization. □

Corollary 3.7. *Let $M = f(N_1, \ldots, N_k)$ be an equation between normalized DBMs $M, N_1, \ldots, N_k$, using the operators $\mathsf{Pre}_{\mathsf{time}}$, $\mathsf{Post}_{\mathsf{time}}$, $\mathsf{Unreset}_R$, $\cap$, shrink and shrink^+ and let $P_1, \ldots, P_k$ be SMs. Then, there exists a shrunk DBM (M', Q) with $\mathsf{M}' = \mathsf{M}$, $M \subseteq M'$ and $(M', Q) = f\big((N_1, P_1), \ldots, (N_k, P_k)\big)$. The shrunk DBM (M', Q) and a corresponding upper bound on δ can be computed in polynomial time.*

Corollary 3.7 shows that shrinking matrices are propagated when usual operations are applied on DBMs. Using this, one can also compute an upper bound on δ, under which a given equation between shrunk DBMs hold. Figure 4 illustrates some of these operations. Notice also that the lemma always gives *normalized* shrunk DBMs (M', Q).

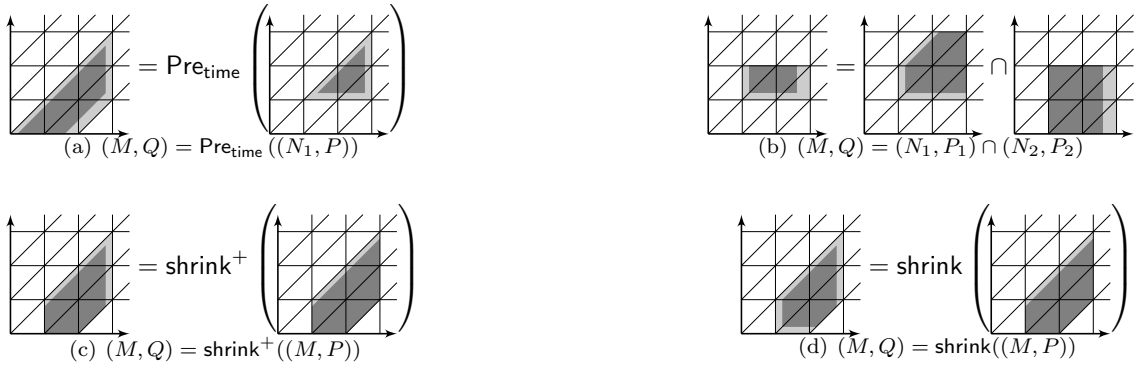


Figure 4: On the right of Fig. 4(a), the gray area represents a zone N_1, while the dark gray area is some shrinking (N_1, Q). On the left, the dark area is the shrunk zone (M, P) where $M = \mathsf{Pre}_{\mathsf{time}}(N_1)$. Similarly, Fig. 4(b) to 4(d) illustrate the intersection of two shrunk zones and the shrinking of zones.

It is easily observed that $\mathsf{shrink}_{[0,\delta]}(\mathsf{Pre}_{\mathsf{time}}(M))$ is the set of states that can reach M after a delay of length at least δ: $\mathsf{shrink}_{[0,\delta]}(\mathsf{Pre}_{\mathsf{time}}(M)) = \{v \mid \exists d \geq \delta, v + d \in M\}$. This observation is used in the construction of the abstract game in Section 4.

Let us extend the computation of upper bounds on δ in Lemma 3.6 for an inclusion relation, which will be useful later in the proofs (proof of Lemma 3.17).

Lemma 3.8. *Assume the equation $(M, P) \cap N \subseteq (N, Q)$ between normalized shrunk DBMs (M, P) and (N, Q). One can compute $\delta_0 > 0$, in polynomial time, such that $(M - \delta P) \cap N \subseteq N - \delta Q$ for all $\delta \in [0, \delta_0]$.*

Proof. Let $N' = M \cap N$. Let Q' be the SM given by Lemma 3.6 such that $(N', Q') = (M, P) \cap N$ and $\delta'_0 > 0$ the corresponding bound on δ. Then, the inclusion is equivalent to $(N', Q') \subseteq (N, Q)$. Since both shrunk DBMs are normalized, holds if for all $x, y \in \mathcal{C}_0$, either $N'_{x,y} = N_{x,y}$ and $Q'_{x,y} \geq Q_{x,y}$, or $N'_{x,y} < N_{x,y}$. In the former case, $N'_{x,y} - \delta Q'_{x,y} \leq N_{x,y} - \delta Q_{x,y}$ holds for all $\delta > 0$. In the latter case, it holds for all $\delta < \left|\frac{N_{x,y} - N'_{x,y}}{Q_{x,y} - Q'_{x,y}}\right|$. Thus, we choose

$$\delta_0 = \min\left(\delta'_0, \min \left|\frac{N_{x,y} - N'_{x,y}}{Q_{x,y} - Q'_{x,y}}\right|\right),$$

where the inner min is over all pairs $x, y \in \mathcal{C}_0$ for which $N'_{x,y} < N_{x,y}$ and $Q_{x,y} \neq Q'_{x,y}$. □

3.4. Shrinking constraints

We now apply the previous developments to our controller synthesis problem. We will explain the different steps of the computations on the example of Fig. 5. Here, a timed automaton with a single transition is given. From state $x = y = 0$, Controller has no choice but to suggest a delay of 1. Thus, from region r_0, the game can reach regions r_1, r_2, r_3, depending on the move of Perturbator. Assuming ℓ and ℓ' are not the target state, in order to win, Controller needs a winning strategy at ℓ' from all three regions. The idea of our algorithm is to run a forward exploration in order to identify these successor regions r_1, r_2, r_3 from which winning strategies need to be found. We will prove that given winning strategies from these successors, one can construct a winning strategy for ℓ as well.

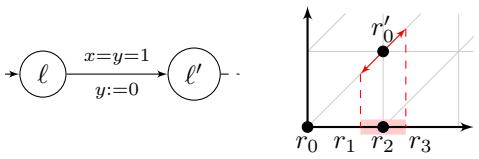


Figure 5: Perturbing one transition.

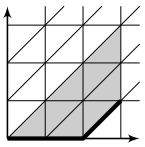
However, winning strategies generally require shrinking, as demonstrated in Fig. 3. It is easy to see that not all shrinkings of these regions provide a winning strategy from r_0; in fact, r_1 (resp. r_3) should not shrink from the right (resp. left) side: their union should include the shaded area, thus points that are arbitrarily close to r_2. The goal of this section is to study *shrinking constraints* in order to express this kind of requirements on the shape of the shrinkings of regions. In the next section, we will show how these constraints can be computed by a forward exploration of the regions of a timed automaton, and in Section 3.6, we will show how a winning strategy can be derived once we have these required constraints.

A shrinking constraint is a matrix that specifies which facet of a given zone can be shrunk by a positive parameter, and which ones cannot be shrunk at all. We now give the formal definition.

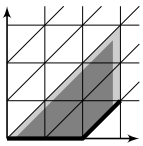
Definition 3.9. *Let M be a DBM. A* shrinking constraint *for M is a $|\mathcal{C}_0| \times |\mathcal{C}_0|$ matrix over $\{0, \infty\}$. A shrinking matrix P is said to* respect *a shrinking constraint S if $P \leq S$, where the comparison is component-wise. A pair $\langle M, S\rangle$ of a DBM and a shrinking constraint is called a* constrained DBM.

Shrinking constraints specify which facets of a given zone one is (or is not) allowed to shrink (see Fig. 6). More precisely, these specify if a component of considered shrinking matrices *must* be 0 or it can also be positive.

$$\begin{array}{c} \\ 0 \\ x \\ y \end{array} \begin{array}{c} \begin{array}{ccc} 0 & x & y \end{array} \\ \begin{pmatrix} (0,\leq),0 & (0,<),\infty & (0,<),0 \\ (3,<),\infty & (0,\leq),0 & (2,<),0 \\ (3,<),\infty & (0,<),\infty & (0,\leq),0 \end{pmatrix} \end{array}$$



(a) A constrained DBM $\langle M, S\rangle$ and its representation



(b) A shrinking of $\langle M, S\rangle$

Figure 6: Consider a zone defined by $0 < x < 3$, $0 < y < 3$, and $0 < x - y < 2$. Let the shrinking constraint S be defined by $S_{0,y} = 0$, $S_{x,y} = 0$, and $S_{z,z'} = \infty$ for other components. The resulting $\langle M, S\rangle$ is depicted on the left, as a matrix (where, for convenience, we merged both matrices into a single one) and as a constrained zone (where a thick segment is drawn for any boundary that is not "shrinkable", i.e., with $S_{z,z'} = 0$). On the right, the dark gray area represents a shrinking of M that satisfies S.

We will focus our attention to *well shrinking constraints* defined below, show that shrinking constraints have a normal form (Subsection 3.4.1), and then show how shrinking constraints can be propagated in order to deduce new shrinking constraints as in the example of Fig. 5 (Subsection 3.4.2).

3.4.1. Normalization and Well-Shrinking Constraints.

A shrinking constraint S for a DBM M is said to be *well* if for any SM $P \leq S$, (M, P) is non-empty. A *well constrained DBM* is a constrained DBM given with a well shrinking constraint. We say that a shrinking constraint S for a DBM M is *normalized* if it is the minimum among all equivalent shrinking constraints: for any shrinking constraint S' if for all SMs P, $P \leq S \Leftrightarrow P \leq S'$, then $S \leq S'$. Similarly to the normalization of DBMs or SMs, normalized shrinking constraints contain the tightest constraints implied by the original shrinking constraint. They can be normalized by a procedure that is slightly different from the one used for the normalization of DBMs. This is defined formally in the following lemma.

Lemma 3.10. *Let M be a normalized DBM. For any shrinking constraint S for M, there exists a minimum shrinking constraint S' for M such that $S' \leq S$, and for any normalized non-empty shrunk zone (M, P), $P \leq S$ if, and only if, $P \leq S'$.*

Moreover, S' can be obtained as follows. Start with $S' = S$. For every pair $x, y \in \mathcal{C}_0$, if $S_{x,y} = 0$, then set $S'_{z,z'}$ to 0 for all edges (z, z') of all paths in $\Pi_{x,y}(\mathsf{G}(M))$. Also assign $S'_{x,y} = 0$ whenever $\mathsf{M}_{x,y} = \infty$. If $S = S'$, then S is said to be normalized.

Proof. Consider S' obtained from S by the above procedure. Obviously, $S' \leq S$, so that for any SM $P \leq S'$, we have also $P \leq S$. Conversely, consider any normalized non-empty shrunk DBM (M, P) with $P \not\leq S'$, for instance $P_{x,y} \geq 1$ for some x, y where $S'_{x,y} = 0$. If $S_{x,y} = 0$, then clearly, $P_{x,y} \not\leq S_{x,y}$. Otherwise, $S_{x,y} = \infty$ and $S'_{x,y} = 0$ indicate that there exists $z_1, z_2 \in \mathcal{C}_0$ such that

$S_{z_1,z_2} = 0$ and there is a path in $\Pi_{z_1,z_2}(\mathsf{G}(M))$ that goes through edge (x, y). Let us denote this path with π. Then since P is normalized, $P_{z_1,z_2} \geq w(\pi)$, and since π contains the edge (x, y), we have $P_{z_1,z_2} \geq P_{x,y} \geq 1$, therefore $P \not\leq S$.

For any pair $x, y \in \mathcal{C}_0$ with $S'_{x,y} = \infty$ let P be the matrix with 0's on all components except for $P_{x,y} = 1$. Let P' be the SM such that $(M, P') = \mathsf{norm}((M, P))$. The normalization procedure can only increase the components so $P'_{x,y} \geq 1$. We have $P' \leq S'$, since for any z_1, z_2 such that $S'_{z_1,z_2} = 0$, (x, y) is not on a path of $\Pi_{z_1,z_2}(\mathsf{G}(M))$. So, for any shrinking constraint S'' such that $S''_{x,y} < S_{x,y}$ for some x, y, there exists a SM P with $P \leq S'$ and $P \not\leq S''$. Therefore, S' is minimal. □

Clearly, the normalization of a shrinking constraint depends on the DBM M, since $\Pi_{x,y}(\mathsf{G}(M))$ depends on $\mathsf{G}(M)$; this is also the case for SMs (Lemma 3.4). In the sequel, unless otherwise stated, all shrinking constraints are assumed to be normalized.

The following property of normalized shrinking constraints is easy to prove, using the previous lemma. Given a shrinking constraint S and a SM P, it will allow us to consider that S is normalized, instead of requiring that P is.

Lemma 3.11. *Let M be a normalized DBM and S a normalized shrinking constraint. Let P be any SM such that (M, P) is non-empty, and let $(M', P') = \mathsf{norm}((M, P))$. Then, $P \leq S$ if, and only if, $P' \leq S$.*

Proof. Since the normalization procedure for computing P' increases the components, the reverse implication holds. Conversely, assume that $P \leq S$, and pick $x, y \in \mathcal{C}_0$ such that $S_{x,y} = 0$ (for the other entries, there is nothing to be checked on P'). Then $P_{x,y} = 0$. From Lemma 3.10, $S_{z,z'} = 0$ for all edges of all paths in $\Pi_{x,y}(\mathsf{G}(M))$, so that also $P_{z,z'} = 0$ for these edges. Applying the definition of $P'_{x,y}$ from Lemma 3.4, we get $P'_{x,y} = 0$. □

3.4.2. Propagating Shrinking Constraints.

Let us come back to Fig. 5. Here, starting from the region r_0 given with its well-shrinking constraint, we deduced the shrinking constraints to be satisfied at r_1, r_2, r_3 by manually inspecting the figure. Similarly, in order to find a winning strategy, say, from location ℓ', and region r_1 satisfying the corresponding shrinking constraint, one would need to inspect the edges from ℓ', and derive shrinking constraints for those successor regions.

In this subsection, we show a systematic way of deriving the shrinking constraints along elementary operations on DBMs. Lemma 3.12 studies each elementary operation on DBM; each item is illustrated in Fig. 7.

Lemma 3.12. *Let M, N, O be normalized non-empty DBMs.*

1. *Assume that $M = \mathsf{Pre}_{\mathsf{time}}(N)$. For any normalized well shrinking constraint S for M, there exists a normalized well shrinking constraint S' for N such that for any shrunk DBM (N', Q) with $\mathsf{N}' = \mathsf{N}$, the following holds: $Q \leq S'$ if, and only if, the SM P s.t. $(M', P) = \mathsf{Pre}_{\mathsf{time}}((N', Q))$ satisfies $P \leq S$.*
2. *Assume that $M = N \cap O \neq \emptyset$.*
   (a) *For any normalized well shrinking constraint S for M, there exists a normalized well shrinking constraint S' for N such that for any shrunk DBM (N', Q) with $\mathsf{N}' = \mathsf{N}$ and $N' \cap O \neq \emptyset$, the following holds: $Q \leq S'$ if, and only if the SM P s.t. $(M', P) = (N', Q) \cap O$ satisfies $P \leq S$.*

(b) For any normalized well shrinking constraint S for N, there exists a normalized well shrinking constraint S' for M such that for any shrunk DBM (M', Q) with $\mathsf{M} = \mathsf{M}'$ and $M \subseteq M'$, the following holds: $Q \leq S'$ if, and only if there exists an SM $P \leq S$ such that $(N, P) \cap O \subseteq (M', Q)$. Moreover, if $(N, P) \cap O \neq \emptyset$ for all SMs $P \leq S$, then S' is a well shrinking constraint.

3. Assume that $M = \mathsf{Unreset}_R(N)$. For any normalized well shrinking constraint S for M, there exists a normalized well shrinking constraint S' for N such that for any SM Q, the following holds: $Q \leq S'$ if, and only if the SM P s.t. $(M', P) = \mathsf{Unreset}_R((N, Q))$ satisfies $P \leq S$.

All shrinking constraints S' can be computed in polynomial time.

Proof. ▶ **Pretime.** Observe that M is obtained from N by first setting the first row of N to $(0, \leq)$ (write N' for this intermediary DBM) and then applying normalization. Since N is normalized, and we only consider valuations with non-negative values, we have $N_{0,y} \leq 0$ for all $y \in \mathcal{C}$. Consider $(x, y) \in \mathcal{C} \times \mathcal{C}_0$, and a path in $\Pi_{x,y}(N)$. If that path does not visit the state 0, then it has the same weight in N' as in N. Otherwise, its weight in N' is larger than in N, thus it is larger than $N'_{x,y}$. In both cases, $N'_{x,y}$ will not change during the normalization phase. So the normalization of N can only change the first row.

Let $S'_{x,y} = S_{x,y}$ for $(x, y) \in \mathcal{C} \times \mathcal{C}_0$, $S'_{0,y} = \infty$ for $y \in \mathcal{C}$, and $S'_{0,0} = 0$. We show that S' satisfies the desired property. Consider a shrunk DBM (N', Q) with $Q \leq S'$ and $\mathsf{N} = \mathsf{N}'$, and let P be the shrinking matrix such that $(M', P) = \mathsf{Pre}_{\mathsf{time}}((N', Q))$. Let us write $M'' = \mathsf{Pre}_{\mathsf{time}}(N')$. The shrunk DBM (M', P) is obtained by normalizing the shrunk DBM (M'', Q') where Q' is derived from Q by setting the first row to 0 (Lemma 3.6).

We show that $P \leq S$. Observe that $\mathsf{M} = \mathsf{M}' = \mathsf{M}''$ so the graphs these DBMs define are the same. First suppose that $S_{x,y} = 0$ for some $x, y \in \mathcal{C} \times \mathcal{C}_0$. For such x, y, we have $\mathsf{N}_{x,y} = \mathsf{M}_{x,y}$, and since $N' \subseteq M''$, we have $\Pi_{x,y}(\mathsf{G}(M)) \subseteq \Pi_{x,y}(\mathsf{G}(N))$. Now, we have $P_{x,y} > 0$ if, and only if there is a path in the former set with positive weight in Q' (Lemma 3.4). But this would imply that the same path has positive weight in Q since $Q' \leq Q$, and moreover this path belongs to $\Pi_{x,y}(\mathsf{G}(N))$. This would in turn imply that $Q_{x,y} > 0$, but we have $S_{x,y} = S'_{x,y} = 0$, which contradicts $Q \leq S'$. It follows that $P_{x,y} \leq S_{x,y}$.

Now suppose that $S_{0,y} = 0$ for some $y \in \mathcal{C}$. Since S is assumed to be normalized, along all paths of $\Pi_{0,y}(\mathsf{G}(M))$, all edges (z, z') satisfy $S_{z,z'} = 0$ (Lemma 3.10). Since $Q'_{0,z} = 0$ for all z, and that $Q'_{z,z'} = Q_{z,z'} = 0$ for all $z, z' \neq 0$, we get $P_{0,y} = 0$ (Lemma 3.4). Hence $P \leq S$.

We now show that for any SMs P and Q s.t. $Q \not\leq S'$ and $(M', P) = \mathsf{Pre}_{\mathsf{time}}((N', Q))$, it holds $P \not\leq S$. Consider any SM $Q \not\leq S'$ such that (w.l.o.g.) (N', Q) is normalized. Then $S'_{x,y} = 0$ and $Q_{x,y} \geq 1$ for some x, y. By construction of S', we have $x \neq 0$, and $S_{x,y} = 0$. Moreover, since $x \neq 0$, we have $\mathsf{M}_{x,y} = \mathsf{N}_{x,y}$, and $\mathsf{N}_{x,y} < \infty$ since $Q_{x,y} \geq 1$ and (N', Q) is normalized. Let P denote the normalized SM such that $(M', P) = \mathsf{Pre}_{\mathsf{time}}((N'', Q))$. P is obtained by letting the first row of Q to zero and applying normalization. We get $P_{x,y} \geq Q_{x,y} \geq 1$, therefore $P \not\leq S$.

To show that S' is a well shrinking constraint, assume that for some $Q \leq S'$, (N', Q) is empty. By definition of S', there exists $P \leq S$ such that $(M', P) = \mathsf{Pre}_{\mathsf{time}}((N', Q))$, so (M', P) is also empty, which contradicts the fact that S is a well shrinking constraint for M.

▶ **Intersection (a).** For any SMs P, Q, we have $(M, P) = (N, Q) \cap O$ if, and only if $(M', P) = (N, Q) \cap M$ for some $\mathsf{M}' = \mathsf{M}$. So it suffices to consider the equation $M = N \cap M$ with $M \subseteq N$. Moreover, notice that $\emptyset \neq M = M \cap N'$ for any $N \subseteq N'$.

We let $S'_{x,y} = S_{x,y}$ for any $x, y \in \mathcal{C}_0$ such that $\mathsf{M}_{x,y} = \mathsf{N}_{x,y}$ and $S'_{x,y} = \infty$ otherwise, and make S' normalized. Observe that the weight of any path is smaller or equal in $\mathsf{G}(M)$ than in $\mathsf{G}(N)$, since $M \subseteq N$ and both DBMs are normalized. Consider any shrunk DBM (N', Q) with $\mathsf{N} = \mathsf{N}'$, $N \subseteq N'$ and $Q \leq S'$ and let P such that $(M', P) = (N', Q) \cap M$. Let us write $M'' = N' \cap M$. Again, $\mathsf{G}(M) = \mathsf{G}(M') = \mathsf{G}(M'')$. Here, P is obtained from Q by first defining Q' as $Q'_{x,y} = 0$ if $\mathsf{M}_{x,y} < \mathsf{N}_{x,y}$ and $Q'_{x,y} = Q_{x,y}$ otherwise. Then P is the normalization of Q'. We show, by induction on $n \geq 2$, that for any pair $x, y \in \mathcal{C}_0$ with $S_{x,y} = 0$, any path of $\Pi_{x,y}(\mathsf{G}(M))$ visiting at most n states have weight 0 in Q'. This entails that $P_{x,y} = 0$ whenever $S_{x,y} = 0$.

- When $n = 2$, if $\mathsf{M}_{x,y} = \mathsf{N}_{x,y}$, then $S'_{x,y} = 0$ by definition of S', so $Q_{x,y} = 0$, and $Q'_{x,y} = 0$. If $\mathsf{M}_{x,y} < \mathsf{N}_{x,y}$, we have $Q'_{x,y} = 0$ by the definition of Q'.
- Consider $n \geq 3$. Let $\pi = x_1 x_2 \dots x_n$ be a path in $\Pi_{x,y}(\mathsf{G}(M))$. Since $S_{x,y} = S_{x_1,x_n} = 0$, we also have $S_{x_1,x_2} = 0$ and $S_{x_2,x_n} = 0$ By induction, $Q'_{x_1,x_2} = 0$ and all paths of $\Pi_{x_2,x_n}(\mathsf{G}(M))$ of length at most $n-1$ have weight 0 in Q'. Hence, π has weight 0 in Q'.

Assume now $Q \not\leq S'$, i.e., $Q_{x,y} \geq 1$ and $S'_{x,y} = 0$ for some $x, y \in \mathcal{C}_0$. Then we must have $\mathsf{M}_{x,y} = \mathsf{N}_{x,y}$ and $S_{x,y} = 0$. Let $(M', P) = (N', Q) \cap M$. We know that P is the normalization of Q', and that $Q'_{x,y} = Q_{x,y} \geq 1$. So $P_{x,y} \geq Q'_{x,y} \geq 1$. Hence $P_{x,y} \not\leq S_{x,y}$.

We have that S' is a well shrinking constraint for N, since otherwise this would imply that S is not a well shrinking constraint for M, as in the previous case.

▶ **Intersection (b).** Using the same argument as in the previous case, we can assume that $O = M \subseteq N$. Since M and N are normalized and non-empty, we have $\mathsf{M}_{x,y} \leq \mathsf{N}_{x,y}$ for all $x, y \in \mathcal{C}_0$. We define S'_1 by $S'_{1\,x,y} = S_{x,y}$ if $\mathsf{M}_{x,y} = \mathsf{N}_{x,y}$ and $S'_{1\,x,y} = 0$ otherwise. Let S' be defined as $S'_{x,y} = \max_{\pi \in \Pi_{x,y}(\mathsf{G}(M))} S'_1(\pi)$.

Consider any shrunk DBM (M', Q) with $\mathsf{M}' = \mathsf{M}$, $(M', Q) \subseteq M$, and $Q \leq S'$. Let us show that there is (N, P) with $P \leq S$ with the desired property. Let us first define Q' as follows:

$$Q'_{x,y} = \begin{cases} 0 & \text{if } \mathsf{M}_{x,y} < \mathsf{N}_{x,y} \text{ or } S_{x,y} = 0 \\ \max\limits_{z,z' : (x,y) \in \Pi_{z,z'}(\mathsf{G}(M))} Q_{z,z'} & \text{if } \mathsf{M}_{x,y} = \mathsf{N}_{x,y} \text{ and } S_{x,y} = \infty. \end{cases}$$

Let us write $(N', P) = \mathsf{norm}((N, Q'))$. If $S_{x,y} = 0$ for some $x, y \in \mathcal{C}_0$, then along all edges $(z, z') \in \Pi_{x,y}(\mathsf{G}(N))$, we have $S_{z,z'} = 0$, therefore $Q'_{z,z'} = 0$. Thus, $P_{x,y} = 0$ (from Lemma 3.4) and we get $P \leq S$.

Let (M'', Q'') denote the shrunk DBM defined by $(M'', Q'') = (N', P) \cap M$ with $\mathsf{M}'' = \mathsf{M}$ (Lemma 3.6). Let us show that $Q \leq Q''$, which implies, together with $M'' \subseteq M \subseteq M'$ that $(M'', Q'') \subseteq (M', Q)$ as desired. Q'' is the normalization of the SM P' defined as follows: $P'_{x,y} = P_{x,y}$ if $\mathsf{M}_{x,y} = \mathsf{N}_{x,y}$ and $P'_{x,y} = 0$ otherwise. Let $x, y \in \mathcal{C}_0$.

- Assume that $\mathsf{M}_{x,y} = \mathsf{N}_{x,y}$. If $S'_{x,y} = 0$, then $Q_{x,y} = 0$ so clearly $Q''_{x,y} \geq Q_{x,y}$. Otherwise, there exists a path $\pi \in \Pi_{x,y}(\mathsf{G}(M))$ such that for some edge $(z, z') \in \pi$, $\mathsf{M}_{z,z'} = \mathsf{N}_{z,z'}$ and $S_{z,z'} = \infty$. By definition of Q', we have $Q'_{z,z'} \geq Q_{x,y}$. Moreover $P'_{z,z'} = P_{z,z'} \geq Q_{z,z'}$. The normalization of P' then yields $Q''_{x,y} \geq P'_{z,z'} = P_{z,z'} \geq Q_{z,z'} \geq Q_{x,y}$.
- Otherwise, $\mathsf{M}_{x,y} < \mathsf{N}_{x,y}$. If $S'_{x,y} = 0$, the result is clear. If $S'_{x,y} = \infty$, then there must be a path in $\Pi_{x,y}(\mathsf{G}(M))$ with an edge (z, z') with $\mathsf{M}_{z,z'} = \mathsf{N}_{z,z'}$ and $S_{z,z'} = \infty$. Then, $Q'_{z,z'} \geq Q_{x,y}$. So the normalization of P' yields $Q''_{x,y} \geq P'_{z,z'} = P_{z,z'} \geq Q'_{z,z'} \geq Q_{x,y}$.

We now show the converse direction. Consider any $x, y \in \mathcal{C}_0$ such that $S'_{x,y} = 0$. Let us show that for any (N, P) with $P \leq S$, if (M', Q'') denotes the shrunk DBM such that $(M', Q'') = M \cap (N, P)$, then $Q''_{x,y} = 0$. This proves that if $Q_{x,y} \geq 1$ for some SM Q, then there is no matching $P \leq S$ that satisfies the property.

We show, by induction on $n \geq 2$, that if $S'_{x,y} = 0$, then all paths of $\Pi_{x,y}(\mathsf{G}(M))$ of length at most n have weight zero in P'.

- Let $n = 2$. If $\mathsf{M}_{x,y} = \mathsf{N}_{x,y}$, then $P'_{x,y} = P_{x,y}$. But we have $S_{x,y} = 0$, so $P_{x,y} = 0$. If $\mathsf{M}_{x,y} < \mathsf{N}_{x,y}$, then $P'_{x,y} = 0$ by definition.

- If $n \geq 3$, consider any path $\pi = x_1 x_2 \dots x_n$ in $\Pi_{x,y}(\mathsf{G}(M))$. Suppose that $\mathsf{M}_{x_j,x_{j+1}} = \mathsf{N}_{x_j,x_{j+1}}$ for all $1 \leq j \leq n-1$. Then π is also a shortest path in $\mathsf{G}(N)$, hence $\mathsf{M}_{x,y} = \mathsf{N}_{x,y}$, and $S_{x_j,x_{j+1}} = 0$ for all $1 \leq j \leq n-1$, as S is normalized. It follows that $P_{x_j,x_{j+1}} = 0$, hence also $P'_{x_j,x_{j+1}} = 0$.

  Suppose now that for some $1 \leq j \leq n-1$, $\mathsf{M}_{x_j,x_{j+1}} < \mathsf{N}_{x_j,x_{j+1}}$. We have $P'_{x_j,x_{j+1}} = 0$. On the other hand, $S'_{x_1,x_j} = S'_{x_{j+1},x_n} = 0$ since otherwise we would get $S'_{x,y} = \infty$. By induction, all paths of length at most $n-1$ in $\Pi_{x_1,x_j}(\mathsf{G}(M))$ and $\Pi_{x_{j+1},x_n}(\mathsf{G}(M))$ have weight 0 in P'. So π has weight 0 in P'.

We now show that S' is a well shrinking constraint for M, given the hypothesis that $(N, P) \cap M \neq \emptyset$ for all SMs $P \leq S$. But this immediately implies that S' is a well shrinking constraint, since for any $Q \leq S'$ there exists $P \leq S$ with $\emptyset \subsetneq (N, P) \cap M \subseteq (M, Q)$.

► **Unreset.** Let N_R denote the DBM that defines the (largest) zone satisfying $\wedge_{x \in R} x = 0$. Since we have $M = \mathsf{Unreset}_R(N \cap N_R)$, by the previous case, we may assume that $N \subseteq N_R$. The DBM M is obtained from N by replacing each component (x, y) with $x \in R$ by ∞. For any $y \in \mathcal{C}_0$, we define $S''_{x,y} = S_{x,y}$ if $x \in \mathcal{C} \setminus R$, and $S''_{0,y} = 0$, and $S''_{x,y} = \infty$ if $x \in R$. Then S' is obtained by normalizing S''.

Let Q be any normalized SM with $Q \leq S'$, and P be the (normalized) SM such that $(M', P) = \mathsf{Unreset}_R((N, Q))$. Let Q' denote the SM obtained from Q by setting each component (x, y), with $x \in R$, to zero. Then, P is the normalization of Q'. Let us show that $P \leq S$. Consider any $x, y \in \mathcal{C}_0$ with $S_{x,y} = 0$.

- If $x \in \mathcal{C} \setminus R$ then $S'_{x,y} = S_{x,y} = 0$ and $Q_{x,y} = 0$. So, along all paths $\Pi_{x,y}(\mathsf{G}(N))$, S' is zero, and so are the weights in Q, and also the weights in Q'. Now, clearly, $\Pi_{x,y}(\mathsf{G}(M)) \subseteq \Pi_{x,y}(\mathsf{G}(N))$. So after normalization of Q', we have $P_{x,y} = 0$.

- If $x = 0$, then $S'_{x,y} = 0$ and $Q_{x,y} = 0$. The argument is then similar.

- If $x \in R$, then $M_{x,y} = \infty$. Thus $\Pi_{x,y}(\mathsf{G}(M)) = \emptyset$, so the normalization of Q' yields $P_{x,y} = 0$.

Consider a normalized SM Q with $Q \not\leq S'$. We must have $Q_{x,y} \geq 1$ and $S''_{x,y} = 0$ for some $x, y \in \mathcal{C}_0$, by Lemma 3.10. The latter condition entails that $x \notin R$. Let Q' denote the DBM obtained from Q by replacing all components (x, y) where $x \in R$ with 0. Then the normalization of Q' yields the SM P such that $(M', P) = \mathsf{Unreset}_R((N, Q))$. Since $x \in \mathcal{C}_0 \setminus R$, then $S_{x,y} = S'_{x,y} = 0$ and $Q'_{x,y} = Q_{x,y} \geq 1$. The normalization of Q' can only increase its components, so $P_{x,y} \geq 1$, hence $P \not\leq S$.

We show that S' is a well shrinking constraint for N as in the previous cases. □

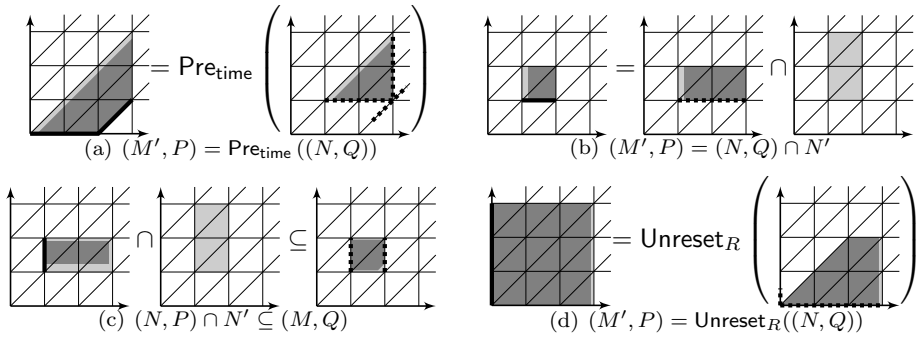


Figure 7: Each of the figures illustrates one item in Lemma 3.12. In each case, DBMs M, N and N' are fixed and satisfy the "unshrunk" equation (that is, when $P = Q = \mathbf{0}$). The thick plain segments represent the fixed shrinking constraint S. The thick dashed segments represent the shrinking constraint S' that is constructed. In each case, if a SM Q is chosen, it holds that $Q \leq S'$ iff there is an SM $P \leq S$ that satisfies the equation. For instance, if $Q_{x,0} \geq 1$ in (a), then the DBM describing the time predecessors of (N, Q) shrinks the diagonal component (x, y), which violates S.

We now extend the previous lemma to the operator shrink^+. In this case, the existence of a shrinking constraint depends on the given constrained DBM. The lemma is illustrated in Fig. 8.

Lemma 3.13. *Let M be a normalized non-empty DBM and S be a normalized shrinking constraint.*

- *If $M_{x,0} < \infty$ and $S_{x,0} = 0$ for some $x \in \mathcal{C}$, then for all SMs Q, the shrunk DBM $(M', P) = \mathsf{shrink}^+((M, Q))$ does not satisfy $P \leq S$.*
- *Otherwise, for all SMs Q, the following holds: $Q \leq S$ if, and only if, the shrunk DBM $(M', P) = \mathsf{shrink}^+((M, Q))$ satisfies $P \leq S$.*

Proof. If $S_{x,0} = 0$ and $\mathsf{M}_{x,0} < \infty$ for some $x \in \mathcal{C}$, since shrink^+ increases each (but one) component of the first column of the SM by one, and since the normalization can only increase components for which $\mathsf{M}_{x,0} < \infty$, there is no $P \leq S$ such that $(M', P) = \mathsf{shrink}^+((M, Q))$.

Assume that $S_{x,0} = \infty$ for all $x \in \mathcal{C}$ with $\mathsf{M}_{x,0} < \infty$. Consider any $Q \not\leq S$. The operator shrink^+ increments all components $(x, 0)$ of the SM by one, and applies normalization. So if $Q_{x,y} > S_{x,y} = 0$ for some $x, y \in \mathcal{C}$, we know that if P denotes the SM such that $(M', P) = \mathsf{shrink}^+((M, Q))$, then $P_{x,y} \geq Q_{x,y} > 0$, since shrink^+ only increases the components of the SM. Thus, for SMs $Q \not\leq S$, there is no corresponding $P \leq S$ with the desired property. Consider now any $Q \leq S$, and P such that $(M', P) = \mathsf{shrink}^+((M, Q))$. Assume that $P_{x,y} \geq 1$ for some $S_{x,y} = 0$. We have $\mathsf{M}_{x,y} < \infty$ since (M', P) is normalized. Since $Q_{x,y} = 0$, there exists a path in $\Pi_{x,y}(\mathsf{G}(M))$ that contains an edge $(z, 0)$ such that $Q_{z,0} = 0$, and $P_{z,0} \geq 1$. But then $S_{z,0} = 0$, which contradicts our assumption. □

Let us comment on Fig. 7, the figure about $\mathsf{Pre}_{\mathsf{time}}$, and how it can be used for our purpose. Assume there is an edge guarded by N (the whole gray area in the right) without resets. In the non-robust setting, this guard can be reached from any point of M (the whole gray area in the left). If we have a shrinking constraint S on M, and we want to synthesize a winning strategy from a shrinking of M satisfying S, then Lemma 3.12 gives the shrinking constraint S' for N, with the following property: given any shrinking (N, Q), we can find $P \leq S$ with $(M, P) = \mathsf{Pre}_{\mathsf{time}}((N, Q))$ (hence, we can delay into (N, Q)), if, and only if Q satisfies $Q \leq S'$. The problem is now "reduced" to finding a winning strategy from $\langle N, S' \rangle$. However, forward-propagating these shrinking constraints

(a) $(M', P) = \mathsf{shrink}^+((M, Q))$ but $P \not\leq S$ (b) $(M', P) = \mathsf{shrink}^+((M, Q))$ with $P \leq S$.

Figure 8: Examples for both cases of Lemma 3.13

is not always that easy. We also need to deal with resets, with the fact that Controller has to choose a delay greater than $\delta > 0$, and also with the case where there are several edges leaving a location. This is the aim of the following developments.

3.5. Neighborhoods

We studied in the previous section the propagation of shrinking constraints along DBM operations. The last step before we can fully treat the propagation of these constraints along a transition is to take into account the perturbations. For instance, in Fig. 5, given Controller's move to the point $x = y = 1$, we need a systematic way of deriving the *neighboring* regions spanned by the perturbations. This will allow us to compute the set of all successor regions reachable under a transition.

We consider constrained regions, which are constrained DBMs in which the DBM represents a region. We define the set of *neighboring regions* of a constrained region $\langle r, S\rangle$ as,

$$\mathcal{N}_{r,S} = \left\{ s \;\middle|\; s \lessdot^* r \text{ or } r \lessdot^+ s, \text{ and } \forall Q \leq S.\ s \cap \mathsf{enlarge}((r, Q)) \neq \emptyset \right\}$$

where $\mathsf{enlarge}((r, Q))$ is the shrunk DBM (M, P) such that $v + [-\delta, \delta] \subseteq M - \delta P$ for every $v \in r - \delta Q$. This is the set of regions that have "distance" at most δ to any shrinking of the constrained region (r, S). We write $\mathsf{neighbor}\langle r, S\rangle = \bigcup_{s \in \mathcal{N}_{r,S}} s$.

Lemma 3.14 (Neighborhood)**.** *Let $\langle r, S\rangle$ be a well constrained region. Then $\mathsf{neighbor}\langle r, S\rangle$ is a zone. If N is the corresponding normalized DBM, there exists a well shrinking constraint S' such that for every SM Q, $Q \leq S'$ iff the SM P defined by $(r', P) = r \cap \mathsf{shrink}((N, Q))$, satisfies $P \leq S$. The pair $\langle N, S'\rangle$ is the* constrained neighborhood *of $\langle r, S\rangle$, and it can be computed in polynomial time. Moreover, the constraint S' is such that $S'_{x,y} = S_{x,y}$ for every $x, y \in \mathcal{C}$, and $S'_{x,0} = S'_{0,x} = \infty$ for every $x \in \mathcal{C}$.*

Constrained neighborhoods are illustrated in Fig. 9.

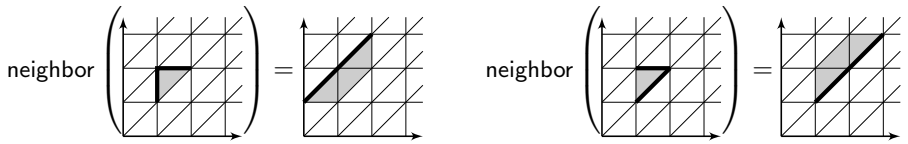


Figure 9: Constrained neighborhood of two constrained regions. Notice that inside any shrinking of the constrained region, there is always a valuation such that a perturbation of $[-\delta, \delta]$ moves the valuation to any region of the neighborhood.

To prove Lemma 3.14 which states the properties of the neighborhoods of constrained regions, we first need to introduce some notations and simple facts about constrained regions.

Let $\langle r, S\rangle$ be a constrained region. There exists a unique partition $X_1, \ldots, X_n$ of the clocks such that for all valuations $\nu \in r$, $\mathsf{frac}(\nu(x)) = \mathsf{frac}(\nu(y))$ for all $x, y \in X_i$, for any $1 \leq i \leq n$, and $\mathsf{frac}(\nu(x)) < \mathsf{frac}(\nu(y))$ if $x \in X_i$ and $y \in X_j$ for any $1 \leq i < j \leq n$. We assume that $\mathsf{frac}(\nu(x)) = 0$ for all $x \in X_1$. So X_1 may be empty, but other X_i's are non-empty. In the sequel, when we consider a region, the above partition is called *partition of clocks according to their fractional values.*

Lemma 3.15. *Let $\langle r, S\rangle$ be a constrained region and consider the partition $X_1, \ldots, X_n$ of clocks according to their fractional values. Then, for all $x, y \in X_i$ for any $1 \leq i \leq n$, we have $S_{x,y} = 0$, and for all $x \in X_1$, we have $S_{x,0} = S_{0,x} = 0$. Moreover,*

- *There exists $2 \leq i_0 \leq n+1$ such that for any $x \in X_i$ with $i_0 \leq i \leq n$, $S_{x,0} = 0$, for any $y \in X_{i'}$ with $i \leq i' \leq n$, $S_{x,y} = 0$, and for any $x \in X_i$ with $1 \leq i < i_0$, $S_{x,0} = \infty$.*
- *There exists $1 \leq j_0 \leq n$ such that for any $x \in X_j$ with $1 \leq j \leq j_0$, $S_{0,x} = 0$, for any $y \in X_{j'}$ with $1 \leq j' < j$, $S_{y,x} = 0$, and for any $x \in X_j$ with $j_0 + 1 \leq j \leq n$, $S_{0,x} = \infty$.*

Proof. This follows from the normalization of S. In fact, suppose $S_{x,0} = 0$ for some $x \in X_i$ with $i \geq 2$, and consider $y \in X_j$ for any $i < j \leq n$. Then $(x, y, 0)$ is a shortest path in $G(r)$. Therefore, we must have $S_{x,y} = S_{y,0} = 0$. One can then choose i_0 as minimal to satisfy this statement, and j_0 maximal. □

Let us make some remarks about the graphs of regions and their shrinking constraints. Consider a region r. For any $x \in X_i$, $y \in X_j$ and $z \in X_k$ with $i < j < k$, (x, y, z) and (z, x, y) are shortest paths in $G(r)$. In fact, $r_{x,z} = r_{x,y} + r_{y,z}$ since $r_{x,y} = r_{x,0} - r_{y,0}$, $r_{y,z} = r_{y,0} - r_{z,0}$, and $r_{x,z} = r_{x,0} - r_{z,0}$. Similarly, $r_{z,x} = r_{z,0} - r_{x,0} + 1$ and $r_{z,y} = r_{z,0} - r_{y,0} + 1$ yield $r_{z,y} = r_{z,x} + r_{x,y}$. A shrinking constraint S with $S_{x,y} = 0$ means that a shrinking of r satisfying S, contains valuations ν with $\mathsf{frac}(\nu(y)) - \mathsf{frac}(\nu(x)) \leq \epsilon$ for any $\epsilon > 0$. If $S_{y,x} = 0$, this means that some valuations ν satisfy $\mathsf{frac}(\nu(y)) - \mathsf{frac}(\nu(x)) \geq 1 - \epsilon$.

Proof of Lemma 3.14. Let us prove that the neighborhood is a zone, and show how to compute it. We characterize the successor regions of r that belong to the neighborhood of $\langle r, S\rangle$. The predecessor regions can be charaterized similarly. Consider the partition of clocks in r ordered according to their fractional parts:

$$0 = \mathsf{frac}(X_1) < \mathsf{frac}(X_2) < \ldots < \mathsf{frac}(X_m) < 1,$$

where X_i's are subsets of clocks having the same fractional part, and X_1 is possibly empty.

- First, assume that $S_{x,0} = \infty$ for $x \in X_m$. Consider the case $X_1 = \emptyset$. Consider any SM $Q \leq S$ such that $Q_{x,0} \geq 2$ for all clocks $x \in X_2 \cup \ldots X_m$. Then for all $\nu \in (r - \delta Q)$, we have $\mathsf{frac}(\nu(x)) \leq 1 - 2\delta$ for all clocks x. We have, for any $t \in [0, \delta]$, $\mathsf{frac}(\nu(x) + t) < 1$. So no region s with $r \lessdot s$ is included in the neighborhood. If $S_{x,0} = \infty$ for $x \in X_m$ but $X_1 \neq \emptyset$, then necessarily $S_{x,0} = S_{0,x} = 0$ for $x \in X_1$ since S is a well shrinking constraint. Then, the neighborhood contains the region s with $r \lessdot s$ but no successor of s, which is shown as above.
- Assume that $S_{x,0} = 0$ for $x \in X_m$. Let $i \in \{1, \ldots, m\}$ be minimum such that X_i is non-empty and $S_{x,0} = 0$ for all $x \in X_j$ and $i \leq j \leq m$. [1] Then, for any $1 \leq j \leq i - 1$

[1] Here, the non-emptiness hypothesis in only significant for $i = 1$.

and $x \in X_j$ $S_{x,0} = \infty$ by Lemma 3.15. Let s be the unique region that satisfies $r \lessdot^+ s$, $s_{x,0} = r_{x,0} + 1$ and $\prec^s_{x,0} = <$ for all $x \in X_i$, and $s_{x,0} = r_{x,0}$ for all $x \in X_{i-1}$. We show that the neighborhood of $\langle r, S\rangle$ contains all regions t such that $r \lessdot^* t \lessdot^* s$, but no region t such that $s \lessdot^+ t$. Let (r, Q, δ_0) be a well-shrinking and consider any $\delta \in [0, \delta_0]$. For any $x \in X_j$ for $\max(2, i) \leq j \leq m$, we have $r_{x,0} - \delta Q_{x,0} = r_{x,0}$, and $-r_{0,x} + \delta Q_{0,x} < r_{x,0}$ (in fact $\prec^r_{x,0} = <$ for these clocks). Let $\epsilon < \min(\delta/2, r_{x,0} - (-r_{0,x} + \delta Q_{0,x}))$ for all $x \in X_j$ and $\max(2, i) \leq j \leq m$, so that $-r_{0,x} + \delta Q_{0,x} < r_{x,0} - \epsilon < r_{x,0}$. By a folklore result [24, Lemma 4], there exists a valuation $\nu \in (r - \delta Q)$ such that $r_{x,0} - \epsilon < \nu(x) < r_{x,0}$ for $x \in X_{\max(2,i)}$. Define $d_j = 1 - \mathsf{frac}(\nu(x))$ for $x \in X_j$ for all $i \leq j \leq m$. We know by the fractional ordering that $0 < d_m < d_{m-1} < \ldots < d_i < \delta/2$. Define also $d'_j = \frac{d_j + d_{j-1}}{2}$ for all $i + 1 \leq j \leq m$, and $d_{m+1} = \frac{d_m}{2}$. If $i = 1$, then we have

$$\mathsf{reg}(\nu) \lessdot \mathsf{reg}(\nu + d_{m+1}) \lessdot \mathsf{reg}(\nu + d_m) \lessdot \mathsf{reg}(\nu + d'_m) \lessdot \mathsf{reg}(\nu + d_{m-1}) \lessdot \ldots \lessdot \mathsf{reg}(\nu + d_i),$$

and for $\epsilon' > 0$ small enough, $\mathsf{reg}(\nu + d_i) \lessdot \mathsf{reg}(\nu + d_i + \epsilon') = s$. If $i > 1$, we remove $\mathsf{reg}(\nu + d_{m+1})$ since this is equal to $\mathsf{reg}(\nu)$. Let us show now that the time-successor of s is not included in the neighborhood. In fact, if $i = 1$, then, the immediate time successor of s is the unique region $r \lessdot^+ t$ such that $-t_{0,x} = t_{x,0} = r_{x,0} + 1$ and $\prec^t_{x,0} = \leq$ for $x \in X_1$. But for all $\nu \in (r - \delta Q)$, we have $\nu(x) = r_{x,0}$, so $\nu(x) + \delta < t_{x,0}$. If $i = 2$ and $X_1 = \emptyset$, then the immediate time successor t of s satisfies $-t_{0,x} = t_{x,0} = r_{x,0} + 1$ for $x \in X_m$. But since for any $\nu \in r$, $\nu(x) + \delta < t_{x,0}$, t is not in the neighborhood. If $i \geq 2$ and $X_{i-1} \neq \emptyset$, then the proof is similar since the immediate time successor of s is the region where clocks in X_{i-1} become integer. The case of the time-predecessors of r is treated similarly.

From the proof above we can directly define the DBM N that represents the neighborhood of r, as follows. N has the same diagonal constraints as r, so $N_{x,y} = r_{x,y}$ for all $x, y \in \mathcal{C}$. For all $x \in X_1$, we let $\mathsf{N}_{x,0} = \mathsf{r}_{x,0} + 1$ and $\prec^N_{x,0} = <$, and if $\mathsf{r}_{0,x} < 0$, we let $\mathsf{N}_{0,x} = \mathsf{r}_{0,x} + 1$ and $\prec^N_{0,x} = <$, otherwise $\mathsf{N}_{0,x} = \mathsf{r}_{0,x} = 0$ and $\prec^N_{0,x} = \leq$. If $S_{x,0} = \infty$ for all $x \in X_2 \cup \ldots \cup X_m$, then $\mathsf{r}_{x,0} = \mathsf{N}_{x,0}$ for all x. Otherwise, let $i \in \{2, \ldots, m\}$ minimum such that $S_{x,0} = 0$ for all $x \in X_j$ for all $i \leq j \leq m$. We let $\mathsf{N}_{x,0} = \mathsf{r}_{x,0} + 1$ and $\prec^N_{x,0} = <$ for all $x \in X_i \cup X_{i+1} \cup \ldots \cup X_m$, and $\mathsf{N}_{x,0} = \mathsf{r}_{x,0}$, $\prec^N_{x,0} = \prec^r_{x,0}$ for all $x \in X_2 \cup \ldots \cup X_{i-1}$. Symmetrically, if $S_{0,x} = \infty$ for all $x \in X_2 \cup \ldots \cup X_m$, we let $\mathsf{N}_{0,x} = \mathsf{r}_{0,x}$ and $\prec^r_{0,x} = \prec^N_{0,x}$. Otherwise, let $i' \in \{2 \ldots, m\}$ be maximum such that $S_{0,x} = 0$ for all $x \in X_j$ for all $2 \leq j \leq i'$. For all $x \in X_j$ and $1 \leq j \leq i'$, if $\mathsf{r}_{0,x} < 0$, we let $\mathsf{N}_{0,x} = \mathsf{r}_{0,x} + 1$ and $\prec^N_{0,x} = <$, and otherwise $\mathsf{N}_{0,x} = 0$ and $\prec^N_{0,x} = \leq$. We let $\mathsf{N}_{0,x} = \mathsf{r}_{0,x}$, $\prec^N_{0,x} = \prec^r_{0,x}$ for all $x \in X_{i'+1} \cup \ldots \cup X_m$.

Let S' be the shrinking constraint given by Lemma 3.12, for which for all SMs Q, $Q \leq S'$ if, and only if there exists $P \leq S$ with $(r', P) = r \cap (N, Q)$ with $\mathsf{r}' = \mathsf{r}$. From the proof of this lemma, S' is the normalization of the following shrinking constraint: we let $S'_{x,y} = S_{x,y}$ for all $x, y \in \mathcal{C}$, since $\mathsf{r}_{x,y} = \mathsf{N}_{x,y}$. For all $x \in \mathcal{C}$, $S'_{x,0} = S_{x,0}$ if $\mathsf{r}_{x,0} = \mathsf{N}_{x,0}$ and $S'_{x,0} = \infty$ otherwise. But by construction of N, $\mathsf{r}_{x,0} = \mathsf{N}_{x,0}$ only if $S'_{x,0} = \infty$ so we get $S'_{x,0} = \infty$ for all clocks. Similarly, we get $S'_{0,x} = \infty$ for all $x \in \mathcal{C}$. We are going to show that S' is already normalized. Consider any $x \in \mathcal{C}$. To get a contradiction, suppose that there exists $z, z' \in \mathcal{C}$ with $S'_{z,z'} = S_{z,z'} = 0$ and a path $x_1 \ldots x_n$ in $\Pi_{z,z'}(\mathsf{G}(N))$ that contains the edge $(x, 0)$ – which would imply that $S'_{x,0}$ becomes 0. Since $\mathsf{r}_{z,z'} = \mathsf{N}_{z,z'}$, we have $\Pi_{z,z'}(\mathsf{G}(N)) \subseteq \Pi_{z,z'}(\mathsf{G}(r))$. Then, $S_{x,0} = 0$ since S is normalized. But this means $\mathsf{N}_{x,0} = \mathsf{r}_{x,0} + 1$ by definition of N, and this path cannot be a shortest path in $\mathsf{G}(N)$ (in fact, $\mathsf{N}_{z,z'} = \mathsf{r}_{z,z'} = \sum_{k=1}^{n-1} r_{x_i, x_{i+1}}$). Consider now $x, y \in \mathcal{C}$ such that $S_{x,y} = \infty$ and let us show that $S'_{x,y} = \infty$ after normalization. Suppose there exists a path $\pi \in \Pi_{z,z'}(\mathsf{G}(N))$ where

$z, z' \neq 0$, $S_{z,z'} = S'_{z,z'} = 0$ and the edge (x, y) belongs to π. If 0 does not appear in π, then this path also belongs to $\Pi_{z,z'}(\mathsf{G}(r))$, since $\mathsf{r}_{z,z'} = \mathsf{N}_{z,z'}$ and all edges have the same weight in both graphs, so we would have $S_{x,y} = 0$. So, assume that some edges $(w, 0)$ and $(0, w')$ belong to π. If $S_{w,0} = S_{0,w} = \infty$, then $\mathsf{r}_{w,0} = \mathsf{N}_{w,0}$ and $\mathsf{r}_{0,w} = \mathsf{N}_{0,w}$ by definition of N, so π is a shortest path in $\mathsf{G}(r)$, which contradicts $S_{x,y} = \infty$. But, if $S_{w,0} = 0$ or $S_{0,w} = 0$ then $\mathsf{N}_{w,0} = \mathsf{r}_{w,0} + 1$ or $\mathsf{N}_{0,w} = \mathsf{r}_{0,w} + 1$, and in this case π is not a shortest path in $\mathsf{G}(N)$ since $\mathsf{N}_{z,z'} = \mathsf{r}_{z,z'}$. Hence, there is no such path π, and we get $S'_{x,y} = \infty$ after normalization.

Now, let Q' be such that $(N', Q') = \mathsf{shrink}((N, Q))$. If Q'' denotes the SM obtained from Q' by incrementing by one each component of the first row and the first column (except cell $(0, 0)$), then $(N', Q') = \mathsf{norm}((N, Q''))$. Moreover, because $S'_{x,0} = S'_{0,x} = \infty$ for all $x \in \mathcal{C}$, we have $Q', Q'' \leq S''$. Then, by definition of S', there exists $P \leq S$ such that $(r', P) = r \cap (N, Q')$. Conversely, if $Q \not\leq S'$, then $Q' \not\leq S'$. So, if there exists P such that $(r', P) = r \cap (N, Q')$, then $P \not\leq S$, by definition of S'.

The fact that S' is a well shrinking constraint for N follows from the fact that S is a well shrinking constraint for r. In fact, for any $Q \leq S'$, there exists $P \leq S$ such that $(r', P) = r \cap \mathsf{shrink}((N, Q))$ and (r', P) is non-empty by hypothesis, so (N, Q) cannot be empty. □

3.6. Controllable Predecessors

Now that we have the notions required to reason with shrinking constraints and neighborhoods, we will show how one can compute *controllable predecessors* given one transition. For instance, in Fig. 5, given winning states at location ℓ' inside regions r_1, r_2, r_3 given as shrunk DBMs satisfying the corresponding shrinking constraints, we will show how to combine these to derive a winning state inside r_0 at location ℓ.

The following lemma characterizes, given a constrained region $\langle r, S\rangle$, the set of constrained regions $\langle s, S_s\rangle$ such that any shrunk region satisying $\langle s, S_s\rangle$ can be reached by delaying from some shrunk region satisfying $\langle r, S\rangle$. These are the sets whose reachability can be ensured by Controller by a delay.

Lemma 3.16. *Let $\langle r, S\rangle$ be a well constrained region, and s be a region such that $r \preccurlyeq^* s$. Then the following properties are equivalent:*

1. *there exists a well shrinking constraint S' (which can be computed in polynomial time) such that for every SM Q, $Q \leq S'$ iff the SM P such that $(r', P) = r \cap \mathsf{shrink}^+(\mathsf{Pre}_{\mathsf{time}}((s, Q)))$ for some $\mathsf{r}' = \mathsf{r}$ satisfies $P \leq S$;*
2. *$\mathsf{neighbor}\langle r, S\rangle \subseteq \mathsf{Pre}_{\mathsf{time}}(s)$;*
3. *define $N = \mathsf{Pre}_{\mathsf{time}}(s)$, and S_N such that for all SM Q, $Q \leq S_N$ iff the SM P defined by $(r', P) = r \cap (N, Q)$ with $\mathsf{r}' = \mathsf{r}$ satisfies $P \leq S$. Then $(S_N)_{x,0} = \infty$ for all $x \in \mathcal{C}$.*

Proof. ▶ **3 ⇒ 1.** Let S' be such that for all SMs Q, $Q \leq S'$ if, and only if there is $Q' \leq S_N$ with $(N', Q') = \mathsf{Pre}_{\mathsf{time}}((s, Q))$ and $\mathsf{N}' = \mathsf{N}$. We will show that S' satisfies the required condition. We first assume that $Q \leq S'$. By Lemma 3.13, there exists $Q'' \leq S_N$ such that $(N'', Q'') = \mathsf{shrink}^+((N', Q')) = \mathsf{shrink}^+(\mathsf{Pre}_{\mathsf{time}}((s, Q)))$ for $\mathsf{N}'' = \mathsf{N}$. And by definition of S_N, there exists $P \leq S$ such that $(r', P) = r \cap (N'', Q'')$ for some $\mathsf{r}' = \mathsf{r}$.

Conversely, if $Q \not\leq S'$, then if Q' denotes the SM such that $(N', Q') = \mathsf{Pre}_{\mathsf{time}}((s, Q))$, then $Q' \not\leq S_N$. By Lemma 3.13, if Q'' denotes the SM such that $(N'', Q'') = \mathsf{shrink}^+((N', Q'))$, then $Q'' \not\leq S_N$. Therefore, there is no SM P such that $(r', P) = r \cap \mathsf{shrink}^+(\mathsf{Pre}_{\mathsf{time}}((s, Q)))$.

▶ **not 2 ⇒ not 1**. Let $\langle V, S'\rangle = \mathsf{neighbor}\langle r, S\rangle$. We assume that $V \not\subseteq N = \mathsf{Pre}_{\mathsf{time}}(s)$. Then, if t denotes the region $s \lessdot t$, we have $s, t \subseteq V$. By definition of the neighborhood, for any SM $P \leq S_r$, when δ is small enough, there exists a valuation $v \in r - \delta P$ such that $v + d' \in t$ for some $0 \leq d' \leq \delta$. But since $s \lessdot t$, $v + d' \in s$ for some $d' \geq 0$ implies that $0 \leq d' < \delta$ (and there exists $0 \leq d' < \delta$ with $v + d' \in r'$). Therefore, for any SM Q, if P denotes the SM such that $(r', P) = r \cap \mathsf{shrink}^+(\mathsf{Pre}_{\mathsf{time}}((s, Q)))$ with $\mathsf{r}' = \mathsf{r}$, then $P \not\leq S$: indeed, if $P \leq S$, then for small enough $\delta > 0$, $v + [0, \delta] \subseteq \mathsf{Pre}_{\mathsf{time}}(s - \delta Q)$ and in particular $v + \delta \in \mathsf{Pre}_{\mathsf{time}}(s - \delta Q)$, which implies that there is $d'' \geq 0$ such that $v + \delta + d'' \in s - \delta Q \subseteq s$, contradicting the above remark.
▶ **2 ⇒ 3**. Let $(V, S') = \mathsf{neighbor}\langle r, S\rangle$. We have $r \subseteq V \subseteq N$. Let (N', Q) be a normalized shrunk DBM with $\mathsf{N} = \mathsf{N}'$, obtained by setting $Q_{x,0} = 1$ for all $x \in \mathcal{C}$ and 0 all other components. Let Q' be the SM such that $(V', Q') = V \cap (N', Q)$, for some $\mathsf{V}' = \mathsf{V}$.

- We show that $Q' \leq S'$. Define $(Q_1)_{x,y} = Q_{x,y}$ if $\mathsf{V}_{x,y} = \mathsf{N}_{x,y}$ and $(Q_1)_{x,y} = 0$ otherwise. Q' is the normalization of Q_1. We have $S'_{x,0} = S'_{0,x} = \infty$ by Lemma 3.14, so $Q'_{x,0} \leq S'_{x,0}$ and $Q'_{0,x} \leq S'_{0,x}$. For any $x, y \in \mathcal{C}$, we assume $S'_{x,y} = 0$. It implies $S_{x,y} = 0$. To get a contradiction, assume that there is a path in $\Pi_{x,y}(\mathsf{G}(V))$ with an edge (z, z') such that $\mathsf{V}_{z,z'} = \mathsf{N}_{z,z'}$ and $Q_{z,z'} \geq 1$. Since $Q_{z,z'} \geq 1$, there must be a path in $\Pi_{z,z'}(\mathsf{G}(N))$ that contains an edge $(\alpha, 0)$. But since $\Pi_{z,z'}(\mathsf{G}(N)) \subseteq \Pi_{z,z'}(\mathsf{G}(V))$, there is a path in $\Pi_{x,y}(\mathsf{G}(V))$ that contains the edge $(\alpha, 0)$. This contradicts $S'_{x,y} = 0$ since we know, by Lemma 3.14 that $S'_{\alpha,0} = \infty$.

- We now show that $Q \leq S_N$, which implies the desired result. By Lemma 3.14, there exists $P \leq S$ such that $(r', P) = r \cap \mathsf{shrink}((V', Q'))$, for some $\mathsf{r}' = \mathsf{r}$. But, if P' denotes the SM such that $(r'', P') = r \cap (V', Q')$, then $P' \leq P \leq S$ because shrink can only increase the components of the SM. Therefore, $(r'', P') = r \cap V \cap (N, Q) = r \cap (N, Q)$. Thus, by Lemma 3.12, we must have $Q \leq S_N$, which implies $(S_N)_{x,0} = \infty$ for all $x \in \mathcal{C}$.

We now assume that the above conditions hold, and prove that S' (of item 1) has the same diagonal constraints as S. By Lemma 3.12, S_N is computed as the normalization of S', which is defined for every $x, y \in \mathcal{C}_0$ as $S'_{x,y} = S_{x,y}$ if $\mathsf{N}_{x,y} = \mathsf{r}_{x,y}$ and ∞ otherwise. Since N and r have the same diagonal constraints, $S_{x,y} = 0$ for $x, y \in \mathcal{C}$ implies $(S_N)_{x,y} = 0$. If $S_{x,y} = \infty$, suppose that $(S_N)_{x,y} = 0$. There exist z, z' such that (x, y) belongs to a path in $\Pi_{z,z'}(\mathsf{G}(N))$, and $S'_{z,z'} = 0$. But this implies that $\mathsf{r}_{z,z'} = \mathsf{N}_{z,z'}$ and $S_{z,z'} = 0$. We have then $\Pi_{z,z'}(\mathsf{G}(N)) \subseteq \Pi_{z,z'}(\mathsf{G}(r))$ since $r \subseteq N$, and this contradicts that $S_{x,y} = \infty$. We now show that S' has the same diagonal components as S_N. In fact, S' is the normalization of S'_N defined by $(S'_N)_{0,x} = \infty$ for all $x \in \mathcal{C}$ and $(S'_N)_{x,y} = (S_N)_{x,y}$ for other $x, y \in \mathcal{C}_0$. Clearly, $(S_N)_{x,y} = 0$ for any $x, y \in \mathcal{C}$ implies $S'_{x,y} = 0$. Assume that $S'_{x,y} = 0$ with $x, y \in \mathcal{C}$. Then (x, y) belongs to a path $\pi \in \Pi_{z,z'}(\mathsf{G}(s))$ with $(S'_N)_{z,z'} = 0$, so necessarily $z \neq 0$ and $\mathsf{s}_{z,z'} = \mathsf{N}_{z,z'}$. If π does not contain the node 0, then $\pi \in \Pi_{z,z'}(\mathsf{G}(N))$ since all other weights are the same in $\mathsf{G}(s)$ and $\mathsf{G}(N)$. If it contains node 0, then π still must be in $\Pi_{z,z'}(\mathsf{G}(N))$. In fact, $(S'_N)_{z,z'} = 0$ means that $N_{z,z'} < \infty$ and since all weights are the same in $\mathsf{G}(s)$ and $\mathsf{G}(N)$ except for the edges $(0, z)$ which can only decrease in s, π is a shortest path in both graphs. In both cases, we get $(S_N)_{x,y} = 0$.

That S' is a well shrinking constraint for s follows from the hypothesis that S is a well shrinking constraint for r. In fact, if for some $Q \leq S'$, (s, Q) is empty, then the corresponding (r, P) (defined in item 1) is empty, which is a contradiction. □

Note that this lemma may not hold for all s with $r \lessdot s$. Consider the constrained region $\langle r, S\rangle$ on the right of Fig. 9, and let s be the first triangle region above r: any valuation arbitrarily close

to the thick segments will be in $r - \delta P$ for any $P \leq S$, but it can only reach s by delaying less than δ time units.

The following lemma describes sets of states from where Controller can ensure reaching a given set, through an action. Intuitively, given a constrained region and its constrained neighborhood, we show how to compute the shrinking constraints on the successor regions (through a given edge), such that Controller can make sure that given shrunk DBMs of the successor regions are reached.

Lemma 3.17. *Let $\langle r, S\rangle$ be a well constrained region, and let $R \subseteq \mathcal{C}$. Let $\mathcal{N} = \{s_1, \ldots, s_m\}$ be the set of neighboring regions of $\langle r, S\rangle$, and define $t_i = s[R \leftarrow 0]$ for each $1 \leq i \leq m$. Then, there exist well shrinking constraints S_{t_i} for all t_i such that for any set of shrunk DBMs $\big((t'_i, Q_{t'_i})\big)_{1\leq i\leq m}$ with $\mathsf{t_i} = \mathsf{t'_i}$, we have that $Q_{t'_i} \leq S_{t_i}$ for all $1 \leq i \leq m$ if, and only if there exists $P \leq S$ such that*

$$(r', P) \subseteq r \cap \mathit{shrink}(\bigcup_{1\leq i\leq m} (s \cap \mathit{Unreset}_R((t'_i, Q_{t'_i})))).$$

for some $\mathsf{r'} = \mathsf{r}$. Moreover, all the $\langle t_i, S_{t_i}\rangle$ can be computed in polynomial time.

Proof. It is useful to remember in this proof that $\mathsf{t_i} = \mathsf{t'_i}$ implies $t_i \subseteq t'_i$ since t_i is a region.

Let $\langle N, S'\rangle$ be the (well) constrained neighborhood of $\langle r, S\rangle$, given by Lemma 3.14. For any region $s \in \mathcal{N}$, let S_s be the well shrinking constraint given by Lemma 3.12, such that for all shrunk DBMs (s', Q) with $\mathsf{s'} = \mathsf{s}$ and $s \subseteq s'$, $Q \leq S_s$ if, and only if there exists $P \leq S'$ with $s \cap (N, P) \subseteq (s', Q)$. Here, S_s is in fact a well shrinking constraint since $s \cap (N, P) \neq \emptyset$ for all $P \leq S'$, by the construction of the neighborhood. Then, let us write $t = s[R \leftarrow 0]$, and let S'_s be the well shrinking constraint given by Lemma 3.12, such that for any shrunk DBM (t', Q_t) with $\mathsf{t} = \mathsf{t'}$ and $t \subseteq t'$, $Q_t \leq S'_s$ if, and only if there exists $P_s \leq S_s$ with $(s', P_s) = s \cap \mathsf{Unreset}_R((t', Q_t))$ for some $\mathsf{s'} = \mathsf{s}$ with $s \subseteq s'$. We let $S_t = \min_{s\in\mathcal{N}:t=s[R\leftarrow 0]} S'_s$, where the minimum is taken componentwise. Then S_t is still a well shrinking constraint. Now, S_t satisfies the following property: for any shrunk DBM (t', Q_t) with $\mathsf{t'} = \mathsf{t}$ and $t \subseteq t'$, $Q_t \leq S_t$ if, and only if for all regions $s \in \mathcal{N}$ with $t = s[R \leftarrow 0]$, there exists $P_s \leq S_s$ such that $(s', P_s) = s \cap \mathsf{Unreset}_R((t', Q_t))$ for some s' with $\mathsf{s'} = \mathsf{s}$ and $s \subseteq s'$. Combining this and S' defined above, we get that S_t has the property that for any shrunk DBMs (t', Q_t) with $\mathsf{t'} = \mathsf{t}$ and $t \subseteq t'$, $Q_t \leq S_t$ if, and only if for all $s \in \mathcal{N}$ with $t = s[R \leftarrow 0]$, there exists $P_s \leq S'$ such that $s \cap (N, P_s) \subseteq s \cap \mathsf{Unreset}_R((t', Q_t))$.

For any family of shrunk DBMs $\{(t'_i, Q_{t'_i})\}_{1\leq i\leq m}$ with $t_i \subseteq t'_i$, $\mathsf{t_i} = \mathsf{t'_i}$ and $Q_{t'_i} \leq S_{t_i}$, consider the set $\{P_s\}_{s\in\mathcal{N}}$ as defined above. Let $P' = \max_{s\in\mathcal{N}}(P_s)$, where the max is componentwise. We have $P' \leq S'$ since $P_s \leq S'$ for all $s \in \mathcal{N}$. We have $s \cap (N, P') \subseteq s \cap \mathsf{Unreset}_R((t'_i, Q_{t'_i}))$ for all $s \in \mathcal{N}$, since $P_s \leq P'$. So,

$$(N, P') = \bigcup_{s\in\mathcal{N}} s \cap (N, P') \subseteq \bigcup_{s\in\mathcal{N}} s \cap \mathsf{Unreset}_R((t'_i, Q_{t'_i})).$$

By Lemma 3.14, there exists $P \leq S$ such that

$$(r', P) = r \cap \mathsf{shrink}((N, P')) \subseteq \bigcup_{s\in\mathcal{N}} s \cap \mathsf{Unreset}_R((t'_i, Q_{t'_i})).$$

This proves the first direction of the lemma.

Consider any SMs $\{Q_{t_i}\}_{1\leq i\leq m}$ such that $(Q_{t_i})_{x,y} \geq 1$ and $(S_t)_{x,y} = 0$ for some i and $x, y \in \mathcal{C}_0$. Then there exists $r_0 \in \mathcal{N}$ with $r_0[R \leftarrow 0] = t$, such that if Q_{r_0} denotes the SM that satisfies

$(r'_0, Q_{r_0}) = r_0 \cap \mathsf{Unreset}_R((t'_i, Q_{t_i}))$ with $\mathsf{r'_0} = \mathsf{r_0}$ and $r_0 \subseteq r'_0$, we have $Q_{r_0} \not\leq S_{r_0}$. So, there is no SM P_{r_0} such that $r_0 \cap (N, P_{r_0}) \subseteq (r'_0, Q_{r_0})$ and $P_{r_0} \leq S'$. It follows that there is no $P' \leq S'$ satisfying $(N, P') \subseteq \bigcup_{s \in \mathcal{N}}(s', Q_s)$ where $(s', Q_s) = s \cap \mathsf{Unreset}_R((t, Q_t))$. In fact, if this would imply that $(N, P') \cap r_0 \subseteq (r'_0, Q_{r_0})$, since the regions in $\mathcal{N}$ are pairwise disjoint. Therefore, if (N, P') satisfies the above inclusion, then $P' \not\leq S'$, so there is no $P \leq S$ such that (r', P) satisfies the statement of the lemma for some $\mathsf{r'} = \mathsf{r}$. □

This lemma gives for instance the shrinking constraints that should be satisfied in r_1, r_2 and r_3, in Fig. 5, once shrinking constraint in r'_0 is known. In this case, the constraint in r'_0 is 0 everywhere since it is a punctual region. The neighborhood $\mathcal{N}$ of r'_0 is composed of r'_0 and two extra regions (defined by $(0 < x < 1) \wedge (x = y)$ and $(1 < x < 2) \wedge (x = y)$). If there are shrinkings of regions r_1, r_2, r_3 satisfying the corresponding shrinking constraints (given in the lemma), and from which Controller wins, then one can derive a shrinking of r'_0, satisfying its constraint, and from which Controller wins. In the next section, we define the game $\mathcal{RG}(\mathcal{A})$ following this idea, and explain how it captures the game semantics for robustness.

4. A finite game abstraction

Let $\mathcal{A} = (\mathcal{L}, \mathcal{C}, \ell_0, E)$ be a timed automaton. We define a finite turn-based game $\mathcal{RG}(\mathcal{A})$ on a graph whose nodes are of two sorts: *square nodes* labelled by (ℓ, r, S_r), where ℓ is a location, r a region, S_r is a well shrinking constraint for r; *diamond nodes* labelled similarly by (ℓ, r, S_r, e) where moreover e is an edge leaving ℓ. Square nodes belong to Controller, while diamond nodes belong to Perturbator. Transitions are defined as follows:

(a) From each square node (ℓ, r, S_r), for any edge $e = (\ell, g, R, \ell')$ of $\mathcal{A}$, there is a transition to the diamond node (ℓ, s, S_s, e) if the following conditions hold:

- (i) $r \lessdot^* s$ and $s \subseteq g$;
- (ii) S_s is such that for all SMs Q, $Q \leq S_s$ iff there exists $P \leq S_r$ with

$$(r', P) = r \cap \mathsf{shrink}^+(\mathsf{Pre}_{\mathsf{time}}((s, Q))),$$

for some $\mathsf{r'} = \mathsf{r}$.

(b) From each diamond node (ℓ, r, S_r, e), where $e = (\ell, g, R, \ell')$ is an edge of $\mathcal{A}$, writing $\mathcal{N}$ for the set of regions in the neighborhood of (r, S_r) and $\mathcal{N}' = \{s[R \leftarrow 0] \mid s \in \mathcal{N}\}$, there are transitions to all square nodes (ℓ', t, S_t) where $t \in \mathcal{N}'$, and $(S_t)_{t \in \mathcal{N}'}$ are such that for all shrunk DBMs $((t', Q_t))_{t \in \mathcal{N}'}$ with $\mathsf{t'} = \mathsf{t}$, it holds $Q_t \leq S_t$ for every $t \in \mathcal{N}'$ iff there exists $P \leq S_r$ such that

$$(r', P) \subseteq r \cap \mathsf{shrink}(\bigcup_{s \in \mathcal{N}}(s \cap \mathsf{Unreset}_R((t', Q_t))), \qquad \text{where } t = s[R \leftarrow 0],$$

for some $\mathsf{r'} = \mathsf{r}$.

Intuitively, the transitions from the square nodes are the decisions of Controller. In fact, it has to select a delay and a transition whose guard is satisfied. Then Perturbator can choose any region in the neighborhood of the current region, and, after reset, this determines the next state.

Note that $\mathcal{RG}(\mathcal{A})$ can be computed, thanks to Lemmas 3.16 and 3.17, and has exponential-size. Observe also that $\mathcal{RG}(\mathcal{A})$ is constructed in a forward manner: we start by the initial constrained

region (*i.e.* the region of valuation $\vec{0}$ with the zero matrix as shrinking constraint), and compute its successors in $\mathcal{RG}(\mathcal{A})$. Then, if Controller has a winning strategy in $\mathcal{RG}(\mathcal{A})$, we construct a winning strategy for $\mathcal{G}_\delta(\mathcal{A})$ by a backward traversal of $\mathcal{RG}(\mathcal{A})$, using Lemmas 3.16 and 3.17. Thus, we construct $\mathcal{RG}(\mathcal{A})$ by propagating shrinking constraints forward, but later do a backward traversal in it. The correctness of the construction is stated as follows.

Proposition 4.1. *For any timed automaton $\mathcal{A}$, Controller has a winning strategy in $\mathcal{RG}(\mathcal{A})$ if, and only if there exists $\delta_0 > 0$ such that Controller wins $\mathcal{G}_\delta(\mathcal{A})$ for all $\delta \in [0, \delta_0]$.*

Note that as we compute a winning strategy for Controller (if any) by Proposition 4.1, we can also compute a corresponding δ_0. The upper bound of Theorem 2.3 is a consequence of the above proposition, since $\mathcal{RG}(\mathcal{A})$ has exponential size and finite reachability games can be solved in time polynomial in the size of the game.

Changing the perturbation parameters.. Consider the semantics where Controller's delays are bounded below by $k\delta$, and the perturbations belong to $[l\delta, m\delta]$ with $l\delta + k\delta \geq 0$, for some rational number $\delta > 0$ and integers $k \geq 0, l, m$ (Observe that any choice of these rational parameters can be written in this manner). The abstract game construction can then be adapted to this case. In fact, it suffices to replace the operator shrink^+ by $\mathsf{shrink}_{[0,k\delta]}$, and the operator shrink by $\mathsf{shrink}_{[l\delta,m\delta]}$ in the construction.

4.1. Proof of Proposition 4.1

4.1.1. Controller wins $\mathcal{RG}(\mathcal{A})$ $\Rightarrow$ Controller wins $\mathcal{G}_\delta(\mathcal{A})$ for all small enough $\delta > 0$.

Assume we are given a reachability objective defined by $\ell_{\smile} \in \mathcal{L}$, and let σ be a memoryless winning strategy for Controller in $\mathcal{RG}(\mathcal{A})$ for reaching $\ell_{\smile}$. Consider the execution tree $\mathcal{T}_\sigma$ of $\mathcal{RG}(\mathcal{A})$, where Controller plays with σ. $\mathcal{T}_\sigma$ is finite by Koenig's Lemma since all branches are winning, thus finite, and all branches end in the target state.

To any square node n of $\mathcal{T}_\sigma$ labelled by (ℓ, r, S_r), we will assign a shrunk DBM $\left(r', P_n\right)$ and $\delta_n > 0$ with $P_n \leq S_r$ and $\mathsf{r} = \mathsf{r}'$, such that Controller wins the game $\mathcal{G}_\delta(\mathcal{A})$ from any state of $\{\ell\} \times (r' - \delta P_n)$ for all $\delta \in [0, \delta_n]$. Remember that $\mathsf{r}' = \mathsf{r}$ implies $r \subseteq r'$ since r is a region. We will define these by a bottom-up traversal of $\mathcal{T}_\sigma$. We start by assigning $P_n = \mathbf{0}$ and $\delta_n = \infty$ to all nodes with $\mathsf{loc}(n) = \ell_{\smile}$ (which are leaves of $\mathcal{T}_\sigma$). These are trivially winning for Controller.

Consider now a square node n of $\mathcal{T}_\sigma$ labelled by (ℓ, r, S_r) whose all square successors have been treated. Then, n has only one successor which is given by σ, we assume it is the diamond node n' labelled by (ℓ, s, S_s, e). We write $e = (\ell, g, R, \ell')$. Let $\mathcal{N}$ be the set of neighboring regions of $\langle s, S_s \rangle$ given by Lemma 3.14. Let $s_1, \ldots, s_m$ be the regions composing $\mathcal{N}$, and let $n'_1, \ldots, n'_m$ denote the successors of n' in $\mathcal{T}_\sigma$, where n'_i is labelled by (ℓ', t_i, S_{t_i}), and such that $t_i = s_i[R \leftarrow 0]$. By construction, shrinking constraints S_{t_i} are given by Lemma 3.17 applied to $\langle s, S_s \rangle$. By induction, for each $1 \leq i \leq m$ there is a shrunk DBM $(t'_i, P_{n'_i})$ and $\delta_{n'_i} > 0$ with $\mathsf{t}'_\mathsf{i} = \mathsf{t}_\mathsf{i}$ and $P_{n'_i} \leq S_{t_i}$, such that Controller wins the game $\mathcal{G}_\delta(\mathcal{A})$ from $\{\ell'\} \times (t'_i - \delta P_{n'_i})$ for all $\delta \leq \delta_{n'_i}$. Now, by Lemma 3.17, there exists a shrunk DBM (s', Q) with $\mathsf{s}' = \mathsf{s}$ and $Q \leq S_s$ such that

$$s' - \delta Q \subseteq s \cap \mathsf{shrink}\Big(\bigcup_{1 \leq i \leq m} s_i \cap \mathsf{Unreset}_R(t'_i - \delta P_{n'_i}) \Big),$$

for all $0 \leq \delta \leq \delta_{n'}$, where $\delta_{n'} \leq \min_i(\delta_{t_i})$ is computed by Lemma 3.8. Then, by construction of the game, there exists $P_n \leq S_r$ and $0 < \delta_n \leq \delta_{n'}$ such that $r' - \delta P_n = r \cap \mathsf{shrink}^+(\mathsf{Pre}_{\mathsf{time}}(s' - \delta Q))$ for

all $0 \leq \delta \leq \delta_n$, for some $\mathsf{r}' = \mathsf{r}$ (Remember that Lemma 3.16 applies here on s' since $\mathsf{s} = \mathsf{s}'$ and $s \subseteq s'$). Here, δ_n can be computed using Lemma 3.6. Controller wins from these states: in fact, it can delay into $s' - \delta Q$, where, after any Perturbator's move, and the clock resets, the next state belongs to one of the states $t'_i - \delta P_{n'_i}$, which are all winning by induction.

Notice that at each step of the computation, we get non-empty shrunk DBMs satisfying the corresponding shrinking constraints. By construction of $\mathcal{RG}(\mathcal{A})$, we have only well shrinking constraints in all nodes, so the computed shrunk DBMs (r', P_n) are always non-empty. The procedure ends in the initial state $(\ell_0, \mathbf{0}, \mathbf{0})$, so Controller wins $\mathcal{G}_\delta(\mathcal{A})$ by projecting the play to $\mathcal{RG}(\mathcal{A})$ and always staying inside set $r' - \delta P_n$ at any node n, which is possible by construction.

4.1.2. Upper bound on δ_0.

We now assume that Controller has a winning strategy and give an upper bound on δ_0 computed by the algorithm. Consider the set $\mathcal{M}$ of all shrunk DBMs that appear when we construct the winning strategy in the proof above, including the shrunk DBMs corresponding to intermediary results. For instance, in the computation above, given the edge $(\ell_i, r_i, S_{r_i}) \to (\ell_i, r'_i, S_{r'_i}, e)$ and $P_{r'_i}$, we compute P_{r_i} such that

$$(r_i, P_{r_i}) = r \cap \mathsf{shrink}^+(\mathsf{Pre}_{\mathsf{time}}\left((r'_i, P_{r'_i})\right)).$$

Then, $\mathcal{M}$ contains the shrunk DBMs (r_i, P_{r_i}) and $(r'_i, P_{r'_i})$, but also (M_1, Q_1) such that $(M_1, Q_1) = \mathsf{Pre}_{\mathsf{time}}((r_i, P_{r_i}))$, and (M_2, Q_2) such that $(M_2, Q_2) = \mathsf{shrink}^+((M_1, Q_1))$. Now, δ_0 is chosen by the algorithm small enough so that all shrunk DBMs of $\mathcal{M}$ are non-empty and normalized, and all such equations that appear in the construction of a winning strategy above hold, for all $\delta \in [0, \delta_0)$. Note that since $\mathcal{RG}(\mathcal{A})$ has exponential size, $\mathcal{M}$ may contain exponentially many shrunk DBMs. We show that δ_0 can be chosen, roughly, as the inverse of the maximal component of all shrinking matrices that appear in all computations.

Proposition 4.2. *Let $m = \max_{(M,P)\in\mathcal{M}} \max_{i,j\in\mathcal{C}_0} P_{i,j}$. Then, one can choose $\delta_0 = \frac{1}{3m}$.*

Proof. To prove this, we need to show for all $\delta \in [0, \delta_0)$ that all shrunk DBMs of $\mathcal{M}$ are normalized, non-empty and satisfy the equations they are involved in. Let us first note that being normalized implies non-emptiness for the shrunk DBMs of $\mathcal{M}$ since all shrinking constraints are well. In fact, a normalized DBM is non-empty if, and only if all its diagonals are 0. Here, for any $(M, P) \in \mathcal{M}$, the diagonal components of M and P must be 0 since M is non-empty and normalized, and $M - \delta P$ is non-empty and normalized for small enough $\delta > 0$, by hypothesis. Hence, if $M - \delta P$ is normalized, it must be non-empty too. Let $(M, P) \in \mathcal{M}$. Normalization condition requires

$$\forall i, j, k \in \mathcal{C}_0^3, \quad M_{i,j} - \delta P_{i,j} \leq M_{i,k} - \delta P_{i,k} + M_{k,j} - \delta P_{k,j},$$

for all $\delta \in [0, \delta_0)$. If $M_{i,j} = M_{i,k} + M_{k,j}$, then we must have $P_{i,j} \geq P_{i,k} + P_{k,j}$ since the above condition holds for small enough $\delta > 0$. But then, for these components, the condition holds for all $\delta > 0$. If $M_{i,j} < M_{i,k} + M_{k,j}$, then the condition holds if $\delta_0 \leq |\frac{M_{i,k}+M_{k,j}-M_{i,j}}{P_{i,k}+P_{k,j}-P_{i,j}}|$, but this is already the case since $\delta_0 < \frac{1}{3m}$. It remains to show that all equations that appear in the computations hold. Equations of the form $(M, P) = \mathsf{Pre}_{\mathsf{time}}((N, Q))$ and $(M, P) = \mathsf{Unreset}_R((N, Q))$ already hold for $\delta \in [0, \delta_0)$ since all involved shrunk DBMs are normalized (see Lemma 3.6 and [36] for details). For an equation of the form $(M, P) = (N_1, Q_1) \cap (N_2, Q_2)$, we have either $(N_1)_{i,j} = (N_2)_{i,j}$ and $P_{i,j} = \max((Q_1)_{i,j}, (Q_2)_{i,j})$, or for example $(N_1)_{i,j} < (N_2)_{i,j}$ and $(M_{i,j}, P_{i,j}) = ((N_1)_{i,j}, (Q_1)_{i,j})$. In the former case, the equation holds for all $\delta > 0$. In the latter case, it holds for all $\delta < |\frac{(N_2)_{i,j}-(N_1)_{i,j}}{(P_2)_{i,j}-(P_1)_{i,j}}|$. This is already the case since $\delta < \frac{1}{3m}$. □

How much can m grow? A quick analysis shows that it is at most doubly exponential in the size of the input. In fact, the size of $\mathcal{RG}(\mathcal{A})$ can be exponential in the size of the automaton, and the number of shrunk DBMs involved in our computations are linear in the size of $\mathcal{RG}(\mathcal{A})$. In fact, we apply, for each edge, a constant number of operations on shrunk DBMs. Whenever we apply such an operation, and say, obtain (M, P), we apply normalization on (M, P), which consists in assigning to each $P_{i,j}$, the P-weight of the longest path in $\mathsf{G}(M)$ from i to j. So at each operation, the maximal constant of shrinking matrices is multiplied at most by $|\mathcal{C}_0|$. Considering the size of $\mathcal{M}$, we get that $m = O(|\mathcal{C}_0|^{2^{|\mathcal{A}|}})$. Note however that this is an extremely pessimistic estimation; it is unlikely that parameters will grow that much at each step, and never be reset along a path. This is confirmed by the experimental results of [35], where shrunk DBMs are used to solve a different problem.

4.1.3. Controller loses $\mathcal{RG}(\mathcal{A})$ $\Rightarrow$ Controller loses $\mathcal{G}_\delta(\mathcal{A})$ for any $\delta > 0$.

To prove the theorem, we will assume the Controller loses in $\mathcal{RG}(\mathcal{A})$, and we will construct a winning strategy for the δ-Perturbator in $\mathcal{G}_\delta(\mathcal{A})$ by looking at the projection of the plays in $\mathcal{RG}(\mathcal{A})$, and by imitating the moves of a winning strategy for the Perturbator in $\mathcal{RG}(\mathcal{A})$. The core idea is to show that the Perturbator can always force the game to be close, by some chosen ϵ, to all boundaries of the current region for which the shrinking constraint is 0. This will ensure that from any state of the play, for any move of Controller, Perturbator can force the game to some state inside any successor in $\mathcal{RG}(\mathcal{A})$.

Let us first formalize what we mean by being close to boundaries.

Definition 4.3. *Let $\langle M, S\rangle$ any constrained DBM. For any $\epsilon \geq 0$, we say that a valuation ν is* ϵ-tight in M w.r.t. S *if $\nu \in M$, and,*

$$\forall x, y \in \mathcal{C}_0, \quad S_{x,y} = 0 \quad \Rightarrow \quad M_{x,y} - \epsilon \leq \nu(x) - \nu(y). \tag{4}$$

We say that M admits tight valuations w.r.t. S *if it admits ϵ-tight valuations for any $\epsilon > 0$.*

We say that $\langle M, S\rangle$ admits tight valuations if for any $P \leq S$, and for all small enough $\delta > 0$, $M - \delta P$ admits tight valuations w.r.t. S.

In the proofs, we will also say that ν is *ϵ-tight in M for a component* $(x, y) \in \mathcal{C}_0^2$ when $M_{x,y} - \epsilon \leq \nu(x) - \nu(y)$.

We show that for any move of Controller in $\mathcal{G}_\delta(\mathcal{A})$ from an ϵ-tight valuation, the corresponding edge exists in $\mathcal{RG}(\mathcal{A})$:

Lemma 4.4. *Fix $\delta > 0$ and $\epsilon \leq \frac{\delta}{2}$. Assume that ν is ϵ-tight in r w.r.t. S, and let $\nu' = (\nu + d)$ for some $d \geq \delta$, and some edge $e = (\ell, g, R, \ell')$ with $\nu' \models g$. Then, $\mathcal{RG}(\mathcal{A})$ has an edge from (ℓ, r, S_r) to $(\ell, r', S_{r'}, e)$ for $r' = \mathsf{reg}(\nu')$ and for some $S_{r'}$.*

Proof. Let $N = \mathsf{Pre}_{\mathsf{time}}(r')$, and consider, by Lemma 3.12, the shrinking constraint S_N such that for any SM Q, $Q \leq S_N$ if, and only if the SM P such that $(r, P) = r \cap (N, Q)$ satisfies $P \leq S_r$. Thanks to Lemma 3.16, it is sufficient to show that $(S_N)_{x,0} = \infty$ for all $x \in \mathcal{C}$. To get a contradiction, assume that $(S_N)_{x,0} = 0$ for some $x \in \mathcal{C}$. By (the proof of) Lemma 3.12, S_N is the normalization of S'_r defined by $(S'_r)_{x,y} = (S_r)_{x,y}$ if $r_{x,y} = N_{x,y}$ and $(S'_r)_{x,y} = \infty$ otherwise. So, $(S_N)_{x,0} = 0$ means that there exists (z, z') such that $(x, 0)$ is on some path π of $\Pi_{z,z'}(\mathsf{G}(N))$, $r_{z,z'} = N_{z,z'}$ and $(S_r)_{z,z'} = 0$. But since $r \subseteq N$ and $r_{z,z'} = N_{z,z'}$ we have $\Pi_{z,z'}(\mathsf{G}(N)) \subseteq \Pi_{z,z'}(\mathsf{G}(r))$. Moreover, all weights along π are the same in r and N. We get $(S_r)_{x,0} = 0$. On the other hand, since $r' \subseteq N$ and $r \preccurlyeq^* r'$, we have $r_{x,0} \leq r'_{x,0} \leq N_{x,0}$. So $r_{x,0} = r'_{x,0}$. By hypothesis, we have $r_{x,0} - \epsilon \leq \nu(x) \leq r_{x,0}$. But $\epsilon \leq \delta/2$ and $d \geq \delta$, so $\nu(x) + d > r_{x,0} = r'_{x,0}$, a contradiction. □

The following lemma shows that ϵ-tightness is preserved by resets.

Lemma 4.5. *Let r, r' be regions such that $r' = r[R \leftarrow 0]$ for some $R \subseteq \mathcal{C}$. Consider shrinking constraints S_r and $S_{r'}$ given by Lemma 3.12, such that for any SM Q, $Q \leq S_{r'}$ iff there the SM P such that $(r, P) = r \cap \mathsf{Unreset}_R((r', Q))$ satisfies $P \leq S_r$. Let $\delta > 0$ small enough so that these equations hold for given P and Q. For any $\epsilon > 0$, and any ν that is ϵ-tight in $r - \delta P$ w.r.t. S_r, we have that $\nu' = \nu[R \leftarrow 0]$ is ϵ-tight in $r' - \delta Q$ w.r.t. $S_{r'}$.*

Proof. Let $N = \mathsf{Unreset}_R(r')$ and S_N be the shrinking constraints such that for any SM Q, $Q \leq S_N$ if and only if the SM P such that $r - \delta P = r \cap (N - \delta Q)$ satisfies $P \leq S_r$. Then $S_{r'}$ is such that for any SM Q, $Q \leq S_{r'}$ if and only if there the SM P such that $N - \delta P = \mathsf{Unreset}_R(r' - \delta Q)$ satisfies $P \leq S_N$. By definition of the reset operation, we have $r'_{x,y} = r_{x,y}$ for all $x, y \notin R$, and $r'_{x,y} = 0$ for all $x, y \in R$ (we assume $0 \in R$). For any $x \in R$ and $y \notin R$, $r'_{x,y} = r_{0,y}$ and $r'_{y,x} = r_{y,0}$.

It suffices to assume that ν is ϵ-tight in $\langle r, S_r \rangle$ and show that $\nu' = \nu[R \leftarrow 0]$ is ϵ-tight in $\langle r', S_{r'} \rangle$. For all $x, y \notin R$ or $x, y \in R$, it is clear that ν' satisfies the ϵ-tightness for these components. Consider $x \in R$ and $y \notin R$ such that $(S_{r'})_{x,y} = 0$. Then $(S_{r'})_{0,y} = 0$, and it suffices to show that $-\nu'(y) \geq {r'}_{0,y} - \epsilon$ since $\nu'(x) = 0$ and $r_{0,y} = r'_{0,y}$. We are going to show that $(S_r)_{0,y} = 0$. If S'_N denotes the shrinking constraint defined by $(S'_N)_{x,0} = (S'_N)_{0,x} = 0$ for all $x \in R$, $(S'_N)_{x,y} = \infty$ whenever $x \in R$ and $y \in \mathcal{C}$ or inversely, and $(S'_N)_{x,y} = (S_N)_{x,y}$ for $x, y \in \mathcal{C}_0 \setminus R$. Then $S_{r'}$ is the normalization of S'_N. Then, $(S_{r'})_{0,y} = 0$, for $y \notin R$, means that there is $z, z' \in \mathcal{C}_0$ such that for some path $\pi \in \Pi_{z,z'}(\mathsf{G}(r'))$, $(0, y)$ belongs to π and $(S'_N)_{z,z'} = 0$. Then either $z, z' \notin R$ and $(S_N)_{z,z'} = 0$, or $z = 0$ and $z' \in R$.

- Consider the first case. We have $r'_{z,z'} = N_{z,z'}$ (by definition of unreset). We show that there is a path $\pi' \in \Pi_{z,z'}(\mathsf{G}(N))$ that contains $(0, y)$ and whose all nodes are outside R. In fact, $r'_{z,0} = N_{z,0}$ and $r'_{y,z'} = N_{y,z'}$ since $z, z', y \notin R$, and these have finite weights (since π is a shortest path). But N is obtained from r' by setting to ∞ edges with an endpoint in R, and applying normalization. So there must be shortest paths from z to 0, and from y to z' in $\mathsf{G}(N)$. Now, since $(S_N)_{z,z'} = 0$, (z, z') belongs to some path $\pi'' \in \Pi_{\alpha,\beta}(\mathsf{G}(N))$ where $r_{\alpha\beta} = N_{\alpha,\beta}$ and $(S_r)_{\alpha,\beta} = 0$. Moreover, from $r \subseteq N$, $r_{\alpha,\beta} = N_{\alpha,\beta}$ and $r_{z,z'} = N_{z,z'}$, it follows that $\pi', \pi'' \in \Pi_{z,z'}(\mathsf{G}(N)) \subseteq \Pi_{z,z'}(\mathsf{G}(r))$. Then, replacing the edge (z, z') in π'' by the path π', we still get a shortest path in $\mathsf{G}(r)$, that contains $(0, y)$. Therefore, we must have $(S_r)_{0,y} = 0$.

- Assume now that $z = 0$ and $z' \in R$. Assume that π does not contain nodes in R other than z' (if it does, we can shorten π and change z'). Let us write $\pi = z_1 z_2 \dots z_m$ where $z_1 = z$ and $z_m = z'$. Since $r'_{0,z_m} = r'_{z_m,0} = 0$, $\pi' = z_1 \dots z_{m-1}$ is a cycle with weight 0. But since all nodes in π' are outside R, this is also a path in $G(r)$. Therefore $(S_r)_{z_i,z_{i+1}} = 0$ along all edges, and in particular $(S_r)_{0,y} = 0$.

By hypothesis, we have $-\nu(y) \geq r_{0,y} - \epsilon$, and since $r'_{0,y} = r_{0,y}$, we get $-\nu'(y) \geq r'_{0,y} - \epsilon$. The proof of the symmetric case $x \notin R$ and $y \in R$ is similar. □

The next lemma is the main lemma of the proof of the second direction of the theorem. It shows that, starting from an ϵ-tight valuation of $\langle r, S \rangle$, and given a Controller's move, Perturbator can, not only choose any successor available in $\mathcal{RG}(\mathcal{A})$ but also make sure that the resulting valuation is $(\epsilon + \epsilon')$-tight, for arbitrary $\epsilon' > 0$.

Lemma 4.6. *Consider a valuation ν that is ϵ-tight in r w.r.t. S. Let $\nu' = (\nu + d)$ for some $d \geq \delta$, let $e = (\ell, g, R, \ell')$ an edge of $\mathcal{A}$ with $\nu' \models g$. Consider the edge of $\mathcal{RG}(\mathcal{A})$ from (ℓ, r, S_r)*

to $(\ell, r', S_{r'}, e)$ *for* $r' = \mathsf{reg}(\nu')$ *and for some* $S_{r'}$*, from Lemma 4.4. Then, for any region* s *in the* $\langle N, S_N\rangle = \mathsf{neighbor}\langle r', S_{r'}\rangle$*, and any* $\epsilon' > 0$*, there exists* $d' \in [d-\delta, d+\delta]$ *such that* $\nu + d'$ *is* $(\epsilon+\epsilon')$*-tight in* s *w.r.t.* S_s *where* S_s *is such that for all SMs* Q*,* $Q \leq S_s$ *if, and only if there is* $P \leq S_N$ *with* $(N, P) \cap s \subseteq (s, Q)$*.*

Before proving Lemma 4.6, let us first prove the second direction of Proposition 4.1.

Proof of Proposition 4.1 (Second direction). We fix $\delta > 0$, and consider an arbitrary strategy σ for Controller in $\mathcal{G}_\delta(\mathcal{A})$. We are going to define an infinite play π, where Controller follows strategy σ, and the target location is not reached. This proves that Perturbator wins $\mathcal{G}_\delta(\mathcal{A})$ against any strategy of Controller since σ is chosen arbitrarily.

By hypothesis, Perturbator has a winning strategy γ in $\mathcal{RG}(\mathcal{A})$. We fix $\epsilon \in [0, \delta/2]$, and we define the sequence $\epsilon_i = \sum_{1\leq j\leq i} \frac{\epsilon}{2^j}$ for $i \geq 1$, which is positive and bounded above by ϵ. We construct in parallel a play $\big((\ell_i, r_i, S_{r_i}), (\ell_i, r'_i, S_{r'_i}, e_i)\big)_{i\geq 1}$ of $\mathcal{RG}(\mathcal{A})$ where Perturbator plays with strategy γ. The play $\pi = (\ell_i, \nu_i)_{i\geq 1}$ will satisfy the following invariant:

$$\text{For each state } (\ell_i, \nu_i), \text{ we have } \nu_i \in r_i \text{ and } \nu_i \text{ is } \epsilon_i\text{-tight in } r_i \text{ w.r.t. } S_{r_i}.$$

Initially, we have (ℓ_1, ν_1) with $\nu_1 = \mathbf{0}$, and the initial state $(\ell_1, \mathbf{0}, \mathbf{0})$ of $\mathcal{RG}(\mathcal{A})$ satisfies the invariant. For $i \geq 2$, let us assume that state (ℓ_i, ν_i) of the play satisfies the invariant for the state (ℓ_i, r_i, S_{r_i}) of $\mathcal{RG}(\mathcal{A})$. Let $d \geq \delta$ be the delay and $e_i = (\ell_i, g_i, R_i, \ell_{i+1})$ the edge prescribed by σ from state (ℓ_i, ν_i) given the current history. Let $\nu'_i = \nu_i + d$ and r'_i be the region of ν'_i. Lemma 4.4 shows that $\mathcal{RG}(\mathcal{A})$ has the corresponding edge. Let $(\ell_{i+1}, r_{i+1}, S_{r_{i+1}})$ be the successor of $(\ell_i, r'_i, S_{r'_i}, e_i)$ in $\mathcal{RG}(\mathcal{A})$, given by γ. Let s be a region in $N = \mathsf{neighbor}\langle r'_i, S'\rangle$ such that $s[R \leftarrow 0] = r_{i+1}$. From Lemmas 4.6 and 4.5, it follows that first s, then r_{i+1} are reachable from ν'_i by Perturbator's move, and that the resulting valuation is ϵ_{i+1}-tight in r_{i+1} w.r.t. $S_{r_{i+1}}$ by choosing $\epsilon' = \epsilon/2^{i+1}$ □

Proof of Prop. 4.6. Consider such valuations $\nu \in r$ and $\nu' \in r'$. For any $x, y \in \mathcal{C}$, $(S_{r'})_{x,y} = 0$ implies that $r'_{x,y} - \epsilon \leq \nu'(x) - \nu'(y) \leq r'_{x,y}$, since S_r and $S_{r'}$ have the same diagonal components and time delays do not change the quantities $\nu(x) - \nu(y)$. Hence, ν' is ϵ-tight in r' w.r.t. $S_{r'}$ for all diagonal components. Assume that $r' \lessdot^* s$. The other case is similar. Let S_s be the shrinking constraint such that for all SMs Q, $Q \leq S_s$ if and only if there is $P \leq S_N$ with $(N, P) \cap s \subseteq (s, Q)$. Let us write the clocks ordered according to their fractional values in r':

$$0 = \mathsf{frac}(X_1) < \mathsf{frac}(X_2) < \ldots < \mathsf{frac}(X_m) < 1,$$

where each X_i is a set of clocks having the same fractional value, and X_1 can be empty.

1. Assume that $(S_{r'})_{x,0} = \infty$ for all $x \in X_2 \cup \ldots \cup X_m$. Then, by definition of the neighborhood, either $r' = s$ or $r' \lessdot s$, where the latter case is possible if $X_1 \neq \emptyset$.

   - If $r' = s$, then Perturbator perturbs by 0. Since $(S_s)_{x,0} = \infty$ and $(S_s)_{x,y} = (S_{r'})_{x,y}$ for all $x, y \in \mathcal{C}$, by Lemma 4.7 (below), the valuation ν' is ϵ-tight (s, S_s) for these components. If $(S_s)_{0,x} = 0$ for some $x \in \mathcal{C}$, then $N_{0,x} < r'_{0,x}$ by Lemma 4.7 since $s_{0,x} = r_{0,x}$, so we must have $(S_{r'})_{0,x} = 0$, and ν' is also ϵ-tight for components $(0, x)$. Note that the ϵ-tightness of diagonal components follow from the ϵ-tightness of ν in r' w.r.t. $S_{r'}$.

- If $r' \lessdot s$, we let Perturbator perturb by a positive amount $0 < d' < \epsilon'$, so that the valuation is in s. Since $(S_s)_{x,0} = \infty$ and $(S_s)_{x,y} = (S_{r'})_{x,y}$ for all $x, y \in \mathcal{C}$, by Lemma 4.7, for these components, $\nu' + d'$ is ϵ-tight in (s, S_s). If $(S_s)_{0,x} = 0$ for some $x \in \mathcal{C}$, then $N_{0,x} < r'_{0,x}$ by Lemma 4.7, so we must have $(S_{r'})_{0,x} = 0$. We have, by hypothesis, $-\nu'(x) \geq r'_{0,x} = s_{0,x} - \epsilon$, so $-(\nu'(x) + d') \geq s_{0,x} - \epsilon - \epsilon'$. $\nu' + d'$ is also ϵ-tight for diagonal components since ν' is.

2. Otherwise, consider the minimum $k \in \{2, \ldots, m\}$ such that $(S_{r'})_{x,0} = 0$ for all $x \in X_k \cup \ldots \cup X_m$. Since s is a successor region of r', there exists $k' \in \{k, \ldots, m\}$ such that either clocks in $X_{k'}$ are integers, or all clocks $X_{k'+1} \cup \ldots \cup X_m$ have changed their integer parts and only these. We treat each case separately:

   (a) Assume $s_{x,0} = r'_{x,0}$, $\prec^s_{x,0} = \leq$ for $x \in X_{k'}$; $s_{x,0} = r'_{x,0} + 1$ and $\prec^s_{x,0} = <$ for all $x \in X_{k'+1} \cup \ldots \cup X_m$; $s_{x,0} = r'_{x,0}$ for all $x \in X_1 \cup \ldots \cup X_{k'-1}$. We define Perturbator's move as $d' = 1 - \mathsf{frac}(\nu'_i(x))$ for $x \in X_{k'}$. It is clear that $\nu' + d' \in s$, and that $d' \leq \epsilon$ since ν' is ϵ-tight. Let us show that the $\nu' + d'$ is ϵ-tight in s w.r.t. S_s. By Lemma 4.7, all diagonal constraints in S_s are the same as in $S_{r'}$ so the property is satisfied for these components.

   – Let us show ϵ-tightness for components $(x, 0)$ (upper bounds). For all $x \in X_{k'+1} \cup \ldots X_m$, we have $(S_s)_{x,0} = \infty$ by Lemma 4.7, since $s_{x,0} = r'_{x,0} + 1 = N_{x,0}$. We have $(\nu' + d')(x) = s_{x,0} = -s_{0,x}$ for $x \in X_{k'}$, so $\nu' + d'$ is ϵ-tight for component $(x, 0)$. For all $x \in X_k \cup \ldots \cup X_{k'-1}$, we have $s_{x,0} = r'_{x,0}$, and $\nu'(x) \geq r'_{x,0} - \epsilon$, which implies $\nu'(x) + d' \geq s_{x,0} - \epsilon$ as required. For all $x \in X_1 \cup \ldots \cup X_{k-1}$, we have $(S_s)_{x,0} = \infty$. In fact, $(S_s)_{x,0} = 0$ means, by Lemma 4.7 that $s_{x,0} < N_{x,0}$. But since $r' \lessdot^* s$, we would have $r' \leq s_{x,0} < N_{x,0}$, and this implies that $(S_{r'})_{x,0} = 0$ by definition of N and Lemma 4.7. But by the choice of k, and by Proposition 3.15, this is a contradiction.

   – We now show ϵ-tightness for components $(0, x)$ (lower bounds). For all $x \in X_{k'+1} \cup \ldots \cup X_m$, we have, $-(\nu'(x) + d) \geq s_{0,x} - \epsilon$ since $d' \leq \epsilon$. The property is again clearly satisfied by $\nu'(x) + d'$ for $x \in X_{k'}$ since $\nu'(x) + d' = s_{x,0}$. Consider now $x \in X_1 \cup \ldots \cup X_{k'-1}$ such that $(S_s)_{0,x} = 0$. We have $(S_s)_{0,y} = 0$ for $y \in X_{k'}$, and $(S_s)_{0,x} = 0$, which implies that $(S_s)_{y,x} = 0$ since paths $0, y, x$ and $0, x$ belong to $\mathsf{G}(s)$, and S_s is normalized. Then, we also have $(S_{r'})_{y,x} = 0$, so $\nu'(y) - \nu'(x) \geq s_{y,x} - \epsilon$. means that $\mathsf{frac}(\nu'(x)) + d' \leq \epsilon$. The following figure illustrates this, assuming $x \in X_i$.

$$\underbrace{0 < X_1 < \ldots < X_i}_{\mathsf{frac}(\nu'(x))} \underbrace{< X_{i+1} < \ldots < X_{k'}}_{\geq 1-\epsilon} \underbrace{< \ldots < 1}_{=d'}$$

   Therefore $\nu' + d'$ satisfies $\nu'(x) + d' \leq -s_{0,x} + \epsilon$, for all $x \in \mathcal{C}$ such that $(S_s)_{0,x} = 0$.

   (b) $s_{x,0} = r'_{x,0} + 1$ and $\prec^s_{x,0} = <$ for all $x \in X_{k'} \cup \ldots \cup X_m$; $s_{x,0} = r'_{x,0}$ and $\prec^s_{x,0} = <$ for all $x \in X_1 \cup \ldots \cup X_{k'-1}$. In this case, we first delay to the immediate time predecessor of s as in the previous case, then add a positive delay d' of at most ϵ'. Then, $\nu' + d'$ is $(\epsilon + \epsilon')$-tight in s w.r.t. S_s.

□

The following technical lemma is used in the proof of Lemma 4.6 above.

Lemma 4.7. *Let $\langle N, S_N \rangle$ denote the constrained neighborhood of some constrained region. Let r be any region included in N. Let S_r denote the shrinking constraint such that for all SMs Q, $Q \leq S_r$*

if, and only if there is $P \leq S_N$ *with* $(N, P) \cap r \subseteq (r, Q)$ *for all small* $\delta > 0$*. Then, for all* $x, y \in \mathcal{C}$*,* $(S_r)_{x,y} = 0$ *implies that* $(S_N)_{x,y} = 0$*; for all* $x \in \mathcal{C}$*,* $(S_r)_{x,0} = 0$ *implies* $r_{x,0} < N_{x,0}$ *and* $(S_r)_{0,x} = 0$ *implies* $r_{0,x} < N_{0,x}$*.*

Proof. By Lemma 3.12, S_r is defined as follows. Let S_1 obtained by $(S_1)_{x,y} = S_{x,y}$ if $r_{x,y} = N_{x,y}$ and 0 otherwise. Then, S_r is the normalization (in the sense of SMs) of S_1, so $S_1 \leq S_r$. But all diagonal constraints are the same in r and N, and $r_{x,0} = N_{x,0}$ implies $(S_1)_{0,x} = S_{x,0} = \infty$, and similarly $r_{0,x} = N_{0,x}$ implies $(S_1)_{0,x} = S_{0,x} = \infty$. The result follows. □

We also note the following corollary of Lemma 4.6 and Lemma 4.5, which will be used in Section 5.

Corollary 4.8. *Let* (ℓ, r, S_r) *be a square node of* $\mathcal{RG}(\mathcal{A})$*. Then* $\langle r, S_r \rangle$ *admits tight valuations.*

Proof. We prove this for each node $n = (\ell, r, S_r)$ by induction on the shortest path from the initial node of $\mathcal{RG}(\mathcal{A})$ to n. It is true for the initial node $(\ell_0, \vec{0}, \vec{0})$. Assume it is the case for $n = (\ell, r, S_r)$, and pick any square node $n'' = (\ell', t, S_t)$ reached via a diamond node $n' = (\ell, s, S_s, e)$. For all successor $(\ell', t', S_{t'})$ of n', pick any SM $Q_{t'} \leq S_{t'}$, and consider $P' \leq S_s$ given by Lemma 3.17 such that

$$(s, P') \subseteq s \cap \mathsf{shrink}(\bigcup_{s' \in \mathcal{N}} (s' \cap \mathsf{Unreset}_R((t', Q_{t'})))),$$

where $\mathcal{N}$ is the neighborhood of $\langle s, S_s \rangle$, and we write $t' = s'[R \leftarrow 0]$ for any $s' \in \mathcal{N}$. Further, let $P_r \leq S_r$ such that

$$(r', P_r) = r \cap \mathsf{shrink}^+(\mathsf{Pre}_{\mathsf{time}}((s, P'))).$$

Choose $\delta > 0$ small enough so that all equations hold. Let ν be any ϵ-tight valuation in $r' - \delta P$. By the above equations, by a joint move of Controller and Perturbator, the game can proceed to any of the set $t' - \delta Q_{t'}$. In particular, by Lemmas 4.6 and 4.5, the resulting valuation can be chosen by Perturbator to be inside $t - \delta Q_t$ as $(\epsilon + \epsilon')$-tight for any $\epsilon' > 0$. Since Q_t was chosen arbitrarily, $\langle t, S_t \rangle$ admits tight valuations.

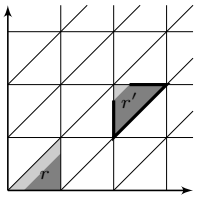
□

5. Extension to Turn-based Timed Games

In this section, we extend the reachability algorithm to turn-based timed games.

The correctness of the abstract game for timed automata (Proposition 4.1) was based on 1) the ability of the controller to always delay inside shrinkings of regions it visits, 2) while not being able to avoid visiting tight valuations. While the former ensured Controller to win $\mathcal{G}_\delta(\mathcal{A})$ if a winning strategy of the abstract game is given, the latter ensured the symmetric property for Perturbator.

The situation is different in a game setting since from locations $\mathcal{L}_P$, Perturbator has no reason to delay into particular shrinkings. In fact, consider Figure 10 which shows the states that can be reached by a delay inside region r', at a location $\mathcal{L}_P$, starting from a shrinking of r, with $r \lessdot^+ r'$. Perturbator can thus delay to valuations close to borders. In fact, if the play arrives to $\mathcal{L}_P$ inside $r - \delta P$, then $r' \cap \mathsf{Post}_{\mathsf{time}}(r - \delta P)$ is the set of valuations that that can be reached inside r' after a delay. While this can still be expressed as a shrunk region (by Lemma 3.6), this set does not admit tight valuations, so the previous proof cannot be carried out directly. We will show however that this set is always included in the union of at most two shrunk regions that admit tight valuations (See Fig. 10(b)). This property will allow us to adapt the abstract game construction and prove its correctness following the same proof as for timed automata.



(a) A region r and its time successors inside r'.

(b) The shrinking of r' on the left can be decomposed as two shrinkings that admit tight valuations. Any actual delay of Perturbator will belong to one of these two shrinkings.

Figure 10: At Perturbator's locations, the delays cannot be expected to end in shrinkings of a particular form. In this example, if Perturbator delays in region r' from r, the valuation can be closed to the borders of the region, and the shrinking expressing the reachable valuations do not admit tight valuations (on the left). We will rather decompose this shrunk region into two shrunk regions that do admit tight valuations (on the right).

For the proofs, we first need the following characterization of constrained regions that admit tight valuations. Intuitively, a constrained region does not admit tight valuations 1) if we have $S_{x,y} = S_{y,x} = 0$ for some pair $x, y \in \mathcal{C}_0$ with $r_{x,y} \neq -r_{y,x}$ since this would require a valuation v to be close to two distant lines, and 2) if the zeros of S are "transitive". The characterization is formally proved in the next lemma.

Lemma 5.1. *Let $\langle r, S_r\rangle$ be a constrained region. Then, $\langle r, S_r\rangle$ admits tight valuations if, and only if the two following conditions hold:*

1. *For any $x, y \in \mathcal{C}_0$, if $r_{x,y} \neq -r_{y,x}$ and $(S_r)_{x,y} = 0$, then $(S_r)_{y,x} = \infty$,*
2. *For any $x, y \in \mathcal{C}_0$, if there is $x_1 x_2 \dots x_n \in \Pi_{x,y}(\mathsf{G}(r))$ with $(S_r)_{x_i,x_{i+1}} = 0$ for $1 \leq i \leq n-1$, then $(S_r)_{x,y} = 0$.*

Proof. Assume that the first condition is not satisfied. If $r_{x,y} \neq -r_{x,y}$ and $(S_r)_{x,y} = (S_r)_{y,x} = 0$, then for any $\nu \in r$, $\nu(x) - \nu(y) > r_{x,y} - \epsilon$ implies $\nu(x) - \nu(y) > -r_{x,y} + \epsilon$, for small enough $\epsilon > 0$. So no valuation ν is tight in r. Assume now that the second condition is not satisfied. If $(S_r)_{x_i,x_{i+1}} = 0$ for all $1 \leq i \leq n-1$ and $(S_r)_{x_1,x_n} = \infty$, then consider any SM P with $P_{x_1,x_n} = 1$. If ν is ϵ-tight in $r - \delta P$, then we have $r_{x_i,x_{i+1}} - \epsilon \leq \nu(x_i) - \nu(x_{i+1})$ for all $1 \leq i \leq n-1$. Summing these yield $\sum_{i=1}^{n-1} r_{x_i,x_{i+1}} - n\epsilon \leq \nu(x_1) - \nu(x_n)$. We have $\sum_{i=1}^{n-1} r_{x_i,x_{i+1}} = r_{x_1,x_n}$ by hypothesis, and also $\nu(x_1) - \nu(x_n) \leq r - \delta$. Now, this means $r_{x_1,x_n} - \epsilon n \leq r_{x_1,x_n} - \delta$, which is a contradiction for small enough $\epsilon > 0$.

Consider the fractional ordering of the clocks in r: $X_1, \dots, X_n$ and let $2 \leq i_0 \leq n+1$ and $1 \leq j_0 \leq n$ be the indices given by Lemma 3.15. By hypothesis, we must have $j_0 < i_0$, since otherwise $j_0 \geq 2$, and $r_{x,0} \neq -r_{0,x}$ for $x \in X_{j_0}$, but $(S_r)_{x,0} = (S_r)_{0,x} = 0$, contradicting the hypothesis.

Observe that if $x \in X_k$ and $y \in X_l$ with $j_0 < k < l < i_0$, then $(S_r)_{y,x} = \infty$, since $(y, 0, x) \in \Pi_{y,x}(\mathsf{G}(r))$ and $(S_r)_{y,0} = \infty$. Furthermore, unless $i_0 = j_0 + 1$, there exists

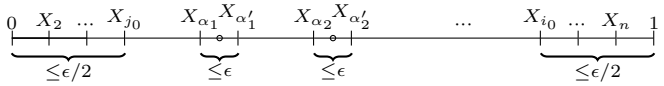
$$j_0 + 1 = \alpha_1 \leq \alpha'_1 < \alpha_2 \leq \alpha'_2 < \ldots < \alpha_m \leq \alpha'_m = i_0 - 1,$$

such that for all $x \in X_k$ and $y \in X_l$ with $\alpha_i \leq k < l \leq \alpha'_i$, we have $(S_r)_{x,y} = 0$, and if $\alpha_i \leq k \leq \alpha'_i$ and $l > \alpha'_i$ then $(S_r)_{x,y} = \infty$. In fact, for any pair $x \in X_k$ and $y \in X_l$ with $j_0 < k < l < i_0$,

$(S_r)_{x,y} = 0$ implies that $(S_r)_{x',y'} = 0$ for all $x' \in X_{k'}$ and $y' \in X_{l'}$ with $k \leq k' < l' \leq l$, because S_r is normalized. Moreover, for such a pair x, y, if for $z \in X_{l+1}$, we have $(S_r)_{y,z} = 0$, then also $(S_r)_{x,z} = 0$. This follows from the hypothesis since $x, y, z \in \Pi_{x,z}(\mathsf{G}(r))$ and has S_r-weight equal to 0. Thus, the sequence $(\alpha_i, \alpha'_i)_i$ is well-defined. Consider $x \in X_k$ and $y \in X_l$ for $k \in [\alpha_i, \alpha'_i]$ and $y \in [\alpha_j, \alpha'_j]$ with $i < j$, then $(S_r)_{x,y} = \infty$. In fact, for any clock $z \in X_{\alpha'_i}$ and $z' \in X_{\alpha_j}$, we have $x, z, z', y \in \Pi_{x,y}(\mathsf{G}(r))$, and $(S_r)_{z,z'} = \infty$ by definition of the sequence α, α'.

We now show that (r, S_r) has ϵ-tight valuations. Consider any normalized SM $P \leq S_r$ and $1/\delta > 2(|\mathcal{C}| + 2) \max_{x,y \in \mathcal{C}_0}(P_{x,y})$.

Fix any $0 < \epsilon < \delta/2$. We define a valuation ν as follows. We let $\nu(x) = r_{x,0}$ for $x \in X_1$. We choose the values $\nu(x)$ for $x \in X_2 \cup \ldots \cup X_{j_0}$ as $-r_{0,x} < \nu(x) < -r_{0,x} + \epsilon/2$, such that they respect the fractional ordering. Similarly, we choose $r_{x,0} - \epsilon/2 < \nu(x) < r_{x,0}$ for $x \in X_{i_0} \cup \ldots \cup X_n$, respecting the fractional ordering. Define $A_i = \frac{i}{|\mathcal{C}|+2}$ for $1 \leq i \leq m$. Define $X_{i,j} = X_i \cup X_{i+1} \cup \ldots X_j$. For each set X_{α_i, α'_i} for $1 \leq i \leq m$, we choose values in the interval $[A_i - \epsilon/2, A_i + \epsilon/2]$ respecting the fractional orderings. The valuation is illustrated in the following figure.



We show that ν is ϵ-tight in $r - \delta P$. We verify at the same time that ν belongs to $r - \delta P$, and that it is ϵ-tight. We first consider the constraints between clocks $X_2 \cup \ldots X_{j_0}$ and $X_{i_0} \cup \ldots \cup X_n$, then between clocks in $x_{j_0+1} \cup \ldots \cup X_{i_0-1}$, and finally for pairs between the two sets.

1. For all $x \in X_2 \cup \ldots \cup X_{j_0}$, we have, by definition $(S_r)_{0,x} = 0$ and $-r_{0,x} < \nu(x) < -r_{0,x} + \epsilon/2$, and $(S_r)_{x,0} = \infty$. By the choice of δ, $\nu(x) < -r_{0,x} + \epsilon/2 < r_{x,0} - \delta Q_{x,0}$. For any pair $x \in X_k$ and $y \in X_l$ with $1 \leq k < l \leq j_0$, we have $(S_r)_{x,y} = 0$ by normalization since $0, x, y$ is in $\Pi_{0,y}(\mathsf{G}(r))$, but the previous inequality implies that $r_{x,y} - \epsilon/2 < \nu(x) - \nu(y) < r_{x,y}$. We have, therefore, $(S_r)_{y,x} = \infty$, and $\nu(y) - \nu(x) < -r_{x,y} + \epsilon/2 < r_{y,x} - \delta P_{x,y}$ by the choice of δ.

2. Similarly, for all $x \in X_{i_0} \cup \ldots \cup X_n$, we have $(S_r)_{0,x} = \infty$, $(S_r)_{x,0} = 0$ and $r_{x,0} - \epsilon/2 < \nu(x) < r_{x,0}$. For any pair $x \in X_k$ and $y \in X_l$ with $i_0 \leq k < l \leq n$, we have $(S_r)_{x,y} = 0$ as before, and $r_{x,y} - \epsilon/2 < \nu(x) - \nu(y) < r_{x,y}$. We have $(S_r)_{y,x} = \infty$, and $\nu(y) - \nu(x) < -r_{x,y} + \epsilon/2 < r_{y,x} - \delta P_{y,x}$ by the choice of δ and ϵ.

3. For any pair $x \in X_2 \cup \ldots \cup X_{j_0}$ and $y \in X_{i_0} \cup \ldots \cup X_n$, we have $(S_r)_{y,x} = 0$ by hypothesis, since $y, 0, x$ is in $\Pi_{y,x}(\mathsf{G}(r))$ and all edges have S_r-weight 0. Thus, $\nu(y) - \nu(x) < r_{y,x} - \delta P_{y,x}$, and we have $r_{y,x} - \epsilon < \nu(y) - \nu(x)$ since $\nu(x) > r_{x,0} - \epsilon/2$ and $\nu(y) < -r_{0,y} + \epsilon/2$. Furthermore, $-r_{x,y} + \delta P_{x,y} < \nu(y) - \nu(x)$ since $-r_{x,y} + \delta P_{x,y} < r_{y,x} - \epsilon$ by the choice of δ and ϵ.

4. Let us now consider constraints between clocks in $X_{j_0+1} \cup \ldots X_{i_0-1}$. For any $x \in X_k$ and $y \in X_l$ with $j_0 < k < l < i_0$, we have $(S_r)_{y,x} = \infty$ since $(y, 0, x)$ belongs to $\Pi_{y,x}(\mathsf{G}(r))$ and $(S_r)_{y,0} = \infty$. We have $\nu(y) - \nu(x) < r_{y,x} - \delta P_{y,x}$ since $\mathsf{frac}(\mathsf{frac}(\nu(y)) - \mathsf{frac}(\nu(x))) \leq \frac{m-1}{|\mathcal{C}|+2} + \epsilon \leq 1 - \delta P_{y,x}$. Assume that $\alpha_i \leq k < l \leq \alpha'_i$, for some i. We have $(S_r)_{x,y} = 0$ and $r_{x,y} - \epsilon < \nu(x) - \nu(y) < r_{x,y}$ by definition of ν. If $k \in [\alpha_i, \alpha'_i]$ and $l \in [\alpha_j, \alpha'_j]$ for $i < j$, then $(S_r)_{x,y} = \infty$. Since $\frac{1}{|C|+2} + \epsilon > \delta P_{x,y}$, we have $\nu(x) - \nu(y) < r_{x,y} - \delta P_{x,y}$.

5. Consider now clock $x \in X_2 \cup \ldots \cup X_{j_0}$ and $y \in X_{j_0+1} \cup \ldots \cup X_{i_0-1}$. We have $(S_r)_{x,y} = \infty$ by hypothesis. In fact, $(0, x, y) \in \Pi_{0,y}(\mathsf{G}(r))$ and $(S_r)_{x,y} = 0$ would imply $(S_r)_{0,y} = 0$. Also,

$(S_r)_{y,x} = \infty$ because $(S_r)_{y,0} = \infty$. We have $-r_{y,x} + \delta P_{y,x} < \nu(x) - \nu(y) < r_{x,y} - \delta P_{x,y}$ by $\frac{1}{|C|+2} + \epsilon > \delta \max_{x,y} P_{x,y}$.

□

We need to extend Lemma 3.12 to the $\mathsf{Post}_{\mathsf{time}}()$ operation.

Lemma 5.2. *Let M, N be normalized non-empty DBMs such that $N = \mathit{Post}_{\mathit{time}}(M)$ and $S \leq \mathit{Well}(M)$. There exists a shrinking constraint S' for N such that for all SMs Q, $Q \leq S'$ if, and only if there exists $P \leq S$ with $\mathit{Post}_{\mathit{time}}((M,P)) = (N',Q)$ with $\mathsf{N}' = \mathsf{N}$.*

Proof. Recall that N is obtained from M by setting all components $(x,0)$ to $(\infty,<)$ and that the resulting DBM is already normalized. Define $(S_1)_{x,0} = \infty$ for all $x \in \mathcal{C}$ and $(S_1)_{x,y} = S_{x,y}$ for all $(x,y) \in \mathcal{C}_0 \times \mathcal{C}$. We define S' as the normalization of S_1.

For any $P \leq S$, let (N,Q) denote the shrunk DBM such that $(N,Q) = \mathsf{Post}_{\mathsf{time}}((M,P))$. Let us show that $Q \leq S'$. Here, Q is the normalization of Q' defined by $Q'_{x,y} = P_{x,y}$ if $y \neq 0$ and $Q'_{x,y} = 0$ if $y = 0$. Since $N_{x,0} = \infty$ for all $x \in \mathcal{C}$, we only need to verify that $Q'_{x,y} \leq S'_{x,y}$ when $y \neq 0$. For these components, we have $N_{x,y} = M_{x,y}$ and $M \subseteq N$, therefore $\Pi_{x,y}(\mathsf{G}(N)) \subseteq \Pi_{x,y}(\mathsf{G}(M))$. So Q' is already normalized because P is, and $Q = Q'$. By definition of S_1 above, we have $Q \leq S_1$, which implies $Q \leq S'$ (since Q is normalized). This also shows that if $Q \not\leq S'$, then for all $P \leq S$, we have $\mathsf{Post}_{\mathsf{time}}(M - \delta P) \not\subseteq N - \delta Q$. In fact, for any such P, as we just showed, there exists $Q' \leq S'$ such that $\mathsf{Post}_{\mathsf{time}}(M - \delta P) = N - \delta Q'$. But $N - \delta Q' \subseteq N - \delta Q$ if, and only if $Q \leq Q'$, and $Q' \leq S'$ implies $Q \leq S$.

For any $Q \leq S'$, let us define P such that $(M,P) = M \cap (N,Q)$. We have $\mathsf{Post}_{\mathsf{time}}((M,P)) = (N,Q)$, since $\mathsf{Post}_{\mathsf{time}}(M \cap (N,Q)) = (N,Q)$. Let us show that $P \leq S$. P is the normalization of P' defined by $P'_{x,y} = Q_{x,y}$ if $y \neq 0$ and $P'_{x,y} = 0$ otherwise. Consider (x,y) such that $S_{x,y} = 0$.

- If $y \neq 0$, then $S'_{x,y} = 0$, by definition of S'. Consider any path $x_1 \dots x_n$ in $\Pi_{x,y}(\mathsf{G}(M))$. We have $S_{x_j,x_{j+1}} = 0$ for all $1 \leq j \leq n-1$. But, if $x_{j+1} = 0$, then $P'_{x_j,x_{j+1}} = 0$ by definition, and otherwise $P'_{x_j,x_{j+1}} = 0$ because $S'_{x_j,x_{j+1}} \leq S_{x_j,x_{j+1}} = 0$. Therefore, $P_{x,y} \leq S_{x,y}$.

- For the case $y = 0$, one can notice that since $N_{x,0} = \infty$ for all $x \in \mathcal{C}$, for all normalized SMs P, we have $P_{x,0} = 0$ by definition. □

The following lemma gives the construction illustrated in Fig. 10. The idea is the following. We know that a time successor region of a constrained region admitting tight valuations does not admit tight valuations in general (see Fig. 10). Nevertheless, we show, in the following lemma, that such a time successor can be over-approximated by the union of at most two constrained regions admitting tight valuations (see the two regions on the right, in Fig. 10). This property will allow us to carry out the same proof as in the previous section, based on tight valuations. The idea of the proof consists in showing that, given $\langle r, S_r \rangle$ and a successor region r', the time successor valuations of r that are in r' are either close to the first facet of r' they cross when entering r', or to the last facet they cross before leaving r'. These facets can be defined looking at the order in which the fractional values of the clocks evolve during a delay, as in the figures below.

Lemma 5.3. *Let $\langle r, S_r \rangle$ be any constrained region that admits tight valuations, and r' such that $r \lessdot^* r'$. There exist, and one can compute, S^1, S^m with $m \in \{1,2\}$, such that each $\langle r', S^i \rangle$ admits tight valuations and for any SMs $(Q_i)_{1 \leq i \leq m}$, we have $Q_i \leq S^i$ for all $1 \leq i \leq m$, if, and only if*

there exists an SM $P \leq S_r$ with

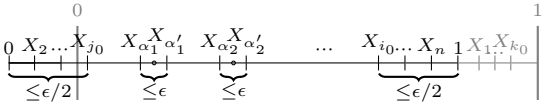
$$\mathsf{Post}_{\mathsf{time}}(r - \delta P) \cap r' \subseteq \bigcup_{i=1}^{m} (r' - \delta Q_i),$$

for all small $\delta > 0$. Moreover, in this case, for any ϵ-tight valuation $v \in (r, P)$ and any $i \in \{1, 2\}$ and $\epsilon' > 0$, there exists $d \geq 0$ such that $v + d$ is $(\epsilon + \epsilon')$-tight in (r', Q_i).

Proof. Consider the clock ordering of a constrained region with tight valuations as in the proof of Lemma 5.1. We assume the indices $1 \leq i_0, j_0 \leq n$ and α_i, α'_i's are defined (see figure below). We distinguish cases according to the last clock that crosses an integer value during the delay from r to r': There exists $1 \leq k_0 \leq n$ and $m \in \mathbb{N}$ such that for all $x \in X_1 \cup \ldots \cup X_{k_0}$, $-r'_{0,x} = -r_{0,x} + m$, while for all $x \in X_{k_0+1} \cup \ldots \cup X_n$, we have $-r'_{0,x} = -r_{0,x} + m + 1$. Notice that $-r_{0,x}$ is the integer part of the clock x in region r. If all integer parts have grown by m, then we let $k_0 = 1$. In the following figure, we represent by a vertical gray line some possible values of the index k_0.

We distinguish several cases according to the relative position of k_0.

1. Assume that $1 \leq k_0 \leq j_0 - 1$. The clock ordering before the delay is illustrated in the following figure in black, whereas the ordering after the delay is given between the gray 0 and 1.



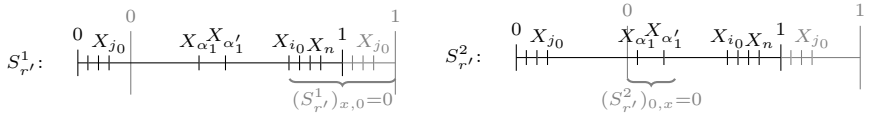
   Let us consider the shrinking constraint $S_{r'}$ given by Lemmas 3.12 and 5.2, so that for all SMs Q, we have $Q \leq S_{r'}$ iff there is $P \leq S$ with $\mathsf{Post}_{\mathsf{time}}(r - \delta P) \cap r' \subseteq r' - \delta Q$. The constraint $S_{r'}$ can be obtained from S_r by only changing the following components: for all $x \in X_1 \cup \ldots \cup X_{k_0}$ $(S_{r'})_{x,0} = 0$, and either $r'_{x,0} = -r'_{x,0}$ and $(S_{r'})_{0,x} = 0$ or $(S_{r'})_{0,x} = \infty$. In fact, since $r' - \delta Q$ contains the successors of $r - \delta P$, the diagonal constraints are the same. Moreover, the partition of the clocks given by i_0, j_0 and α_i, α'_i's are preserved by delays; while time is only translated. So the clocks $X_0, \ldots, X_{k_0}$ will be mixed with $X_{i_0}, \ldots, X_n$ but the constraints on the rest of the clocks will remain. By Lemma 5.1, $(r', S_{r'})$ has tight valuations.

   Consider SMs Q and P such that $\mathsf{Post}_{\mathsf{time}}(r - \delta P) \cap r' \subseteq r' - \delta Q$ for all small enough $\delta > 0$. Let $\nu \in r - \delta P$ be an ϵ-tight valuation in (r, S_r). Define $d = m + 1 - \nu(x_{k_0+1})$ if x_{k_0+1} has integer value in r' and $d = m + 1 - \frac{\nu(x_{k_0}) + \nu(x_{k_0+1})}{2}$ otherwise (if $k_0 = n$, then let us assume $x_{n+1} = 1$). Then, according to the definition of k_0 and m, $\nu + d$ belongs to r'. Therefore, also $\nu + d \in r - \delta Q$. We show that $\nu + d$ is ϵ-tight in $(r', S_{r'})$. Diagonal components stay unchanged both in shrinking constraints and in valuations, so we only need to verify the constraints $(S_{r'})_{x,0}$ and $(S_{r'})_{0,x}$. Since the distances between the first j_0 clocks are at most ϵ in ν (i.e. $\nu(x_{j_0}) - (-r_{0,x_{j_0}}) \leq \epsilon$), we have that $r'_{x,0} - \epsilon \leq \nu(x) + d \leq r'_{x,0}$ for all $x \in X_1 \cup \ldots \cup X_{k_0}$, and $-r'_{0,x} \leq \nu(x) + d \leq -r'_{0,x} + \epsilon$ for all $x \in X_{k_0+1} \cup \ldots \cup X_{j_0}$. It remains to verify that $r'_{y,0} - \epsilon \leq \nu(y) + d \leq r'_{y,0}$ for $y \in X_{i_0} \cup \ldots \cup X_n$. This follows from the fact that $r_{y,x} - \epsilon \leq \nu(y) - \nu(x) \leq r_{y,x}$ (which holds by hypothesis on (r, S_r)).

   The case where $k_0 \geq i_0 + 1$ is symmetric.

2. Assume that $j_0 \leq k_0 < \alpha_1$. Note that we may have $X_{j_0} = \emptyset$ or $X_{i_0} = \emptyset$, or both. We define two shrinking constraints $S^1_{r'}$ and $S^2_{r'}$, such that $(r, S^i_{r'})$ admits tight valuations for both $i = 1, 2$. Both have the same diagonal components as S_r, since these do not change along delays. The first one corresponds to delaying to points where clocks X_{k_0} can be very close to their upper integer values (Fig. 11(a)), and the second one to points where clocks X_{α_1} are very close to their integer parts (Fig. 11(b)). We will show that for any pair $Q^1 \leq S^1_{r'}$ and $Q^2 \leq S^2_{r'}$, there is a corresponding $P \leq S_r$ such that any valuation obtained by delaying from $r - \delta P$ inside r' lies either in $(r' - \delta Q^1)$ or $(r' - \delta Q^2)$, and moreover ϵ-tight valuations are reached in these shrunk regions if one starts at an ϵ-tight valuation in (r, S_r).

   To define $(S^i_{r'})$, we modify S_r as follows. We let $(S^1_{r'})_{x,0} = 0$ and $(S^1_{r'})_{0,x} = \infty$ for all $x \in X_1 \cup \ldots \cup X_{j_0}$. Then $(S^1_{r'})$ admits ϵ-tight valuations by Lemma 5.1. We let $(S^2_{r'})_{x,0} = (S^2_{r'})_{0,x} = \infty$ for all $x \in X_1 \cup \ldots \cup X_{j_0}$ and $x \in X_{i_0} \cup \ldots \cup X_n$, and $(S^2_{r'})_{0,x} = 0$ and $(S^2_{r'})_{x,0} = \infty$ for all $x \in X_{\alpha_i} \cup \ldots \cup X_{\alpha'_i}$. Then, similarly, $(S^2_{r'})$ admits ϵ-tight valuations. The definitions are illustrated in Figure 11.



(a) $S^1_{r'}$ (b) $S^2_{r'}$

Figure 11: The ordering of the clock partition $X_1, \ldots, X_n$ in region r is shown in black in both figures. The new ordering inside r' is shown in gray.

Consider any pair of SMs $Q_i \leq (S^i_{r'})$ for $i \in \{1, 2\}$. We define P satisfying S_r as follows. We let $P_{x,0} = 0$ whenever $(S_r)_{x,0} = 0$ or $(S_{r'})_{x,0} = 0$, and $P_{0,x} = 0$ whenever $(S_r)_{0,x} = 0$. We choose arbitrary values for other components satisfying S_r and the following constraints:

(a) $P_{x,y} \geq \max(Q^1_{x,y}, Q^2_{x,y})$ for all $x, y \in \mathcal{C}$. Notice here that $Q^1_{x,y} = Q^2_{x,y} = 0$ whenever $(S_r)_{x,y} = 0$ by definition of $S^1_{r'}$ and $S^2_{r'}$.

(b) For all $x \in X_{\alpha_1} \cup \ldots \cup X_{i_0 - 1}$, $P_{x,0} \geq \max(Q^1_{x,0}, Q^2_{x,0})$. Notice that $(S_r)_{x,0} = \infty$ for these components.

(c) For all $x \in \mathcal{C} \setminus (X_{\alpha_1} \cup \ldots \cup X_{\alpha'_1})$, $P_{\alpha_1, x} \geq Q^2_{0,x}$. Notice that $(S_r)_{x,\alpha_1} = \infty$ for these components.

(d) For all $x \in \mathcal{C} \setminus (X_{\alpha_1} \cup \ldots \cup X_{\alpha'_1})$, and $y \in X_{\alpha_1} \cup \ldots \cup X_{\alpha'_1}$, $P_{x,\alpha_1} \geq \max(Q^1_{0,x}, Q^2_{x,0} + Q^1_{0,y})$. We have again $(S_r)_{x,\alpha_1} = \infty$.

This shows that one can choose values for P while satisfying S_r. Note also that normalization of P can only increase its components, so all lower bounds given above will still be satisfied.

Consider any $\nu \in r - \delta P$. Let us show that for any $d \geq 0$ with $\nu + d \in r'$, we have $\nu + d \in r' - \delta Q^i$ for $i = 1$ or $i = 2$. For diagonal components $(x, y) \in \mathcal{C}^2$, we have $-r'_{y,x} + \delta(Q_i)_{y,x} \prec^{r'}_{y,x} \nu(x) + d - (\nu(y) + d) \prec^{r'}_{x,y} r'_{x,y} - \delta(Q_i)_{x,y}$ for small enough $\delta > 0$. In fact, either $(S_{r'})_{x,y} = 0$ and $(Q_i)_{x,y} = 0$ for $i = 1, 2$, since $S^i_{r'}$ has the same diagonal components as S_r, or $P_{x,y} \geq \max((Q_1)_{x,y}, (Q_2)_{x,y})$.

It remains to choose $i \in \{1, 2\}$ such that the constraints on the components $(x, 0)$ and $(0, x)$ are satisfied. Let us write $\nu' = \nu + d$.

(a) Assume that for all $x \in X_1 \cup \ldots X_n$, $-r'_{0,x}+\delta Q^1_{0,x} \prec^{r'}_{0,x} \nu'(x)$. For all $x \in X_{i_0} \cup \ldots \cup X_n$ and $x \in X_1 \cup \ldots \cup X_{j_0}$, we have $Q^1_{x,0} = 0$ so $\nu(x) \prec^{r'}_{x,0} r'_{x,0} - \delta Q^1_{x,0}$. For any $x \in X_{\alpha_1} \cup \ldots X_{i_0-1}$, we have $P_{x,0} \geq Q^1_{x,0}$, so $\nu'(x) \prec^{r'}_{x,0} r'_{x,0} - \delta Q^1_{x,0}$.

(b) Assume that for some $1 \leq i \leq n$, for clocks $x \in X_i$, $-r'_{0,x} + \delta Q^1_{0,x} \not\prec^{r'}_{0,x} \nu'(x)$. We show that $\nu' \in r' - \delta Q^2$. Let us first show that we have $i \in \{\alpha_1, \ldots, \alpha'_1\}$. This follows from the fact that $P_{x,\alpha_1} \geq Q^1_{0,x}$ for all clocks $x \in \mathcal{C} \setminus (X_{\alpha_1} \cup \ldots \cup X_{\alpha'_1})$. In fact, this ensures that the constraint $-r'_{0,x} + \delta Q^1_{0,x} \prec^{r'}_{0,x} \nu'(x)$ is always satisfied for these clocks, hence $i \in \{\alpha_1, \ldots, \alpha'_1\}$.

We first show that the upper bounds hold. Let $y \in X_{\alpha_1}$ and $y' \in X_i$. We have $\nu'(y) \leq -r'_{0,y} + \delta Q^1_{0,y'}$ since $\nu'(y') \leq -r'_{0,y'} + \delta Q^1_{0,y'}$ by definition of i, and $\mathsf{frac}(\nu(y)) \leq \mathsf{frac}(\nu(y'))$. For any $x \in X_{i_0} \cup \ldots \cup X_n \cup X_1 \cup \ldots \cup X_{j_0}$, we also have $\nu'(x) - \nu'(y) \prec^{r'}_{x,y} r'_{x,y} - \delta P_{x,y}$, which together yields $\nu'(x) \prec^{r'}_{x,0} r'_{x,0} - \delta(P_{x,y} - Q^1_{0,y'})$. This implies the desired bound since $P_{x,y} \geq Q^2_{x,0} + Q^1_{0,y'}$. For any clock $x \in X_{\alpha_1} \cup \ldots \cup X_{i_0-1}$, we have $P_{x,0} \geq Q^2_{x,0}$, which yields $\nu'(x) \prec^{r'}_{x,0} -\delta Q^2_{x,0}$.

It remains to show the lower bounds. We have $Q^2_{0,x} = 0$ for all $x \in X_{\alpha_1} \cup \ldots \cup X_{\alpha'_1}$ so $-r'_{0,x} + \delta Q^2_{0,x} \prec^{r'}_{0,x} \nu'(x)$. For all $x \in \mathcal{C} \setminus (X_{\alpha_1} \cup \ldots \cup X_{\alpha'_1})$, we have by definition $P_{\alpha_1,x} \geq Q^2_{0,x}$. Then, $-r'_{\alpha_1,x} + \delta P_{\alpha_1,x} \prec^{r'}_{\alpha_1,x} \nu'(x) - \nu'(\alpha_1)$. Combining with the fact that $\nu'(\alpha_1) \geq -r'_{0,\alpha_1}$ we get $-r'_{0,x} + \delta Q^2_{0,x} \prec^{r'}_{0,x} \nu'(x)$.

It is now easy to see that from ϵ-tight valuations in (r, S_r), one can reach $(\epsilon + \epsilon')$-tight valuations in both $(r', S^1_{r'})$ and $(r, S^2_{r'})$. In fact, for any $\epsilon' > 0$, the delay $d \geq 0$ can be chosen such that $r'_{x,0} - \nu'(x) \leq \epsilon'$, for $x \in X_{j_0}$, but also such that $\nu'(x) + r'_{0,x} \leq \epsilon'$ for $x \in X_{\alpha_1}$. In the former case, ν' is $(\epsilon + \epsilon')$-tight in $(r', S^1_{r'})$, and in the latter case, ν' is $(\epsilon + \epsilon')$-tight in $(r', S^2_{r'})$.

3. The cases where $\alpha'_i \leq k_0 < \alpha_{i+1}$ for $i > 1$ or $\alpha'_m \leq k_0 < i_0$ are treated similarly to the previous case: Here α'_i has the role of j_0, and α_{i+1} has the role of α_1.

4. Assume that $\alpha_i \leq k_0 < \alpha'_i$. Then $S_{r'}$ has the same diagonal components as S_r, and for any $x \in X_{\alpha_i} \cup \ldots \cup X_{k_0}$, $(S_{r'})_{x,0} = 0$ and $(S_{r'})_{0,x} = \infty$, while for any $x \in X_{k_0+1} \cup \ldots \cup X_{\alpha'_i}$, $(S_{r'})_{x,0} = \infty$ and $(S_{r'})_{0,x} = 0$. Furthermore, for any other clock $x \in \mathcal{C}$, $(S_{r'})_{x,0} = (S_{r'})_{0,x} = \infty$.

If we fix SMs P and Q such that $\mathsf{Post}_{\mathsf{time}}(r - \delta P) \cap r' \subseteq r' - \delta Q$, then for any ϵ-tight valuation $\nu \in r - \delta P$, any delay d with $\nu + d \in r'$ is ϵ-tight in $(r', S_{r'})$. In fact, the diagonal components stay unchanged during delays, and the fractional values of clocks $X_{\alpha_i} \cup \ldots \cup X_{\alpha'_i}$ differ by at most ϵ in ν.

5. Assume that $j_0 + 1 = i_0$ (there is no α_i, α'_i). Consider the case $k_0 = j_0$. The proof follows again the same ideas as before, but is simpler. We give the details. We define $(S^1_{r'})_{x,y} = (S^2_{r'})_{x,y} = (S_r)_{x,y}$ for all $x, y \in \mathcal{C}$. We let $(S^1_{r'})_{x,0} = 0$ and $(S_{r'})_{0,x} = \infty$ for all $x \in \mathcal{C}$, and $(S^2_{r'})_{0,x} = \infty$ and $(S_{r'})_{0,x} = 0$ for all $x \in \mathcal{C}$. $S^1_{r'}$ is represented in Figure 11(a) (with the difference that there is no α_i's).

Let $Q^i \leq S^i_{r'}$ for $i = 1, 2$. Let $j'_0 = \min\{j \mid 1 \leq j \leq j_0, X_j \neq \emptyset\}$. Define P, where $P_{x,y} \geq Q^1_{0,x}$ for all $x \in X_{i_0} \cup \ldots \cup X_n$ and $y \in X_{j'_0}$. Let $P_{x,i_0} \geq Q^1_{0,y} + Q^2_{x,0}$ for all $x \in \mathcal{C} \setminus X_{i_0}$ and $y \in \mathcal{C}$. Consider $\nu \in r - \delta P$. Let $\nu' = \nu + d$ such that $\nu' \in r'$. If $-r'_{0,x} + \delta \prec^{r'}_{0,x} \nu'(x)$ for all $x \in \mathcal{C}$, then $\nu' \in r' - \delta Q^1$ since $(Q^1)_{x,0} = 0$ and $P_{x,y} \geq Q^1_{x,y}$ for all $x, y \in \mathcal{C}$.

Otherwise, we have $\nu'(y') \leq -r'_{0,y'} + \delta Q^1_{0,y'}$ for some $y' \in \mathcal{C}$, so $\nu'(i_0) \leq -r'_{0,i_0} + \delta Q^1_{0,y'}$ since $\mathsf{frac}(\nu'(i_0)) \leq \mathsf{frac}(\nu'(y'))$. This means that $\nu'(i_0) \prec^{r'}_{i_0,0} r'_{i_0,0} - \delta Q^2_{x,0}$ for small enough $\delta > 0$. For all $x \in \mathcal{C} \setminus X_{i_0}$, we have $\nu'(x) - \nu'(i_0) \prec^{r'}_{x,i_0} r'_{x,i_0} - \delta P_{x,i_0}$. This implies, by the above constraint on $\nu'(i_0)$ that $\nu'(x) \prec^{r'}_{x,0} r'_{x,0} - \delta(P_{x,i_0} + Q^1_{0,y'})$. This implies $\nu'(x) \prec^{r'}_{x,0} r'_{x,0} - \delta Q^2_{x,0}$ since $P_{x,i_0} \geq Q^1_{0,y'} + Q^2_{x,0}$. We are done since $Q^2_{0,x} = 0$ for all $x \in \mathcal{C}$.

It is clear that from ϵ-tight valuations in $r - \delta P$, one can delay to $(\epsilon + \epsilon')$-tight valuations both in $r' - \delta Q^1$ and $r' - \delta Q^2$, for any $\epsilon' > 0$.

□

We can now generalize the abstract game to turn-based timed games, and prove the main theorem for these games.

Let us fix a timed game $\mathcal{A} = (\mathcal{L}_C \cup \mathcal{L}_P, \mathcal{C}, \ell_0, E)$. We define $\mathcal{RG}(\mathcal{A})$ as follows. The states of $\mathcal{RG}(\mathcal{A})$ is again given as a set of square nodes (ℓ, r, S_r), where $\ell \in \mathcal{L}_C$ and $\langle r, S_r\rangle$ is a well constrained region. The diamond nodes are now either of the form (ℓ, r, S_r, e) with $\ell \in \mathcal{L}_C$ and e is an edge leaving ℓ, or (ℓ, r, S_r) with $\ell \in \mathcal{L}_P$. As before, square nodes belong to Controller, and diamond nodes belong to Perturbator. The edges leaving square nodes and the diamond nodes of the form (ℓ, r, S_r, e) are defined as for timed automata. There is an edge from a diamond node (ℓ, r, S_r) to (ℓ', t, S_t), if there is an edge $e = (\ell, g, R, \ell')$ in $\mathcal{A}$, and the following conditions hold:

i) There is a region s with $r \lessdot^* s$ and $t = s[R \leftarrow 0]$,

ii) Let $m \in \{1, 2\}$ and (s, S^i) for $i = \{1, m\}$ be the constrained region(s) given by Lemma 5.3 applied to $\langle r, S_r\rangle$. There exists $i \in \{1, 2\}$ such that for all SMs P, $P \leq S_t$ iff there exists an SM $Q \leq S^i$ with $(s', Q) = \mathsf{Unreset}_R((t, P))$, for some $\mathsf{s'} = \mathsf{s}$.

Notice that (t, S_t) can be computed in polynomial time thanks to Lemmas 5.3 and 3.12. Also, all constrained regions of the square nodes admit tight valuations by Lemma 5.3 and Corollary 4.8.

Theorem 2.3, for turn-based timed games, follows from the following proposition.

Proposition 5.4. *For any turn-based timed game $\mathcal{A}$, Controller has a winning strategy in $\mathcal{RG}(\mathcal{A})$ if, and only if there exists $\delta_0 > 0$ such that Controller wins $\mathcal{G}_\delta(\mathcal{A})$ for all $\delta \in [0, \delta_0]$.*

Proof. Assume that Controller wins $\mathcal{RG}(\mathcal{A})$. As in the proof of Proposition 4.1, we assign to nodes $n = (\ell, r, S_r)$ in $\mathcal{RG}(\mathcal{A})$, shrunk DBMs (r', P_n) with $\mathsf{r'} = \mathsf{r}$, describing the set of winning states from the corresponding locations in $\mathcal{G}_\delta(\mathcal{A})$. The construction and the proof are the same for square nodes. If n is a diamond node of the form (ℓ, r, S_r), then fix any s with $r \lessdot^* s$ and an edge $e = (\ell, g, R, \ell')$ from n with $s \subseteq g$. Assume, by induction hypothesis, that we have shrunk DBMs (t, P_{n_1}) and (t, P_{n_2}) describing winning states from the successors $n_1 = (\ell', t, S^1_t)$ and $n_2 = (\ell', t, S^2_t)$ of n through the edge e, where $t = s[R \leftarrow 0]$ (Assume $n_1 = n_2$ if there is only one such successor). Consider (s, S^1) and (s, S^2) given by Lemma 5.3 applied to $\langle r, S_r\rangle$ (assume again that $S^1 = S^2$ if there is only one such constrained region). By construction, for each $i \in \{1, 2\}$, $\langle t, S^i_t\rangle$ is defined by Lemma 3.12, such that for all SMs P_{n_i}, $P_{n_i} \leq S^i_t$ iff there exists SM $Q^i \leq S^i$ with $(s_i, Q^i) = \mathsf{Unreset}_R((t, P_{n_i}))$, for some $\mathsf{s_i} = \mathsf{s}$ and $s \subseteq s_i$. Then, by Lemma 5.3, there exists P^s_e such that

$$\mathsf{Post}_{\mathsf{time}}((r^s_e, P^s_e)) \cap s \subseteq \bigcup_{i=1}^{m} (s_i, Q_i),$$

for some $\mathsf{r}_\mathsf{e}^\mathsf{s} = \mathsf{r}$ with $r \subseteq r_e^s$. Here, (r_e^s, P_e^s) describes a set of winning states from location ℓ assuming Perturbator chooses the edge e and delays anywhere in the region s. We define (r_e, P_e) as the intersection of all (r_e^s, P_e^s) with $r \preccurlyeq^* s$, and define (r', P_n) as the intersection of all (r_e, P_e) for all edges e leaving n. This concludes one direction of the proof.

Now, assume that Perturbator wins $\mathcal{RG}(\mathcal{A})$. The proof of Proposition 4.1 is based on the construction of a strategy for $\mathcal{G}_\delta(\mathcal{A})$, that ensured that the valuation is always ϵ-tight inside any visited constrained region. When one projects the play in $\mathcal{RG}(\mathcal{A})$, this property allows Perturbator to choose valuations in $\mathcal{G}_\delta(\mathcal{A})$ belonging to any successor it would choose in $\mathcal{RG}(\mathcal{A})$ in order to win. Thus, Perturbator's strategy simply consists in following in $\mathcal{G}_\delta(\mathcal{A})$ its winning strategy in $\mathcal{RG}(\mathcal{A})$. The same proof carries over to timed games since Lemma 5.3 shows that given a ϵ-tight valuation of $\langle r, S_r\rangle$, there exist delays that lead to $(\epsilon + \epsilon')$-tight valuations inside $\langle s, S^1\rangle$ and $\langle s, S^2\rangle$ for any $\epsilon' > 0$. □

6. Hardness Result

The hardness result is shown by reduction from the halting problem in linear-bounded alternating Turing machines. These are Turing machines whose computations do not use more tape cells than the size of the input, and checking whether there is an accepting computation is EXPTIME-complete. Each transition is either an instruction, a disjunction, or a conjunction. Disjunction and conjunction transitions are given with pairs of successor states. Given an input, the set of accepting states are defined inductively as follows. There is a distinguished accepting state which is accepting by definition. A state with an instruction is accepting if its successor is. A disjunctive state accepts if, and only if *one* of the successors is accepting, and a conjunctive state accepts if, and only if *both* of its successors are accepting. The acceptance condition can thus be formalized by a two-player game played on the Turing machine; the first player controls the disjunctive states and instructions, and the second player controls the conjunctive states.

Proposition 6.1. *The robust reachability problem is* EXPTIME*-hard.*

The reduction has the following idea. We adapt a standard encoding of linearly bounded Turing machines for timed automata in order to take care of imprecisions. In addition, the gadget of Figure 14 allows the simulation of the alternation, thanks to the ability of distinguishing positive and negative perturbations. We now give the detailed proofs.

Proof. We use a reduction from the halting problem in linear-bounded alternating Turing machines over a two-letter alphabet $\Sigma = \{a, b\}$. Let $\mathcal{M}$ be such a Turing machine, and write n for the bound on the tape length. We assume w.l.o.g. that $n \geq 3$ and that instructions are of the form:

- (disjunction) $\delta(q) = q' \vee q''$
- (conjunction) $\delta(q) = q' \wedge q''$
- (instruction) $\delta(q) = (\gamma, \gamma', \mathsf{dir}, q')$ where $\gamma, \gamma' \in \{a, b\}$ and $\mathsf{dir} \in \{\leftarrow, \rightarrow\}$. Such a transition reads γ in the current cell, writes γ' and follows direction given by dir.

Our encoding of $\mathcal{M}$ uses the set of $n + 2$ clocks $X = \{x_i \mid i = 1 \dots n\} \cup \{y, z\}$. The content of cell i is encoded by clock x_i: it is an a if the value of clock x_i is $n - i$, and a b if the value of clock x_i is bounded below by $2n - i$. Due to robustness concerns, these guards will be relaxed a bit in the construction.

We assume the content of the tape is represented by a word w over alphabet Σ of length n. Let $k \in \mathbb{N}$ and $\epsilon \geq 0$. We say that a valuation v over X is a *k-shift encoding of w with precision ϵ* whenever $v(y) = v(z) = 0$, and for every $1 \leq i \leq n$:

- $w_i = a$ iff $-\epsilon \leq v(x_i) - (n-i) - k \leq \epsilon$
- $w_i = b$ iff $-\epsilon \leq v(x_i) - (2n-i) - k$

We encode the instructions as follows.

► **Regular instruction.** A transition $\delta(q) = (\gamma, \gamma', \mathsf{dir}, q')$ is mimicked thanks to modules $\mathsf{instr}^{i,k}_{\delta(q)=(\gamma,\gamma',\mathsf{dir},q')}$ for every $1 \leq i \leq n$ and $0 \leq k < n$. Such a module is depicted on Figure 13: it is a sequence of n modules, the ith one being of a special shape. The initial state is $(q, i, k, 1)$. We write I for the interval $[n-1, n+1]$ and I' for the interval $[2n-1, +\infty)$, and adopt the notation $S + k = \{b + k \mid b \in S\}$ for any set S.

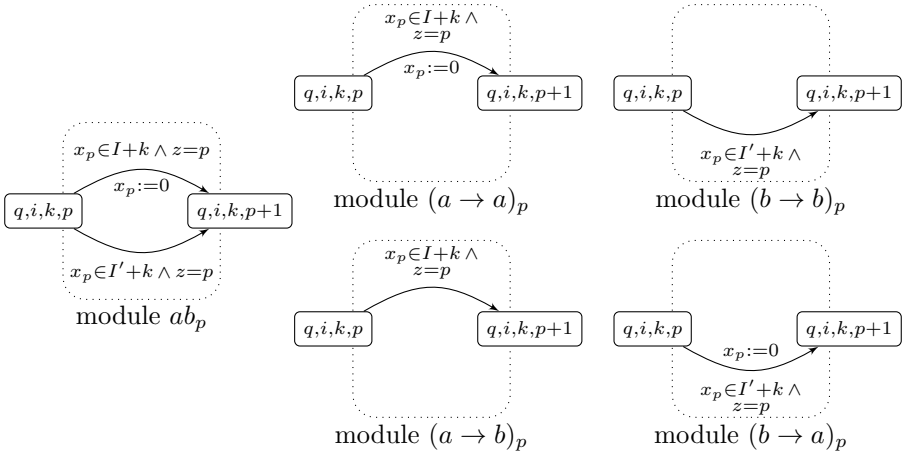


Figure 12: Intermediary modules: when traversing module ab_p, either clock x_p belongs to $I + k$ while $z = p$, corresponding to an a at position p. Clock x_p is then reset, so that position p still contains an a; or clock x_p is in $I' + k$ when $z = p$, encoding a b at position p. In that case, clock x_p is not reset, and the content of cell p is preserved. Using similar ideas, modules $(\gamma \to \gamma')_p$, with $\gamma, \gamma' \in \{a, b\}$, check that cell p initially contains γ, and replace it with γ'.

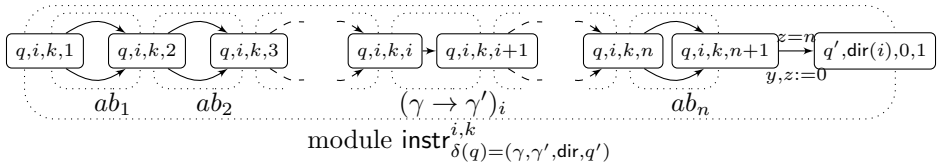


Figure 13: Module $\mathsf{instr}^{i,k}_{\delta(q)=(\gamma,\gamma',\mathsf{dir},q')}$ for the simulation of the regular instruction $\delta(q) = (\gamma, \gamma', \mathsf{dir}, q')$, and its constituent submodules. If $\mathsf{dir} = \rightarrow$, $\mathsf{dir}(i) = i + 1$ if $1 \leq i < n$, and undefined otherwise. If $\mathsf{dir} = \leftarrow$, $\mathsf{dir}(i) = i - 1$ if $1 < i \leq n$, and undefined otherwise.

The correctness of this module is given by the following lemma:

Lemma 6.2. *Let $0 \leq \epsilon \leq 1$ and $0 \leq \delta \leq 1$. Let $w \in \Sigma^n$ such that $w_i = \gamma$. Assume module $\mathsf{instr}^{i,k}_{\delta(q)=(\gamma,\gamma',\mathsf{dir},q')}$ is entered with valuation v which is a k-shift encoding of w with precision ϵ. The Controller has a unique strategy in this module, and for every response of the δ-perturbator, the valuation v' when leaving the module is a 0-shift encoding of $w[w_i \leftarrow \gamma']$ (the word obtained from w by replacing w_i with γ') with precision 2δ.*

Proof. There is a unique strategy for Controller, which is to play from state (q, i, k, j) when z reaches j with the unique possible transition: if initially $|v(x_j) - (n-j) - k| \leq \epsilon$, then he will choose the top-most transition, and if initially $v(x_j) \geq 2n - j + k - \epsilon$, then it will choose the bottom-most transition. Indeed note that in the first case, since $\epsilon \leq 1$, the value of clock x_j when z reaches j lies within $I + k$. When clock x_j is reset clock z is almost j (more precisely it lies between $j - \delta$ and $j + \delta$), whereas clock z is almost n when leaving the module (more precisely it lies between $n - \delta$ and $n + \delta$). In the second case, the value of x_j is increased by almost n. This straightforwardly implies the mentioned property. □

Remark 6.3. *Note that if the module above is entered while $w_i \neq \gamma$, then it reaches a deadlock. This could be avoided using extra transitions to a sink state.*

▶ **Conjunction.** $\delta(q) = q' \wedge q''$ is mimicked thanks to modules $\mathsf{conj}^{i,k}_{\delta(q)=q'\wedge q''}$ (for every $1 \leq i \leq n$ and $0 \leq k < n$) on Figure 14. As in the previous module, the Controller has no other choice than selecting the next transition when the constraint is satisfied. For the first transition, the δ-Perturbator can choose to do it a bit earlier, or a bit later, and depending on this, the controller will next choose either $y = 1 \wedge z \leq 2$ or $y = 1 \wedge z > 2$.

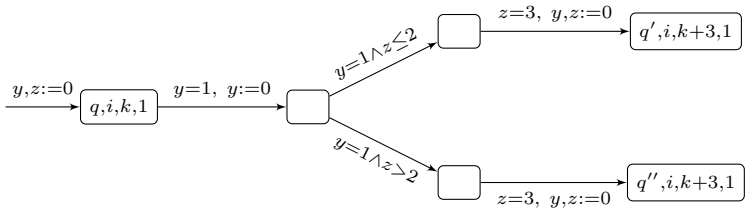


Figure 14: Module $\mathsf{conj}^{i,k}_{\delta(q)=q'\wedge q''}$ for conjunctive transition

Lemma 6.4. *Let $0 \leq \epsilon \leq 1$ and $0 \leq \delta \leq 1$. Let $w \in \Sigma^n$. Assume module $\mathsf{conj}^{i,k}_{\delta(q)=q'\wedge q''}$ is entered with valuation v which is a k-shift encoding of w with precision ϵ. The Controller has a unique strategy in this module, and the δ-Perturbator can choose to reach either $(q', i, k+3, 1)$ or $(q'', i, k+3, 1)$. In both cases, the valuation v' when leaving the module is a $(k+3)$-shift encoding of w with precision $\epsilon + \delta$.*

Proof. The Controller has no other choice than satisfying the next constraint of the next transition. On the other hand, the δ-Perturbator can either choose to postpone the first transition, or to fire it earlier. In the first case, the Controller has then to choose the bottom-most transition, and in the second case, the Controller has to choose the top-most transition. Globally the values of the clocks are increased by 3 (plus or minus δ), which yields the expected property. □

▶ **Disjunction.** $\delta(q) = q' \vee q''$ is mimicked thanks to modules $\mathsf{disj}^{i,k}_{\delta(q)=q'\vee q''}$ (for every $1 \leq i \leq n$ and $0 \leq k < n$) on Figure 15.

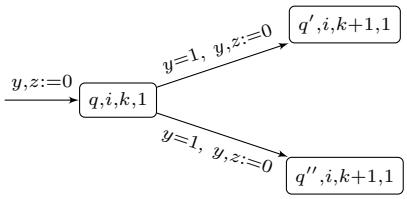


Figure 15: Module $\mathsf{disj}^{i,k}_{\delta(q)=q'\wedge q''}$ for disjunctive transition

Lemma 6.5. *Let $\epsilon \geq 0$ and $\delta \geq 0$. Let $w \in \Sigma^n$. Assume module $\mathsf{disj}^{i,k}_{\delta(q)=q'\vee q''}$ is entered with valuation v which is a k-shift encoding of w with precision ϵ. The Controller can choose to reach either $(q', i, k+1, 1)$ or $(q'', i, k+1, 1)$. In both cases, the valuation v' when leaving the module is a $(k+1)$-shift encoding of w with precision $\epsilon + \delta$.*

Proof. Similar to the previous proof. □

▶ **Reset module.** We fix an integer $0 \leq k < n$. Shifts encodings accumulate when stacking disjunctive and conjunctive instructions. We present a module $\mathsf{reset}^{i,k}_q$ which resets the shift from state q, position i.

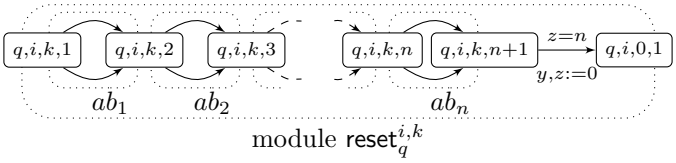


Figure 16: Module $\mathsf{reset}^{i,k}_q$ which resets the shift in the encoding.

Lemma 6.6. *Let $0 \leq \epsilon \leq 1$ and $0 \leq \delta \leq 1$. Let $w \in \Sigma^n$. Assume module $\mathsf{reset}_{q,i,k}$ is entered with valuation v which is a k-shift encoding of w with precision ϵ. The Controller has a unique strategy in this module, and for every response of the δ-Perturbator, the valuation v' when leaving the module is a 0-shift encoding of w with precision 2δ.*

Proof. This proof is similar to the proof of Lemma 6.2. □

▶ **Global reduction.** It remains to glue all the modules together. An easy solution is to apply the reset module after each conjunctive or disjunctive instruction (though there are some more thrifty solutions). The reset module allows both to reset the shift and to reinitialize the imprecision. We write $\mathcal{A}$ for the resulting timed automaton. The halting location of the Turing machine is called final in $\mathcal{A}$.

One can easily check that in $\mathcal{A}$, letting $0 \leq \delta < \frac{1}{2}$, the Controller has a winning strategy against the δ-Perturbator to reach location final if, and only if, the Turing machine $\mathcal{M}$ halts. □

7. Conclusion

We considered a game-based approach to robust reachability in timed automata, by modelling the semantics as a game between the controller and its environment. We proved that a bound on the imprecisions, and a corresponding robust strategy for reachability objectives in turn-based timed games can be synthesized, and that the existence of such a bound and a strategy is EXPTIME-complete. The problem is thus harder than classical reachability [4]. A similar semantics was studied in [15], and our result partially answers to an open problem posed there for parity objectives on general timed games.

A natural continuation of this work would be to look at zone-based on-the-fly algorithms, as for usual timed games in [13]. Note that solving usual timed games is also an EXPTIME-complete problem and on-the-fly algorithms allowed efficient implementations.

Developing algorithms for safety objectives, that is, ensuring infinite executions that avoid some given state would require the use of different techniques and seems to be a challenging problem. In fact, in that setting, one has to deal with the accumulation of the imprecisions over infinite runs, which cannot be avoided by adjusting δ.

We are currently studying a different model in which the controller is required to choose delays for which the guard of the chosen edge is satisfied no matter what the perturbation is, which is the parameterized version of the framework of [15]. We believe both semantics are interesting and allow different ways of incorporating perturbations in the model. Another interesting perturbation model is to assume that imprecisions are probabilistic, that is, once Controller has chosen a delay d, the actual delay is chosen from $[d - \delta, d + \delta]$ following a probability distribution.

8. References

[1] Yasmina Abdeddaïm, Eugene Asarin, and Oded Maler. Scheduling with timed automata. *Theoretical Computer Science*, 354(2):272–300, 2006.

[2] Parosh Aziz Abdulla, Pavel Krčál, and Wang Yi. Sampled universality of timed automata. In *Proc. 10th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS'07)*, LNCS 4423, p. 2–16. Springer, 2007.

[3] Parosh Aziz Abdulla, Pavel Krčál, and Wang Yi. Sampled semantics of timed automata. *Logical Methods in Computer Science*, 6(3:14), 2010.

[4] Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

[5] Rajeev Alur, Thomas A. Henzinger, and Moshe Y. Vardi. Parametric real-time reasoning. In *Proc. 25th Annual ACM Symposium on Theory of Computing (STOC'93)*, p. 592–601. ACM, 1993.

[6] Eugene Asarin, Oded Maler, Amir Pnueli, and Joseph Sifakis. Controller synthesis for timed automata. In *Proc. IFAC Symposium on System Structure and Control*, p. 469–474. Elsevier Science, 1998.

[7] Johan Bengtsson and Wang Yi. Timed automata: Semantics, algorithms and tools. In *Proc. 4th Advanced Course on Petri Nets (ACPN'03)*, LNCS 3098, p. 87–124. Springer, 2004.

[8] Patricia Bouyer, Kim G. Larsen, Nicolas Markey, Ocan Sankur, and Claus Thrane. Timed automata can always be made implementable. In *Proc. 22nd International Conference on Concurrency Theory (CONCUR'11)*, LNCS 6901, p. 76–91. Springer, 2011.

[9] Patricia Bouyer, Nicolas Markey, and Pierre-Alain Reynier. Robust model-checking of timed automata. In *Proc. 7th Latin American Symposium on Theoretical Informatics (LATIN'06)*, LNCS 3887, p. 238–249. Springer, 2006.

[10] Patricia Bouyer, Nicolas Markey, and Pierre-Alain Reynier. Robust analysis of timed automata via channel machines. In *Proc. 11th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS'08)*, LNCS 4962, p. 157–171. Springer, 2008.

[11] Patricia Bouyer, Nicolas Markey, and Ocan Sankur. Robust model-checking of timed automata via pumping in channel machines. In *Proc. 9th International Conference on Formal Modeling and Analysis of Timed Systems (FORMATS'11)*, LNCS 6919, p. 97–112. Springer, 2011.

[12] Patricia Bouyer, Nicolas Markey, and Ocan Sankur. Robust reachability in timed automata: A game-based approach. In *Proc. 39th International Colloquium on Automata, Languages and Programming (ICALP'12)*, LNCS 7392, p. 128–140. Springer, 2012.

[13] Franck Cassez, Alexandre David, Emmanuel Fleury, Kim G. Larsen, and Didier Lime. Efficient on-the-fly algorithms for the analysis of timed games. In *Proc. 16th International Conference on Concurrency Theory (CONCUR'05)*, LNCS 3653, p. 66–80. Springer, 2005.

[14] Franck Cassez, Thomas A. Henzinger, and Jean-François Raskin. A comparison of control problems for timed and hybrid systems. In *Proc. 5th International Workshop on Hybrid Systems: Computation and Control (HSCC'02)*, LNCS 2289, p. 134–148. Springer, 2002.

[15] Krishnendu Chatterjee, Thomas A. Henzinger, and Vinayak S. Prabhu. Timed parity games: Complexity and robustness. *Logical Methods in Computer Science*, 7(4:8), 2010.

[16] Conrado Daws and Piotr Kordy. Symbolic robustness analysis of timed automata. In *Proc. 4th International Conference on Formal Modeling and Analysis of Timed Systems (FORMATS'06)*, LNCS 4202, p. 143–155, 2006.

[17] Martin De Wulf, Laurent Doyen, Nicolas Markey, and Jean-François Raskin. Robust safety of timed automata. *Formal Methods in System Design*, 33(1–3):45–84, 2008.

[18] Martin De Wulf, Laurent Doyen, and Jean-François Raskin. Almost ASAP semantics: From timed models to timed implementations. *Formal Aspects of Computing*, 17(3):319–341, 2005.

[19] Martin Fränzle. Analysis of hybrid systems: An ounce of realism can save an infinity of states. In *Proc. 13th International Workshop on Computer Science Logic (CSL'99)*, LNCS 1683, p. 126–139. Springer, 1999.

[20] Ronald L. Graham. Bounds on multiprocessing timing anomalies. *SIAM Journal on Applied Mathematics*, 17(2):416–429, 1969.

[21] Vineet Gupta, Thomas A. Henzinger, and Radha Jagadeesan. Robust timed automata. In *Proc. International Workshop on Hybrid and Real-Time Systems (HART'97)*, LNCS 1201, p. 331–345. Springer, 1997.

[22] Thomas A. Henzinger and Jean-François Raskin. Robust undecidability of timed and hybrid systems. In *Proc. 3rd International Workshop on Hybrid Systems: Computation and Control (HSCC'00)*, LNCS 1790, p. 145–159. Springer, 2000.

[23] Thomas A. Henzinger and Joseph Sifakis. The embedded systems design challenge. In *Proc. 14th International Symposium on Formal Methods (FM'06)*, LNCS 4085, p. 1–15. Springer, 2006.

[24] Frédéric Herbreteau, Dileep Kini, B. Srivathsan, and Igor Walukiewicz. Using non-convex approximations for efficient analysis of timed automata. In *Proc. 30th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'11)*, LIPIcs 13, p. 78–89. Leibniz-Zentrum für Informatik, 2011.

[25] Thomas Hune, Judi Romijn, Mariëlle Stoelinga, and Vaandrager Frits W. Linear parametric model-checking of timed automata. In *Proc. 7th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'01)*, LNCS 2031, p. 189–203. Springer, 2001.

[26] Rémi Jaubert and Pierre-Alain Reynier. Quantitative robustness analysis of flat timed automata. In *Proc. 14th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS'11)*, LNCS 6604, p. 229–244. Springer, 2011.

[27] Aleksandra Jovanovic, Sébastien Faucou, Didier Lime, and Olivier Henri Roux. Real-time control with parametric timed reachability games. In *11th International Workshop on Discrete Event Systems (WODES'12)*, p. 323–330, 2012.

[28] Hermann Kopetz. *Real-Time Systems: Design Principles for Distributed Embedded Applications*. Springer, 2011.

[29] Kim G. Larsen, Axel Legay, Louis-Marie Traonouez, and Andrzej Wasowski. Robust specification of real time components. In *Proc. 9th International Conference on Formal Modeling and Analysis of Timed Systems (FORMATS'11)*, LNCS 6919, p. 129–144. Springer, 2011.

[30] Kim G. Larsen, Paul Pettersson, and Wang Yi. UPPAAL in a nutshell. *International Journal on Software Tools for Technology Transfer*, 1(1–2):134–152, 1997.

[31] Nicolas Markey. Robustness in real-time systems. In *Proc. 6th IEEE International Symposium on Industrial Embedded Systems (SIES'11)*, p. 28–34. IEEE Computer Society Press, 2011.

[32] Anuj Puri. Dynamical properties of timed automata. *Discrete Event Dynamic Systems*, 10(1-2):87–113, 2000.

[33] Jan Reineke, Björn Wachter, Stephan Thesing, Reinhard Wilhelm, Ilia Polian, Jochen Eisinger, and Bernd Becker. A definition and classification of timing anomalies. In *Proc. 6th International Workshop on Worst-Case Execution Time Analysis (WCET'06)*, 2006.

[34] Ocan Sankur. Untimed language preservation in timed systems. In *Proc. 36th International Symposium on Mathematical Foundations of Computer Science (MFCS'11)*, LNCS 6907, p. 556–567. Springer, 2011.

[35] Ocan Sankur. Shrinktech: A tool for the robustness analysis of timed automata. In *Proc. 25th International Conference on Computer Aided Verification (CAV'13)*, LNCS 8044, p. 1006–1012. Springer, 2013.

[36] Ocan Sankur, Patricia Bouyer, and Nicolas Markey. Shrinking timed automata. In *Proc. 30th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'11)*, LIPIcs 13, p. 375–386. Leibniz-Zentrum für Informatik, 2011.

[37] Ocan Sankur, Patricia Bouyer, Nicolas Markey, and Pierre-Alain Reynier. Robust controller synthesis in timed automata. In *Proc. 24th International Conference on Concurrency Theory (CONCUR'13)*, LNCS 8052, p. 546–560. Springer, 2013.

[38] Sergio Yovine. Kronos: A verification tool for real-time systems. *International Journal on Software Tools for Technology Transfer*, 1(1–2):123–133, 1997.

Appendix A. Modelling of our scheduling example

The scheduling problem described in Fig. 1 contains two processes: Process 1 executing A and B, and Process 2 executing C, D, and E. Both processes need the same resources (machines M_1 and M_2), which can only be used exclusively. This system can be modelled by a network of timed automata, as follows. We define a timed automaton $\mathcal{P}_1$ corresponding to Process 1, and $\mathcal{P}_2$ corresponding to Process 2. A third timed automaton $\mathcal{M}$ will make sure the mutual exclusion for both machines: no more than one task is executed on a machine at any time. Moreover, we will also use $\mathcal{M}$ in order to impose a greedy scheduling: this automaton will block time if some task is waiting for execution on a machine that is free (more details follow below). Each automaton $\mathcal{P}_i$ makes sure that its tasks are executed respecting the dependences and timing constraints. All three timed automata are given in figures below. Note that we use additional features such as invariants, boolean variables and synchronization, which we didn't define, but it can be seen that an equivalent system can be defined as an ordinary timed automaton.

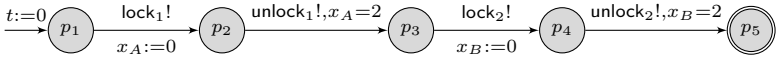


Figure A.17: Timed automaton $\mathcal{P}_1$ modeling Process 1.

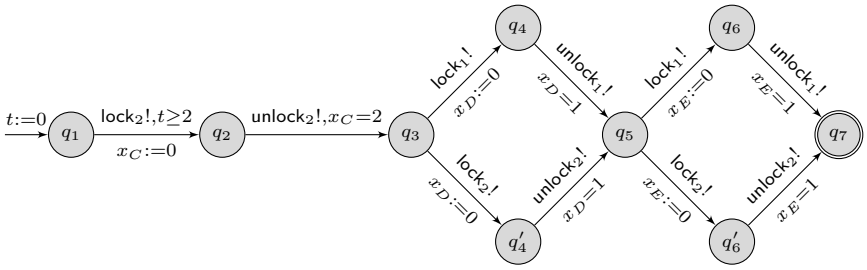


Figure A.18: Timed automaton $\mathcal{P}_2$ modeling Process 2.

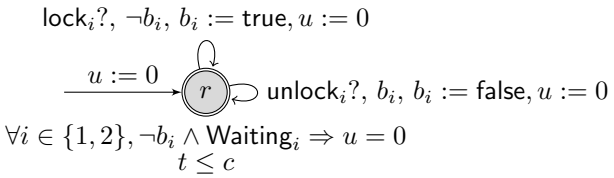
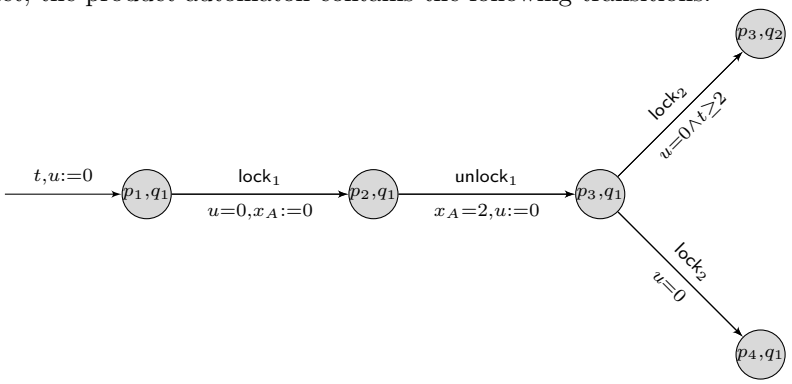


Figure A.19: Timed automaton $\mathcal{M}$ ensuring mutual exclusion and greedy scheduling.

We use synchronization on labels lock_i: the meaning of $\mathsf{lock}_i!$ is that the corresponding edge can only be taken if another component takes an edge labeled with $\mathsf{lock}_i?$, and vice versa. For instance, in automaton $\mathcal{P}_1$, the task A can be executed on M_1 (first edge), only if lock_1 can be taken. The lock is released upon the termination of the task (second edge). Automaton $\mathcal{M}$ has two boolean variables b_1, b_2, where b_i is set to true if machine M_i is busy. The two self-loops are defined for $i = 1, 2$. For instance, the automaton $\mathcal{M}$ synchronizes with action lock_i only if b_i is not set to true. It has the invariants $\forall i \in \{1, 2\}, \neg b_i \wedge \mathsf{Waiting}_i \Rightarrow u = 0$ and $t \leq c$. The first one ensures greedy

scheduling, while the second one blocks the time above c time units. [2] Here, $\mathsf{Waiting}_i$ is a formula stating that some task is ready for execution on machine i. We omit the definition of $\mathsf{Waiting}_i$ here but it can be easily defined once we add a boolean variable for each task that is true if, and only if the task has finished. Hence, automaton $\mathcal{M}$ blocks time in the whole system if some task is ready for execution on a machine that is free.

Now, the reachability of the location (p_5, q_7, r) in the product of $\mathcal{P}_1$, $\mathcal{P}_2$ and $\mathcal{M}$ means that the tasks are schedulable in c time units. For instance, this location is reachable for $c = 6$ in the usual semantics as shown in Section 2.2. A strategy that ensures the reachability of this location gives a greedy scheduler. However, this location is not robustly reachable for $c = 6$, but it is only for $c = 8$. In fact, the product automaton contains the following transitions.



The automaton must indeed start by scheduling the task A on M_1. At state (p_3, q_1), it can choose between scheduling C (upper transition) and scheduling B (lower transition). The former one is always enabled in the usual semantics, and yields a total scheduling time of 6. But in the game semantics, the lower transition may be the only option if we have a negative perturbation during the transition $(p_2, q_1) \to (p_3, q_1)$. This can only yield a scheduling time of 8 as shown before.

[2] Note that we do not really need to consider invariants. We could as well add this constraint as guards in the edges of $\mathcal{M}$.