Robust Reachability in Timed Automata: A Game-based Approach

Patricia Bouyer, Nicolas Markey, Ocan Sankur

May 2012

Research report LSV-12-07

Laboratoire Spécification & Vérification

École Normale Supérieure de Cachan
61, avenue du Président Wilson
94235 Cachan Cedex France

Robust Reachability in Timed Automata: A Game-based Approach *

Patricia Bouyer, Nicolas Markey, and Ocan Sankur

LSV, CNRS & ENS Cachan, France.
{bouyer,markey,sankur}@lsv.ens-cachan.fr

Abstract. Reachability checking is one of the most basic problems in verification. By solving this problem, one synthesizes 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 a timed automaton; with "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.

1 Introduction

Timed automata [2] 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 [2], and efficient algorithms have then been developed and implemented [12, 17].

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 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 [10, 11].

* This work has been partly supported by project ImpRo (ANR-10-BLAN-0317)

In this work, we are interested in the synthesis of robust strategies in timed automata for reachability objectives, taking into account response times and imprecisions. We propose to model the problem as a game between a controller (that will guide the system) and its environment. In our semantics, which is parameterized by some $0 < \delta_P \leq \delta_R$, the controller chooses to delay an amount $d \geq \delta_R$, and the system delays d', where d' is chosen by the environment satisfying $|d - d'| \leq \delta_P$. We say that a given location is *robustly reachable* if there exist parameters $0 < \delta_P \leq \delta_R$ such that the controller has a winning strategy ensuring that the location is reached against any strategy of the environment. If δ_P and δ_R are fixed, this can be solved using techniques from control theory [3]. However δ_P, δ_R are better seen as parameters here, representing imprecisions in the implementation of the system (they may depend on the digital platform on which the system is implemented), and whose values may not be available in the design phase. 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 (by adapting the shrink operator in Section 3).

Note that this semantics was studied in [5] for timed games with *fixed* parameters, where the parameterized version was presented as a challenging open problem. We solve this problem for reachability objectives in timed automata: 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 (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 recently [16]. 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 (see Section 4).

2 Robust reachability in timed automata

2.1 Timed automata and robust reachability

Given a finite set of clocks $\mathcal{C}$, we call *valuations* the elements of $\mathbb{R}^{\mathcal{C}}_{\geq 0}$. 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 $\mathbf{0}$ 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 $v(x)$.

Definition 1 ([2]). *A* timed automaton $\mathcal{A}$ *is a tuple* $(\mathcal{L}, \mathcal{C}, \ell_0, E)$*, consisting of finite sets* $\mathcal{L}$ *of locations,* $\mathcal{C}$ *of clocks,* $E \subseteq \mathcal{L} \times \Phi_{\mathcal{C}} \times 2^{\mathcal{C}} \times \mathcal{L}$ *of edges, and where* $\ell_0 \in \mathcal{L}$ *is the initial location. An edge* $e = (\ell, g, R, \ell')$ *is also written as* $\ell \xrightarrow{g,R} \ell'$*.*

Standard semantics of timed automata is usually given as a timed transition system. To capture robustness, we define the semantics as a game where perturbations in delays are uncontrollable. Given a timed automaton $\mathcal{A} = (\mathcal{L}, \mathcal{C}, \ell_0, E)$ and $\delta > 0$, we define the *perturbation game* of $\mathcal{A}$ w.r.t. δ as a two-player turn-based timed game $\mathcal{G}_\delta(\mathcal{A})$ between players *Controller* and *Perturbator*. The state space of $\mathcal{G}_\delta(\mathcal{A})$ is partitioned into $V_C \cup V_P$ where $V_C = \mathcal{L} \times \mathbb{R}^{\mathcal{C}}_{\geq 0}$ is the set of states that belong to Controller and $V_P = \mathcal{L} \times \mathbb{R}^{\mathcal{C}}_{\geq 0} \times \mathbb{R}_{\geq 0} \times E$ is the set of states that belong to Perturbator. The initial state is $(\ell_0, \mathbf{0})$ 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 $\epsilon \in [-\delta, \delta]$.

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 $(\ell_0, \mathbf{0})$. 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 (ℓ, v, d, e)). 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 1 (Parameterized Robust Reachability). Given a timed automaton $\mathcal{A}$ 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.

Notice that we are interested in the parameterized problem: δ is not fixed in advance. For fixed parameter, the problem can be formulated as a usual timed game, see [5]. Our main result is the decidability of this parameterized 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:

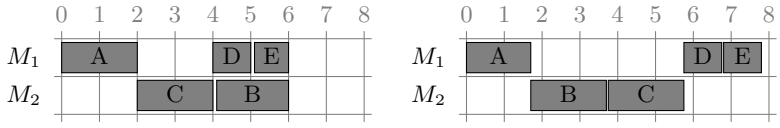
Theorem 2. *Parameterized robust reachability is* EXPTIME*-complete.*

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, so 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. 4). So, Controller must also be able to win from these new 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).

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 [8].

Consider the scheduling problem described in Fig. 1, inspired by [15]. 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 guarantee a greedy scheduling policy 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.



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

Fig. 1. Consider tasks A, B, C of duration 2 and D, E of duration 1. 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, task C cannot be scheduled before 2 time units (which could be modelled using an extra task). 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.

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.

2.3 Related work: robustness in timed automata and games

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, see [13] for a survey.

The approach initiated in [14, 7, 6] is the closest to our framework/proposition. It consists in *enlarging* all clocks 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. This 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.

Another work that is close to ours is that of [5]. The authors consider general two-player (concurrent) games with a fixed lower bound on delays, where chosen delays can be changed by some fixed value δ. It is then shown that winning strategies can be synthesized: In fact, when δ is fixed, the semantics can simply 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 partially solve this open problem here, under the assumption that there is a single player with a reachability objective. The extension to two-player games (with reachability objective) is ongoing work, and we believe the techniques presented in this paper can be used for that purpose.

Finally, [9] studies a topological and language-based approach to robustness, where (roughly) a timed word is accepted by the automaton if, and only if, one of its neighborhoods is accepted. This is not related to our formalization.

3 Shrinking DBMs

3.1 Regions, zones and DBMs

We assume familiarity with the notions of regions and zones (see [4]). 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 $(\mathbb{R} \times \{<, \leq\}) \cup \{(\infty, <)\}$. 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 $x, y \in \mathcal{C}_0$, writing $(M_{x,y}, \prec_{x,y})$ for the (x,y)-entry of M, it holds $v(x) - v(y) \prec_{x,y} M_{x,y}$ (where $v(0) = 0$). For any DBM M, let $\mathsf{G}(M)$ denote the graph over nodes $\mathcal{C}_0$, where the weight of the edge $(x,y) \in \mathcal{C}_0^2$ is $(M_{x,y}, \prec_{x,y})$. The normalization of M corresponds to assigning to each edge (x,y) the weight of the shortest path in $\mathsf{G}(M)$. We say that M is *normalized* when it is stable under normalization.

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 $X \cap (x \leq 2 - \delta)$. Then, this shrinking is backward propagated to ℓ_1: the winning states are $Y \cap (x \leq 2 - 2\delta)$, where we "shrink" Y by 2δ in order to compensate for a possible perturbation.

An important observation here is that when $\delta > 0$ is small enough, so that both $X \cap (x \leq 2 - \delta)$ and $Y \cap (x \leq 2 - 2\delta)$ are non-empty, these sets precisely describe the winning states. Thus, we have a *uniform* description of the winning states for "all small enough $\delta > 0$". We now define *shrunk DBMs*, a data structure we introduced in [16], 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

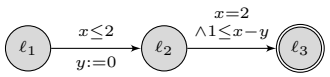


Fig. 2. Automaton $\mathcal{A}$

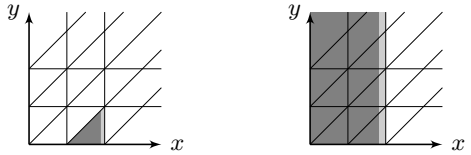


Fig. 3. Winning states in ℓ_2 (left) and in ℓ_1 (right)

shrinking matrix [16]. 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 sequel, we abusively use "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 a shrunk DBM (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 $\delta > 0$.

It was shown in [16] that when usual operations are applied on shrunk DBMs, one always obtain shrunk DBMs, whose shrinking matrices can be computed. We refer to [4, 16] for the formal definitions of these operations.

Lemma 3 ([16]). *Let $M = f(N_1, \ldots, N_k)$ be an equation between normalized DBMs $M, N_1, \ldots, N_k$, using the operators Pre_{time}, $\mathsf{Unreset}_R$, $\cap$, shrink and shrink^+ and let $P_1, \ldots, P_k$ be SMs. Then, there exists a SM Q such that (M, Q) is normalized and $(M, Q) = f\big((N_1, P_1), \ldots, (N_k, P_k)\big)$. Moreover, Q and the corresponding upper bound on δ can be computed in polynomial time.*

3.4 Shrinking constraints

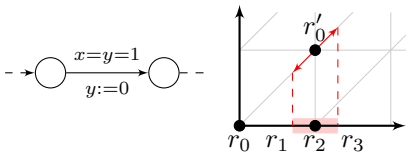
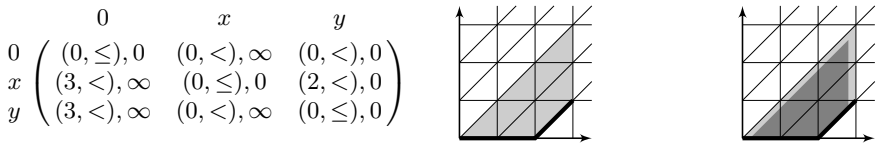


Fig. 4. Perturbing one transition

Consider a transition of a timed automaton, as depicted on the figure at right. From region r_0, the game can reach regions r_1, r_2, r_3, depending on the move of Perturbator. Therefore, in order to win, Controller needs a winning strategy from all three regions. One can then inductively look for winning strategies from these regions; this will generally require shrinking, as exemplified in Fig. 3. However, 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. In order to define the shrinkings that are useful to us, we introduce shrinking constraints.

Definition 4. *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 (not) allowed to shrink (see Fig. 5). 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



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

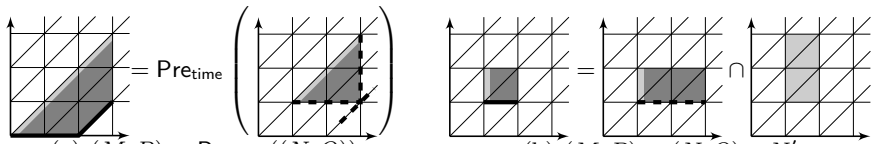
Fig. 5. Consider a zone defined by $0 < x < 3$, $0 < y < 3$, and $0 < x - y < 2$. Let the shrinking constraint S be defined by $S_{0,y} = 0$, $S_{x,y} = 0$, and $S_{z,z'} = \infty$ for other components. The resulting $\langle M, S\rangle$ is depicted on the left, as a matrix (where, for convenience, 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.

P, $P \leq S \Leftrightarrow P \leq S'$, then $S \leq S'$. One can show that any shrinking constraint can be made normalized, by a procedure similar to the normalization of DBMs. Lemma 5 shows that shrinking constraints can be propagated along operations on DBMs. This is illustrated in Fig. 6.

Lemma 5. *Let M, N, N' be normalized non-empty DBMs.*

1. *Assume that $M = \mathsf{Pre}_{\mathsf{time}}(N)$, $M = N \cap N'$, or $M = \mathsf{Unreset}_R(N)$. Then, for any normalized well shrinking constraint S for M, there exists a well shrinking constraint S' for N such that for any SM Q, the following holds: $Q \leq S'$ iff the SM P s.t. $(M, P) = \mathsf{Pre}_{\mathsf{time}}((N, Q))$ (respectively, $(M, P) = (N, Q) \cap N'$ or $(M, P) = \mathsf{Unreset}_R((N, Q))$) satisfies $P \leq S$.*
2. *Assume that $M = N \cap N'$. For any well shrinking constraint S for N, there exists a shrinking constraint S' for M such that for any SM Q, the following holds: $Q \leq S'$ iff a SM $P \leq S$ s.t. $(N, P) \cap N' \subseteq (M, Q)$. Moreover, if $(N, P) \cap N' \neq \emptyset$ for all SMs $P \leq S$, then S' is well.*

Let us comment on Fig. 6(a), 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



(a) $(M, P) = \mathsf{Pre}_{\mathsf{time}}((N, Q))$ (b) $(M, P) = (N, Q) \cap N'$

Fig. 6. The figures illustrate the first item in Lemma 5. In each case, DBMs M, N and N' are fixed and satisfy the "unshrunk" equation. The thick plain segments represent the fixed shrinking constraint S. The dashed segments represent resulting constraint S'. For any SM Q, we have $Q \leq S'$ iff there is an SM $P \leq S$ that satisfies the equation.

want to synthesize a winning strategy from a shrinking of M satisfying S, then Lemma 5 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 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 now consider constrained regions, which are constrained DBMs in which the DBM represents a region. Fig. 4 shows that if Controller plays to a region, then Perturbator can reach some of the surrounding regions, shown by the arrows. To characterize these, we define the set of *neighboring regions* of $\langle r, S \rangle$ as,

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

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 $\mathsf{neighbor}\langle r, S \rangle = \bigcup_{r' \in \mathcal{N}_{r,S}} r'$.

Lemma 6 (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.*

Constrained neighborhoods are illustrated in Fig. 7.

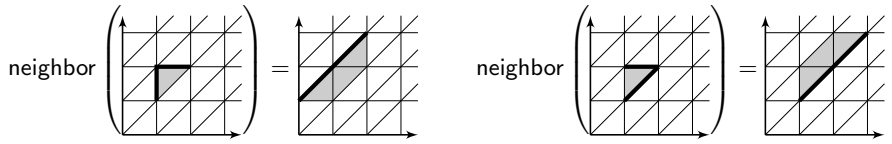


Fig. 7. Constrained neighborhood of two constrained regions. Notice that inside any shrinking of the constrained region, there is always a valuation such that a perturbation of $[-\delta, \delta]$ moves the valuation to any region of the neighborhood.

3.6 Two crucial properties for the construction of the abstraction

The following lemma characterizes, given a constrained region $\langle r, S \rangle$, the set of constrained regions $\langle r', S_{r'} \rangle$ such that any shrunk region satisying $\langle r', S_{r'} \rangle$ can be reached by delaying from some shrunk region satisfying $\langle r, S \rangle$.

Lemma 7. *Let $\langle r, S\rangle$ be a well constrained region, and r' be a region such that $r \lessdot^* r'$. 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 $(r, P) = r \cap \mathsf{shrink}^+(\mathsf{Pre}_{\mathsf{time}}((r', Q)))$, satisfies $P \leq S$;*
2. *$\mathsf{neighbor}\langle r, S\rangle \subseteq \mathsf{Pre}_{\mathsf{time}}(r')$;*

Note that this lemma may not hold for all r' with $r \lessdot r'$. Consider the constrained region $\langle r, S\rangle$ on the right of Fig. 7, and let r' 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 r' by delaying less than δ time units.

Lemma 8. *Let $\langle r, S\rangle$ be a well constrained region, and let $R \subseteq \mathcal{C}$. Let $\mathcal{N}$ be the set of neighboring regions of $\langle r, S\rangle$, and $\mathcal{N}' = \{r'[R \leftarrow 0] \mid r' \in \mathcal{N}\}$. Then, there exist well shrinking constraints $S_{r''}$ for all $r'' \in \mathcal{N}'$ such that for any $(Q_{r''})_{r'' \in \mathcal{N}'}$, we have $Q_{r''} \leq S_{r''}$ for all $r'' \in \mathcal{N}'$ iff there exists $P \leq S$ such that*

$$(r, P) \subseteq r \cap \mathsf{shrink}(\bigcup_{r' \in \mathcal{N}} (r' \cap \mathsf{Unreset}_R((r'', Q_{r''})))).$$

with $r'' = r'[R \leftarrow 0]$. Moreover, all $\langle r'', S_{r''}\rangle$ can be computed in polynomial time.

This lemma gives for instance the shrinking constraints that should be satisfied in r_1, r_2 and r_3, in Fig. 4, 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 $(\ell, r', S_{r'}, e)$ if the following conditions hold:
  (i) $r \lessdot^* r'$ and $r' \subseteq g$;
  (ii) $S_{r'}$ is such that for all SMs Q, $Q \leq S_{r'}$ iff there exists $P \leq S_r$ with

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

(*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 $\mathcal{N}' = \{r'[R \leftarrow 0] \mid r' \in \mathcal{N}\}$, there are transitions to all square nodes $(\ell', r'', S_{r''})$ with $r'' \in \mathcal{N}'$, and $(S_{r''})_{r'' \in \mathcal{N}'}$ are such that for all SMs $(Q_{r''})_{r'' \in \mathcal{N}'}$, it holds $Q_{r''} \leq S_{r''}$ for every $r'' \in \mathcal{N}'$ iff there exists $P \leq S_r$ such that

$$(r, P) \subseteq r \cap \mathsf{shrink}(\bigcup_{r' \in \mathcal{N}} (r' \cap \mathsf{Unreset}_R((r'', Q_{r''}))) \qquad (\text{where } r'' = r'[R \leftarrow 0])$$

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 7 and 8, 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 $\mathbf{0}$ 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 7 and 8. 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 9. *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 9, we can also compute a corresponding δ_0. One can show, by a rough estimation, that $1/\delta_0$ is at worst doubly exponential in the size of $\mathcal{A}$.

Let us point out an interesting intermediary result of the proof: given a winning strategy for Perturbator in $\mathcal{RG}(\mathcal{A})$, we show that there is a winning strategy for Perturbator in $\mathcal{G}_\delta(\mathcal{A})$ that keeps the compatible runs close to borders of regions where shrinking constraints are 0.

The upper bound of Theorem 2 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. The EXPTIME lower bound is obtained by simulating an alternating-time linear-bounded Turing machine. Simulation of the transitions is rather standard in timed-automata literature (though we must be careful here as delays can be perturbed). The difficult point is to simulate conjunctions: this is achieved using the module of Fig. 8. From the initial state, Controller has no choice but to play the first transition when $y = 1$. Perturbator can either anticipate or delay this transition, which will determine which of the dashed or dotted transitions is available next. This way, Perturbator decides by which end the module is exited.

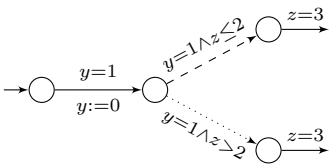


Fig. 8. Conjunction

5 Conclusion

We considered a game-based approach to robust reachability in timed automata. We proved that robust schedulers for reachability objectives can be synthesized, and that the existence of such a scheduler is EXPTIME-complete (hence harder than classical reachability [2]). We are currently working on a zone-based version of the algorithm, and on extending the techniques of this paper to the synthesis of robust controllers in timed games, which will answer an open problem posed in [5] for reachability objectives. Natural further works also include the synthesis of robust schedulers for safety objectives. This seems really challenging, and the abstraction we have built here is not correct in this case (it requires at least a notion of *profitable cycles à la* [6]). Another interesting direction for future work is to assume imprecisions are probabilistic, that is, once Controller has chosen a delay d, the real delay is chosen in a stochastic way in the interval $[d-\delta, d+\delta]$.

References

1. Y. Adbeddaïm, E. Asarin, and O. Maler. Scheduling with timed automata. *TCS*, 354(2):272–300, 2006.
2. R. Alur and D. L. Dill. A theory of timed automata. *TCS*, 126(2):183–235, 1994.
3. E. Asarin, O. Maler, A. Pnueli, and J. Sifakis. Controller synthesis for timed automata. In *SSSC'98*, pp. 469–474. Elsevier, 1998.
4. J. Bengtsson and W. Yi. Timed automata: Semantics, algorithms and tools. In *ACPN'03*, vol. 2098 of *LNCS*, pp. 87–124. Springer, 2004.
5. K. Chatterjee, T. A. Henzinger, and V. S. Prabhu. Timed parity games: Complexity and robustness. *LMCS*, 7(4), 2010.
6. M. De Wulf, L. Doyen, N. Markey, and J.-F. Raskin. Robust safety of timed automata. *FMSD*, 33(1-3):45–84, 2008.
7. M. De Wulf, L. Doyen, and J.-F. Raskin. Almost ASAP semantics: From timed models to timed implementations. *FAC*, 17(3):319–341, 2005.
8. R. L. Graham. Bounds on multiprocessing timing anomalies. *SIAM J. Applied Maths*, 17(2):416–429, 1969.
9. V. Gupta, T. A. Henzinger, and R. Jagadeesan. Robust timed automata. In *HART'97*, vol. 1201 of *LNCS*, pp. 331–345. Springer, 1997.
10. T. A. Henzinger and J. Sifakis. The embedded systems design challenge. In *FM'06*, vol. 4085 of *LNCS*, pp. 1–15. Springer, 2006.
11. H. Kopetz. *Real-Time Systems: Design Principles for Distributed Embedded Applications*. Springer, 2011.
12. K. G. Larsen, P. Pettersson, and W. Yi. UPPAAL in a nutshell. *Intl J. STTT*, 1(1-2):134–152, 1997.
13. N. Markey. Robustness in real-time systems. In *SIES'11*, pp. 28–34. IEEE Comp. Soc. Press, 2011.
14. A. Puri. Dynamical properties of timed automata. *DEDS*, 10(1-2):87–113, 2000.
15. J. Reineke, B. Wachter, S. Thesing, R. Wilhelm, I. Polian, J. Eisinger, and B. Becker. A definition and classification of timing anomalies. In *WCET'06*, 2006.
16. O. Sankur, P. Bouyer, and N. Markey. Shrinking timed automata. In *FSTTCS'11*, vol. 13 of *LIPIcs*, pp. 375–386. LZI, 2011.
17. S. Yovine. Kronos: A verification tool for real-time systems. *Intl J. STTT*, 1(1-2):123–133, 1997.

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.

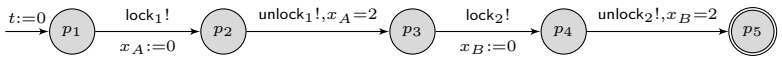


Fig. 9. Timed automaton $\mathcal{P}_1$ modeling Process 1.

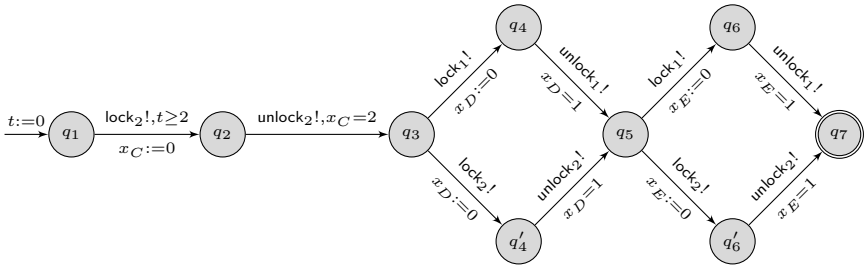


Fig. 10. Timed automaton $\mathcal{P}_2$ modeling Process 2.

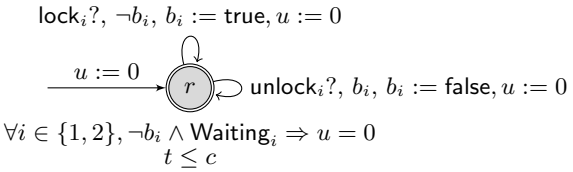
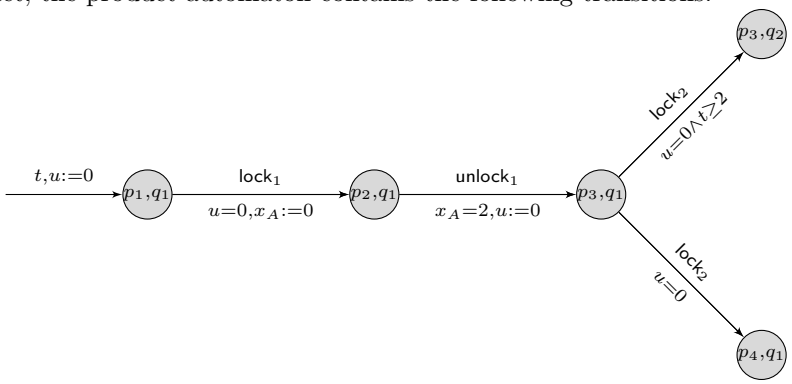


Fig. 11. Timed automaton $\mathcal{M}$ ensuring mutual exclusion and greedy scheduling.

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. [1] 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.

B Shrinking matrices and shrinking constraints

B.1 Shrunk DBMs

Lemma 3 shows that shrinking matrices are propagated when usual operations are applied on DBMs. Using this lemma, one can also compute an upper bound on δ,

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

under which a given equation between shrunk DBMs hold. Figure 12 illustrates some of these operations.

Let us extend the computation of upper bounds on δ in Lemma 3 for an inclusion relation, which will be useful later in the proofs (proof of Lemma 8).

Lemma 10. *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 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 $Q_{x,y} \neq Q'_{x,y}$. □

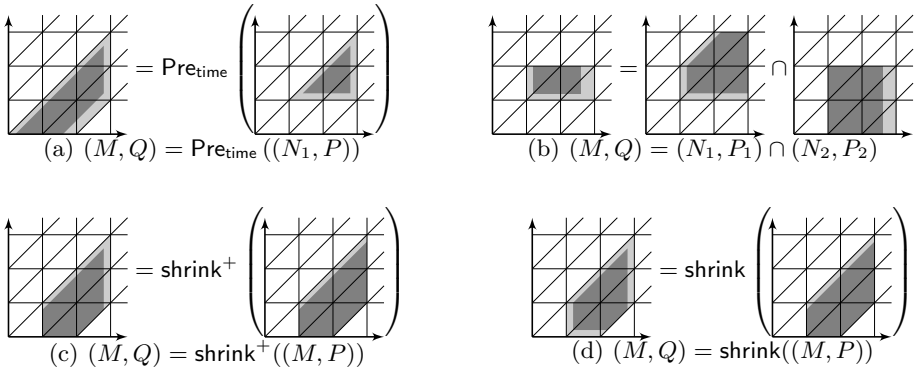


Fig. 12. On the right of Fig. 12(a), the gray area represents a zone N_1, while the dark gray area is some shrinking (N_1, Q). On the left, the dark area is the shrunk zone (M,P) where $M = \mathsf{Pre}_{\mathsf{time}}(N_1)$. Similarly, Fig. 12(b) to 12(d) illustrate the intersection of two shrunk zones and the shrinking of zones.

Notice that Lemma 3 always gives *normalized* shrunk DBMs (M,Q), which means that $M - \delta Q$ is normalized for all small enough $\delta > 0$. The normalization of shrunk DBMs is stated explicitly in the following lemma.

Lemma 11 ([2, Lemma 6]). *Let M be a normalized DBM and P be a shrinking matrix such that (M, P) is non-empty. Then, there exists a unique shrinking matrix P', and $\delta_0 > 0$ such that $M - \delta P'$ is normalized and defines the same zone as $M - \delta P$ for all $\delta \in [0, \delta_0]$. Both P' and δ_0 are computable in polynomial time. We then write $(M, P') = \mathsf{norm}((M, P))$.*

We then say that (M, P') is *normalized*. If M is clear from context, we may say that P' is normalized.

In the proofs, we will need a good characterization of shrunk DBMs that are normalized. Similarly to the normalization of DBMs, this can be done using the finite shortest paths in the graph $\mathsf{G}(M)$ (defined in Section 3) of a given DBM M.

Definition 12. *Let M be a normalized DBM. For any $x, y \in \mathcal{C}_0$, we define $\mathit{\Pi}_{x,y}(\mathsf{G}(M))$ as the set of paths with least (and finite) weight from x to y in $\mathsf{G}(M)$, where edges are weighted by M.*

In the sequel, we consider paths $\pi = \pi_1 \dots \pi_n \in \mathit{\Pi}_{x,y}(\mathsf{G}(M))$ *weighted by* a given matrix or DBM. The *P-weight* of π (or the weight of π in P) is the sum of the weights $P_{\pi_j, \pi_{j+1}}$ for $1 \leq j \leq n-1$. Finite shortest paths can be used to characterize the non-emptiness and the normalization of shrunk DBMs:

Lemma 13. *Consider a normalized DBM M and shrinking matrices P and P' such that $(M, P') = \mathsf{norm}((M, P))$. Then, (M, P') is non-empty if, and only if, for all $x \in \mathcal{C}_0$, there is no path in $\mathit{\Pi}_{x,x}(\mathsf{G}(M))$ with positive P-weight. In this case, P' is such that, for any $x, y \in \mathcal{C}_0$,*

$$P'_{x,y} = \max_{\pi \in \mathit{\Pi}_{x,y}(\mathsf{G}(M))} \sum_{1 \leq i \leq |\pi|-1} P_{\pi_i, \pi_{i+1}}.$$

Proof. In the proof of Lemma 11 (see [2]), the normalization algorithm is as follows: first set P^0 to P, and for any (x, y) such that $M_{x,y} < \infty$, set

$$P^{i+1}_{x,y} := \max(P^i_{x,y}, \max_z(P^i_{x,z} + P^i_{z,y})),$$

(where z ranges over $\mathcal{C}_0$ such that $M_{x,z} + M_{z,y} = M_{x,y}$), until stabilization. If $M_{x,y} = \infty$, then set $P^{i+1}_{x,y} = 0$. When the fixpoint is reached, the resulting matrix P' is such that

$$\begin{cases} P'_{x,y} = \max(P'_{x,y}, \max_z(P'_{x,z} + P'_{z,y})) & \text{if } M_{x,y} < \infty, \\ P'_{x,y} = 0 & \text{otherwise.} \end{cases}$$

Consider a cycle $x_1 x_2 \dots x_n$ in $\mathit{\Pi}_{x,x}(\mathsf{G}(M))$ (with $x_1 = x_n = x$), and assume that it has positive weight in P. Then there exists $1 \leq i \leq n-1$ such that $P_{x_i, x_{i+1}} \geq 1$. Since $x_1 x_2 \dots x_n$ is a shortest cycle when weighted by M, we have $0 = M_{x_1,x_1} = M_{x_1,x_2} + M_{x_2,x_3} + \dots + M_{x_{n-1},x_n}$. We also have $M_{x_i,x_{i+2}} = M_{x_i,x_{i+1}} + M_{x_{i+1},x_{i+2}}$ (because M is normalized, and because $x_1 x_2 \dots x_n$ is a shortest path), so the normalization procedure gives

$$P'_{x_i,x_{i+2}} \geq P^1_{x_i,x_{i+2}} \geq P_{x_i,x_{i+1}} + P_{x_{i+1},x_{i+2}} \geq 1.$$

By repeating this argument, we get $P'_{x_i,x_n} \geq 1$. Finally, since $M_{x_1,x_n} = M_{x_1,x_i} + M_{x_i,x_n}$, we get $P'_{x_1,x_n} \geq 1$. But then $(M - \delta P')_{x_1,x_n} = M_{x_1,x_n} - \delta P'_{x_1,x_n} = -\delta P'_{x_1,x_n}$, which is strictly less than $(0, \leq)$, so that $(M - \delta P')$ is empty for all $\delta > 0$.

Suppose now that all cycles in $\Pi_{x,x}(\mathsf{G}(M))$ have weight 0 in P. We show that the following invariant is satisfied by the normalization procedure: at step $i \geq 1$, if $M_{x,y} = \infty$, then $P^i_{x,y} = 0$, and otherwise $P^i_{x,y}$ is (A) equal to the weight in P of some path of $\Pi_{x,y}(\mathsf{G}(M))$, and (B) at least the maximal P-weight of the paths of $\Pi_{x,y}(\mathsf{G}(M))$ having length at most i.

This clearly holds for $i = 1$, since all edges (x, y) such that $M_{x,y} < \infty$ are finite shortest paths. Consider $i \geq 2$.

- Assume first that some path of $\Pi_{x,y}(\mathsf{G}(M))$ of greatest P-weight among those of length at most i has length less than i. By induction, $P'_{x,y}$ satisfies Property (B) at the previous iteration, so it also satisfies (B) after step $i + 1$ since $P^i_{x,y}$ can only increase when i increases. Moreover, for any $z \in \mathcal{C}$ such that $M_{x,z} + M_{z,y} = M_{x,y}$, if $P^i_{x,y} = P^i_{x,z} + P^i_{z,y}$, then, by induction, there are finite shortest paths from x to z and from z to y of respective P-weights $P^i_{x,z}$ and $P^i_{z,y}$. Then, by removing any cycle in the concatenation of these paths, we get Property (A).
- Assume now that the path of $\Pi_{x,y}(\mathsf{G}(M))$ of greatest P-weight among those of length at most i has length exactly i. Let v denote the node just before y in this path. By induction, there is a shortest path of weight $P^i_{x,v}$ from x to v, and its weight is at least the greatest weight from x to v in at most $i-1$ steps. Also, $P^i_{v,y} \geq P_{v,y}$. Then, the algorithm ensures $P_{x,y} \geq P_{x,v} + P_{v,y}$ since $M_{x,y} = M_{x,v} + M_{v,y}$. This proves Property (B). Property (A) is proved as in the previous case.

When the algorithm stops (after k iterations), for some $\delta_0 > 0$, $M - \delta P^k$ is normalized for all $\delta \in [0, \delta_0]$ [2]. By assumption, there is no cycle in $\Pi_{x,x}(\mathsf{G}(M))$ with positive weight in P, so each $P^k_{x,y}$ is equal to the weight of the path in $\Pi_{x,y}(\mathsf{G}(M))$ with the greatest weight in P. Now, $M - \delta P$ is empty for all $0 < \delta < \delta_0$ if, and only if, there exists $x \in \mathcal{C}_0$ such that $(M - \delta P^k)_{x,x} < (0, \leq)$ for all $0 < \delta < \delta_0$. But since $M_{x,x} = 0$, this means $P^k_{x,x} \geq 1$, which contradicts our assumption. Therefore, there exists $\delta_0 > 0$ such that $M - \delta P'$ is non-empty for all $\delta \in [0, \delta_0]$. □

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:

Proposition 14. *For any DBM M, it holds*

$$\mathit{shrink}_{[0,\delta]}(\mathit{Pre}_{\mathit{time}}(M)) = \{v \mid \exists d \geq \delta, v + d \in M\}.$$

B.2 Shrinking constraints

Normalization and Well-Shrinking Constraints. A shrinking constraint is *normalized* if it is minimum among all "equivalent" shrinking constraints.

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 15. *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 $M_{x,y} = \infty$. 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 11). 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 16. *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 15, $S_{z,z'} = 0$ 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 13, we get $P'_{x,y} = 0$. □

Propagating Shrinking Constraints. In the core of the paper, Lemma 5 states how shrinking constraints are propagated along equation on DBMs. We restate this lemma in detail, and then prove it. Each item is illustrated in Fig. 13 below.

Lemma 17. *Let M, N, N' be normalized non-empty DBMs.*

1. *Assume that $M = \mathsf{Pre}_{time}(N)$. For any normalized well shrinking constraint S for M, there exists a well shrinking constraint S' for N such that for any SM Q, the following holds: $Q \leq S'$ if, and only if, the normalized SM P s.t. $(M, P) = \mathsf{Pre}_{time}((N, Q))$ satisfies $P \leq S$.*
2. *Assume that $M = N \cap N'$.*
   (a) *For any normalized well shrinking constraint S for M, there exists a shrinking constraint S' for N such that for any SM Q, the following holds: $Q \leq S'$ if, and only if the normalized SM P s.t. $(M, P) = (N, Q) \cap N'$ satisfies $P \leq S$.*
   (b) *For any normalized well shrinking constraint S for N, there exists a shrinking constraint S' for M such that for any SM Q, the following holds: $Q \leq S'$ if, and only if there exists a normalized SM P s.t. $(N, P) \cap N' \subseteq (M, Q)$ satisfies $P \leq S$. Moreover, if $(N, P) \cap N' \neq \emptyset$ 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 well shrinking constraint S' for N such that for any SM Q, the following holds: $Q \leq S'$ if, and only if the normalized 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 step 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 SM $Q \leq S'$, and let P be the shrinking matrix such that $(M, P) = \mathsf{Pre}_{\mathsf{time}}((N, Q))$. 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 [2].

We show that $P \leq S$. First suppose that $S_{x,y} = 0$ for some $x, y \in \mathcal{C} \times \mathcal{C}_0$. For such x, y, we have $N_{x,y} = M_{x,y}$, $\prec^N_{x,y} = \prec^M_{x,y}$; 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 13). But this would imply that the same path has positive weight in Q since $Q' \geq Q$, 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 15). 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 13). 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 $M_{x,y} = N_{x,y}$, $\prec^M_{x,y} = \prec^N_{x,y}$, and $N_{x,y} < \infty$ 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 $(M, P) = (N, Q) \cap N'$ if, and only if $(M, P) = (N, Q) \cap M$. So it suffices to consider the equation $M = N \cap M$ with $M \subseteq N$. We let $S'_{x,y} = S_{x,y}$ for any $x, y \in \mathcal{C}_0$ such that $M_{x,y} = N_{x,y}$ 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 SM $Q \leq S'$ and let P such that $(M, P) = (N, Q) \cap M$. P is obtained from Q by first defining Q' as $Q'_{x,y} = 0$ if $M_{x,y} < N_{x,y}$ 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 $M_{x,y} = N_{x,y}$, then $S'_{x,y} = 0$ by definition of S', so $Q_{x,y} = 0$, and $Q'_{x,y} = 0$. If $M_{x,y} < N_{x,y}$, 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 $M_{x,y} = N_{x,y}$ 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 $N' = M \subseteq N$. Since M and N are normalized and non-empty, we have $M_{x,y} \leq N_{x,y}$ for all $x, y \in \mathcal{C}_0$. We define S'_1 by $S'_{1\,x,y} = S_{x,y}$ if $M_{x,y} = N_{x,y}$ 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 $Q \leq S'$. Let us show that there is a SM $P \leq S$ with the desired property. We define Q' as follows:

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

Let P be the normalization of Q' in N. 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 13) and we get $P \leq S$.

Let Q'' denote the normalized DBM defined by $(M, Q'') = M \cap (N, P)$ (Lemma 3). Let us show that $Q \leq Q''$, which implies $(M, Q'') \subseteq (M, Q)$ as desired. Q'' is the normalization of the SM P' defined as follows: $P'_{x,y} = P_{x,y}$ if $M_{x,y} = N_{x,y}$ and $P'_{x,y} = 0$ otherwise. Let $x, y \in \mathcal{C}_0$.

- Assume that $M_{x,y} = N_{x,y}$. 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 $(z, z') \in \pi$, $M_{z,z'} = N_{z,z'}$ 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, $M_{x,y} < N_{x,y}$. 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 $M_{z,z'} = N_{z,z'}$ 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 $P \leq S$, if Q'' denotes the unique normalized SM 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 $M_{x,y} = N_{x,y}$, then $P'_{x,y} = P_{x,y}$. But we have $S_{x,y} = 0$, so $P_{x,y} = 0$. If $M_{x,y} < N_{x,y}$, 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 $M_{x_j,x_{j+1}} = N_{x_j,x_{j+1}}$ for all $1 \leq j \leq n-1$. Then π is also a shortest path in $\mathsf{G}(N)$, hence $M_{x,y} = N_{x,y}$, 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 $1 \leq j \leq n-1$, $M_{x_j,x_{j+1}} < N_{x_j,x_{j+1}}$. 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 $(N, P) \cap N' \neq \emptyset$ 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 $\emptyset \subsetneq (N, P) \cap N' \subseteq (M, Q)$.

▶ **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 15. 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 $(M, P) = \mathsf{Unreset}_R((M, Q))$. 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. □

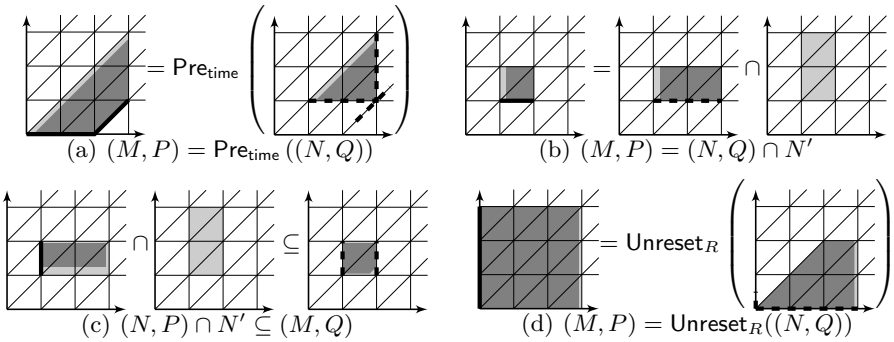
We now extend the previous lemma to the operator shrink^+. In this case, the existence of a shrinking constraint depends on the given constrained DBM. The lemma is illustrated in Fig. 14.

Lemma 18. *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 normalized SM P such that $(M, P) = \mathsf{shrink}^+((M, Q))$ does not satisfy $P \leq S$.*
- *Otherwise, for all SMs Q, the following holds: $Q \leq S$ if, and only if, the normalized SM P such that $(M, P) = \mathsf{shrink}^+((M, Q))$ satisfies $P \leq S$.*

Proof. If $S_{x,0} = 0$ and $M_{x,0} < \infty$ 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 $M_{x,0} < \infty$, 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 $M_{x,0} < \infty$. 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

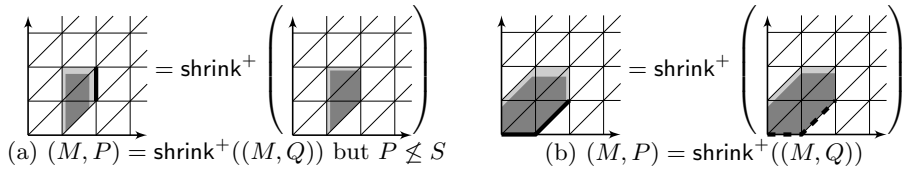


(a) $(M, P) = \mathsf{Pre}_{\mathsf{time}}((N, Q))$ (b) $(M, P) = (N, Q) \cap N'$

(c) $(N, P) \cap N' \subseteq (M, Q)$ (d) $(M, P) = \mathsf{Unreset}_R((N, Q))$

Fig. 13. Each of the figures illustrates one item in Lemma 17. 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.

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 $M_{x,y} < \infty$ 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. □



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

Fig. 14. Examples for both cases of Lemma 18

B.3 Neighborhoods

To prove Lemma 6 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*. Note that this is a partition of all clocks; we do not restrict to clocks with bounded values.

Proposition 19. *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 $S_{x,0} = 0$ for all $x \in X_{i_0} \cup \ldots \cup X_n$ and $S_{x,0} = \infty$ for all $x \in X_2 \cup \ldots \cup X_{i_0-1}$. Notice that when $i_0 = n+1$, we have $S_{x,0} = \infty$ for all x.*
- *There exists $1 \leq j_0 \leq n$ such that $S_{0,x} = 0$ for all $x \in X_1 \cup \ldots \cup X_{j_0}$ and $S_{0,x} = \infty$ for all $x \in X_{j_0+1} \cup \ldots \cup X_n$.*

Proof. This follows from the normalization of S. □

Lemma 6 (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 $x \in \mathcal{C}$.

Proof. 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 $i \leq j \leq m$. [2] Then, for any $1 \leq j \leq i-1$ and $x \in X_j$ $S_{x,0} = \infty$ by Proposition 19. Let r' be the unique region that satisfies $r \lessdot^+ r'$, $r'_{x,0} = r_{x,0} + 1$ and $\prec^{r'}_{x,0} = <$ for all $x \in X_i$, and $r'_{x,0} = r_{x,0}$ for all $x \in X_{i-1}$. We show that the neighborhood of $\langle r, S\rangle$ contains all regions s such that $r \lessdot^* s \lessdot^* r'$, but no region s such that $r' \lessdot^+ s$. 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 [1, 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, $\mathsf{reg}(\nu + d_i) \lessdot \mathsf{reg}(\nu + d_i + \epsilon') = r'$. 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 r' is not included in the neighborhood. In fact, if $i = 1$, then, the immedaite time successor of r' is the unique region $r \lessdot^+ r''$ such that $-r''_{0,x} = r''_{x,0} = r_{x,0} + 1$ and $\prec^{r''}_{x,0} = \leq$ for $x \in X_1$. But for all $\nu \in (r - \delta Q)$, we have $\nu(x) = r_{x,0}$, so $\nu(x) + \delta < r''_{x,0}$. If $i = 2$ and $X_1 = \emptyset$, then the immediate time successor r'' of r' satisfies $-r''_{0,x} = r''_{x,0} = r_{x,0} + 1$ for $x \in X_m$. But since for any $\nu \in r$, $\nu(x) + \delta < r''_{x,0}$, r'' 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 r' 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 $N_{x,0} = r_{x,0} + 1$ and $\prec^N_{x,0} = <$, and if $r_{0,x} < 0$, we let $N_{0,x} = r_{0,x} + 1$ and $\prec^N_{0,x} = <$, otherwise $N_{0,x} = r_{0,x} = 0$ and $\prec^N_{0,x} = \leq$. If $S_{x,0} = \infty$ for all $x \in X_2 \cup \ldots \cup X_m$, then $r_{x,0} = N_{x,0}$ 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 $N_{x,0} = r_{x,0} + 1$ and $\prec^N_{x,0} = <$ for all $x \in X_i \cup X_{i+1} \cup \ldots \cup X_m$, and $N_{x,0} = r_{x,0}$, $\prec^N_{x,0} = \prec^r_{x,0}$ 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 $N_{0,x} = r_{0,x}$ 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 $r_{0,x} < 0$, we let $N_{0,x} = r_{0,x} + 1$ and $\prec^N_{0,x} = <$,

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

and otherwise $N_{0,x} = 0$ and $\prec^N_{0,x} = \leq$. We let $N_{0,x} = r_{0,x}$, $\prec^N_{0,x} = \prec^r_{0,x}$ for all $x \in X_{i'+1} \cup \ldots \cup X_m$.

Let S' be the shrinking constraint given by Lemma 17, for which for all SMs Q, $Q \leq S'$ if, and only if there exists $P \leq S$ with $(r, P) = r \cap (N, Q)$. 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 $r_{x,y} = N_{x,y}$. For all $x \in \mathcal{C}$, $S'_{x,0} = S_{x,0}$ if $r_{x,0} = N_{x,0}$ and $S'_{x,0} = \infty$ otherwise. But by construction of N, $r_{x,0} = N_{x,0}$ 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 $r_{z,z'} = N_{z,z'}$, 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 $N_{x,0} = r_{x,0} + 1$ by definition of N, and this path cannot be a shortest path in $\mathsf{G}(N)$ (in fact, $N_{z,z'} = r_{z,z'} = \sum_{k=1}^{n-1} r_{x_i,x_{i+1}}$). 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 $r_{z,z'} = N_{z,z'}$ 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 $r_{w,0} = N_{w,0}$ and $r_{0,w} = N_{0,w}$ 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 $N_{w,0} = r_{w,0} + 1$ or $N_{0,w} = r_{0,w} + 1$, and in this case π is not a shortest path in $\mathsf{G}(N)$ since $N_{z,z'} = r_{z,z'}$. Hence, there is no such path π, and we get $S'_{x,y} = \infty$ after normalization.

Now, let Q' be such that $(N, Q') = \mathsf{shrink}((N, Q))$. Since $S'_{x,0} = S'_{0,x} = \infty$ for all $x \in \mathcal{C}$ and that $\mathsf{shrink}((N, Q))$ is obtained by incrementing each $Q_{0,x}$ and $Q_{x,0}$ by one, and applying normalization, we have $Q' \leq S'$. 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. □

B.4 Two crucial properties for the construction of the abstraction

We reinforce the first lemma with an extra characterization, which makes the proof easier, and will be useful later.

Lemma 20. *Let $\langle r, S\rangle$ be a well constrained region, and r' be a region such that $r \lessdot^* r'$. 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 $(r, P) = r \cap \mathsf{shrink}^+(\mathsf{Pre}_{\mathsf{time}}((r', Q)))$, satisfies $P \leq S$;*

2. $\mathit{neighbor}\langle r, S\rangle \subseteq \mathit{Pre}_{\mathit{time}}(r')$;
3. *define* $N = \mathit{Pre}_{\mathit{time}}(r')$, *and* S_N *such that for all SM* Q, $Q \leq S_N$ *iff the SM* P *defined by* $(r, P) = r \cap (N, Q)$ *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 $(N, Q') = \mathsf{Pre}_{\mathsf{time}}((r', Q))$. We will show that S' satisfies the required condition. We first assume that $Q \leq S'$. By Lemma 18, there exists $Q'' \leq S_N$ such that $(N, Q'') = \mathsf{shrink}^+((N, Q')) = \mathsf{shrink}^+(\mathsf{Pre}_{\mathsf{time}}((r', Q)))$. And by definition of S_N, there exists $P \leq S$ such that $(r, P) = r \cap (N, Q'')$. Conversely, if $Q \not\leq S'$, then if Q' denotes the SM such that $(N, Q') = \mathsf{Pre}_{\mathsf{time}}((r', Q))$, then $Q' \not\leq S_N$. By Lemma 18, 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 $(r, P) = r \cap \mathsf{shrink}^+(\mathsf{Pre}_{\mathsf{time}}((r', Q)))$.
▶ **not 3 ⇒ not 1**. Let $(V, S') = \mathsf{neighbor}\langle r, S\rangle$. We assume that $V \not\subseteq N$. Then, if r'' denotes the region $r' \lessdot r''$, we have $r', r'' \subseteq V$. 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 $v + d' \in r''$ for some $0 \leq d' \leq \delta$. But since $r' \lessdot r''$, $v + d' \in r'$ 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 $(r, P) = r \cap \mathsf{shrink}^+(\mathsf{Pre}_{\mathsf{time}}((r', Q)))$, then $P \not\leq S$: indeed, if $P \leq S$, then for small enough $\delta > 0$, $v + [0, \delta] \subseteq \mathsf{Pre}_{\mathsf{time}}(r' - \delta Q)$ and in particular $v + \delta \in \mathsf{Pre}_{\mathsf{time}}(r' - \delta Q)$, which implies that there is $d'' \geq 0$ such that $v + \delta + d'' \in r' - \delta Q \subseteq r'$, contradicting the above remark.
▶ **2 ⇒ 3**. Let $(V, S') = \mathsf{neighbor}\langle r, S\rangle$. We have $r \subseteq V \subseteq N$. Let Q be a SM with $Q_{x,0} = 1$ for all $x \in \mathcal{C}$ and 0 all other components, and apply normalization for (N, Q). Let Q' be the SM such that $(V, Q') = V \cap (N, Q)$.

- We show that $Q' \leq S'$. Define $(Q_1)_{x,y} = Q_{x,y}$ if $V_{x,y} = N_{x,y}$ 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 6, 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 $\mathit{\Pi}_{x,y}(\mathsf{G}(V))$ with an edge (z, z') such that $V_{z,z'} = N_{z,z'}$ and $Q_{z,z'} \geq 1$. Since $Q_{z,z'} \geq 1$, there must be a path in $\mathit{\Pi}_{z,z'}(\mathsf{G}(N))$ that contains an edge $(\alpha, 0)$. But since $\mathit{\Pi}_{z,z'}(\mathsf{G}(N)) \subseteq \mathit{\Pi}_{z,z'}(\mathsf{G}(V))$, there is a path in $\mathit{\Pi}_{x,y}(\mathsf{G}(V))$ that contains the edge $(\alpha, 0)$. This contradicts $S'_{x,y} = 0$ since we know, by Lemma 6 that $S'_{\alpha,0} = \infty$.
- We now show that $Q \leq S_N$, which will imply the expected result. By Lemma 6, there exists $P \leq S$ such that $(r, P) = r \cap \mathsf{shrink}((V, Q'))$. 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 17, 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 17, 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

$N_{x,y} = r_{x,y}$ 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 exists 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 $r_{z,z'} = N_{z,z'}$ 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 $\pi \in \Pi_{z,z'}(\mathsf{G}(r'))$ with $(S'_N)_{z,z'} = 0$, so necessarily $z \neq 0$ and $r'_{z,z'} = N_{z,z'}$. 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 $\mathsf{G}(r')$ 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 $\mathsf{G}(r')$ and $\mathsf{G}(N)$ except for the edges $(0, z)$ which can only decrease in r', π is a shortest path in both graphs. In both cases, we get $(S_N)_{x,y} = 0$.

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

Lemma 8. *Let $\langle r, S\rangle$ be a well constrained region, and let $R \subseteq \mathcal{C}$. Let $\mathcal{N}$ be the set of neighboring regions of $\langle r, S\rangle$, and $\mathcal{N}' = \{r'[R \leftarrow 0] \mid r' \in \mathcal{N}\}$. Then, there exist well shrinking constraints $S_{r''}$ for all $r'' \in \mathcal{N}'$ such that for any $(Q_{r''})_{r'' \in \mathcal{N}'}$, we have $Q_{r''} \leq S_{r''}$ for all $r'' \in \mathcal{N}'$ iff there exists $P \leq S$ such that*

$$(r, P) \subseteq r \cap \mathit{shrink}(\bigcup_{r' \in \mathcal{N}} (r' \cap \mathit{Unreset}_R((r'', Q_{r''})))).$$

with $r'' = r'[R \leftarrow 0]$. Moreover, all $\langle r'', S_{r''}\rangle$ can be computed in polynomial time.

Proof. Let $\langle N, S'\rangle$ be the (well) constrained neighborhood of $\langle r, S\rangle$, given by Lemma 6. For any region $r' \in \mathcal{N}$, let $S_{r'}$ be the well shrinking constraint given by Lemma 17, such that for all SMs Q, $Q \leq S_{r'}$ if, and only if there exists $P \leq S'$ with $r' \cap (N, P) \subseteq (r', Q)$. Here, $S_{r'}$ is in fact a well shrinking constraint since $r' \cap (N, P) \neq \emptyset$ for all $P \leq S'$, by the construction of the neighborhood. Then, let $S'_{r'}$ be the well shrinking constraint given by Lemma 17, such that for any SM $Q_{r''}$ $Q_{r''} \leq S'_{r'}$ if, and only if there exists $P_{r'} \leq S_{r'}$ with $(r', P_{r'}) = r' \cap \mathsf{Unreset}_R((r'', Q_{r''}))$. We let $S_{r''} = \min_{r' \in \mathcal{N} : r'' = r'[R \leftarrow 0]} S'_{r'}$, where the minimum is taken componentwise. Then $S_{r''}$ is still a well shrinking constraint. Now, $S_{r''}$ satisfies the following property: for any SM $Q_{r''}$, $Q_{r''} \leq S_{r''}$ if, and only if for all regions $r' \in \mathcal{N}$ with $r'' = r'[R \leftarrow 0]$, there exists $P_{r'} \leq S_{r'}$ such that $r' - \delta P_{r'} = r' \cap \mathsf{Unreset}_R(r'' - \delta Q_{r''})$ for small enough $\delta > 0$. Combining this and S' defined above, we get that $S_{r''}$ has the property that for any SM $Q_{r''}$, $Q_{r''} \leq S_{r''}$ if, and only if for all $r' \in \mathcal{N}$ with $r'' = r'[R \leftarrow 0]$, there exists $P_{r'} \leq S'$ such that $r' \cap (N, P_{r'}) \subseteq r' \cap \mathsf{Unreset}_R((r'', Q_{r''}))$.

For any such family $\{Q_{r''}\}_{r'' \in \mathcal{N}'}$ and $\{P_{r'}\}_{r' \in \mathcal{N}}$, let P' be defined by $P' = \max_{r' \in \mathcal{N}}(P_{r'})$, where the max is componentwise. We have $P' \leq S'$ since $P_{r'} \leq S'$

for all $r' \in \mathcal{N}$. We have $r' \cap (N, P') \subseteq r' \cap \mathsf{Unreset}_R((r'', Q_{r''}))$, for all $r' \in \mathcal{N}$, since $P_{r'} \leq P'$. So,

$$(N, P') = \bigcup_{r' \in \mathcal{N}} r' \cap (N, P') \subseteq \bigcup_{r' \in \mathcal{N}} r' \cap \mathsf{Unreset}_R((r'', Q_{r''})).$$

By Lemma 6, there exists $P \leq S$ such that

$$(r, P) = r \cap \mathsf{shrink}((N, P')) \subseteq \bigcup_{r' \in \mathcal{N}} r' \cap \mathsf{Unreset}_R((r'', Q_{r''})).$$

This proves the first direction of the lemma.

Consider SMs $\{Q_{r''}\}_{r'' \in \mathcal{N}'}$ such that $(Q_{r''})_{x,y} \geq 1$ and $(S_{r''})_{x,y} = 0$ for some $r'' \in \mathcal{N}'$ and $x, y \in \mathcal{C}_0$. Then there exists $r_0 \in \mathcal{N}$ with $r_0[R \leftarrow 0] = r''$, such that if Q_{r_0} denotes the SM that satisfies $(r_0, Q_{r_0}) = r_0 \cap \mathsf{Unreset}_R((N, Q_{r''}))$ 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 $(N, P') \subseteq \bigcup_{r' \in \mathcal{N}} (r', Q_{r'})$ where $(r', Q_{r'}) = r' \cap \mathsf{Unreset}_R((N, Q_{r''}))$. 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.

□

C A finite game abstraction

In this section, we prove the following theorem:

Proposition 9. *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 $\delta \in [0, \delta_0]$.*

C.1 Proof of the EXPTIME upper-bound

► **Controller wins $\mathcal{RG}(\mathcal{A}) \Rightarrow$ Controller wins $\mathcal{G}_\delta(\mathcal{A})$ for all small enough $\delta > 0$.** Let σ be a memoryless winning strategy for Controller in $\mathcal{RG}(\mathcal{A})$ for reaching some location $\ell_{\smile}$. 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 t of $\mathcal{T}_\sigma$ labelled by (ℓ, r, S_r), we will assign a shrunk DBM $\big(r, P_t\big)$ and $\delta_t > 0$ with $P_t \leq S_r$, such that Controller wins the game $\mathcal{G}_\delta(\mathcal{A})$ from any state of $\{\ell\} \times (r - \delta P_t)$ for all $\delta \in [0, \delta_t]$. We will define these by a bottom up traversal of $\mathcal{T}_\sigma$. We start by assigning $P_t = \mathbf{0}$ and $\delta_t = \infty$ to all nodes with $\mathsf{loc}(t) = \ell_{\smile}$ (which are leaves of $\mathcal{T}_\sigma$). These are trivially winning for Controller.

Consider now a square node t of $\mathcal{T}_\sigma$ labelled by (ℓ, r, S_r) whose all square successors have been treated. Then, t has only one successor which is given by σ, we assume it is the diamond node t' labelled by $(\ell, r', S_{r'}, e)$. We write $e = (\ell, g, R, \ell')$. Let $\mathcal{N}$ be the set of neighboring regions of $\langle r', S_{r'} \rangle$ given by

Lemma 6. Let $r_1, \ldots, r_m$ be the regions composing $\mathcal{N}$, and let $t'_1, \ldots, t'_m$ denote the successors of t' in $\mathcal{T}_\sigma$, where t'_i is labelled by $(\ell', r'_i, S_{r'_i})$, and such that $r'_i = r_i[R \leftarrow 0]$. By construction, shrinking constraints $S_{r'_i}$ are given by Lemma 8 applied to $\langle r', S_{r'} \rangle$. By induction, for each $1 \leq i \leq m$ there is a shrunk DBM $(r'_i, P_{r'_i})$ and $\delta_{r'_i} > 0$ with $P_{r'_i} \leq S_{r'_i}$, such that Controller wins the game $\mathcal{G}_\delta(\mathcal{A})$ from $\{\ell'\} \times (r'_i - \delta P_{r'_i})$ for all $\delta \leq \delta_{r'_i}$. Now, by Lemma 8, there exists a SM $Q \leq S_{r'}$ such that

$$r' - \delta Q \subseteq r' \cap \mathsf{shrink}(\bigcup_{1 \leq i \leq m} r_i \cap \mathsf{Unreset}_R(r'_i - \delta P_{t_i})),$$

for all $0 \leq \delta \leq \delta_{t'}$, where $\delta_{t'} \leq \min_i(\delta_{r'_i})$ is computed by Lemma 10. Then, by construction of the game, there exists $P_t \leq S_r$ and $0 < \delta_t \leq \delta_{t'}$ such that $r - \delta P_t = r \cap \mathsf{shrink}^+(\mathsf{Pre}_{\mathsf{time}}(r' - \delta Q))$ for all $0 \leq \delta \leq \delta_t$. Here, δ_t can be computed using Lemma 3. Controller wins from these states: in fact, it can delay into $r' - \delta Q$, where, after Perturbator's any move, and the clock resets, the next state belongs to one of the states $r'_i - \delta P_{t_i}$, 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_t) 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 $r - \delta P_t$ at any node t.

► **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 21. *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 and [2] 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.

► **Controller loses $\mathcal{RG}(\mathcal{A})$ ⇒ 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 $\mathcal{RG}(\mathcal{A})$.

Definition 22. *Consider any constrained DBM $\langle M, S\rangle$. We say that a valuation ν is* ϵ-tight in $\langle M, S\rangle$ *if $\nu \in M$, and*

$$\forall x, y \in \mathcal{C}_0, \quad S_{x,y} = 0 \quad \Rightarrow \quad M_{x,y} - \epsilon \leq \nu(x) - \nu(y). \tag{1}$$

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

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

Lemma 23. *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 17, 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 $P \leq S_r$. Then, for any valuation ν that is ϵ-tight in $\langle r, S_r\rangle$, valuation $\nu' = \nu[R \leftarrow 0]$ is ϵ-tight in $\langle r', S_{r'}\rangle$.*

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 $r'_{y,x} = r_{y,0}$.

For all $x, y \notin R$ or $x, y \in R$, it is clear that ν' satisfies the property. 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. □

Before proceeding to the proof of the theorem, we need the following technical lemma about intersection, which follows easily from Lemma 17, when applied to the constrained neighborhood of a constrained region.

Lemma 24. *Let (N, S_N) 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 17, 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 are now ready to prove the other direction of Propsition 9: 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: for each state (ℓ_i, ν_i), we have $\nu_i \in r_i$ and ν_i is ϵ_i-tight in $\langle r_i, S_{r_i} \rangle$. This property is central in our proof. It is used to show that Perturbator can imitate, in $\mathcal{G}_\delta(\mathcal{A})$, the move prescribed for $\mathcal{RG}(\mathcal{A})$ by γ. In fact, we prove below that when the game is in a diamond node, Perturbator can choose, in $\mathcal{G}_\delta(\mathcal{A})$, a valuation corresponding to each of the successors of that node.

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. The following lemma shows that $\mathcal{RG}(\mathcal{A})$ has a corresponding edge.

Lemma 25. *$\mathcal{RG}(\mathcal{A})$ has an edge from (ℓ_i, r_i, S_{r_i}) to $(\ell_i, r'_i, S_{r'_i}, e_i)$ for some $S_{r'_i}$.*

Proof. Let $N = \mathsf{Pre}_{\mathsf{time}}(r'_i)$, and consider, by Lemma 17, the shrinking constraint S_N such that for any SM Q, $Q \leq S_N$ if, and only if the SM P such that

$(r_i, P) = r_i \cap (N, Q)$ satisfies $P \leq S_{r_i}$. Thanks to Lemma 20, 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 17, S_N is the normalization of S'_{r_i} defined by $(S'_{r_i})_{x,y} = (S_{r_i})_{x,y}$ if $(r_i)_{x,y} = N_{x,y}$ and $(S'_{r_i})_{x,y} = \infty$ otherwise. So, $(S_N)_{x,0} = 0$ means that there exists (z, z') such that $(x, 0)$ is on some path π of $\Pi_{z,z'}(\mathsf{G}(N))$, $(r_i)_{z,z'} = N_{z,z'}$ and $(S_{r_i})_{z,z'} = 0$. But since $r_i \subseteq N$ and $(r_i)_{z,z'} = N_{z,z'}$ we have $\Pi_{z,z'}(\mathsf{G}(N)) \subseteq \Pi_{z,z'}(\mathsf{G}(r_i))$. Moreover, all weights along π are the same in r_i and N. We get $(S_{r_i})_{x,0} = 0$. On the other hand, since $r'_i \subseteq N$ and $r_i \lessdot^* r'_i$, we have $(r_i)_{x,0} \leq (r'_i)_{x,0} \leq N_{x,0}$. So $(r_i)_{x,0} = (r'_i)_{x,0}$. By induction hypothesis, we have $(r_i)_{x,0} - \epsilon_i \leq \nu_i(x) \leq (r_i)_{x,0}$. But $\epsilon_i \leq \delta/2$ and $d \geq \delta$, so $\nu_i(x) + d > (r_i)_{x,0} = (r'_i)_{x,0}$, a contradiction. □

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 $s[R \leftarrow 0] = r_{i+1}$. It then remains to realize that first s, then r_{i+1} are reachable from ν'_i by Perturbator's move, and that the resulting valuations are ϵ_i-tight in their respective constrained DBMs.

For any $x, y \in \mathcal{C}$, $(S_{r'_i})_{x,y} = 0$ implies that $(r'_i)_{x,y} - \epsilon_i \leq \nu'_i(x) - \nu'_i(y) \leq (r'_i)_{x,y}$, since S_{r_i} and $S_{r'_i}$ have the same diagonal components and time delays do not change the quantities $\nu_i(x) - \nu_i(y)$. Hence, ν'_i is ϵ_i-tight in $\langle S_{r'_i}, r'_i\rangle$ for all diagonal components (this suffices for the rest of the proof). Assume that $r'_i \lessdot^* s$. 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'_i:

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

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

1. Assume that $(S_{r'_i})_{x,0} = \infty$ for all $x \in X_2 \cup \ldots \cup X_m$. Then, by definition of the neighborhood, either $r'_i = s$ or $r'_i \lessdot s$, where the latter case is possible if $X_1 \neq \emptyset$.
   - If $r'_i = s$, then Perturbator perturbs by 0. Since $(S_s)_{x,0} = \infty$ and $(S_s)_{x,y} = (S_{r'_i})_{x,y}$ for all $x, y \in \mathcal{C}$, by Lemma 24, the valuation ν'_i is ϵ_i-tight (s, S_s) for these components. If $(S_s)_{0,x} = 0$ for some $x \in \mathcal{C}$, then $N_{0,x} < (r'_i)_{0,x}$ by Lemma 24 since $s_{0,x} = r_{0,x}$, so we must have $(S_{r'_i})_{0,x} = 0$, and ν'_i is also ϵ_i-tight for components $(0, x)$. Note that the ϵ_i-tightness of diagonal components follow from the ϵ_i-tightness of ν_i in $\langle S_{r'_i}, r'_i\rangle$. Then, $\nu''_i = \nu'_i[R \leftarrow 0]$ satisfies the invariant by Lemma 23.
   - If $r'_i \lessdot s$, we let Perturbator perturb by a positive amount $0 < d' < \epsilon/2^{i+1}$, so that the valuation is in s. Since $(S_s)_{x,0} = \infty$ and $(S_s)_{x,y} = (S_{r'_i})_{x,y}$ for all $x, y \in \mathcal{C}$, by Lemma 24, for these components, $\nu'_i + d'$ is ϵ_i-tight in (s, S_s). If $(S_s)_{0,x} = 0$ for some $x \in \mathcal{C}$, then $N_{0,x} < (r'_i)_{0,x}$ by Lemma 24, so we must have $(S_{r'_i})_{0,x} = 0$. We have, by hypothesis, $-\nu'_i(x) \geq (r'_i)_{0,x} = s_{0,x} - \epsilon_i$, so $-(\nu'_i(x) + d') \geq s_{0,x} - \epsilon_{i+1}$. $\nu'_i + d'$ is also ϵ_i-tight (thus, ϵ_{i+1}-tight) for diagonal components since ν'_i is. Then, $\nu''_i = (\nu'_i + d')[R \leftarrow 0]$ satisfies the invariant by Lemma 23.

2. Otherwise, consider the minimum $k \in \{2, \ldots, m\}$ such that $(S_{r'_i})_{x,0} = 0$ for all $x \in X_k \cup \ldots \cup X_m$. Since s is a successor region of r'_i, 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 $s_{x,0} = (r'_i)_{x,0}$, $\prec^s_{x,0} = \leq$ for $x \in X_{k'}$; $s_{x,0} = (r'_i)_{x,0} + 1$ and $\prec^s_{x,0} = <$ for all $x \in X_{k'+1} \cup \ldots \cup X_m$; $s_{x,0} = (r'_i)_{x,0}$ 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 $\nu'_i + d' \in s$, and that $d' \leq \epsilon_i$ since ν'_i is ϵ_i-tight. Let us show that the $\nu'_i + d'$ is ϵ_i-tight in (s, S_s). By Lemma 24, all diagonal constraints in S_s are the same as in $S_{r'_i}$ so the property is satisfied for these components.
   - Let us show ϵ_i-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 24, since $s_{x,0} = (r'_i)_{x,0} + 1 = N_{x,0}$. We have $(\nu'_i + d')(x) = s_{x,0} = -s_{0,x}$ for $x \in X_{k'}$, so $\nu'_i + d'$ is ϵ_i-tight for component $(x, 0)$. For all $x \in X_k \cup \ldots \cup X_{k'-1}$, we have $s_{x,0} = (r'_i)_{x,0}$, and $\nu'_i(x) \geq (r'_i)_{x,0} - \epsilon_i$, which implies $\nu'_i(x) + d' \geq s_{x,0} - \epsilon_i$ 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 24 that $s_{x,0} < N_{x,0}$. But since $r'_i \lessdot^* s$, we would have $r'_i \leq s_{x,0} < N_{x,0}$, and this implies that $(S_{r'_i})_{x,0} = 0$ by definition of N and Lemma 24. But by the choice of k, and by Proposition 19, this is a contradiction.
   - We now show ϵ_i-tightness for components $(0, x)$ (lower bounds). For all $x \in X_{k'+1} \cup \ldots \cup X_m$, we have, $-(\nu'_i(x) + d) \geq s_{0,x} - \epsilon_i$ since $d' \leq \epsilon_i$. The property is again clearly satisfied by $\nu'_i(x) + d'$ for $x \in X_{k'}$ since $\nu'_i(x) + d' = s_{x,0}$. 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 $(S_{r'_i})_{y,x} = 0$, so $\nu'_i(y) - \nu'_i(x) \geq s_{y,x} - \epsilon_i$. means that $\mathsf{frac}(\nu'_i(x)) + d' \leq \epsilon_i$. The following figure illustrates this, assuming $x \in X_i$.

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

   Therefore $\nu'_i + d'$ satisfies $\nu'_i(x) + d' \leq -s_{0,x} + \epsilon_i$, for all $x \in \mathcal{C}$ such that $(S_s)_{0,x} = 0$. We conclude by Lemma 23.
   (b) $s_{x,0} = (r'_i)_{x,0} + 1$ and $\prec^s_{x,0} = <$ for all $x \in X_{k'} \cup \ldots \cup X_m$; $s_{x,0} = (r'_i)_{x,0}$ 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 $\epsilon/2^{i+1}$. Then, $\nu'_i + d'$ is ϵ_{i+1}-tight in (s, S_s) and we conclude by Lemma 23.

C.2 Proof of the EXPTIME lower-bound

Proposition 26. *The robust reachability problem is* EXPTIME*-hard.*

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 a γ in the current cell, write a 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 16: 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.

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

Lemma 27. *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 he 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. ◻

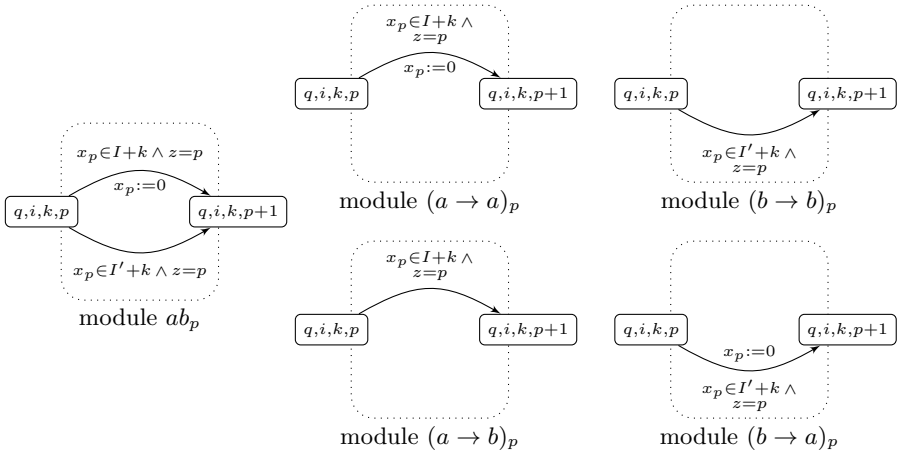


Fig. 15. Intermediary modules: when traversing module ab_p, either clock x_p belongs to $I + k$ while $z = p$, corresponding to an a at position p. Clock x_p is then reset, so that position p still contains an a; or clock x_p is in $I' + k$ when $z = p$, encoding a b at position p. In that case, clock 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'.

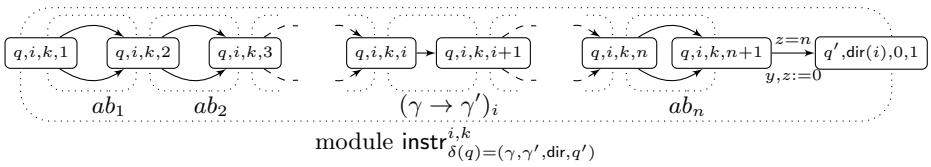


Fig. 16. Module $\mathsf{instr}^{i,k}_{\delta(q)=(\gamma,\gamma',\mathsf{dir},q')}$ for the simulation of the regular instruction $\delta(q) = (\gamma, \gamma', \mathsf{dir}, q')$, and its constituent submodules. If $\mathsf{dir} = \to$, $\mathsf{dir}(i) = i+1$ if $1 \leq i < n$, and undefined otherwise. If $\mathsf{dir} = \leftarrow$, $\mathsf{dir}(i) = i - 1$ if $1 < i \leq n$, and undefined otherwise.

Remark 1. 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 17. As in the previous module, the Controller has no other choice that 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 to choose either $y = 1 \wedge z \leq 2$ or $y = 1 \wedge z > 2$.

Lemma 28. *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*

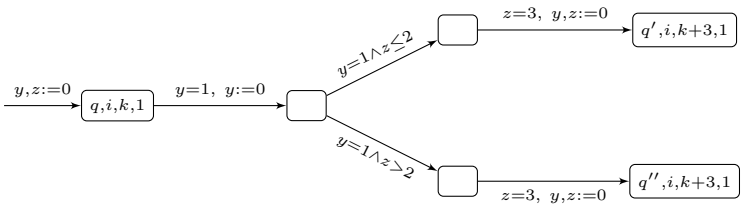


Fig. 17. Module $\mathsf{conj}^{i,k}_{\delta(q)=q'\wedge q''}$ for conjunctive transition

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 18.

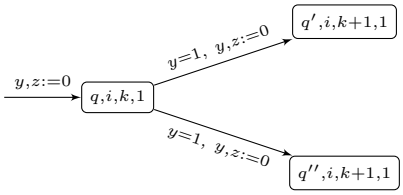


Fig. 18. Module $\mathsf{disj}^{i,k}_{\delta(q)=q'\wedge q''}$ for disjunctive transition

Lemma 29. *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 $\mathsf{reset}_q^{i,k}$ which resets the shift from state q, position i.

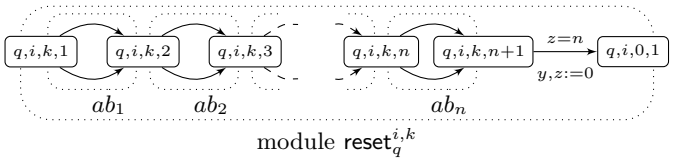


Fig. 19. Module $\mathsf{reset}_q^{i,k}$ which resets the shift in the encoding.

Lemma 30. *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 27. □

▶ **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. □

References

1. F. Herbreteau, D. Kini, B. Srivathsan, and I. Walukiewicz. Using non-convex approximations for efficient analysis of timed automata: Extended version. Technical Report cs.LO/1110.3704v1, arXiv – Computing Research Repository, 2011.
2. O. Sankur, P. Bouyer, and N. Markey. Shrinking timed automata. In *FSTTCS'11*, vol. 13 of *LIPIcs*, pp. 375–386. LZI, 2011.