VACSIM

Validation de la commande des systèmes critiques par couplage simulation et méthodes d'analyse formelles

Tâche 4
Validation formelle de propriétés quantitatives : approche par automates

Sous-tâche 4.1 : Analyse quantitative des automates temporisés

Livrable L4.1

Etude de techniques d'analyse quantitative des modèles temporisés

Version A

Appel :	**PROGRAMME INGENIERIE NUMERIQUE & SECURITE 2011**
Numéro d'agrément :	**ANR-11-INSE-004**
Thématique:	**Systèmes embarqués et ingénierie du logiciel**
Objectif:	**Validation formelle de propriétés quantitatives : approche par automates**
Date de démarrage du projet:	**01.10.2011**
Durée :	**42 mois**

Synthèse

Le projet **VACSIM** (Validation de la commande des systèmes critiques par couplage simulation et méthodes d'analyse formelle), référencé ANR-11-INSE-004, étudie les avantages respectifs des techniques de simulation, en incluant des modèles des processus commandés, et des méthodes d'analyse formelles, pour la validation de la commande des systèmes critiques. Ce projet est structuré en 6 tâches.

La tâche 4 « Validation formelle de propriétés quantitatives : approche par automates» a pour but de contribuer à l'avancée de techniques de validation de systèmes temporisés. Elle est divisée en trois sous-tâches complémentaires, visant chacune une problématique particulière de validation : l'analyse quantitative des automates temporisés, les techniques de validation à l'exécution (test, monitoring et enforcement), et la vérification d'automates temporisés communicants.

Ce livrable L4.1 du projet VACSIM est issu des travaux de la sous-tâche T4.1 relative à l'analyse quantitative des automates temporisés. Dans cette sous-tâche, nous proposons de développer des techniques de vérification pour ces aspects quantitatifs des automates temporisés, et de les appliquer à la validation de systèmes temps-réel. Il est naturel d'associer à chaque exécution d'un système une valeur représentant, par exemple son coût, ou toute autre quantité. Dans ce but, nous nous sommes focalisé sur la quantification des éventuelles défaillances d'un système temps-réel en exprimant pour chacune de ses exécutions la portion de temps passé dans des états critiques. Cette notion de fréquence est étudiée dans le but de développer des algorithmes calculant l'ensemble des fréquences pour un automate temporisé, voire la fréquence moyenne qui exprimera à quel point un système est défaillant.

L'ensemble des méthodes mises en place pourra ensuite être exploité dans d'autres champs de la validation formelle que la vérification, comme par exemple le test, le diagnostic ou le contrôle.

Ce livrable L4.1 a été initialement rédigé par INRIA Rennes- Bretagne Atlantique.

Sommaire

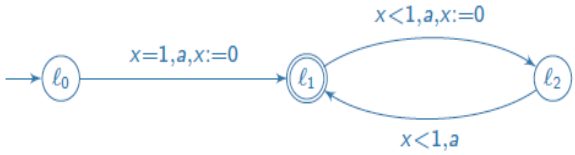
Table des Figures

1. Fréquences dans les automates temporisés *forgetful*

1.1. Les automates temporisés

Le modèle des automates temporisés (Timed Automata) a été introduit par Alur et Dill [AD94] pour représenter les comportements des systèmes temps-réel, c'est à dire des systèmes où les contraintes de temps ont une influence directe sur leurs comportements. Intuitivement, un automate temporisé est un automate ayant un ensemble fini de localités, dont certaines sont acceptantes, un alphabet fini d'actions, et muni d'un ensemble fini d'horloges continues évoluant de manière synchrone (*i.e.* à la même vitesse). Chaque transition entre deux localités de l'automate est étiquetée par une action, une garde exprimant des contraintes sur les valeurs des horloges (et conditionnant le tir de la transition), et une remise à zéro (reset) d'un sous-ensemble d'horloge. Habituellement, la sémantique des automates temporisés pour les mots temporisés infinis est la sémantique de Büchi (également présentée dans [AD94]), qui définit le langage d'un automate temporisé par l'ensemble des mots temporisés dont une exécution visite infiniment souvent des localités acceptantes.

Exemple : pour l'automate temporisé de la Figure 1, dont la localité acceptante ℓ_1 est représentée par un double cercle, nous donnons une de ses exécutions infinies, et le mot temporisé infini accepté correspondant dans la sémantique de Büchi.



$$(\ell_0, 0) \xrightarrow{1,a} (\ell_1, 0)\left(\xrightarrow{0.5,a} (\ell_2, 0) \xrightarrow{0.25,a} (\ell_1, 0.25)\right)^{\omega}$$

$$(1, a)(1.5, a)(1.75, a)(2.25, a)(2.5, a)\cdots$$

Accepted run: run going infinitely often through accepting locations

Every run of this timed automaton is accepted

Accepted timed word: timed word with an accepted run

Figure 1: Automate temporisé et sémantique de Büchi.

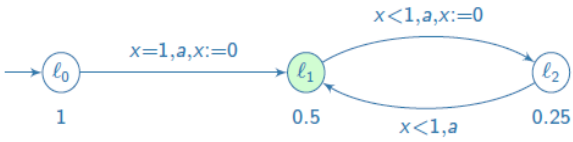
Récemment, plusieurs travaux ont proposé d'aborder des aspects quantitatifs dans les problèmes de vérification, en ajoutant au modèle des coûts [ALP01, BFH*01] ou des probabilités [KNS*02, BBB*08], et en calculant des quantités relatives à ces aspects.

1.2. Fréquences pour l'analyse quantitative des automates temporisés

Plutôt que de modifier le modèle et y ajoutant des aspects quantitatifs, une direction alternative que nous avons explorée consiste à garder le modèle standard des automates temporisés, mais à affiner

la condition d'acceptation de Büchi, en prenant en compte la proportion du temps écoulé dans les localités acceptantes. Dans le cadre du projet ANR Testec, nous avions introduit dans [BBB*11] une sémantique quantitative des automates temporisés pour les mots temporisés infinis, basée sur cette notion de fréquence. Plus précisément, un mot temporisé infini appartient au langage s'il a une fréquence positive, c'est à dire si la proportion de temps passée dans des localités acceptantes durant son exécution sur l'automate est positive.

Exemple : Pour le même automate qu'en Figure 1, la Figure 2 explicite le temps de séjour dans les localités acceptantes, la sémantique des fréquences d'une exécution, et son calcul pour la même exécution prise en exemple dans la sémantique de Büchi: le temps total passé en ℓ_1 étant double du temps passé en ℓ_2 , et non-borné, la proportion de temps passée dans l'état initial est alors négligeable pour une exécution infinie, et la fréquence est de 2/3. On peut également montrer que l'ensemble des fréquences que peuvent couvrir les exécutions de cet automate est l'intervalle [0,1] tout entier.
La Figure 3 présente deux autres exécutions de l'automate : la première a une fréquence d'1/2, l'exécution passant le même temps (non-borné) dans les 2 localités de la boucle, et le temps de séjour dans la localité initiale devenant négligeable à l'infini; la seconde présente une exécution Zénon, le temps de séjour dans chaque localité de la boucle étant le même, et tendant vers 1 à l'infini, la proportion du temps passé dans la localité initiale n'est plus négligeable, et la fréquence est alors de 1/3. Notons que la proportion de temps passé dans une localité peut être nulle bien que le mot soit accepté au sens de Büchi, par exemple sur l'automate de la Figure 2, en alternant des délais 1 dans la localité non-acceptante avec des délais tendant vers 0 dans la localité acceptante.



Frequency of a run $\rho = ((\ell_i, v_i) \xrightarrow{\tau_i, a_i})_{i \geq 0}$:

$$\mathrm{freq}(\rho) = \limsup_{n \to \infty} \frac{\sum_{i \leq n \mid \ell_i \in F} \tau_i}{\sum_{i \leq n} \tau_i}$$

$$\mathrm{freq}\Big((\ell_0, 0) \xrightarrow{1,a} (\ell_1, 0)\big(\xrightarrow{0.5,a} (\ell_2, 0) \xrightarrow{0.25,a} (\ell_1, 0.25)\big)^{\omega}\Big) = \frac{2}{3}$$

Set of frequencies $= [0, 1]$

Semantics with positive frequency (or with threshold)
- Accepted run: run whose frequency is positive
- Accepted timed word: timed word with an accepted run

Figure 2: Sémantique des fréquences.

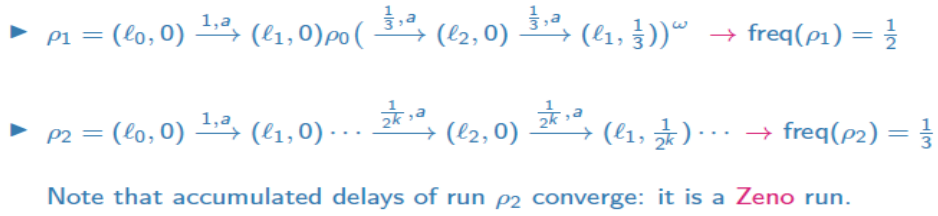


Figure 3: Exemples de fréquences.

Nous avions également envisagé des variantes de sémantiques avec seuil, un mot serait alors accepté s'il a une fréquence supérieure (ou inférieure) au seuil.

Dans le cadre de la sémantique des fréquences, on s'intéresse à des questions classiques sur les langages, telles que « le langage est-il vide ? » ou «le langage est-il universel ? ». Afin de répondre à ces questions, nous étudions l'ensemble des fréquences pour tous les mots dans un automate temporisé. Dans [BBB*11], nous avions montré que les bornes inférieures et supérieures de l'ensemble des fréquences des automates temporisés, à une seule horloge, peuvent être calculées en utilisant l'abstraction des coins (*corner-point abstraction)* introduite dans [BBL08]. Cette abstraction est un raffinement de la classique abstraction des régions. En plus de considérer les classes d'équivalence de valuations des horloges dans l'abstraction des régions (deux valuations sont dans la même région si, dans le futur, elle satisferont les mêmes gardes), dans l'abstraction des coins, on considère une abstraction des délais, ce qui permet d'approcher la notion de fréquence par une notion de ratio.

> **Exemple :** la Figure 4 décrit un automate temporisé et son abstraction des coins : à chaque région est attachée une information du positionnement approximatif dans la région (schématisé par un point), et abstrayant les délais concrets dans [0,1] par un entier 0 ou 1. Les transitions sont étiquetées par une action ou l'écoulement du temps (ε), et le ratio coût/récompense qui abstrait le ratio (temps écoulé dans la localité si acceptante/temps écoulé).

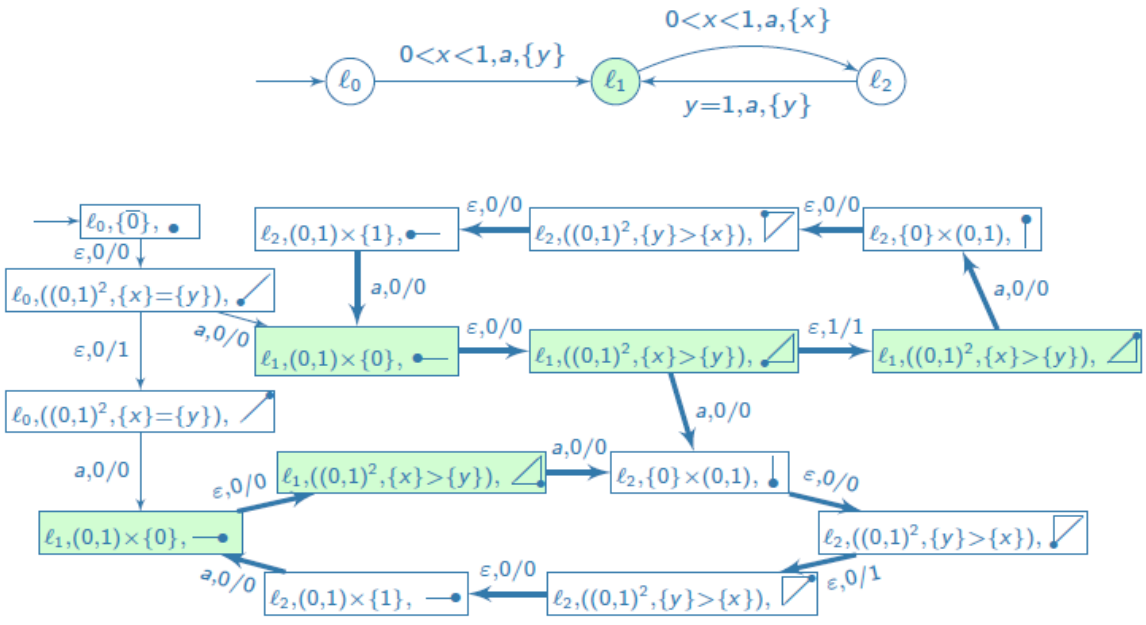


Figure 4: Automate temporisé à deux horloges et son abstraction des coins.

Le calcul des bornes inférieures et supérieures des fréquences peut alors être utilisé pour décider de la vacuité ou de l'universalité des langages avec fréquences positives (ou satisfaisant le seuil) pour les automates temporisés déterministes. Dans le cas plus général des automates non déterministes, le problème de l'universalité s'avère cependant être non primitif récursif (*i.e.* à la limite de la décidabilité) pour les automates temporisés avec une seule horloge, et même indécidable pour les automates à plusieurs horloges.

Les techniques utilisées dans [BBB*11] pour étudier l'ensemble des fréquences ne s'étendent donc malheureusement pas aux automates temporisés avec plusieurs horloges. Mais on peut remarquer que tous les contre-exemples à cette extension que nous avons pu construire s'appuient sur un phénomène de convergence entre les horloges le long des cycles. Notre travail est donc parti de l'étude fine de ces phénomènes de convergence.

Au delà de la *zenoness* (lorsque le temps converge le long d'une exécution), d'autres phénomènes de convergence entre les horloges ont été abordés pour la première fois dans [CHR02] et semblent être au cœur du problème pour l'étude des fréquences dans les automates temporisés à plusieurs horloges.

> **Exemple :** considérons à nouveau l'automate de la Figure 4, et la localité acceptante ℓ_1. Cet automate n'a aucune exécution Zénon (la durée d'un cycle de l'automate est d'une unité de temps), pourtant les contraintes des gardes dans le cycle imposent que les temps de séjour dans la localité acceptante ℓ_1 décroissent à chaque nouvelle visite. En conséquence, la fréquence d'une exécution infinie est de la forme 1-ε avec ε dans]0,1], l'ensemble des fréquences des exécutions étant alors l'intervalle [0,1[.

Comme pour la *zenoness*, ces phénomènes de convergence correspondent à des comportements non réalistes du point de vue de l'implémentabilité. En effet, l'observation de ces phénomènes de convergence des horloges nécessiterait une précision d'horloge infinie. Il est donc naturel de vouloir éliminer ces comportements dans les problèmes de vérification. Un moyen de détecter les cycles ne possédant pas de telles convergences (appelés cycles *forgetful*) a été récemment présenté dans [BA11]. Cette notion de *forgetfulness* est utilisée dans cet article afin de caractériser les langages temporisés ayant une entropie non-dégénérée, c'est à dire les langages dont le volume croît exponentiellement en fonction de la longueur des mots.

1.3. Automates temporisés forgetful

Dans le travail présenté ici, nous proposons d'étudier comment l'hypothèse de *forgetfulness* peut être utilisée pour calculer l'ensemble des fréquences dans les automates temporisés. Tout d'abord, nous montrons que la *forgetfulness* d'un cycle dans un automate temporisé à une horloge correspond au fait de ne pas forcer la convergence de l'horloge, c'est à dire que l'horloge est soit réinitialisée, soit non bornée le long de ce cycle, de sorte qu'on peut oublier sa valeur d'un cycle à l'autre. Plus généralement, un cycle d'un automate temporisé (à plusieurs horloges) est forgetful si sa projection dans l'abstraction des coins est fortement connexe. Un automate est *strongly forgetful* si tous ses cycles sont *forgetful* et *forgetful* si tous ses cycles simples sont *forgetful*. Notez que la *forgetfulness* n'implique pas que toutes les exécutions soient non-Zénon.

Exemple : l'automate à deux horloges de la Figure 4, dont on a expliqué les problèmes de convergence précédemment, est un contre-exemple à la *forgetfulness*. En effet, la projection du cycle de cet automate est le sous-graphe en traits gras dans son abstraction des coins, qui n'est pas une composante fortement connexe. *A contrario*, l'automate de la Figure 5 est *forgetful*, la projection de son cycle dans l'abstraction des coins (à droite sur la figure) étant fortement connexe.

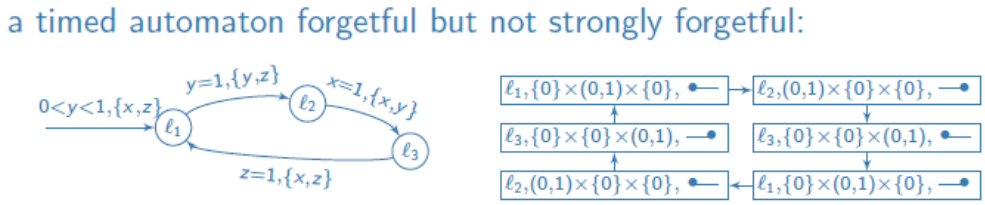


Figure 5: Exemple d'automate forgetful mais non fortement forgetful et son abstraction des coins.

Le point intéressant est que sous l'hypothèse de *strong forgetfulness* et quand le temps diverge nécessairement le long des exécutions (*strongly* non-Zénon), l'ensemble des fréquences peut être calculé de manière exacte, en utilisant l'abstraction des coins.

Ce résultat pour les automates temporisés *strongly forgetful* est aussi constructif que le théorème de [BBB*11] pour les automates temporisés à une seule horloge. Plus précisément, pour toute exécution σ de l'abstraction des coins et de ratio r (équivalent à la fréquence dans l'abstraction des coins), on sait construire une exécution dans l'automate temporisé qui se projette sur σ et ayant r comme fréquence. Ce résultat est donc plus précis que l'inclusion de l'ensemble des ratios de l'abstraction des coins dans l'ensemble des fréquences de l'automate temporisé. En revanche, pour pouvoir relâcher l'hypothèse de *strong forgetfulness* et traiter les automates temporisés *forgetful*, la preuve repose sur un ensemble d'exécutions canoniques de l'abstraction des coins dont les ratios couvrent l'ensemble de tous les ratios des exécutions de l'abstraction des coins.

Notre contribution peut également être comparée à celle de [BBL08] qui concerne les automates temporisés à doubles prix, *i.e.* les automates temporisés dont les localités sont munis de coûts et de récompenses s'accumulant proportionnellement au temps passé dans les localités. En effet, les automates temporisés avec fréquences en sont un cas particulier où les coûts sont 1 dans les localités acceptantes et nulles ailleurs, et les récompenses sont constantes. Dans [BBL08], soit une exécution de ratio minimal, soit une famille ε-optimale (*i.e.* des exécutions optimales pour tout $\varepsilon > 0$) est calculée, alors que, en supposant la *forgetfulness*, l'ensemble exact des fréquences peut être calculé. Nos techniques pour utiliser la *forgetfulness* pourraient ainsi être utiles pour les automates temporisés à doubles prix et peut-être plus généralement dans d'autres contextes.

L'article qui suit en annexe est la contribution majeure de notre travail sur l'analyse quantitative des automates temporisés. Il est publié dans

> A. Stainer. Frequencies in Forgetful Timed Automata. In proceedings of the 10th International Conference on Formal Modeling and Analysis of Timed Systems (Formats'12), LNCS, Volume 7595, London, UK, September 2012. http://www.irisa.fr/vertecs/Publis/Ps/Stainer-Format12.pdf

Les preuves sont disponibles dans le rapport de recherche :

> [S12]. Stainer, A.: Frequencies in forgetful timed automata. Research Report 8009, INRIA, Rennes, France (July 2012), http://hal.inria.fr/hal-00714262

L'article est structuré comme suit. L'introduction résume les éléments ci-dessus. Dans la section 2 *Preliminaries*, nous introduisons le modèle des automates temporisés, la sémantique quantitative basée sur les fréquences, la *forgetfulness* et l'abstraction *corner-point* comme outil pour l'étude des fréquences. Dans la section 3 *Computation of the Set of Frequencies in a One-Clock Timed*

Automaton, nous proposons une caractérisation de la *forgetfulness* pour les automates temporisés à une seule horloge, et fournissons une expression pour l'ensemble des fréquences dans cette classe restreinte. Enfin, la section 4 *Extension to n-Clock Timed Automata* traite des automates temporisés à plusieurs horloges et présente comment utiliser la *forgetfulness* pour s'assurer que, lorsque le temps diverge, l'ensemble des fréquences d'un automate temporisé est exactement l'ensemble des ratios de son abstraction des coins.

2. Références bibliographiques

[AD94]. Alur, R., Dill, D.L.: A theory of timed automata. Theoretical Computer Science 126(2), 183–235 (1994)
[ALT*01]. Alur, R., La Torre, S., Pappas, G.J.: Optimal Paths in Weighted Timed Automata. In: Di Benedetto, M.D., Sangiovanni-Vincentelli, A.L. (eds.) HSCC 2001. LNCS, vol. 2034, pp. 49–62. Springer, Heidelberg (2001)
[BBB*08]. Baier, C., Bertrand, N., Bouyer, P., Brihaye, Th., Größer, M.: Almost-sure model checking of infinite paths in one-clock timed automata. In: Proceedings of the 23rd Annual IEEE Symposium on Logic in Computer Science (LICS 2008), pp. 217–226. IEEE (2008)
[BA11]. Basset, N., Asarin, E.: Thin and Thick Timed Regular Languages. In: Fahrenberg, U., Tripakis, S. (eds.) FORMATS 2011. LNCS, vol. 6919, pp. 113–128. Springer, Heidelberg (2011)
[BFH*01]. Behrmann, G., Fehnker, A., Hune, T., Larsen, K.G., Pettersson, P., Romijn, J.M.T., Vaandrager, F.W.: Minimum-Cost Reachability for Priced Timed Automata. In: Di Benedetto, M.D., Sangiovanni-Vincentelli, A.L. (eds.) HSCC 2001. LNCS, vol. 2034, pp. 147–161. Springer, Heidelberg (2001)
[BBB*11]. Bertrand, N., Bouyer, P., Brihaye, T., Stainer, A.: Emptiness and Universality Problems in Timed Automata with Positive Frequency. In: Aceto, L., Henzinger, M., Sgall, J. (eds.) ICALP 2011, Part II. LNCS, vol. 6756, pp. 246–257. Springer, Heidelberg (2011)
[BBL08]. Bouyer, P., Brinksma, E., Larsen, K.G.: Optimal infinite scheduling for multi-priced timed automata. Formal Methods in System Design 32(1), 3–23 (2008)
[CHR02]. Cassez, F., Henzinger, T.A., Raskin, J.-F.: A Comparison of Control Problems for Timed and Hybrid Systems. In: Tomlin, C.J., Greenstreet, M.R. (eds.) HSCC 2002. LNCS, vol. 2289, pp. 134–148. Springer, Heidelberg (2002)
[KNS*02]. Kwiatkowska, M.Z., Norman, G., Segala, R., Sproston, J.: Automatic verification of real-time systems with discrete probability distributions. Theoretical Computer Science 282, 101–150 (2002).
[LMS04]. Laroussinie, F., Markey, N., Schnoebelen, P.: Model Checking Timed Automata with One or Two Clocks. In: Gardner, P., Yoshida, N. (eds.) CONCUR 2004. LNCS, vol. 3170, pp. 387–401. Springer, Heidelberg (2004)
[OW04]. Ouaknine, J., Worrell, J.: On the language inclusion problem for timed automata: Closing a decidability gap. In: Proceedings of the 19th IEEE Symposium on Logic in Computer Science (LICS 2004), pp. 54–63. IEEE (2004)
[P00]. Puri, A.: Dynamical properties of timed automata. Discrete Event Dynamic Systems 10(1-2), 87–113 (2000)
[S12]. Stainer, A.: Frequencies in forgetful timed automata. Research Report 8009, INRIA, Rennes, France (July 2012), `http://hal.inria.fr/hal-00714262`

3. Annexe

A. Stainer. Frequencies in Forgetful Timed Automata. In proceedings of the 10th International Conference on Formal Modeling and Analysis of Timed Systems (Formats'12), LNCS, Volume 7595, London, UK, September 2012. http://www.irisa.fr/vertecs/Publis/Ps/Stainer-Format12.pdf

Frequencies in Forgetful Timed Automata

Amélie Stainer

University of Rennes 1, Rennes, France

Abstract. A quantitative semantics for infinite timed words in timed automata based on the frequency of a run is introduced in [6]. Unfortunately, most of the results are obtained only for one-clock timed automata because the techniques do not allow to deal with some phenomenon of convergence between clocks. On the other hand, the notion of forgetful cycle is introduced in [4], in the context of entropy of timed languages, and seems to detect exactly these convergences. In this paper, we investigate how the notion of forgetfulness can help to extend the computation of the set of frequencies to n-clock timed automata.

1 Introduction

Timed automata have been introduced in [1]. This model is commonly used to represent real-time systems. A timed automaton is roughly a finite automaton equipped with a finite set of continuous clocks which evolve synchronously, are used in guards and can be reset along the transitions. The usual semantics of timed automata for infinite timed words is the Büchi semantics also presented in [1]. Recently, several works propose to add quantitative aspects in verification problems, such as costs [2,5] or probabilities [9,3].

In particular, one can refine the acceptance condition by considering the proportion of time elapsed in accepting locations. A quantitative semantics for infinite timed words based on this notion of frequency has thus been introduced in [6]. Lower and upper bounds of the set of frequencies of one-clock timed automata are computed using the corner-point abstraction, a refinement of the classical region abstraction, introduced in [7]. These bounds can be used to decide the emptiness of the languages with positive frequencies and the universality for deterministic timed automata. Furthermore, the universality problem is proved to be non primitive recursive for non-deterministic timed automata with one clock and undecidable with several clocks. The techniques from [6] do not extend to timed automata with several clocks, and all counterexamples rely on some phenomenon of convergence between clocks along cycles. Beyond zenoness (when time converges along a run), other convergence phenomena between clocks were first discussed in [8]. Similarly to zenoness, these convergences correspond to behaviors that are unrealistic from an implementability point of view. A way to detect cycles with no such convergences (called forgetful cycles) has been recently introduced in [4]. This notion of forgetfulness was used to characterize timed languages with a non-degenerate entropy.

M. Jurdziński and D. Ničković (Eds.): FORMATS 2012, LNCS 7595, pp. 236–251, 2012.

In this paper, we naturally propose to investigate how forgetfulness can be exploited to compute frequencies. First, we show that forgetfulness of a cycle in a one-clock timed automaton is equivalent to not forcing the convergence of the clock, that is the clock is reset or not bounded. Note that forgetfulness does not imply that all runs are non-Zeno. With this assumption, the set of frequencies can be exactly computed using the corner-point abstraction. Then, we show that in n-clock forgetful timed automata where time diverges necessarily along a run, the set of frequencies can also be computed thanks to the corner-point abstraction. On the one hand, the result for timed automata for which all cycles are forgetful (strong forgetfulness) is as constructive as the theorem of [6] over one-clock timed automata. On the other hand, to relax strong forgetfulness and consider timed automata whose *simple* cycles are forgetful, the proof relies on a set of canonical runs whose frequencies cover the set of all frequencies in the timed automaton.

Our contribution can also be compared with that of [7] on double priced timed automata, that is, timed automata with costs and rewards. Indeed, frequencies are a particular case of cost and reward functions. In [7], either a run of minimal ratio or an optimal family (*i.e.* ε-optimal runs for all $\varepsilon > 0$) is computed, whereas, assuming forgetfulness, the exact set of frequencies can be computed, not only the optimal ones. Our techniques might thus prove useful for double priced timed automata and maybe more generally in other contexts.

The paper is structured as follows. In the next section we introduce the model of timed automata, the quantitative semantics based on frequencies, forgetfulness and the corner-point abstraction as a tool to study frequencies. In Section 3, we propose a characterization of forgetfulness in one-clock timed automata and provide an expression for the set of frequencies for this restricted class. Last, Section 4 deals with n-clock timed automata and explains how to use forgetfulness to ensure that, when time diverges, the set of frequencies of a timed automaton and the set of ratios in its corner-point abstraction are equal.

All the details omitted, due to space constraints, in this paper, can be found in the research report [13].

2 Preliminaries

In this section, we recall the definition of timed automata with the quantitative semantics based on frequencies introduced in [6]. Then the corner-point abstraction is presented, firstly to define forgetfulness, and secondly as a tool to compute frequencies in timed automata.

2.1 Timed Automata and Frequencies

Given a finite set X of clocks, a *valuation* is a mapping $v : X \to \mathbb{R}_+$. The valuation associating 0 with all clocks is written $\overline{0}$ and $v + t$ is the valuation defined, for every clock x of X by $(v + t)(x) = v(x) + t$. For $X' \subseteq X$, $v_{[X' \leftarrow 0]}$ denotes the valuation equal to $\overline{0}$ for the clocks of X' and equal to v for the other

clocks. On the other hand, a *guard* over X is a finite conjunction of constraints of the form $x \sim c$ where $x \in X$, $c \in \mathbb{N}$ and $\sim \in \{<, \leq, =, \geq, >\}$. The set of guards over X is noted $G(X)$. Moreover, for a valuation v and a guard g, v *satisfies* g with the usual definition, is written $v \models g$.

Definition 1 (timed automaton). *A* timed automaton *is a tuple* $\mathcal{A} = (L, L_0, F, \Sigma, X, E)$ *such that:* L *is a finite set of locations,* $L_0 \subseteq L$ *is the set of initial locations,* $F \subseteq L$ *is the set of accepting locations,* Σ *is a finite alphabet,* X *is a finite set of clocks and* $E \subseteq L \times G(X) \times \Sigma \times 2^X \times L$ *is a finite set of edges.*

The *semantics* of a timed automaton $\mathcal{A}$ is given as a timed transition system $\mathcal{T}_{\mathcal{A}} = (S, S_0, S_F, (\mathbb{R}_+ \times \Sigma), \rightarrow)$ where $S = L \times \mathbb{R}_+^X$ is the set of states, $S_0 = L_0 \times \{\overline{0}\}$ is the set of initial states, $S_F = F \times \mathbb{R}_+^X$ is the set of accepting states and $\rightarrow \subseteq S \times (\mathbb{R}_+ \times \Sigma) \times S$ is the transition relation composed of all moves of the form $(\ell, v) \xrightarrow{\tau, a} (\ell', v')$ such that $\tau > 0$ and there exists an edge $(\ell, g, a, X', \ell') \in E$ with $v + \tau \models g$ and $v' = (v + \tau)_{[X' \leftarrow 0]}$.

A *run* of a timed automaton $\mathcal{A}$ is a finite or infinite sequence of moves starting in an initial state. In the sequel, unless otherwise stated, the run is assumed to be infinite. Thus, an infinite run $\rho = s_0 \xrightarrow{\tau_0, a_0} s_1 \xrightarrow{\tau_1, a_1} s_2 \xrightarrow{\tau_2, a_2} \cdots$ is said to be *Zeno* if $(\sum_{0 \leq j \leq i} \tau_j)_{i \in \mathbb{N}}$ is bounded.

Definition 2 (frequency). *Given* $\mathcal{A} = (L, L_0, F, \Sigma, X, E)$ *a timed automaton and* $\rho = (\ell_0, v_0) \xrightarrow{\tau_0, a_0} (\ell_1, v_1) \xrightarrow{\tau_1, a_1} (\ell_2, v_2) \cdots$ *an infinite run of* $\mathcal{A}$*, the* frequency *of* F *along* ρ*, denoted* $\mathsf{freq}_{\mathcal{A}}(\rho)$*, is defined as* $\limsup_{n \to \infty} \frac{\sum_{\{i \leq n \mid \ell_i \in F\}} \tau_i}{\sum_{i \leq n} \tau_i}$.

Note that, as in [6], the limit sup is an arbitrary choice. In the sequel, our goal is to compute the set of frequencies of the infinite runs of $\mathcal{A}$, which is written $\mathsf{Freq}(\mathcal{A})$. To do so, we sometimes distinguish $\mathsf{Freq}_Z(\mathcal{A})$ and $\mathsf{Freq}_{nZ}(\mathcal{A})$ which respectively denote the sets of frequencies of the Zeno and non-Zeno runs of $\mathcal{A}$. For example, Fig. 1 represents a timed automaton $\mathcal{A}$ with $F = \{\ell_1\}$ (accepting locations are colored in gray), such that $\mathsf{Freq}(\mathcal{A}) = \mathsf{Freq}_{nZ}(\mathcal{A}) = [0, 1[$. Indeed, there is no Zeno runs in $\mathcal{A}$ and there is an underlying constraint along the cycle which ensures that delays elapsed in the accepting location are decreasing. This implies that frequencies of an infinite run in $\mathcal{A}$ is of the form $\frac{1-\varepsilon}{\varepsilon}$ with $\varepsilon \in]0, 1]$.

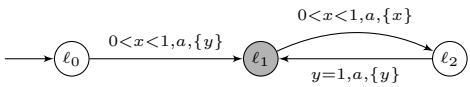


Fig. 1. A timed automaton $\mathcal{A}$ to illustrate the notion of frequency

2.2 Corner-Point Abstraction and Forgetfulness

Given the maximal constant M appearing in a timed automaton $\mathcal{A}$, the usual region abstraction forms a partition of the valuations over X, the clocks of $\mathcal{A}$.

In the following definition, $\lfloor t \rfloor$ and $\{t\}$ are respectively the integer part and the fractional part of the real t. The *region equivalence* $\equiv_{\mathcal{A}}$ over valuations of X is defined as follows: $v \equiv_{\mathcal{A}} v'$ if (i) for every clock $x \in X$, $v(x) \leq M$ iff $v'(x) \leq M$; (ii) for every clock $x \in X$, if $v(x) \leq M$, then $\lfloor v(x) \rfloor = \lfloor v'(x) \rfloor$ and $\{v(x)\} = 0$ iff $\{v'(x)\} = 0$ and (iii) for every pair of clocks $(x, y) \in X^2$ such that $v(x) \leq M$ and $v(y) \leq M$, $\{v(x)\} \leq \{v(y)\}$ iff $\{v'(x)\} \leq \{v'(y)\}$. The equivalence classes of this relation are called *regions* and $Reg_{\mathcal{A}}$ denotes the set of regions for the timed automaton $\mathcal{A}$. For each valuation v of the clocks of $\mathcal{A}$, there is a single region containing v, denoted by $R(v)$. A region R' is a *time-successor* of a region R if there exists $v \in R$ and $t \in \mathbb{R}_+$ such that $v + t \in R'$ and $R' \neq R$. The set of the time-successors of a region is naturally ordered, and the mapping $timeSucc : Reg_{\mathcal{A}} \to Reg_{\mathcal{A}}$ associates with any region, its first time-successor. The particular case of the region $\{\perp^X\}$ where all the clocks are larger than M is fixed as follows : $timeSucc(\{\perp^X\}) = \{\perp^X\}$.

Given a timed automaton, one can build a timed automaton having only region guards while preserving the set of frequencies. In fact, we need to extend the guards with constraints of the form $x - y \sim c$ where $x, y \in X$, $c \in \mathbb{N}$ and $\sim \in \{<, \leq, =, \geq, >\}$, but both models are known to be equivalent. In the sequel, timed automata are thus assumed to be split in regions and all the transitions can be fired. Moreover, in order to take into account that zero delays are not allowed in the semantics, transitions with a constraint of the form $x = 0$ are removed and transitions with a punctual constraint (of the form $x = c$) arrive directly in the time-successor (with constraint $x > c$) if x is not reset.

The corner-point abstraction is a refinement of the region abstraction, where states are formed of a region with one of its extremal points. Thus, an *$\mathcal{A}$-pointed region* (pointed region for short) is a pair (R, α) where R is a region and α an integer valuation ($\in (\mathbb{N}_{\leq M} \cup \perp)^X$, $\perp$ if the clock is not bounded in this region) belonging to the closure of R (for the usual topology), in this case, α is said to be a corner of R. The set of $\mathcal{A}$-pointed regions is written $Reg\bullet_{\mathcal{A}}$. The operations defined on the valuations of a set of clocks are extended in a natural way to the corners, with the convention that $M + 1 = \perp$ and $\perp + 1 = \perp$. Then the *timeSucc* function can be extended to pointed regions:

$$timeSucc(R, \alpha) = \begin{cases} (R, \alpha + 1) & \text{if } \alpha + 1 \text{ is a corner of } R \\ (timeSucc(R), \alpha') & \text{otherwise} \end{cases}$$

where $\forall x$, $\alpha'(x) = \alpha(x)$ if x is bounded in $timeSucc(R)$ and else $\alpha'(x) = \perp$.

Using this mapping, the construction of the corner-point abstraction is very similar to the usual region automaton.

Definition 3 (corner-point abstraction). *The* corner-point abstraction *of a timed automaton $\mathcal{A}$ (corner-point of $\mathcal{A}$ for short) is the finite automaton $\mathcal{A}_{cp} = (L_{cp}, L_{0,cp}, F_{cp}, \Sigma_{cp}, E_{cp})$ where $L_{cp} = L \times Reg\bullet_{\mathcal{A}}$ is the set of states, $L_{0,cp} = L_0 \times \{(\{\overline{0}\}, \overline{0})\}$ is the set of initial states, $F_{cp} = F \times Reg\bullet_{\mathcal{A}}$ is the set of accepting states, $\Sigma_{cp} = \Sigma \cup \{\varepsilon\}$, and $E_{cp} \subseteq L_{cp} \times \Sigma_{cp} \times L_{cp}$ is the finite set of edges defined as the union of discrete transitions and idling transitions:*

- discrete transitions: $(\ell, R, \alpha) \xrightarrow{a} (\ell', R', \alpha')$ *if there exists a transition* $\ell \xrightarrow{g,a,X'} \ell'$ *in* $\mathcal{A}$*, such that* $R = g$ *and* $(R', \alpha') = (R_{[X' \leftarrow 0]}, \alpha_{[X' \leftarrow 0]})$,
- idling transitions: $(\ell, R, \alpha) \xrightarrow{\varepsilon} (\ell, R', \alpha')$ *if* $(R', \alpha') = timeSucc(R, \alpha)$.

In particular, as a consequence of $\bot + 1 = \bot$, there is an idling loop on each state whose region is $\{\bot^X\}$. The *projection* of a (finite or infinite) run $\rho = (\ell_0, v_0) \xrightarrow{\tau_0, a_0} (\ell_1, v_1) \xrightarrow{\tau_1, a_1} \cdots$ of $\mathcal{A}$, denoted by $\mathsf{Proj}(\rho)$, is the set of runs of $\mathcal{A}_{cp}$ such that for all indices i, the i-th discrete transition goes from a state $(\ell_i, R(v_i + \tau_i), \alpha)$ to a state $(\ell_{i+1}, R(v_{i+1}), \alpha')$ and for all clocks $x \in X$, the number $\mu_i(x)$ of idling transitions of the form $(\ell, R, \alpha) \xrightarrow{\varepsilon} (\ell, R, \alpha + 1)$ since the last reset of x has to be equal to $\lfloor v_i(x) + \tau_i \rfloor$ or $\lceil v_i(x) + \tau_i \rceil$. Note that if x is bounded in a region $R(v_i + \tau_i)$, then $\mu_i(x)$ can be recovered from the associated corner α. Given $\varepsilon > 0$, we say that a (finite or infinite) run ρ *mimics up to* $\varepsilon > 0$ a (finite or infinite) run π in $\mathsf{Proj}(\rho)$ if, for all indices i, the i-th discrete transition of π goes from a state $(\ell_i, R(v_i), \alpha)$ such that, for all clock $x \in X$, if $\alpha(x) \neq \bot$ then $|v_i(x) + \tau_i - \alpha(x)| < \varepsilon$ and otherwise $|v_i(x) + \tau_i - \mu_i(x)| < \varepsilon$ (written $||v_i + \tau_i - \alpha|| < \varepsilon$ abusing notations).

In the sequel we often consider cycles of the graph of $\mathcal{A}$ (cycles of $\mathcal{A}$ for short), that is some sequences $\ell_0 \ell_1 \cdots \ell_n = \ell_0$ such that for all $0 \leq i \leq n-1$ there exists an edge from ℓ_i to ℓ_{i+1} in $\mathcal{A}$. Similarly to runs, we define the *projection of a cycle* C of $\mathcal{A}$, denoted by $\mathsf{Proj}(C)$. If C is a simple cycle with no region $\bot^X$, $\mathsf{Proj}(C)$ is the subgraph of $\mathcal{A}_{cp}$ covered by the projection of any finite run of $\mathcal{A}$ along C. If C is a simple cycle with some regions $\bot^X$, we simply add the idling loops associated with each states of the form $(\ell, \{\bot^X\}, \bot^X)$. To define the projection of a cycle C which is not simple, we first unfold the timed automaton $\mathcal{A}$ to obtain an equivalent simple cycle.

Forgetfulness was originally defined in [4] using the orbit graph. We choose here to give an alternative definition of forgetfulness based on the corner-point abstraction, which is less succinct, but will show useful for computing frequencies.

Definition 4 (forgetfulness)

- *A cycle* C *in a timed automaton is* forgetful *if* $\mathsf{Proj}(C)$ *is strongly connected;*
- *A timed automaton is* forgetful *if all its simple cycles are forgetful;*
- *A timed automaton is* strongly forgetful *if all its cycles are forgetful.*

Roughly speaking, forgetful cycles are cycles where some choices of current delays cannot impact forever on the future delays. These cycles can forget previous delays in their long term behaviors. Fig. 1 represents a timed automaton, inspired by [8], that is not forgetful. Indeed, the projection of the single cycle of this timed automaton is the subgraph with bold edges in its corner-point represented in Fig. 2, it is clearly not strongly connected. In fact, if from location ℓ_1 an a is read with x close to 0, it becomes impossible to read an a with x close to 1 in the future. More precisely, delays in ℓ_1 are smaller and smaller. Note that in Fig. 2, we did not draw the edges labelled by ε which lead to states from which no discrete transition can be fired in the future.

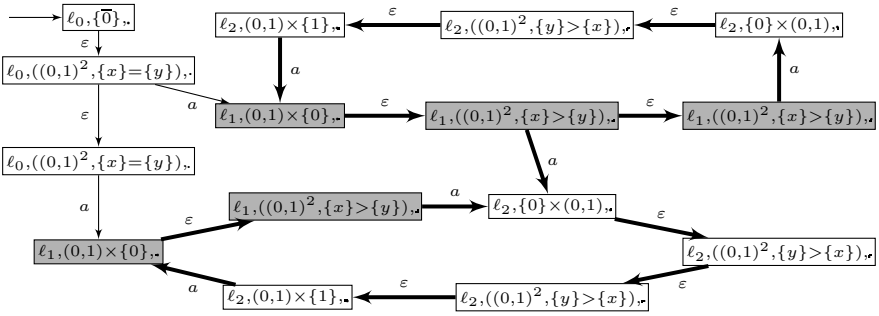


Fig. 2. Corner-point of the timed automaton from Fig. 1

We then define the notion of aperiodicity of a forgetful cycle and forgetful aperiodic timed automata.

Definition 5 (aperiodicity)

- *A forgetful cycle C in a timed automaton is* aperiodic *if for all $k \in \mathbb{N}$, the cycle obtained by the concatenation of k iterations of C is forgetful.*
- *A forgetful timed automaton is* aperiodic *if all its simple cycles are aperiodic;*

Strong forgetfulness trivially implies aperiodicity, whereas forgetfulness does not. Indeed Fig. 3 represents a timed automaton which is forgetful and periodic. The summary of its corner-point illustrates the periodicity. The cycle formed of two iterations of the simple cycle is not strongly connected, it has two distinct connected components. The projection of a forgetful cycle C in $\mathcal{A}_{cp}$ is strongly connected, then given any state s of $\mathcal{A}_{cp}$ in $\mathsf{Proj}(C)$, there are some simple cycles containing s. Intuitively, such a cycle corresponds to a number of iterations of C in $\mathcal{A}$, this is the number of non-consecutive occurrences of states sharing the same location of $\mathcal{A}$ as s. Thus, we can characterize the aperiodicity of a forgetful cycle by a notion of pseudo aperiodicity of its projection.

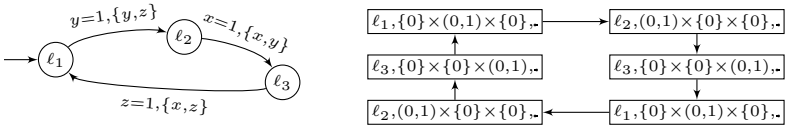


Fig. 3. A forgetful and periodic timed automaton

Proposition 1. *A forgetful cycle C is aperiodic if and only if* $(*)$ *the greatest common divisor, over the simple cycles D of $\mathsf{Proj}(C)$, of the numbers of iterations of C corresponding to D, is* 1.

The characterization $(*)$ of aperiodicity allows one to check it in the corner-point abstraction. The notion of aperiodicity will be a key for the relaxation of strong forgetfulness in the second part of Section 4.3.

2.3 The Corner-Point Abstraction as a Tool for Frequencies

In the corner-point, the idling transitions which do not change the current region correspond to an elapse of one time unit. In the same way as in [6], these abstract delays are used to abstract the frequencies in a timed automaton by ratios in its corner-point abstraction. To do so, the corner-point is equipped with costs and rewards as follows:

- the *reward* of a transition is 1 if it is of the form $(\ell, R, \alpha) \xrightarrow{\varepsilon} (\ell, R, \alpha')$ and 0 otherwise;
- the *cost* of a transition is 1 if the reward is 1 and the location ℓ is accepting and 0 otherwise.

In particular, the loops on the states whose region is $\{\perp^X\}$ have reward 1. Thanks to these costs and rewards, the *ratio* of an infinite run of the corner-point can be defined, similarly to the frequency in the timed automaton, as the limit sup of the ratios of the accumulated costs over the accumulated rewards. An infinite run in the corner-point is said *reward-converging* (resp. *reward-diverging*) if the accumulated reward is finite (resp. not bounded). This notion is close to zenoness of runs in a timed automaton even if some Zeno runs could be projected to reward-diverging runs in the corner-point abstraction and, the other way around, non-Zeno runs could be projected to reward-converging runs. Thus, we write $\mathsf{Rat}(\mathcal{A}_{cp})$, $\mathsf{Rat}_{r-d}(\mathcal{A}_{cp})$ and $\mathsf{Rat}_{r-c}(\mathcal{A}_{cp})$ for the sets of the ratios of the infinite runs in $\mathcal{A}_{cp}$, the reward-diverging runs in $\mathcal{A}_{cp}$ and the reward-converging ones. We also say *reward-diverging* for a cycle of $\mathcal{A}_{cp}$ whose accumulated reward is positive.

A cycle of $\mathcal{A}_{cp}$ is said *accepting* (resp. *non-accepting*) if all its locations are accepting (resp. non-accepting) and it is said *mixed* if it has both accepting and non-accepting locations.

In the sequel, we often use the following results established in [6] and [7].

Lemma 1 ([6]). *For every run ρ in a one-clock timed automaton $\mathcal{A}$, there are two runs π and π' in $\mathsf{Proj}(\rho)$ respectively minimizing and maximizing the ratio such that: $\mathsf{Rat}(\pi) \leq \mathsf{freq}_{\mathcal{A}}(\rho) \leq \mathsf{Rat}(\pi')$.*
These runs are respectively called the contraction and the dilatation of ρ.

Lemma 2 ([6]). *Let $\{S_1, \cdots, S_k\}$ be the set of SCCs of $\mathcal{A}_{cp}$. The set $\mathsf{Rat}_{r-d}(\mathcal{A}_{cp})$ of ratios of reward-diverging runs in $\mathcal{A}_{cp}$ is equal to $\bigcup_{S_i \in SCC}[m_i, M_i]$ where m_i and M_i are the minimal and the maximal ratios for reward-diverging cycles in S_i.*
Moreover, if $\mathcal{A}$ has a single clock, then $\mathsf{Freq}_{nZ}(\mathcal{A}) = \mathsf{Rat}_{r-d}(\mathcal{A}_{cp})$.

Lemma 3 ([7]). *Consider a transition $(\ell, R, \alpha) \to (\ell', R', \alpha')$ in $\mathcal{A}_{cp}$, take a valuation $v \in R$ such that $\delta(v) < \varepsilon$ and $|v(x) - \alpha(x)| = \mu_v(x)$ with $\mu_v(x) = \min\{|v(x) - p| \ p \in \mathbb{N}\}$, $\nu_v(x, y) = \min\{|v(x) - v(y) - p| \ p \in \mathbb{N}\}$ and $\delta(v) = \max(\{\mu_v(x)\} \cup \{\nu_v(x, y)\})$. There exists a valuation $v' \in R'$ such that $(\ell, v) \to (\ell', v')$ in $\mathcal{A}$, $\delta(v') < \varepsilon$ and $|v'(x) - \alpha'(x)| = \mu_{v'}(x)$.*

In particular, the latter lemma implies by induction that any run in $\mathcal{A}_{cp}$ can be mimicked in $\mathcal{A}$ up to any $\varepsilon > 0$.

3 Computation of the Set of Frequencies in a One-Clock Timed Automaton

One-clock timed automata have simpler clock behaviors than the general model. In fact, having a single clock in a timed automaton is quite close to forgetfulness in the sense that each time the clock is reset, the timed automaton forgets all the timing information. In this section, we present a new characterization of forgetfulness and we show the equivalence between forgetfulness and strong forgetfulness when there is a single clock. Last, we propose an expression for the set of frequencies of forgetful one-clock timed automata.

In a one-clock timed automaton, a reset of the clock along a cycle is linked to forgetfulness. The following lemma states the precise characterization of forgetful cycles inspired by this observation.

Proposition 2. *Let C be a cycle of a one-clock timed automaton. Then, C is forgetful if and only if the clock is reset or not bounded along C.*

In fact, Proposition 2 implies that the cycle obtained by concatenation of any sequence of forgetful cycles in a one-clock timed automaton is also forgetful. Indeed, if the clock is reset or not bounded along each cycle of the sequence, it is clearly the case for the sequence itself.

Corollary 1. *A one-clock timed automaton is forgetful iff it is strongly forgetful.*

Recall that, as illustrated in Fig. 1, Corollary 1 (as well as Proposition 2) does not hold for n-clock timed automata.

Let us now consider the set of the frequencies in a one-clock timed automaton. By Lemma 2, if there are only non-Zeno runs in a timed automaton, then the set of the frequencies equals to the set of the ratios in the corner-point. Firstly, the particular case where a timed automaton has a reward-converging cycle in its corner-point containing both accepting and non-accepting locations is easy to treat as stated in the following proposition.

Proposition 3. *Let $\mathcal{A}$ be a forgetful one-clock timed automaton. If there is a mixed reward-converging cycle in its corner-point $\mathcal{A}_{cp}$, then* $\mathsf{Freq}_Z(\mathcal{A}) =]0,1[$ *and* $\mathsf{Freq}_{nZ}(\mathcal{A}) = [0,1]$.

Now, for the general case, it is possible to consider only timed automata which do not have such cycles in their corner-point. This allows us to give a general expression for the set of frequencies of a forgetful one-clock timed automaton. For readability, let us define some notations. Given C a cycle of $\mathcal{A}$ having a reward-converging cycle in its projection, we write $p(C)$ for the set of ratios of cycle-free prefixes ending in reward-converging cycles of $\mathsf{Proj}(C)$ and $c(C)$ for the set of ratios of co-reachable reward-diverging cycles. By convention, we let $\max(\emptyset) = -1$ and $\min(\emptyset) = 2$. Then, we define $M(C) = \max(p(C) \cup c(C))$, $m(C) = \min(p(C) \cup c(C))$,

$$M(\mathcal{A}_{cp}) = \max\{M(C) \mid C \text{ acc. cycle of } \mathcal{A} \text{ with a r.-c. cycle in } \mathsf{Proj}(C)\} \text{ and}$$

$$m(\mathcal{A}_{cp}) = \min\{m(C) \mid C \text{ non-acc. cycle of } \mathcal{A} \text{ with a r.-c. cycle in } \mathsf{Proj}(C)\}.$$

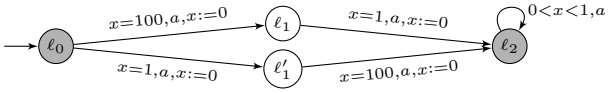


Fig. 4. A non-forgetful counterexample for Theorem 1

Theorem 1. *Let $\mathcal{A}$ be a forgetful one-clock timed automaton. If there is a mixed reward-converging cycle in $\mathcal{A}_{cp}$, then* $\mathsf{Freq}_Z(\mathcal{A}) =]0,1[$ *and* $\mathsf{Freq}_{nZ}(\mathcal{A}) = [0,1]$. *Otherwise:* $\mathsf{Freq}(\mathcal{A}) = \mathsf{Rat}_{r-d}(\mathcal{A}_{cp}) \cup \big[0, M(\mathcal{A}_{cp})\big[\cup \big]m(\mathcal{A}_{cp}), 1\big]$.

Proof. The first part of Theorem 1 is established in Proposition 3, let us now assume that there is no mixed reward-converging cycles in $\mathcal{A}_{cp}$. By Lemma 2, for non-Zeno runs: $\mathsf{Freq}_{nZ}(\mathcal{A}) = \mathsf{Rat}_{r-d}(\mathcal{A}_{cp})$. The rest of the proof is based on the following lemma dealing with plain reward-converging cycles.

Lemma 4. *Let C be a cycle of a one-clock forgetful timed automaton $\mathcal{A}$:*

- *If* $\mathsf{Proj}(C)$ *contains a non-accepting reward-converging cycle, then the set of frequencies of the infinite runs of $\mathcal{A}$ ending in C is* $\big[0, M(C)\big[$.
- *If* $\mathsf{Proj}(C)$ *contains an accepting reward-converging cycle , then the set of frequencies of the infinite runs of $\mathcal{A}$ ending in C is* $\big]m(C), 1\big]$.

Back to the proof of the second part of Theorem 1, the inclusion from right to left is straightforward from the non-Zeno case and Lemma 4.

Thanks to the equality in the non-Zeno case, the inclusion from left to right is only needed for the subset $\mathsf{Freq}_Z(\mathcal{A})$. Let thus ρ be a Zeno run. It can be projected on a reward-converging run in the corner-point. This projection necessarily ends in a strongly connected subgraph of the corner-point having zero rewards and containing only accepting locations or only non-accepting locations. We study the case where all the locations of the end are non-accepting, the other case is symmetric. By Lemma 4, the prefix of ρ corresponding to the prefix of the projection before the infinite suffix in the subgraph has a frequency smaller than $M(C)$ for a cycle C having a reward-converging projection. To conclude, the frequency of ρ is smaller than the prefix because all the locations of the suffix are non-accepting. □

Note that if the timed automaton is not forgetful, the form of the set of the frequencies can be very different from the expression given in Theorem 1. Fig. 4 gives an example of non-forgetful timed automaton such that $\mathsf{Freq}(\mathcal{A}) = \big]\frac{1}{101}, \frac{2}{102}\big[\cup \big]\frac{100}{101}, \frac{101}{102}\big[$. There is no reward-diverging run in $\mathcal{A}_{cp}$, $M(\mathcal{A}_{cp}) = -1$ because there is no accepting reward-converging cycle in $\mathcal{A}_{cp}$ and $m(\mathcal{A}_{cp}) = \frac{1}{101}$, hence the expected set of frequencies would be $]\frac{1}{101}, 1]$. The difference with forgetful timed automata is that the accumulated delays in ℓ_2 cannot diverge, therefore it is not possible to increase the frequency as much as necessary. In particular, there is no infinite run of frequency 1. More generally, this example illustrates a simple manner to obtain, for the set of frequencies, any finite union of open intervals included in $[0,1]$.

4 Extension to n-Clock Timed Automata

There is a real gap between one-clock timed automata and n-clock timed automata. For example, in [10], the reachability problem for one-clock timed automata is proved to be NLOGSPACE-complete, whereas it becomes NP-hard with two clocks. As an other example, the language inclusion problem which is undecidable in the general case [1], becomes decidable with at most one clock [11]. In this section, we use forgetfulness and time divergence to compute the set of frequencies in n-clock timed automata. Note that these assumptions are strong but can be justified by implementability concerns.

4.1 Forgetfulness in n-Clock Timed Automata

The goal is to find some reasonable assumptions to obtain a class of timed automata whose sets of frequencies are exactly sets of ratios of their corner-point abstractions. We do not want to complexity our problem dealing with Zeno runs as we did in one-clock timed automata for which the Zeno case is already non-trivial. More precisely, we want to extend the result $\mathsf{Freq}_{nZ}(\mathcal{A}) = \mathsf{Rat}_{r-d}(\mathcal{A}_{cp})$ of [6], from one-clock timed automata to n-clock timed automata. To do so, we first assume that timed automata are strongly non-Zeno, that is in every cycle there is one clock which is reset and lower guarded by a positive constant. This implies that there is no reward-converging run in its corner-point (strong reward-divergence [7]). For one-clock timed automata strong non-zenoness is strictly stronger than forgetfulness and implies that $\mathsf{Freq}(\mathcal{A}) = \mathsf{Freq}_{nZ}(\mathcal{A}) = \mathsf{Rat}_{r-d}(\mathcal{A}_{cp})$. Unfortunately, this assumption is not sufficient for n-clock timed automata. For example, the timed automaton in Fig. 5, taken from [6], is strongly non-Zeno and such that $\mathsf{Freq}(\mathcal{A}) =]0,1] \neq \{0\} \cup \{1\} = \mathsf{Rat}(\mathcal{A}_{cp})$. In fact, this timed automaton is a typical example of forgetful timed automaton. Delays in ℓ_1 have to be larger and larger along cycles, which ensures that frequency 0 cannot be reached in $\mathcal{A}$. On the contrary, in $\mathcal{A}_{cp}$, either the accumulated reward in ℓ_1 is 0 (ratio 0) or there is one idling transition with reward 1 from a state of $\mathcal{A}_{cp}$ with location ℓ_1 and in the future, there always are such transitions in states of the form (ℓ_1, R, α) (ratio 1). Therefore, except over one-clock timed automata, forgetfulness and strong reward-divergence are not comparable.

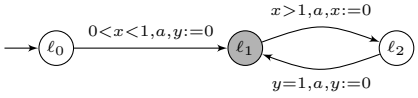


Fig. 5. A non-forgetful strongly non-Zeno timed automaton

The following theorem is the first illustration of the utility of forgetfulness to compute the set of frequencies in timed automata with several clocks.

Theorem 2. *Let $\mathcal{A}$ be a strongly non-Zeno and forgetful timed automaton. Then* $\mathsf{Freq}(\mathcal{A}) \subseteq \mathsf{Rat}(\mathcal{A}_{cp})$.

Proof (sketch). The idea is that the infinite run consisting in an infinite iteration of the cycle of minimal ratio in $\mathcal{A}_{cp}$ has a ratio smaller than the frequency of any infinite run in $\mathcal{A}$. Symmetrically, there is a run of ratio larger than the frequency of any infinite run in $\mathcal{A}$. The theorem is thus straightforward if there is a single SCC in $\mathcal{A}_{cp}$. Otherwise, forgetfulness is the key to obtain the inclusion $\mathsf{Freq}(\mathcal{A}) \subseteq \mathsf{Rat}(\mathcal{A}_{cp})$ instead of simple bounds. Indeed, given an infinite run ρ ending in an SCC of $\mathcal{A}$, by forgetfulness of the cycles, all the projections of ρ end in the same SCC of $\mathcal{A}_{cp}$. The proof can thus be done in this SCC by neglecting the prefix thanks to time divergence. □

In the sequel, we see how strongly non-zenoness and forgetfulness can be useful to obtain the other inclusion. The problem is not trivial even under these assumptions and the proof techniques could certainly be interesting in different contexts. This section allows to understand several subtleties of forgetfulness.

4.2 Techniques to Compute the Frequencies

In this section, we explain the technical aspects which allow the extension to n-clock timed automata. First, thanks to Lemma 3 we know that any infinite run in a corner-point $\mathcal{A}_{cp}$ can be mimicked up to any $\varepsilon > 0$. This lemma implies that respective lower and upper bounds of the sets of ratios and frequencies are equal, but as seen with the timed automaton in Fig. 5, $\mathsf{Freq}(\mathcal{A})$ can be very different from $\mathsf{Rat}(\mathcal{A}_{cp})$ when $\mathcal{A}$ is not forgetful. Second, the following lemma established in [12] expresses the preservation of some barycentric relations between valuations along cycles.

Lemma 5 ([12]). *Let $\mathcal{A}$ be a timed automaton and an edge (ℓ, g, a, X', ℓ') such that $(\ell, v) \to (\ell', v')$ and $(\ell, w) \to (\ell', w')$ with $R(v) = R(w)$ and $R(v') = R(w')$, then for any $\lambda \in [0, 1]$ $(\ell, \lambda v + (1-\lambda)w) \to (\ell', \lambda v' + (1-\lambda)w')$.*

Naturally, this lemma can be extended to finite sequences of edges by induction.

The combination of both lemmas helps us to prove that if along a given cycle one can go from every corner to a fixed corner α, then along this cycle one can go as close to α as necessary in $\mathcal{A}$. This way of reducing the distance to corners is the key for the extension to n-clock timed automata. Indeed, when time diverges (non-zenoness), if an infinite run ρ in $\mathcal{A}$ mimics an infinite run π of $\mathcal{A}_{cp}$ up to ε converging to 0 along ρ (*i.e.* for all ε there is a suffix of ρ which mimics the corresponding suffix of π up to ϵ), then $\mathsf{freq}_{\mathcal{A}}(\rho) = \mathsf{Rat}(\pi)$.

Lemma 6. *Let $\mathcal{A}$ be a timed automaton and $\rho = (\ell_0, v_0) \xrightarrow{\tau_0, a_0} (\ell_1, v_1) \xrightarrow{\tau_1, a_1} \cdots \xrightarrow{\tau_{n-1}, a_{n-1}} (\ell_0, v_n)$ with $R(v_0) = R(v_n) =: r$ be a finite run of $\mathcal{A}$. If given a corner α_n of the region r and for all (ℓ_0, r, α) there is a finite run from (ℓ_0, r, α) to (ℓ_0, r, α_n) in $\mathsf{Proj}(\rho)$, then for all $\varepsilon > 0$, there exists $\rho' = (\ell_0, v_0) \xrightarrow{\tau'_0, a_0} (\ell_1, v'_1) \cdots \xrightarrow{\tau'_{n-1}, a_{n-1}} (\ell_0, v'_n)$ such that $\mathsf{Proj}(\rho') = \mathsf{Proj}(\rho)$ and $||v'_n - \alpha_n|| < \varepsilon$.*

Proof. Let us start by fixing, for all corners α, a valuation v_α^ε in r which is very close to α. Thanks to these valuations, we then define a barycentric expression

for v_0. Let Ω_r the set of the corners of r. As the closure of r is the convex hull of Ω_r (for the usual topology of $\mathbb{R}^X$), there exists $(v^\varepsilon_\alpha \in r)_{\alpha\in\Omega_r}$ and $(\lambda_\alpha \in [0,1])_{\alpha\in\Omega_r}$ such that $\sum_{\alpha\in\Omega_r}\lambda_\alpha = 1$, $v_0 = \sum_{\alpha\in\Omega_r}\lambda_\alpha v^\varepsilon_\alpha$ and $||v^\varepsilon_\alpha - \alpha|| < \varepsilon$. By assumptions, there are some paths in $\mathsf{Proj}(\rho)$ going from each α to α_n. Thanks to Lemma 3, there are some finite runs from each of our valuations v^ε_α very close to the corners α to some valuations $v^\varepsilon_{\alpha,\alpha_n}$ very close to a common corner α_n. Formally, there exists $(v^\varepsilon_{\alpha,\alpha_n} \in r)_{\alpha\in\Omega_r}$ and some finite runs $(\rho_\alpha)_{\alpha\in\Omega_r}$ from $(\ell_0, v^\varepsilon_\alpha)$ to $(\ell_0, v^\varepsilon_{\alpha,\alpha_n})$ in $\mathcal{A}$ with $||v^\varepsilon_{\alpha,\alpha_n} - \alpha_n|| < \varepsilon$. Then, by Lemma 5, there is a finite run ρ' from (ℓ_0, v_0) to the state with location ℓ_0 and the valuation equal to the barycenter of the valuations $v^\varepsilon_{\alpha,\alpha_n}$ which is very close to α_n by the triangle inequality. Formally, there is a finite run ρ' from $(\ell_0, v_0 = \sum_{\alpha\in\Omega_r}\lambda_\alpha v^\varepsilon_\alpha)$ to $(\ell_0, \sum_{\alpha\in\Omega_r}\lambda_\alpha v^\varepsilon_{\alpha,\alpha_n})$. To conclude, ρ' is as needed because, by the triangle inequality, $||\sum_{\alpha\in\Omega_r}\lambda_\alpha v^\varepsilon_{\alpha,\alpha_n} - \alpha_n|| \leq \sum_{\alpha\in\Omega_r}\lambda_\alpha||v^\varepsilon_{\alpha,\alpha_n} - \alpha_n|| < \varepsilon$. □

To use Lemma 6, we need to find a cycle in $\mathcal{A}_{cp}$ which allows, in some sense, to synchronize all the corners of a region to a common one. Indeed, each run in $\mathcal{A}_{cp}$ corresponds to a run in $\mathcal{A}$, the existence of ρ is not a real constraint. Moreover, Lemma 6 does not depend on forgetfulness of timed automata. The following lemma illustrates how forgetfulness can help to use Lemma 6.

Lemma 7. *Let $\mathcal{A}$ be a timed automaton, X its set of clocks and a sequence $(c_i)_{1\leq i\leq K}$ with $K = 2^{|X|+1}$, of forgetful cycles containing the location ℓ of $\mathcal{A}$ such that all the cycles obtained by concatenation of the cycles of a subsequence $(c_k)_{1\leq i\leq k\leq j\leq K}$ are forgetful. Then for all pairs of corners (α, α') of the region R associated to ℓ, there is a finite run of the form $(\ell, R, \alpha) \xrightarrow{\pi_1} (\ell, R, \alpha_1) \xrightarrow{\pi_2} \cdots (\ell, R, \alpha_{K-1}) \xrightarrow{\pi_K} (\ell, R, \alpha')$ such that for all indices i, π_i corresponds to one iteration of c_i.*

Proof. Abusing notations we write $\pi \in \mathsf{Proj}(c)$ for "π corresponds to one iteration of c" Consider the subset construction with $s_0 = \{(\ell, R, \alpha)\}$ and $s_{i+1} = \{(\ell, R, \beta') \,|\, \exists(\ell, R, \beta) \in s_i, \exists\pi'_i \in \mathsf{Proj}(c_i), \text{ s.t. } (\ell, R, \beta) \xrightarrow{\pi'_i} (\ell, R, \beta')\}$.

First, there are at most $|X| + 1$ corners in R, hence there are at most $K = 2^{|X|+1}$ subsets of $(\ell, R, \text{all}) := \{(\ell, R, \alpha) \,|\, \alpha \text{ corner of } R\}$. Second, by forgetfulness of the c_i's, if $s_i = (\ell, R, \text{all})$ then for all $j > i$, $s_j = (\ell, R, \text{all})$. Third, there is no other cycles in the subset construction. Indeed, if there exists indices $i < j$ such that $s_i = s_j \neq (\ell, R, \text{all}) := \{(\ell, R, \alpha) \,|\, \alpha \text{ corner of } R\}$ then the cycle obtained by concatenation of cycles $c_{i+1}, \cdots, c_j$ is not forgetful, which contradicts strong forgetfulness.

As a consequence, the subset construction loops in (ℓ, R, all) forever after a cycle-free prefix whose length is thus smaller than K. Hence, there is a finite run of the form $(\ell, R, \alpha) \xrightarrow{\pi_1} (\ell, R, \alpha_1) \xrightarrow{\pi_2} \cdots (\ell, R, \alpha_{K-1}) \xrightarrow{\pi_K} (\ell, R, \alpha')$ such that for all indices i, $\pi \in \mathsf{Proj}(c_i)$. □

In the next sections we use these two lemmas to prove that our assumptions are sufficient to ensure the existence of such synchronizing cycles along infinite runs that we want to mimic in $\mathcal{A}$.

4.3 Frequencies in n-Clock Forgetful Timed Automata

We first consider the case of strongly forgetful timed automata. Thanks to Lemma 6 and by observing the consequences of forgetfulness of all the cycles in a timed automaton, we obtain a theorem which is as constructive as the corresponding result for one-clock timed automata from [6].

Theorem 3. *Let $\mathcal{A}$ be a strongly non-Zeno strongly forgetful timed automaton. Then, for every infinite run π in the corner-point of $\mathcal{A}$, there exists an infinite run ρ_π in $\mathcal{A}$ such that $\pi \in \mathsf{Proj}(\rho_\pi)$ and $\mathsf{freq}_{\mathcal{A}}(\rho_\pi) = \mathsf{Rat}(\pi)$.*

The idea is to prove, for every run π in $\mathcal{A}_{cp}$, the existence of synchronizing cycles infinitely often along π which allow to mimic it up to an ε converging to 0.

Proof. Along the infinite run π of $\mathcal{A}_{cp}$, there is a pair (ℓ, R) which appears infinitely often, possibly with different corners. Let $(\ell, R, \alpha_i)_{i\in\mathbb{N}}$ be a sequence of the occurrences of (ℓ, R) and $(\pi_i)_{i\in\mathbb{N}}$ the sequence of factors of π leading respectively from (ℓ, R, α_i) to (ℓ, R, α_{i+1}). Each π_i corresponds to a forgetful cycle c_i in $\mathcal{A}$ hence by Lemma 7, for all pairs (α, α') of corners of the region R, there is a finite run of the form $(\ell, R, \alpha) \xrightarrow{\pi'_1} (\ell, R, \alpha_1) \xrightarrow{\pi'_2} \cdots (\ell, R, \alpha_{K-1}) \xrightarrow{\pi'_K} (\ell, R, \alpha')$ with $K = 2^{|X|+1}$ and such that for all indices i, π corresponds to one iteration of c_i. In particular, this finite run belongs to the projections of exactly the same runs as $\pi_1 \cdot \pi_2 \cdots \pi_K$. As a consequence, for any finite run $\rho = (\ell, v_0) \xrightarrow{\tau_0,a_0} (\ell_1, v_1) \xrightarrow{\tau_1,a_1} \cdots \xrightarrow{\tau_{n-1},a_{n-1}} (\ell, v_n)$ with $R(v_0) = R(v_n) = R$ and such that $\pi_0.\pi_1 \cdots \pi_K \in \mathsf{Proj}(\rho)$, for any corner β_n of the region R and for all (ℓ, R, α) there is a finite run from (ℓ, R, α) to (ℓ, R, β_n) in $\mathsf{Proj}(\rho)$. Hence, Lemma 6 can be applied to such finite runs. Then, for any ε and given ρ^i a mimicking of π until (ℓ, R, α_i), Lemma 6 ensures the existence of a extension of ρ^i to ρ^{i+K} mimicking π until (ℓ, R, α_{i+K}), such that $||v - \alpha_{i+K}|| < \varepsilon$ where v is the last valuation of ρ^{i+K}. In words, it is possible to fix some finite factors along π which allow to go as close as necessary from a corner of π along a mimicking ρ. Out of these factors, the distance to the corners of π can be preserved (Lemma 3). To conclude, these factors can be placed infinitely often to allow the convergence of the distance to the corner of π to 0, but as rarely as necessary to be neglected in the computation of the frequency. □

Theorem 3 implies that the set of ratios $\mathsf{Rat}(\mathcal{A}_{cp})$ is included in the set of frequencies $\mathsf{Freq}(\mathcal{A})$. This implies, together with Theorem 2, that if $\mathcal{A}$ is a strongly non-Zeno and strongly forgetful timed automaton, then $\mathsf{Freq}(\mathcal{A})$ is equal to $\mathsf{Rat}(\mathcal{A}_{cp})$. Strong forgetfulness is a realistic assumption from an implementability point of view, but is not satisfactory because of its difficulties to be checked. Indeed, checking if a cycle is forgetful can be done thanks to the corner-point, but there is an unbounded number of cycles in a timed automaton and we do not know any property which would allow, in general, to avoid to check them all. As a consequence, it is important to relax this assumption. We did not succeed in proving that strong forgetfulness can be relaxed in Theorem 3. Nevertheless, the inclusion $\mathsf{Rat}(\mathcal{A}_{cp}) \subseteq \mathsf{Freq}(\mathcal{A})$ still holds when strong forgetfulness is replaced by

forgetfulness and aperiodicity, both of which can be checked on the corner-point abstraction.

Theorem 4. *Let $\mathcal{A}$ be a strongly non-Zeno, forgetful and aperiodic timed automaton. Then,* $\mathsf{Rat}(\mathcal{A}_{cp}) \subseteq \mathsf{Freq}(\mathcal{A})$.

Proof (sketch). The idea is to prove that, for every $\mathsf{rat} \in \mathsf{Rat}(\mathcal{A}_{cp})$, there exists an infinite run π_{rat} in $\mathcal{A}_{cp}$ of ratio rat and such that there exists a infinite run ρ_π of $\mathcal{A}$ with $\mathsf{freq}_{\mathcal{A}}(\rho_\pi) = \mathsf{Rat}(\pi_{\mathsf{rat}})$ and $\pi_{\mathsf{rat}} \in \mathsf{Proj}(\rho_\pi)$. Thanks to Lemma 2 and by reward-divergence, we have the following expression for the set of the ratios $\mathsf{Rat}(\mathcal{A}_{cp}) = \mathsf{Rat}_{r-d}(\mathcal{A}_{cp}) = \bigcup_{S_i \in SCC}[m_i, M_i]$. In fact, for all i, each value $\mathsf{rat} \in [m_i, M_i]$ is the ratio of a run π_{rat} in $\mathcal{A}_{cp}$ which alternates with the suitable proportions some cycles c_i of ratio m_i and C_i of ratio M_i in S_i. The prefix to go to c_i and the finite runs to go from a cycle to the other are neglected in the computation of the ratio by performing sufficiently many iterations at each step. Such a π_{rat} can be mimicked up to any $\varepsilon > 0$ (Lemma 3). We thus use Lemmas 7 and 6 to decrease ε. The finite runs to go from C_i to c_i are simply concatenated with $2^{|X|+1}$ iterations of c_i. The cycle c_i corresponds, in $\mathcal{A}_{cp}$ to a cycle (simple or a concatenation of a single simple cycle) $\hat{c}_i$. Aperiodicity entails that the concatenations of $\hat{c}_i$ are forgetful. Hence, Lemma 7 ensures that the finite run constituted of $2^{|X|+1}$ iterations of c_i is synchronizing and Lemma 6 that ε can decrease each time that π_{rat} goes from C_i to c_i. □

We thus obtain the following result as a corollary of Theorems 2 and 4.

Corollary 2. *Let $\mathcal{A}$ be a strongly non-Zeno, forgetful and aperiodic timed automaton. Then,* $\mathsf{Freq}(\mathcal{A}) = \mathsf{Rat}(\mathcal{A}_{cp})$.

Strong forgetfulness implies aperiodicity, hence Theorem 3 cannot help to established this equality in a more general case. However, note that Theorem 4 does not imply Theorem 3. In Theorem 3, not only the inclusion $\mathsf{Rat}(\mathcal{A}_{cp}) \subseteq \mathsf{Freq}(\mathcal{A})$ is established, but also for all infinite runs π in $\mathcal{A}_{cp}$ there exists an infinite run ρ in $\mathcal{A}$ with $\pi \in \mathsf{Proj}(\rho)$ and $\mathsf{freq}_{\mathcal{A}}(\rho) = \mathsf{Rat}(\pi)$. In Theorem 4, this is only proved for some infinite runs π of $\mathcal{A}_{cp}$.

4.4 Discussion About Assumptions

As explained above, our will to relax the strong forgetfulness is due to its difficulties to be checked. Strong forgetfulness clearly implies at once forgetfulness and aperiodicity, but a first open question is whether the other implication is true. Indeed, we did not find any example of forgetful aperiodic timed automaton which is non-strongly forgetful. We think that, either there are some one but probably with more than two clocks which is difficult to visualize, or the implication is true and proving this statement could lead to fundamental advances in the understanding of the corner-point abstraction.

An other open question is whether the hypothesis of aperiodicity in Theorem 4 can be relaxed. We use this hypothesis in the proof, but could not find counterexamples. We built some examples of periodic timed automata as in Fig. 3 ,

but periodic timed automata seem to be degenerated and in particular, based on punctual guards which implies bijections between runs in the timed automaton and those in its corner-point abstraction.

5 Conclusion

A quantitative semantics based on frequencies has recently been proposed for timed automata in [6]. In this paper, we used the notion of forgetfulness introduced in [4] to extend the results about frequencies in timed automata. On the one hand, thanks to forgetfulness we can compute the set of frequencies in one-clock timed automata even with Zeno behaviors, whereas only the bounds of this set was computed in [6]. On the other hand, with forgetfulness and time-divergence inspired by [7], we compute the set of frequencies in a class of n-clock timed automata, whereas techniques of [6] were not applicable. In the future, we would like to investigate more deeply the difference between forgetfulness and strong forgetfulness with the hope to extend Theorem 3. Moreover, Theorem 2 is less constructive than the equivalent result for one-clock timed automata which use notions of contraction and dilatation of a run. It would be interesting to see if forgetfulness could help to extend these constructions to n-clock timed automata. Finally, our main tool presented in Lemma 6 can be easily used for the scheduling problem in timed automata with costs and rewards studied in [7]. Thus, we can prove that in strongly non-Zeno forgetful timed automata, there is always an infinite run whose ratio is optimal. We hope that Lemma 6, which is fundamental, will be useful for a lot of problems for which the corner-point is suitable.

Acknowledgements. I am very grateful to Nathalie Bertrand for useful discussions and detailed proofreadings of this paper. Moreover, I would like to thank the reviewers for their very interesting remarks.

References

1. Alur, R., Dill, D.L.: A theory of timed automata. Theoretical Computer Science 126(2), 183–235 (1994)
2. Alur, R., La Torre, S., Pappas, G.J.: Optimal Paths in Weighted Timed Automata. In: Di Benedetto, M.D., Sangiovanni-Vincentelli, A.L. (eds.) HSCC 2001. LNCS, vol. 2034, pp. 49–62. Springer, Heidelberg (2001)
3. Baier, C., Bertrand, N., Bouyer, P., Brihaye, Th., Größer, M.: Almost-sure model checking of infinite paths in one-clock timed automata. In: Proceedings of the 23rd Annual IEEE Symposium on Logic in Computer Science (LICS 2008), pp. 217–226. IEEE (2008)
4. Basset, N., Asarin, E.: Thin and Thick Timed Regular Languages. In: Fahrenberg, U., Tripakis, S. (eds.) FORMATS 2011. LNCS, vol. 6919, pp. 113–128. Springer, Heidelberg (2011)

5. Behrmann, G., Fehnker, A., Hune, T., Larsen, K.G., Pettersson, P., Romijn, J.M.T., Vaandrager, F.W.: Minimum-Cost Reachability for Priced Timed Automata. In: Di Benedetto, M.D., Sangiovanni-Vincentelli, A.L. (eds.) HSCC 2001. LNCS, vol. 2034, pp. 147–161. Springer, Heidelberg (2001)
6. Bertrand, N., Bouyer, P., Brihaye, T., Stainer, A.: Emptiness and Universality Problems in Timed Automata with Positive Frequency. In: Aceto, L., Henzinger, M., Sgall, J. (eds.) ICALP 2011, Part II. LNCS, vol. 6756, pp. 246–257. Springer, Heidelberg (2011)
7. Bouyer, P., Brinksma, E., Larsen, K.G.: Optimal infinite scheduling for multi-priced timed automata. Formal Methods in System Design 32(1), 3–23 (2008)
8. Cassez, F., Henzinger, T.A., Raskin, J.-F.: A Comparison of Control Problems for Timed and Hybrid Systems. In: Tomlin, C.J., Greenstreet, M.R. (eds.) HSCC 2002. LNCS, vol. 2289, pp. 134–148. Springer, Heidelberg (2002)
9. Kwiatkowska, M.Z., Norman, G., Segala, R., Sproston, J.: Automatic verification of real-time systems with discrete probability distributions. Theoretical Computer Science 282, 101–150 (2002)
10. Laroussinie, F., Markey, N., Schnoebelen, P.: Model Checking Timed Automata with One or Two Clocks. In: Gardner, P., Yoshida, N. (eds.) CONCUR 2004. LNCS, vol. 3170, pp. 387–401. Springer, Heidelberg (2004)
11. Ouaknine, J., Worrell, J.: On the language inclusion problem for timed automata: Closing a decidability gap. In: Proceedings of the 19th IEEE Symposium on Logic in Computer Science (LICS 2004), pp. 54–63. IEEE (2004)
12. Puri, A.: Dynamical properties of timed automata. Discrete Event Dynamic Systems 10(1-2), 87–113 (2000)
13. Stainer, A.: Frequencies in forgetful timed automata. Research Report 8009, INRIA, Rennes, France (July 2012), http://hal.inria.fr/hal-00714262