Durations and Parametric Model-Checking in Timed Automata

VÉRONIQUE BRUYÈRE
Université de Mons-Hainaut
and
EMMANUEL DALL'OLIO and JEAN-FRANÇOIS RASKIN
Université Libre de Bruxelles

We consider the problem of model-checking a parametric extension of the logic TCTL over timed automata and establish its decidability. Given a timed automaton, we show that the set of durations of runs starting from a region and ending in another region is definable in Presburger arithmetic (when the time domain is discrete) or in a real arithmetic (when the time domain is dense). Using this logical definition, we show that the parametric model-checking problem for the logic TCTL can be solved algorithmically; the proof of this result is simple. More generally, we are able to effectively characterize the values of the parameters that satisfy the parametric TCTL formula with respect to the given timed automaton.

Categories and Subject Descriptors: D.2.4 [**Software Engineerings**]: Software/Program Verification; F.3.1 [**Logics and Meaning of Programs**]: Specifying and Verifying and Reasoning about Programs; F.4.1 [**Mathematical Logics and Formal Languages**]: Mathematical Logic

General Terms: Theory, Verification

Additional Key Words and Phrases: Timed Automata, Model-Checking, Presburger Arithmetic

1. INTRODUCTION

For more than twenty years, temporal logics [Pnueli 1977; Clarke et al. 1986] and automata theory play a central role in the foundations for the formal verification of reactive systems. Reactive systems often have to meet real-time constraints. As a consequence, in the early nineties, temporal logics and automata theory have been extended to explicitly refer to real-time. Timed automata [Alur and Dill 1994], an extension of usual automata with clocks, have been proposed as a natural model

This research was supported by the FRFC project "Centre Fédéré en Vérification" funded by the Belgian National Science Foundation (FNRS) under grant nr 2.4530.02.
A preliminary version of this paper appeared in the Proceedings of the 20th Annual Symposium on Theoretical Aspects of Computer Science, STACS'03, *Lecture Notes in Computer Science* 2607, Springer, 2003, pp. 687-698.
Authors'addresses: V. Bruyère, Institut d'Informatique, Université de Mons-Hainaut, Le Pentagone, Avenue du Champ de Mars 6, B-7000 Mons, Belgium. E. Dall'Olio and J.-F. Raskin, Département d'Informatique, Université Libre de Bruxelles, Boulevard du Triomphe CP 212, B-1050-Bruxelles, Belgium.

for systems that must respect real-time constraints. Temporal logics have also been extended to express quantitative real-time properties [Alur et al. 1990; Alur et al. 1996; Henzinger 1998].

The model-checking problem, given a timed automaton $\mathcal{A}$ and a real-time temporal logic formula ϕ, consists in answering by Yes or No if the executions of $\mathcal{A}$ satisfy the property expressed by ϕ. The *Response property* is a typical example of a real-time property: "if every request reaches the server within two time units, then every request must be accepted or rejected within four time units". This property is easily expressed in TCTL as follows:

$$\forall\Box(\texttt{send} \to \forall\Diamond_{\leq 2}\texttt{request}) \to \forall\Box(\texttt{send} \to \forall\Diamond_{\leq 4}\texttt{granted} \vee \texttt{rejected})$$

The main drawback with this formulation of the response property is that it refers to fixed constants. In the context of real-time systems, it is often more natural to use parameters instead of constants to express the delays involved in the model. In particular, we would like to formalize the above property in a more abstract way like "if every request reaches the server within α time units, then there should exist a bound of time β within which every request must be accepted or rejected, that bound should be less than 2α". In this way, the property is less dependent on the particular model on which it is checked. This is particularly important when the model is not completely known a priori. The parametric property above can be formalized using an extension of TCTL with parameters and quantification over parameters as follows:

$$\begin{aligned}\forall\alpha\exists\beta\cdot\beta \leq 2\alpha\wedge\ [&\forall\Box(\texttt{send} \to \forall\Diamond_{\leq\alpha}\texttt{request})\\ &\to \forall\Box(\texttt{send} \to \forall\Diamond_{\leq\beta}\texttt{granted} \vee \texttt{rejected})]\end{aligned}$$

The use of parameters can also be very useful to *learn* about a model. For example, parameters in conjunction with temporal logics can be used to express questions that are not expressible in the usual temporal logics. As an example, consider the following question: "Is there a bound on the delay to reach a σ state from an initial state ?". This property is easily expressed using the TCTL syntax extended with parameters: $\exists\alpha\cdot\forall\Diamond_{\leq\alpha}\sigma$. The answer to the question is Yes if the formula is true on the given timed model. We can even be more ambitious and ask to characterize the set of parameter valuations for which a formula is true on the model. As an example, consider the following TCTL formula with parameter β: $\forall\Diamond_{\leq\beta}\sigma$. If there is no bound on the time needed to reach a σ state from an initial state, then the answer to the parameter valuation problem would be "empty", while it would be all the valuations v such that $v(\alpha) \geq c$ if all paths starting from an initial state reach a σ state within c time units.

We show in this paper that the problems outlined above are decidable for discrete- and dense-timed automata (when parameters range over natural numbers). While those results were partially known, see [Wang 1995; Wang and Hsiung 1997; Emerson and Trefler 1999], the main contributions of this paper are as follows. First, we show that the *natural* and *intuitive* notion of *duration* of runs (of a timed automaton starting from a region and ending to another region) is central and sufficient to answer the parametric TCTL verification problem. Second, we show that Presburger arithmetic PA, the first-order arithmetic RA over the structure $\langle\mathbb{R}, +, <, \mathbb{N}, 0, 1\rangle$ of the real numbers (with $\mathbb{N}$ being the predicate "is a natural number"), and classical

automata are *elegant* tools to formalize this notion of run durations. Finally, using those simple and clean concepts from logic and automata theory, we are able to prove the decidability of the model-checking problem for a parametric version of TCTL that strictly subsumes, to the best of our knowledge, all the parametric extensions of TCTL proposed so far. Furthermore, we are able to effectively characterize the values of the parameters that satisfy the parametric TCTL formula. Finally, it should be noted that the technique to establish these results is very similar if the time domain is discrete or dense.

Structure of the paper. The paper is organized as follows. In Section 2, we review some basic notions about timed automata. We consider both discrete and dense time domains. In Section 3, we introduce the Parametric TCTL logic. In Section 4, we detail known results about the finite region graph underlying any timed automaton. In Section 5, we define the notion of duration of a run starting from a region and ending to another region. We show that the set of possible run durations between two regions is effectively definable in Presburger arithmetic when considering a discrete time domain, and effectively definable in the RA arithmetic when considering a dense time domain. In Section 6, we show that the model-checking problem for Parametric TCTL can be solved using the notion of run duration and its effective formalization into Presburger arithmetic (or in the RA arithmetic). This is possible by a logical characterization of the values of the parameters that satisfy the Parametric TCTL formula. In Section 7, we study in details the complexity of our method. We compare our work with existing related works in the last section.

A preliminary version of this paper appeared in the Proceedings of the 20th Annual Symposium on Theoretical Aspects of Computer Science, STACS'03, *Lecture Notes in Computer Science* 2607, Springer-Verlag, 2003, pp. 687-698.

2. TIMED AUTOMATA

In this section, we recall the classical notion of timed automaton [Alur and Dill 1994].

Notations. Throughout the paper, we denote by $X = \{x_1, \ldots, x_n\}$ a set of n clocks. We use the same notation $x = (x_1, \ldots, x_n)$ for the *variables* and for an *assignment* of values to these variables. Depending on whether the timed automata are *dense*-timed or *discrete*-timed, the values of the variables are taken in domain $\mathbb{T}$ equal to the set $\mathbb{R}^+$ of nonnegative reals or to the set $\mathbb{N}$ of natural numbers. Given a clock assignment x and $\tau \in \mathbb{T}$, $x + \tau$ is the clock assignment $(x_1 + \tau, \ldots, x_n + \tau)$. A *simple guard* is an expression of the form $x_i \sim c$ where x_i is a clock, $c \in \mathbb{N}$ is an integer constant, and $\sim$ is one of the symbols $\{<, \leq, =, >, \geq\}$. A *guard* is any finite conjunction of simple guards. We denote by $\mathcal{G}$ the set of guards. Let g be a guard and x be a clock assignment, the notation $x \models g$ means that x satisfies g. A *reset* function r indicates which clocks are reset to 0. Thus either $r(x)_i = 0$ or $r(x)_i = x_i$. The set of reset functions is denoted by $\mathcal{R}$. We use the notation Σ for the set of *atomic propositions*.

Definition 2.1. A *timed automaton* $\mathcal{A} = (L, X, E, \mathcal{I}, \mathcal{L})$ has the following components: (i) L is a finite set of *locations*, (ii) X is a set of *clocks*, (iii) $E \subseteq L \times \mathcal{G} \times \mathcal{R} \times L$ is a finite set of *edges*, (iv) $\mathcal{I} : L \to \mathcal{G}$ assigns an *invariant* to each location, (v)

$\mathcal{L} : L \to 2^{\Sigma}$ is the *labeling* function.

Definition 2.2. A timed automaton $\mathcal{A} = (L, X, E, \mathcal{I}, \mathcal{L})$ generates a *transition system* $T_{\mathcal{A}} = (Q, \to)$ with a set of *states* Q equal to $\{(l, x) \mid l \in L, x \in \mathbb{T}^n, x \models \mathcal{I}(l)\}$ and a *transition relation* $\to \; = \; \bigcup_{\tau \in \mathbb{T}} \xrightarrow{\tau}$ defined as follows:

$$(l, x) \xrightarrow{\tau} (l', x')$$

— if $\tau > 0$, then $l = l'$ and $x' = x + \tau$ (*elapse of time* at location l)
— if $\tau = 0$, then $(l, g, r, l') \in E$, $x \models g$ and $r(x) = x'$ (*instantaneous switch*)

The states (l, x) of $T_{\mathcal{A}}$ are shortly denoted by q. Given $q = (l, x) \in Q$ and $\tau \in \mathbb{T}$, we denote by $q + \tau$ the state $(l, x + \tau)$. Note that if $\mathcal{A}$ is a discrete-timed automaton, the elapse of time is discrete with $\tau = 0, 1, 2, \cdots$. If $\mathcal{A}$ is a dense-timed automaton, then τ is any positive real number. Note also that given a transition $q \to q'$ of $T_{\mathcal{A}}$, it is easy to compute the unique τ such that $q \xrightarrow{\tau} q'$.

Definition 2.3. Given a transition system $T_{\mathcal{A}}$, a *run* $\rho = (q_i)_{i \geq 0}$ is an infinite path in $T_{\mathcal{A}}$

$$\rho = q_0 \xrightarrow{\tau_0} q_1 \xrightarrow{\tau_1} q_2 \to \cdots \to q_i \xrightarrow{\tau_i} q_{i+1} \to \cdots$$

such that $\sum_{i \geq 0} \tau_i = \infty$ (*progress* of time). A *finite run* $\rho = (q_i)_{0 \leq i \leq j}$ is any finite path in $T_{\mathcal{A}}$. A *position* in ρ is a state $q_i + \tau$, where $i \geq 0$ and $\tau \in \mathbb{T}$ such that $0 \leq \tau < \tau_i$. The *duration* $t = \mathrm{D}_\rho(q)$ *at position* $q = q_i + \tau$ is equal to $t = \tau + \sum_{0 \leq i' < i} \tau_{i'}$.

The duration $\mathrm{D}_\rho(q)$ indicates the accumulated time till position q. If $\mathcal{A}$ is a discrete-timed automaton, then D_ρ is a function which assigns a natural number to any position of the run ρ. If $\mathcal{A}$ is a dense-timed automaton, the assignment is in $\mathbb{R}^+$. As we allow *several* consecutive instantaneous switches in the definition of a run, different positions q can have the same duration $\mathrm{D}_\rho(q)$. The set of positions in a run ρ can be totally ordered as follows. Let $q = q_i + \tau$ and $q' = q_{i'} + \tau'$ be two positions of ρ. Then $q < q'$ iff either $i < i'$ or $i = i'$ and $\tau < \tau'$.

3. PARAMETRIC TIMED CTL LOGIC

Let P be a fixed finite set of *parameters* (in the sense of variables for positive integers). A parameter of P is denoted by θ. A *parameter valuation* for P is a function $v : P \to \mathbb{N}$ which assigns a natural number to each parameter $\theta \in P$. In the sequel, α means any element of $P \cup \mathbb{N}$ and β means any linear term $\sum_{i \in I} c_i \theta_i + c$, with $c_i, c \in \mathbb{N}$ and $\{\theta_i \mid i \in I\} \subseteq P$. A parameter valuation v is naturally extended to linear terms by defining $v(c) = c$ for any $c \in \mathbb{N}$.

The *syntax* of *Parametric Timed CTL logic*, PTCTL logic for short, is defined in two steps: we first define formulae of first type, and then formulae of second type. We propose two logics, the discrete PTCTL logic and the dense PTCTL logic. The first one is dedicated to discrete-timed automata whereas the second one is dedicated to dense-timed automata. The notation σ means any atomic proposition $\sigma \in \Sigma$.

Definition 3.1. *Discrete* PTCTL formulae φ of *first type* are given by the following grammar

$$\varphi ::= \sigma \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists \bigcirc \varphi \mid \varphi \exists \mathrm{U}_{\sim\alpha} \varphi \mid \varphi \forall \mathrm{U}_{\sim\alpha} \varphi$$

and formulae f of *second type* by

$$f ::= \varphi \mid \theta \sim \beta \mid \neg f \mid f \vee f \mid \exists \theta f$$

Dense PTCTL formulae are defined in the same way, except that operator $\exists\bigcirc$ is forbidden.

The *existential* fragment $\exists$PTCTL of PTCTL logic is the fragment where formulae are of the form $\exists\theta_1 \cdots \exists\theta_m\ f$ where f is a PTCTL formula without quantifiers over parameters.

In this definition, we assume that if a parameter θ has an occurrence in a formula f which is under the scope of a quantifier, then it cannot have another occurrence in f which is not under the scope of a quantifier (this is always possible up to a renaming of the parameters). In second type formula $\exists\theta f$, it is assumed that θ is a free parameter of formula f. The set of free parameters of f is denoted by P_f. Note that the usual operators $\exists$U and $\forall$U are obtained as $\exists \mathrm{U}_{\geq 0}$ and $\forall \mathrm{U}_{\geq 0}$. We now give the *semantics* of PTCTL logic.

Definition 3.2. Let $T_{\mathcal{A}}$ be the transition graph of a discrete-timed automaton $\mathcal{A} = (L, X, E, \mathcal{I}, \mathcal{L})$ and $q = (l, x)$ be a state of $T_{\mathcal{A}}$. Let f be a discrete PTCTL formula and v be a parameter valuation on P_f. Then the *satisfaction* relation $q \models_v f$ is defined inductively in the following way.

(1) First type formulae:
— $q \models_v \sigma$ iff there exists a run $\rho = (q_i)_{i\geq 0}$ in $T_{\mathcal{A}}$ with $q = q_0$ and $\sigma \in \mathcal{L}(l)$
— $q \models_v \neg\varphi$ iff $q \not\models_v \varphi$
— $q \models_v \varphi \vee \psi$ iff $q \models_v \varphi$ or $q \models_v \psi$
— $q \models_v \exists\bigcirc \varphi$ iff there exists a run $\rho = (q_i)_{i\geq 0}$ in $T_{\mathcal{A}}$ with $q = q_0$ and $q_0 \xrightarrow{\tau} q_1$ satisfying $\tau = 0$ or $\tau = 1$, such that $q_1 \models_v \varphi$
— $q \models_v \varphi \exists \mathrm{U}_{\sim\alpha}\, \psi$ iff there exists a run $\rho = (q_i)_{i\geq 0}$ in $T_{\mathcal{A}}$ with $q = q_0$, there exists a position p in ρ such that $\mathrm{D}_\rho(p) \sim v(\alpha)$, $p \models_v \psi$ and $p' \models_v \varphi$ for all $p' < p$
— $q \models_v \varphi \forall \mathrm{U}_{\sim\alpha}\, \psi$ iff for any run $\rho = (q_i)_{i\geq 0}$ in $T_{\mathcal{A}}$ with $q = q_0$, there exists a position p in ρ such that $\mathrm{D}_\rho(p) \sim v(\alpha)$, $p \models_v \psi$ and $p' \models_v \varphi$ for all $p' < p$

(2) Second type formulae:
— $q \models_v \theta \sim \beta$ iff there exists a run $\rho = (q_i)_{i\geq 0}$ in $T_{\mathcal{A}}$ with $q = q_0$ and $v(\theta) \sim v(\beta)$
— $q \models_v \neg f$ iff $q \not\models_v f$
— $q \models_v f \vee g$ iff $q \models_v f$ or $q \models_v g$
— $q \models_v \exists\theta f$ iff there exists $c \in \mathbb{N}$ such that $q \models_{v'} f$ where v' is defined on P_f by $v' = v$ on $P_{\exists\theta f}$ and $v'(\theta) = c$

If $\mathcal{A}$ is a dense-timed automaton and f is a dense PTCTL formula, then the satisfaction relation is defined in the same way, except that the formula $q \models_v \exists\bigcirc \varphi$ is cancelled.

Problem 3.3. The *model-checking* problem is the following. Given a timed automaton $\mathcal{A}$, a state q of $T_{\mathcal{A}}$, and a PTCTL formula f, is there a parameter valuation v on P_f such that $q \models_v f$? The model-checking problem is called *discrete* if $\mathcal{A}$ is a discrete-timed automaton and f a discrete PTCTL formula. It is called *dense* if $\mathcal{A}$ is a dense-timed automaton and f a dense PTCTL formula.

Problem 3.4. The *parameter synthesis* problem is the following. Given a timed automaton $\mathcal{A}$, a state q of $T_{\mathcal{A}}$, and a PTCTL formula f, compute a symbolic representation[1] of the set of parameter valuations v on P_f such that $q \models_v f$.

4. REGION GRAPHS

In this section we recall the definition of region graph [Alur and Dill 1994]. It is usually given for dense-timed automata. It can also be applied to discrete-timed automata. Thus in the sequel $\mathcal{A} = (L, X, E, \mathcal{I}, \mathcal{L})$ is a discrete- or a dense-timed automaton.

We first recall the usual equivalence on clock assignments and its extension to states of the transition system $T_{\mathcal{A}}$ generated by $\mathcal{A}$. For clock x_i, let c_i be the largest constant that x_i is compared with in any guard of E and any invariant of $\mathcal{I}$. For $\tau \in \mathbb{T}$, $\text{fract}(\tau)$ denotes its fractional part and $\lfloor \tau \rfloor$ denotes its integral part.

Definition 4.1. Two clock assignments x and x' are equivalent, $x \approx x'$, iff the following conditions hold

- For any i, $1 \leq i \leq n$, either $\lfloor x_i \rfloor = \lfloor x'_i \rfloor$ or $x_i, x'_i > c_i$
- For any $i \neq j$, $1 \leq i, j \leq n$, $\text{fract}(x_i) \leq \text{fract}(x_j)$ iff $\text{fract}(x'_i) \leq \text{fract}(x'_j)$
- For any i, $1 \leq i \leq n$, $\text{fract}(x_i) = 0$ iff $\text{fract}(x'_i) = 0$

The equivalence relation $\approx$ is extended to the states of $T_{\mathcal{A}}$ as follows. Let $q = (l, x)$, $q' = (l', x')$ be two states. Then

$$q \approx q' \quad \text{iff} \quad l = l' \text{ and } x \approx x'.$$

We use $[x]$ (resp. $[q]$) to denote the equivalence class to which x (resp. q) belongs. A *region* is an equivalence class $[q]$. Hence, a region is fully specified by a location, the integral parts of the clock assignments, and the ordering of the fractional parts of the clock assignments. The set of all regions is denoted by R. This set can be considered to be *finite*, because the exact value of the integral part of a clock x_i must be recorded only if it is smaller than or equal to c_i. A region $[q]$ is called *boundary* if $q + \tau \not\approx q$ for any $\tau > 0$ (in other words if $[q] \ni (l, x)$ with $\text{fract}(x_i) = 0$ for some i). It is called *unbounded* if it satisfies $q = (l, x)$ with $x_i > c_i$ for all i.

It is well-known [Alur and Dill 1994] that $\approx$ is *back-stable* on Q: if $q \approx q'$ and $p \to q$, then $\exists p' \approx p$ such that $p' \to q'$.

The next lemma will be useful in the Section 5.

LEMMA 4.2. *Let x be a clock assignment such there exists i, $1 \leq i \leq n$, with $0 < \text{fract}(x_i) < 1$. Then for all y, $0 < y < 1$, there exists a clock assignment x' such that $x \approx x'$ and $\text{fract}(x'_i) = y$.*

PROOF. The equivalence class $[x]$ of x is fully specified by the integral parts of the clock assignments and the ordering of the fractional parts of the clock assignments. Since $0 < \text{fract}(x_i) < 1$ and $0 < y < 1$, it follows that there exists a clock assignment x' in $[x]$ with the same integral parts as for x, and with the fractional parts satisfying the same ordering as for x such that $\text{fract}(x'_i) = y$. □

[1]For instance this representation could be given in a decidable logical formalism, like Presbruger arithmetic, which explicitely gives numerical constraints over the parameters.

It is proved in [Alur et al. 1990] that states belonging to the same region satisfy the same set of TCTL formulae. The proof is easily adapted to PTCTL formulae.

PROPOSITION 4.3. *Let q, q' be two states of $T_{\mathcal{A}}$ such that $q \approx q'$. Let f be a* PTCTL *formula and v be a parameter valuation. Then $q \models_v f$ iff $q' \models_v f$.*

It is important to note that this proposition holds because a parameter valuation $v : P \to \mathbb{N}$ assigns a *natural* value to each parameter. The result is no longer true if the valuation assigns a real number to each parameter.

As a consequence of Proposition 4.3, Problems 3.3 and 3.4 can be restated as follows.

Problem 4.4. Given a timed automaton $\mathcal{A}$, a region r of R, and a PTCTL formula f, is there a parameter valuation v on P_f such that $r \models_v f$? Is it possible to compute a symbolic representation of the set of parameter valuations v on P_f such that $r \models_v f$?

We proceed with the definition of region graph. Given two regions $r = [q]$, $r' = [q']$ such that $r \neq r'$, we say that r' is a *successor* of r, $r' = \mathrm{succ}(r)$, if $\exists \tau \in \mathbb{T}$, $q + \tau \in r'$, and $\forall \tau' < \tau$, $q + \tau' \in r \cup r'$.

Definition 4.5. Let $\mathcal{A}$ be a timed automaton. The *region graph* $R_{\mathcal{A}} = (R, F)$ is a finite graph with R as vertex set and its edge F defined as follows. Given two regions $r, r' \in R$, the edge (r, r') belongs to F: (i) if $q \xrightarrow{\tau} q'$ in $T_{\mathcal{A}}$, with $\tau = 0$, $r = [q]$ and $r' = [q']$, or (ii) if $r' = \mathrm{succ}(r)$, or (iii) if $r = r'$ is an unbounded region.

We recall that the size $|R_{\mathcal{A}}|$ of the region graph, i.e. its number of regions, is in $\mathcal{O}(2^{|\mathcal{A}|})$ where $|\mathcal{A}|$ takes into account the number of locations of $\mathcal{A}$ and the length of the binary representation of the constants appearing in $\mathcal{A}$ [Alur and Dill 1994].

There is a simple correspondence between runs of $T_{\mathcal{A}}$ and paths in $R_{\mathcal{A}}$. More precisely, let $\rho = (q_i)_{i \geq 0}$ be a run. Consider $q_i \xrightarrow{\tau_i} q_{i+1}$. If $\tau_i = 0$ or if $[q_i] = [q_{i+1}]$ is an unbounded region, then $([q_i], [q_{i+1}])$ is an edge of $R_{\mathcal{A}}$. Otherwise, there is a path $(r_{i,k})_{1 \leq k \leq n_i}$ in $R_{\mathcal{A}}$ such that $r_{i,1} = [q_i]$, $r_{i,n_i} = [q_{i+1}]$ and $r_{i,k+1} = \mathrm{succ}(r_{i,k})$ for all k, $1 \leq k < n_i$. This path can be empty (when $[q_i] = [q_{i+1}]$). In this way, a unique infinite path of $R_{\mathcal{A}}$, that we denote by $\pi(\rho)$, corresponds to the run ρ of $T_{\mathcal{A}}$. We say that $\pi(\rho)$ is the path *associated* to ρ. Any path $\pi(\rho)$ associated to a run ρ is called *progressive* since time progresses without bound along ρ (see Definition 2.3). On the other hand, for any progressive path of $R_{\mathcal{A}}$, we can find a corresponding run of $T_{\mathcal{A}}$ [Alur and Dill 1994]. This run is not unique.

Let us look at the region graph $R_{\mathcal{A}}$ when $\mathcal{A}$ is discrete-timed. In Definition 4.1, only the first condition is useful. Thus given an equivalence class $[x]$, the clock assignment x_i in $[x]$ is either one among the values 1, 2, ..., c_i, or it belongs to the set $c_i^+ = \{c \in \mathbb{N} \mid c > c_i\}$. If r, r' are two regions such that $r' = \mathrm{succ}(r)$, then any clock assignment is increased by 1 from r to r'.

5. DURATIONS

To solve the model-checking problem (see Problem 3.3), we want an algorithm that, given a timed automaton $\mathcal{A}$, a region r of $R_{\mathcal{A}}$, and a PTCTL formula f, checks whether there exists a parameter valuation v on P_f such that $r \models_v f$.

For the discrete time, our approach is the following one. Let $P_f = \{\theta_1, \ldots, \theta_m\}$. We are going to construct a formula $\Delta_{f,r}$ of Presburger arithmetic with free variables $\theta_1, \ldots, \theta_m$ such that $r \models_v f$ for some valuation v iff the sentence $\exists\theta_1 \cdots \exists\theta_m \Delta_{f,r}$ is true. Our algorithm follows because Presburger arithmetic has a decidable theory. The construction of formula $\Delta_{f,r}$ is presented in Section 6.

The *main tool* of our approach is a description, given two regions s, s' of $R_{\mathcal{A}}$, of all the possible values of duration $\mathrm{D}_\rho(q_j)$ for any finite run ρ from q_0 to q_j in $T_{\mathcal{A}}$ such that $[q_0] = s$, $[q_j] = s'$ (see Definition 5.1 below). The description is given by a Presburger arithmetic formula. It is presented in this section. Such a description is useful for constructing the formula $\Delta_{f,r}$ when f contains the operator $\exists \mathrm{U}_{\sim\alpha}$ or $\forall \mathrm{U}_{\sim\alpha}$. Indeed, in these cases, a duration has to be compared with a parameter valuation (see Definition 3.2).

In the case of dense time, the approach is exactly the same but with a first-order arithmetic over the reals instead of Presburger arithmetic. We will see that the description of the durations is very similar to the one obtained in the discrete case, due to a granularity of one of the durations (see Lemma 5.5). It will follow that the construction of the formula $\Delta_{f,r}$ in the dense case is nearly the same as in the discrete case.

Let us now give the definition of the set $\lambda^S_{s,s'}$ of durations.

Definition 5.1. Let $\mathcal{A}$ be a timed automaton and $R_{\mathcal{A}} = (R, F)$ be its region graph. Let $s, s' \in R$ and $S \subseteq R$. Then $\lambda^S_{s,s'}$ is the set of $t \in \mathbb{T}$ for which there exists a finite run $\rho = (q_i)_{0 \le i \le j}$ in $T_{\mathcal{A}}$ such that

— the path $\pi(\rho) = (r_k)_{0 \le k \le k'}$ in $R_{\mathcal{A}}$ associated with ρ satisfies $s = r_0$, $s' = r_{k'}$ and $r_k \in S$ for any k, $0 \le k < k'$,

— the run ρ has duration $\mathrm{D}_\rho(q_j) = t$.

In this definition, s belongs to S and s' may not belong to S.

5.1 Discrete Time

We recall that *Presburger arithmetic*, PA for short, is the set of first-order formulae of $\langle \mathbb{N}, +, <, 0, 1 \rangle$. Terms are built from variables, the constants 0, 1, and the symbol $+$. Atomic formulae are either equations $t_1 = t_2$ or inequations $t_1 < t_2$ between terms t_1, t_2. Formulae are built from atomic formulae using first-order quantifiers and the usual boolean connectives. Formulae are interpreted over the natural numbers, with the usual interpretation of $+$, $<$, 0 and 1. The *existential* fragment of PA, denoted by $\exists$PA, is the set of formulae which are of the form a block of existential quantifiers followed by a PA formula without quantifiers.

Presburger *theory* is the set of all valid sentences of Presburger arithmetic, i.e., formulae without free variables which are true in the standard representation. It is well-known that Presburger theory is decidable with a complexity in 3ExpTime in the size of the sentence, and in NP for the existential fragment of PA [Oppen 1978].

A set $X \subseteq \mathbb{N}^k$ is *definable* by a PA formula $\varphi(x_1, \ldots, x_k)$ if X is exactly the set of k-tuples of assignments of $x_1, \ldots, x_k$ making formula φ true.

Proposition 5.2. *Let $\mathcal{A} = (L, X, E, \mathcal{I}, \mathcal{L})$ be a discrete-timed automaton and $R_{\mathcal{A}} = (R, F)$ be its region graph. Let $s, s' \in R$ and $S \subseteq R$. Then the set $\lambda^S_{s,s'}$ is*

definable by a PA *formula. The construction of the formula is effective.*

PROOF. As $\mathcal{A}$ is a discrete-timed automaton, its region graph satisfies the following property. Consider the edge $(r, r') \in F$. There are three cases. Either it corresponds to an instantaneous switch, which takes no time. Or $r' = \text{succ}(r)$ and any clock has been increased by 1 from r to r'. Or $r' = r$ is an unbounded region for which we can suppose that any clock is increased by 1 along (r, r').

Let a be a fixed symbol meaning an increment by 1 of the clocks. We define an automaton $\mathcal{C}$ in the sense of [Lewis and Papadimitriou 1998], as a particular subgraph of $R_{\mathcal{A}}$. It has $S \cup \{s'\}$ as set of states and $F \cap (S \times (S \cup \{s'\}))$ as set of transitions. Any of its transitions (r, r') is labeled by ϵ (the empty word) if it corresponds to an instantaneous switch, and by a otherwise. Its unique initial state is s and its unique final state is s'. The standard *subset construction* is then applied to $\mathcal{C}$ to get a deterministic automaton without ϵ-transitions. The resulting automaton $\mathcal{C}'$ has the special structure of a "frying pan" automaton [Eilenberg 1974] since the only symbol labeling the transitions is a (see Figure 1). Denote its states by $\{\zeta_0, \ldots, \zeta_k, \ldots, \zeta_l\}$, with ζ_0 the initial state. Note that $\mathcal{C}'$ could be without cycle, i.e., with a set of states equal to $\{\zeta_0, \ldots, \zeta_k\}$.

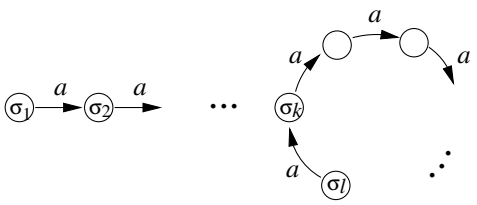

Fig. 1. Frying pan automaton

It is not difficult to check that t belongs to $\lambda^S_{s,s'}$ iff t is the length of a path in $\mathcal{C}'$ starting at ζ_0 and ending at some final state ζ_m. Indeed the duration t is equal to the total number of occurrences of letter a along a path of $\mathcal{C}$ from s to s'. Equivalently, by the subset construction, it is equal to the length of the corresponding path in $\mathcal{C}'$ from ζ_0 to some final state ζ_m containing s'. Hence if $m < k$, then $t = m$ and if $k \leq m \leq l$, then $\exists z \in \mathbb{N}, t = m + cz$ where $c = l - k + 1$ is the length of the cycle. Therefore $\lambda^S_{s,s'}$ is definable by a PA formula given by a disjunction of formulas like $t = m$ or $\exists z\ t = m + cz$ where m, c are constants. □

5.2 Dense Time

To deal with the dense time, we work with the RA *arithmetic*, instead of Presburger arithmetic : it is the set of first-order formulae of $\langle \mathbb{R}, +, <, \mathbb{N}, 0, 1 \rangle$ where $\mathbb{N}$ is a unary predicate. Formulae are interpreted over the real numbers. The interpretation of $\mathbb{N}$ is defined such that $\mathbb{N}(x)$ holds iff x is a natural number. Any PA formula can be translated into an RA formula thanks to predicate $\mathbb{N}$ (the RA formula is simply a copy of the given PA formula such that each variable x is constrained by $\mathbb{N}(x)$). As for Presburger arithmetic, the RA arithmetic has a decidable theory with a complexity in 3EXPTIME in the size of the sentence [Weispfenning 1999].

We do not work with $\langle \mathbb{R}, +, <, 0, 1 \rangle$, i.e. RA without the predicate $\mathbb{N}$. Indeed we need the predicate $\mathbb{N}$ to define the set $\lambda^S_{s,s'}$ by an RA formula in the dense case (see the proof of Proposition 5.3 hereafter). This predicate will again appear in the RA fomulae constructed in the proofs of Proposition 5.7 and Theorem 6.3. Note that $\langle \mathbb{R}, +, <, 0, 1 \rangle$ has a decidable theory with a complexity in 2ExpTime in the size of the sentence [Ferrante and Rackoff 1975].

Proposition 5.3. *Suppose that $\mathcal{A}$ is a dense-timed automaton. Then set $\lambda^S_{s,s'}$ is definable by a* RA *formula. The construction of the formula is effective.*

The proof of Proposition 5.3 is in the same vein as the proof of Proposition 5.2, however with some fitting due to dense time. Before giving this proof, we introduce a new automaton in relation with the dense-timed automaton $\mathcal{A}$ and we prove two lemmas.

The region graph $R_{\mathcal{A}}$ only deals with clock assignments. In a way to keep track of the total time elapsed, we are going to add to $\mathcal{A}$ a new clock x_0 which is reset to 0 each time it reaches the value 1. The new automaton denoted by $\mathcal{A}^0$, is obtained from $\mathcal{A} = (L, X, E, \mathcal{I}, \mathcal{L})$ in the following way. First, for each location l, a new invariant $x_0 \leq 1$ is added to $\mathcal{I}(l)$. Second, let $(l, g, r, l') \in E$ be any edge of $\mathcal{A}$. We replace it by two edges (l, g_1, r_1, l') and (l, g_2, r_2, l') such that

— g_1 is the guard $g \wedge (x_0 < 1)$, r_1 acts as r and lets x_0 unchanged,
— g_2 is the guard $g \wedge (x_0 = 1)$, r_2 acts as r and resets x_0 to 0.

Third, for every location $l \in L$, a new edge (l, g, r, l) is added such that g is the guard $x_0 = 1$, r lets all clocks unchanged except x_0 which is reset to 0, i.e., $r((x_0, x_1, \ldots, x_n)) = (0, x_1, \ldots, x_n)$.

To any (finite) run ρ of $T_{\mathcal{A}}$ corresponds a run denoted ρ^0 of $T_{\mathcal{A}^0}$ such that $x_0 = 0$ at its first state. Indeed, to obtain ρ^0, the value of the clock x_0 enriches ρ by adding to each state of ρ the fractional part of the current total time elapsed[2]. Conversely to any (finite) run of $T_{\mathcal{A}^0}$ corresponds a run of $T_{\mathcal{A}}$ by erasing clock x_0. Note that the run ρ^0 may be longer than ρ as clock x_0 is reset to 0 each time it reaches value 1.

The following property is immediate.

Lemma 5.4. *Let $\rho = (q_i)_{0 \leq i \leq j}$ and $\rho^0 = (q^0_i)_{0 \leq i \leq j^0}$. Suppose that duration $\mathrm{D}_\rho(q_j)$ equals t. Then* fract(t) *is equal to the value of x_0 at state $q^0_{j^0}$ and $\lfloor t \rfloor$ is equal to the number of times x_0 is reset to 0 along ρ^0.*

Note that x_0 is reset to 0 along transition $q^0_i \xrightarrow{\tau_i} q^0_{i+1}$ iff its value is different from 0 at state q^0_i and equal to 0 at state q^0_{i+1}.

The next lemma is the first step to Proposition 5.3. It states the nice property that in dense time, the granularity of the durations is equal to one, as in discrete time. In the proof, we index the equivalence relation $\approx$ (see Definition 4.1) by $\mathcal{A}$ or $\mathcal{A}^0$ to emphasize on the automaton that is used. Clearly if $q^0 \approx_{\mathcal{A}^0} q'^0$, then $q \approx_{\mathcal{A}} q'$.

Lemma 5.5. *Let $t \in \lambda^S_{s,s'}$. If $t \in]c, c+1[$ for some $c \in \mathbb{N}$, then $]c, c+1[\subseteq \lambda^S_{s,s'}$.*

[2]Note that if the value of x_0 has been reset to 0 one or several times between two consecutive states q and q' of ρ, then one or several states indicating those resets are intercalated between the corresponding states q^0 and q'^0 of ρ^0.

PROOF. Let $t \in \lambda^S_{s,s'}$. By hypothesis, $t = c + y$ with $0 < y < 1$. Let $t' = c + y'$ such that $0 < y' < 1$. We have to prove that $t' \in \lambda^S_{s,s'}$.

By Definition 5.1, there exists a finite run $\rho = (q_i)_{0 \leq i \leq j}$ in $T_{\mathcal{A}}$ with duration $t = \mathrm{D}_\rho(q_j)$ such that the associated path $\pi(\rho) = (r_k)_{0 \leq k \leq k'}$ in $R_{\mathcal{A}}$ satisfies the following properties: $s = r_0$, $s' = r_{k'}$ and $r_k \in S$ for any k, $0 \leq k < k'$. Consider $\rho^0 = (q^0_i)_{0 \leq i \leq j^0}$ in $T_{\mathcal{A}^0}$ and $\pi(\rho^0) = (r^0_k)_{0 \leq k \leq k'^0}$ in $R_{\mathcal{A}^0}$.

Suppose that $q^0_{j^0} = (l, x)$. By Lemma 5.4, $x_0 = \mathrm{fract}(t) = y$. We then define $q'^0_{j^0} = (l, x')$ such that $x' \approx_{\mathcal{A}^0} x$ and $x'_0 = y'$ (such an x' exists by Lemma 4.2). As the relation $\approx_{\mathcal{A}^0}$ is back-stable, it follows that there exists a finite run $\rho'^0 = (q'^0_i)_{0 \leq i \leq j^0}$ in $T_{\mathcal{A}^0}$ such that $q'^0_i \approx_{\mathcal{A}^0} q^0_i$, for any i, $0 \leq i \leq j^0$.

Hence $\pi(\rho^0) = \pi(\rho'^0)$. If we forget the additional clock, we get a finite run $\rho' = (q'_i)_{0 \leq i \leq j}$ in $T_{\mathcal{A}}$ such that $\pi(\rho) = \pi(\rho')$.

By construction, the number of times clock x_0 is reset to 0 along ρ'^0 and ρ^0 is identical. By Lemma 5.4, this number is equal to $\lfloor t \rfloor = c$ and $\mathrm{D}_{\rho'}(q'_j) = c + y'$. This shows that $t' \in \lambda^S_{s,s'}$. □

PROOF OF PROPOSITION 5.3. Let $R_{\mathcal{A}^0} = (R^0, F^0)$ be the region graph of $\mathcal{A}^0$. If r is a region of $R_{\mathcal{A}^0}$, we denote by $\overline{r}$ the region of $R_{\mathcal{A}}$ obtained by erasing clock x_0.

As in the proof of Proposition 5.2, we construct a classical automaton $\mathcal{C}$ as a particular subgraph of $R_{\mathcal{A}^0}$. Here symbol a means that clock x_0 has been reset to 0. Formally, the set of states of $\mathcal{C}$ is equal to $\{r \in R^0 \mid \overline{r} \in S \cup \{s'\}\}$ and its set of transitions is equal to $\{(r, r') \in F^0 \mid \overline{r} \in S, \overline{r}' \in S \cup \{s'\}\}$. Any transition (r, r') is labeled by a if x_0 has a non null value at region r and a null value at region r'. Otherwise it is labeled by ϵ. The unique initial state is the region r such that $\overline{r} = s$ and x_0 has value 0. The final states are regions r such that $\overline{r} = s'$. Let $\mathcal{C}'$ be the frying pan automaton obtained when the subset construction is applied to $\mathcal{C}$. We denote its states by $\{\zeta_0, \ldots, \zeta_k, \ldots, \zeta_l\}$, with ζ_0 as initial state (see Figure 1).

Let $t = \lfloor t \rfloor + y \in \lambda^S_{s,s'}$ with $y \in [0, 1[$. By definition of $\lambda^S_{s,s'}$, there exists a run $\rho = (q_i)_{0 \leq i \leq j}$ in $T_{\mathcal{A}}$ such that $t = \mathrm{D}_\rho(q_j)$ and the path $\pi(\rho) = (r_k)_{0 \leq k \leq k'}$ in $R_{\mathcal{A}}$ satisfies $s = r_0$, $s' = r_{k'}$ and $r_k \in S$ for any k, $0 \leq k < k'$. By construction of $\mathcal{C}$, the related path $\pi(\rho^0)$ in $R_{\mathcal{A}^0}$ can be viewed as a path of $\mathcal{C}$ starting at the initial state and ending at some final state that we denote by r_f.

Now, in $\mathcal{C}'$, the latter path leads to a path π' starting at ζ_0 and ending at some final state ζ_m, the length of which is equal to $\lfloor t \rfloor$ (by Lemma 5.4). Moreover if the value of x_0 is different from 0 at state r_f, then we have for any $y' \in]0, 1[$, $t' = \lfloor t \rfloor + y' \in \lambda^S_{s,s'}$ with the same path π' in $\mathcal{C}'$ (see Lemma 5.5).

Therefore the set $\lambda^S_{s,s'}$ is definable by a RA formula given by a disjunction of terms like $t = m$, $\exists z\ \mathbb{N}(z) \wedge t = m + cz$ where m, c are constants in $\mathbb{N}$ (as in the proof of Proposition 5.2) and terms like $m < t < m + 1$, $\exists z\ \mathbb{N}(z) \wedge m + cz < t < m + cz + 1$ (due to the above observation). □

5.3 Additional Sets

To solve the model-checking problem, we need two auxiliary sets that we present now. We begin with the definition of the set $\chi^S_{s,s'}$ which is very close to the definition of $\lambda^S_{s,s'}$.

Definition 5.6. Let $\mathcal{A}$ be a timed automaton and $R_{\mathcal{A}}$ be its region graph. Let $s, s' \in R$ and $S \subseteq R$. Then $\chi^S_{s,s'}$ is the set of $t \in \mathbb{N}$ for which there exists a finite run $\rho = (q_i)_{0 \leq i \leq j}$ in $T_{\mathcal{A}}$ such that

— the path $\pi(\rho) = (r_k)_{0 \leq k \leq k'}$ in $R_{\mathcal{A}}$ associated with ρ satisfies $s = r_0$, $s' = r_{k'}$ and $r_k \in S$ for any k, $0 \leq k < k'$,

— the run ρ has duration $\mathrm{D}_\rho(q_j) = t$,

— any position $p < q_j$ in ρ satisfies $\mathrm{D}_\rho(p) < t$.

The differences with $\lambda^S_{s,s'}$ are the following. The duration t is necessarily a natural number. A third condition has been added : it means that q_j is the first position in ρ with duration equal to t (remember that several positions can have the same duration in a run).

Proposition 5.7. *Let $\mathcal{A}$ be a discrete- or a dense-timed automaton. Then set $\chi^S_{s,s'}$ is definable by a* PA *formula. The construction of the formula is effective.*

Proof. We begin with the case of a discrete-timed automaton $\mathcal{A}$. The proof is similar to the proof of Proposition 5.2 with a construction of automaton $\mathcal{C}$ slightly different, in a way to take into account the third condition of Definition 5.6. We add to $R_{\mathcal{A}} = (R, F)$ a new region s'_c as a copy of s'. We also add a new edge (r, s'_c) as a copy of (r, s') for any $r \in R$ such that clocks have been increased by 1 from r to s'. Then $\mathcal{C}$ is constructed with $S \cup \{s'_c\}$ as set of states and $F \cap S \times (S \cup \{s'_c\})$ as set of transitions. The rest of the proof is identical.

We now treat the dense case. The proof is similar to the proof of Proposition 5.3. The construction of automaton $\mathcal{C}$ is modified as above with a copy of the edges along which clock x_0 has been reset to 0. As $t \in \mathbb{N}$ in the definition of $\chi^S_{s,s'}$, the situation of Lemma 5.5 does not occur. We thus obtain a PA formula. □

We end this subsection with the definition of the set Pr_S.

Definition 5.8. Let $\mathcal{A}$ be a timed automaton and $R_{\mathcal{A}} = (R, F)$ be its region graph. Let $S \subseteq R$. Then Pr_S is the set

$$\{\ s \in S \quad | \quad \text{there exists a progressive path in } R_{\mathcal{A}} \text{ with all its vertices in } S \text{ and its first vertex is equal to } s\ \}.$$

Proposition 5.9. *It is decidable whether $s \in \mathrm{Pr}_S$.*

Proof. Suppose that $\mathcal{A}$ is discrete-timed. The proof is again close to the proof of Proposition 5.2. Here $\mathcal{C}$ has S as set of states and $F \cap S \times S$ as set of transitions. It has a unique initial state equal to s. Then $s \in \mathrm{Pr}_S$ iff the frying pan automaton $\mathcal{C}'$ has a cycle. The proof is similar if $\mathcal{A}$ is dense-timed. □

6. MODEL-CHECKING

In this section, we solve the model-checking problem. Complexity issues are discussed in Section 7.

6.1 Discrete Model-Checking

Theorem 6.1. *Let $\mathcal{A}$ be a discrete-timed automaton and r be a region of $R_{\mathcal{A}}$. Let f be a* PTCTL *formula with $P_f = \{\theta_1, \ldots, \theta_m\}$. Then there exists a* PA *formula*

$\Delta_{f,r}$ such that $r \models_v f$ for some valuation v iff the sentence $\exists\theta_1 \cdots \exists\theta_m \Delta_{f,r}$ is true. The construction of $\Delta_{f,r}$ is effective.

Before giving the proof of this theorem, we make two comments. First, instead of working with the PTCTL grammar given in Definition 3.1, we prefer to work with the following grammar in a way to avoid the universal quantifier of the $\forall$U operator :

$$\varphi ::= \sigma \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists\bigcirc \varphi \mid \varphi \exists \mathrm{U}_{\sim\alpha} \varphi \mid \exists\Box_{\sim\alpha}\varphi \tag{1}$$

The semantics of the new operator $\exists\Box_{\sim\alpha}\varphi$ is defined as follows :

$q \models_v \exists\Box_{\sim\alpha}\varphi$ iff there exists a run $\rho = (q_i)_{i\geq 0}$ in $T_{\mathcal{A}}$ with $q = q_0$ such that $p \models_v \varphi$ for any position p in ρ with duration $\mathrm{D}_\rho(p) \sim v(\alpha)$.

The semantics of the new grammar and the grammar of Definition 3.1 are equivalent. Indeed formula $\varphi\ \forall \mathrm{U}_{\sim\alpha}\ \psi$ is translated in the grammar given in (1) by the formula $\neg\left((\exists\Box_{\sim\alpha}\neg\psi) \vee (\neg\psi\ \exists \mathrm{U}_{\sim\alpha}\ \neg(\varphi \vee \psi))\right)$. Conversely, formula $\exists\Box_{\sim\alpha}\varphi$ is translated by $\neg(\top\ \forall \mathrm{U}_{\sim\alpha}\ \neg\varphi)$.

Second, we use in the next proof the statements of Propositions 5.2, 5.3, 5.7 and 5.9 with the following notation. We denote the PA formula for the set $\lambda^S_{s,s'}$ by $\lambda^S_{s,s'}(t)$. Similarly, we denote the PA formula for the set $\chi^S_{s,s'}$ by $\chi^S_{s,s'}(t)$. Proposition 5.9 states that it is decidable whether $r \in \mathrm{Pr}_R$. Therefore, we use the notation $\mathrm{Pr}_R(r)$ for the PA formula true or false depending on whether r belongs to Pr_R or not.

Proof of Theorem 6.1. We consider a fixed state $q = (l, x)$ of $T_{\mathcal{A}}$ such that $r = [q]$. We use both notations $q \models_v f$ and $r \models_v f$ thanks to Proposition 4.3. The construction of $\Delta_{f,r}$ is done by induction on the formula f. The set of free variables of $\Delta_{f,r}$ is equal to P_f.

We begin with formulae $f = \varphi$ of first type. The construction of $\Delta_{\varphi,r}$ is easy in the first case $\varphi = \sigma$. By Definition 3.2, $q \models_v \varphi$ iff there exists a run $\rho = (q_i)_{i\geq 0}$ in $T_{\mathcal{A}}$ with $q = q_0$ and $\sigma \in \mathcal{L}(l)$. Equivalently there exists a progressive path in $R_{\mathcal{A}}$ starting at r such that $\sigma \in \mathcal{L}(l)$. Thus

$$\Delta_{\varphi,r} = \mathrm{Pr}_R(r) \wedge (\sigma \in \mathcal{L}(l))$$

where $\sigma \in \mathcal{L}(l)$ means the PA formula true or false depending on whether σ belongs to $\mathcal{L}(l)$ or not.

The construction of $\Delta_{\varphi,r}$ is also easy in the next three cases.

$$\begin{array}{ll} \varphi = \neg\psi & \Delta_{\varphi,r} = \neg\Delta_{\psi,r} \\ \varphi = \psi \vee \phi & \Delta_{\varphi,r} = \Delta_{\psi,r} \vee \Delta_{\phi,r} \\ \varphi = \exists\bigcirc \psi & \Delta_{\varphi,r} = \bigvee_{(r,r')\in F} (\Delta_{\psi,r'} \wedge \mathrm{Pr}_R(r')) \end{array}$$

Let us study the case $\varphi = \psi \exists \mathrm{U}_{\sim\alpha}\ \phi$. Suppose that $q \models_v \varphi$. By Definition 3.2, there exists a run $\rho = (q_i)_{i\geq 0}$ in $T_{\mathcal{A}}$ with $q = q_0$, there exists a position p in ρ with $\mathrm{D}_\rho(p) \sim \alpha$ such that $p \models_v \phi$ and $p' \models_v \psi$ for all $p' < p$. If $p = q$, then $q \models_v \phi$. Otherwise let $t = \mathrm{D}_\rho(p)$ and consider the path $\pi(\rho) = (r_k)_{k\geq 0}$ in $R_{\mathcal{A}}$. We have $r_0 = r$ and $r_{k'} = [p]$ for some $k' > 0$. Denote by r' the region $r_{k'}$ and by S the set $\{r_k \mid 0 \leq k < k'\}$. We thus observe that $t \in \lambda^S_{r,r'}$, $r' \models_v \phi$ and $s \models_v \psi$ for all $s \in S$. Moreover, r' is the first vertex of the progressive path $(r_k)_{k\geq k'}$, i.e.,

$r' \in \mathrm{Pr}_R$. Therefore we get for $\Delta_{\varphi,r}$ the next formula where $\lambda^S_{r,r'}(t)$ denotes the PA formula defining the set $\lambda^S_{r,r'}$ (see Proposition 5.2)

$$\begin{array}{rcl}\Delta_{\psi \exists \mathrm{U}_{\sim\alpha}\, \phi,r} & = & (\Delta_{\phi,r} \wedge \mathrm{Pr}_R(r) \wedge (0 \sim \alpha)) \\ & & \vee \bigvee_{r' \in R} \bigvee_{S \subseteq R} \big((\exists t \sim \alpha\ \lambda^S_{r,r'}(t)) \wedge \Delta_{\phi,r'} \wedge \bigwedge_{s \in S} \Delta_{\psi,s} \wedge \mathrm{Pr}_R(r') \big)\end{array}$$

We now turn to the case $\varphi = \exists \Box_{\sim\alpha} \psi$. Since time is discrete, we can restrict the study to $\sim \in \{<, \geq, =\}$.

Suppose that $\varphi = \exists \Box_{<\alpha} \psi$. Then $q \models_v \exists \Box_{<\alpha} \psi$ iff there exists a run $\rho = (q_i)_{i \geq 0}$ with $q = q_0$ such that $p \models_v \psi$ for any position p in ρ with duration $\mathrm{D}_\rho(p) < v(\alpha)$. Consider the *first* position p' in ρ such that $\mathrm{D}_\rho(p') = v(\alpha)$. Define $r' = [p']$ and $S = \{s \in R \mid s = [p],\ p < p'\}$. It follows that $\mathrm{D}_\rho(p') \in \chi^S_{r,r'}$ (see Definition 5.6). Moreover, for any $s \in S$, $s \models_v \psi$ and r' is the first vertex of a progressive path. Hence

$$\Delta_{\exists \Box_{<\alpha} \psi,r} = \bigvee_{r' \in R} \bigvee_{S \subseteq R} \big(\chi^S_{r,r'}(\alpha) \wedge \bigwedge_{s \in S} \Delta_{\psi,s} \wedge \mathrm{Pr}_R(r') \big)$$

Similarly for $\varphi = \exists \Box_{\geq\alpha} \psi$, we have

$$\Delta_{\exists \Box_{\geq\alpha} \psi,r} = \bigvee_{r' \in R} \bigvee_{S \subseteq R} \big(\chi^R_{r,r'}(\alpha) \wedge \bigwedge_{s \in S} \Delta_{\psi,s} \wedge \mathrm{Pr}_S(r') \big)$$

For $\varphi = \exists \Box_{=\alpha} \psi$, remember that there may exist several positions p in ρ such that $\mathrm{D}_\rho(p) = v(\alpha)$. In the next formula, r'' is the region $[p'']$ such that p'' is the first position in ρ with $\mathrm{D}_\rho(p'') = v(\alpha) + 1$.

$$\Delta_{\exists \Box_{=\alpha} \psi,r} = \bigvee_{r',r'' \in R} \bigvee_{S \subseteq R} \big(\chi^R_{r,r'}(\alpha) \wedge \chi^S_{r',r''}(1) \wedge \bigwedge_{s \in S} \Delta_{\psi,s} \wedge \mathrm{Pr}_R(r'') \big)$$

This completes the proof for PTCTL formulae of first type. The construction of $\Delta_{f,r}$ is easily done for formulae f of second type as follows. If $f = \varphi$ (resp. $\theta \sim \beta$, $\neg f_1$, $f_1 \vee f_2$, $\exists \theta f_1$), then $\Delta_{f,r} = \Delta_{\varphi,r}$ (resp. $\theta \sim \beta$, $\neg \Delta_{f_1,r}$, $\Delta_{f_1,r} \vee \Delta_{f_2,r}$, $\exists \theta \Delta_{f_1,r}$). □

Corollary 6.2. *The discrete model-checking problem is decidable.*

6.2 Dense Model-Checking

The proof of Theorem 6.1 is an easy translation to PA of PTCTL semantics, as soon as definability in PA of sets $\lambda^S_{r,r'}$ and $\chi^S_{r,r'}$ is established. The same situation repeats for dense model-checking.

Theorem 6.3. *Let $\mathcal{A}$ be a dense-timed automaton and r be a region of $R_\mathcal{A}$. Let f be a* PTCTL *formula with $P_f = \{\theta_1, \ldots, \theta_m\}$. Then there exists a* RA *formula $\Delta_{f,r}$ such that $r \models_v f$ for some valuation v iff the sentence $\exists \theta_1 \cdots \exists \theta_m \Delta_{f,r}$ is true. The construction of $\Delta_{f,r}$ is effective.*

Proof. The approach is the same as in the proof of Theorem 6.1. It is an induction on the first type formulae of the grammar given in (1) (except that $\exists \bigcirc \varphi$ does not exist).

We focus on operators $\exists \mathrm{U}_{\sim\alpha}$ and $\exists \Box_{\sim\alpha}$ only. The related formulae $\Delta_{\varphi,r}$ are very similar to the formulae for discrete time. They are given in Table I. Some comments are given hereafter about the little modifications due to dense time.

Concerning $\exists \mathrm{U}_{\sim\alpha}$, formula $\Delta_{\varphi,r}$ is identical to the discrete case, except an additional subformula $\neg \mathrm{B}(r') \rightarrow \Delta_{\psi,r'}$. Given a region r', $\mathrm{B}(r')$ is true iff r' is a

Table I. Formulae $\Delta_{\varphi,r}$ for dense time

φ	$\Delta_{\varphi,r}$
$\psi \exists \mathrm{U}_{\sim\alpha}\, \phi$	$\big(\Delta_{\phi,r} \wedge \mathrm{Pr}_R(r) \wedge (0 \sim \alpha)\big) \quad \vee$ $\bigvee_{r' \in R} \bigvee_{S \subseteq R} ((\exists t \sim \alpha\ \lambda^S_{r,r'}(t)) \wedge \Delta_{\phi,r'} \wedge \bigwedge_{s \in S} \Delta_{\psi,s}$ $\wedge\, \mathrm{Pr}_R(r') \wedge (\neg\, \mathrm{B}(r') \rightarrow \Delta_{\psi,r'}))$
$\exists \Box_{\geq\alpha} \psi$	$\bigvee_{r' \in R} \bigvee_{S \subseteq R} \Big(\chi^R_{r,r'}(\alpha) \wedge \bigwedge_{s \in S} \Delta_{\psi,s} \wedge \mathrm{Pr}_S(r')\Big)$
$\exists \Box_{<\alpha} \psi$	$\bigvee_{r' \in R} \bigvee_{S \subseteq R} \Big(\chi^S_{r,r'}(\alpha) \wedge \bigwedge_{s \in S} \Delta_{\psi,s} \wedge \mathrm{Pr}_R(r') \wedge (\neg\, \mathrm{B}(r') \rightarrow \Delta_{\psi,r'})\Big)$
$\exists \Box_{\leq\alpha} \psi$	$\bigvee_{S \subseteq R} \bigvee_{r' \in S} \bigvee_{r'',(r',r'') \in F^>} \Big(\lambda^S_{r,r'}(\alpha) \wedge \bigwedge_{s \in S} \Delta_{\psi,s} \wedge \mathrm{Pr}_R(r'')\Big)$
$\exists \Box_{>\alpha} \psi$	$\bigvee_{S \subseteq R} \bigvee_{r' \in R} \bigvee_{r'',(r',r'') \in F^>}$ $\Big(\lambda^R_{r,r'}(\alpha) \wedge \bigwedge_{s \in S} \Delta_{\psi,s} \wedge \mathrm{Pr}_S(r'') \wedge (\neg\, \mathrm{B}(r') \rightarrow \Delta_{\psi,r'})\Big)$
$\exists \Box_{=\alpha} \psi$	$\bigvee_{S \subseteq R} \bigvee_{r',r'' \in S} \bigvee_{r''',(r'',r''') \in F^>}$ $\Big(\chi^R_{r,r'}(\alpha) \wedge \lambda^S_{r',r''}(0) \wedge \bigwedge_{s \in S} \Delta_{\psi,s} \wedge \mathrm{Pr}_R(r''')\Big)$

boundary region. This new subformula is necessary because if r' is not boundary, then $r' \models_v \psi$. It is decidable whether $\mathrm{B}(r')$ is true[3].

Since time is dense, we can no longer restrict the study of $\exists \Box_{\sim\alpha}$ to $\sim \in \{<, \geq, =\}$.

Formula $\Delta_{\varphi,r}$ for $\varphi = \exists \Box_{\geq\alpha} \psi$ is the same as for discrete time.

If $\varphi = \exists \Box_{<\alpha} \psi$, an additional subformula $\neg\, \mathrm{B}(r') \rightarrow \Delta_{\psi,r'}$ is again necessary.

Let us discuss the new case $\varphi = \exists \Box_{\leq\alpha} \psi$. We need to express that r' is the region $[p']$ such that p' is the *last* position in ρ with $\mathrm{D}_\rho(p') = v(\alpha)$. This is equivalent to say that there is a region $r'' = [p'']$ with p'' some position in ρ, there is an edge (r', r'') in $R_\mathcal{A}$ such that $p' \xrightarrow{\tau} p''$ and τ is *strictly* positive. We denote by $F^>$ the set of edges $(r', r'') \in F$ with $\tau > 0$. Thus, compared with $\Delta_{\exists \Box_{<\alpha} \psi, r}$, two new things appear in $\Delta_{\exists \Box_{\leq\alpha} \psi, r}$: first $\bigvee_{r'',(r',r'') \in F^>}$ as just discussed, second $\bigvee_{r' \in S}$ instead of $\bigvee_{r' \in R}$ to express that $r' \models_v \psi$.

The case $\varphi = \exists \Box_{>\alpha} \psi$ is solved similarly.

For the last case $\varphi = \exists \Box_{=\alpha} \psi$, we express that r' is the region $[p']$ such that p' is the *first* position in ρ with $\mathrm{D}_\rho(p') = v(\alpha)$, and r'' is the region $[p'']$ such that p'' is the *last* position in ρ with $\mathrm{D}_\rho(p'') = v(\alpha)$. In particular, we have $\lambda^S_{r',r''}(0)$.

Finally, it is important to note that to ensure that the values of parameters $\theta_1, \ldots, \theta_m$ are integers, it is necessary to add to the formula $\Delta_{f,r}$ the conjunction with $\bigwedge_{1 \leq i \leq m} \mathbb{N}(\theta_i)$. □

Corollary 6.4. *The dense model-checking problem is decidable.*

6.3 Parameter Synthesis

The proofs given for Theorems 6.1 and 6.3 also solve the parameter synthesis problem (See Problem 3.4). Indeed the formula $\Delta_{f,r}$ is a PA or RA formula with free variables in P_f that exactly defines the values of the parameters of P_f that satisfy formula f at region r. As its construction is effective, we get a symbolic representation of these values, that supports boolean operations, projections and checking emptiness.

Theorem 6.5. *Let $\mathcal{A}$ be a timed automaton and r be a region of $R_\mathcal{A}$. Let f be*

[3]Indeed any region is specified by a location, the integral parts of the clock assignments, and the ordering of the fractional parts of the clock assignments. Thus r' is boundary iff at least one clock value has a fractional part equal to 0.

a PTCTL *formula with* $P_f = \{\theta_1, \ldots, \theta_m\}$*. Then the formula* $\Delta_{f,r}(\theta_1, \ldots, \theta_m)$ *with free variables* $\theta_1, \ldots, \theta_m$ *is an effective characterization is the set of valuations* v *for* P_f *such that* $r \models_v f$*.*

Let us denote by $V(\mathcal{A}, f, r)$ the set of valuations v for P_f such that $r \models_v f$. Since PA and RA have a decidable theory, any question formulated in these logics about the set $V(\mathcal{A}, f, r)$ is therefore decidable. Let us enumerate some interesting such questions[4] and their translation in PA and RA (in the case of RA, as done in the proof of Theorem 6.3, it is necessary to add to the following formulae a conjunction with $\bigwedge_{1 \leq i \leq m} \mathbb{N}(\theta_i)$).

(1) "Is the set $V(\mathcal{A}, f, r)$ not empty ?"

$$\exists \theta_1, \ldots, \exists \theta_m \Delta_{f,r}(\theta_1, \ldots, \theta_m)$$

(2) "Does the set $V(\mathcal{A}, f, r)$ contain every valuation ?"

$$\forall \theta_1, \ldots, \forall \theta_m \Delta_{f,r}(\theta_1, \ldots, \theta_m)$$

(3) "Is the set $V(\mathcal{A}, f, r)$ finite ?"

$$\exists z \forall \theta_1, \ldots, \forall \theta_m \Delta_{f,r}(\theta_1, \ldots, \theta_m) \Rightarrow \bigwedge_{1 \leq i \leq m} \theta_i < z$$

We could also be interested in optimization problems. Assume for example that there is only one parameter θ like in a formula f such $\exists \Box_{\leq \theta} \sigma$. It is interesting to know what is the maximum value z of θ for which this formula is satisfied. Such a maximum value can be defined by $\Delta_{f,r}(z) \wedge \forall \theta (\Delta_{f,r}(\theta) \Rightarrow \theta \leq z)$. Now, PA and RA have an effective quantifier elimination, by adding the congruences $\equiv_n$ for each natural number n in PA and by adding the congruences $\equiv_n$ and the integer part operation $\lfloor \ \rfloor$ in RA (see [Weispfenning 1999]). It follows that we can compute the value of z. More generally, if an optimisation problem can be formulated in these logics, then its solutions can be symbolically and effectively represented by a quantifier-free formula.

7. COMPLEXITY

In this section, we study the complexity of the algorithms that we proposed in the previous section for the model-checking problem. We first establish complexity results for discrete time. We then show that these results remain valid for dense time.

Throughout this section, $\mathcal{A}$ is a timed automaton and $R_{\mathcal{A}}$ its region graph. We suppose that f is a PTCTL formula constructed by induction on the grammar given in (1). We also suppose that S is a subset of R and r, r' are two regions of R.

7.1 Discrete Time

To compute the complexity of the algorithm given in Theorem 6.1, we first study the size of formulae $\lambda^S_{r,r'}(t)$ and $\Delta_{f,r}$ and the time to construct them (Propositions 7.1 and 7.2). By *size* of a formula f, we mean the number of symbols used to write it, and we denote it by $|f|$.

[4]Similar questions are treated in [Alur et al. 1999] for linear time.

PROPOSITION 7.1. *The* PA *formulae* $\lambda^S_{r,r'}(t)$ *and* $\chi^S_{r,r'}(t)$ *have a size, and can be constructed in time, bounded by* $\mathcal{O}(2^{2\cdot|R_\mathcal{A}|})$.

PROOF. To construct the formula $\lambda^S_{r,r'}(t)$, as explained in the proof of Proposition 5.2, we apply the subset construction to some subgraph of $R_\mathcal{A}$. The resulting automaton $\mathcal{C}'$ has at most $\mathcal{O}(2^{|R_\mathcal{A}|})$ states and can be constructed in time bounded by $\mathcal{O}(|R_\mathcal{A}|^2 \cdot 2^{|R_\mathcal{A}|})$ [Champarnaud 2001]. We construct for each final state of $\mathcal{C}'$ a formula of the form $t = m$ or $\exists z\ t = m + cz$. So the number of these formulae is bounded by $|\mathcal{C}'|$. Furthermore, the size[5] of constants c and m are also bounded by $|\mathcal{C}'|$. As a consequence the size of formula $\lambda^S_{r,r'}(t)$ is bounded by $\mathcal{O}(|\mathcal{C}'|^2) = \mathcal{O}(2^{2\cdot|R_\mathcal{A}|})$. To construct formula $\lambda^S_{r,r'}(t)$, we first need to construct the automaton $\mathcal{C}'$. Therefore $\lambda^S_{r,r'}(t)$ can be constructed in time bounded by $\mathcal{O}(|R_\mathcal{A}|^2 \cdot 2^{|R_\mathcal{A}|} + 2^{2\cdot|R_\mathcal{A}|}) = \mathcal{O}(2^{2\cdot|R_\mathcal{A}|})$.

The proof is similar for formula $\chi^S_{r,r'}(t)$. □

PROPOSITION 7.2. *The* PA *formula* $\Delta_{f,r}$ *has a size, and can be constructed in time, bounded by* $\mathcal{O}(2^{4\cdot|R_\mathcal{A}|\cdot|f|})$.

PROOF. The PA formula $\Delta_{f,r}$ is constructed by induction on the structure of f (see the proof of Theorem 6.1). We first prove by induction on f that $|\Delta_{f,r}|$ is bounded by

$$\mathcal{O}(2^{3\cdot|R_\mathcal{A}|\cdot|f|} \cdot |R_\mathcal{A}|^{3\cdot|f|}).$$

For the base case, f is either an atomic proposition σ or of the form $\theta \sim \beta$. Formula $\Delta_{f,r}$ has the expected length.

For the induction case, we only treat the case where f is of the form $\exists\Box_{=\alpha}\psi$ or $\psi\exists \mathrm{U}_{\sim\alpha}\,\phi$ as the other cases are simpler or similar. We recall the definition of $\Delta_{f,r}$

$$\begin{array}{lcl}
\Delta_{\exists\Box_{=\alpha}\psi,r} & = & \bigvee_{r',r''\in R}\bigvee_{S\subseteq R}\left(\chi^R_{r,r'}(\alpha)\wedge\chi^S_{r',r''}(1)\wedge\bigwedge_{s\in S}\Delta_{\psi,s}\wedge \Pr_R(r'')\right)\\
\Delta_{\psi\exists \mathrm{U}_{\sim\alpha}\,\phi,r} & = & (\Delta_{\phi,r}\wedge \Pr_R(r)\wedge(0\sim\alpha)) \qquad \vee\\
 & & \bigvee_{r'\in R}\bigvee_{S\subseteq R}\left((\exists t\sim\alpha\ \lambda^S_{r,r'}(t))\wedge\Delta_{\phi,r'}\wedge\bigwedge_{s\in S}\Delta_{\psi,s}\wedge \Pr_R(r')\right)
\end{array}$$

Using the induction hypothesis and Proposition 7.1, the size of the above formulae are bounded as follows (with $n = |R_\mathcal{A}|$)

$$\begin{array}{l}
|\Delta_{\exists\Box_{=\alpha}\psi,r}|\\
\leq \mathcal{O}\left(n^2 2^n\cdot(2^{2n}+2^{2n}+n|\Delta_{\psi,s}|+1)\right)\\
\leq \mathcal{O}\left(n^3 2^n\cdot(2^{2n}+|\Delta_{\psi,s}|)\right)\\
\leq \mathcal{O}\left(n^3 2^n\cdot(2^{2n}\cdot|\Delta_{\psi,s}|)\right)\\
\leq \mathcal{O}\left(2^{3n\cdot(|\psi|+1)}\cdot n^{3\cdot(|\psi|+1)}\right)\\
\leq \mathcal{O}\left(2^{3n\cdot|f|}\cdot n^{3\cdot|f|}\right).
\end{array}$$

$$\begin{array}{l}
|\Delta_{\psi\exists \mathrm{U}_{\sim\alpha}\,\phi,r}|\\
\leq \mathcal{O}\left(|\Delta_{\phi,r}|+1+1+n2^n\cdot(1+2^{2n}+|\Delta_{\phi,r'}|+n|\Delta_{\psi,s}|+1)\right)\\
\leq \mathcal{O}\left(n^2 2^n\cdot(2^{2n}+|\Delta_{\phi,r'}|+|\Delta_{\psi,s}|)\right)\\
\leq \mathcal{O}\left(n^3 2^n\cdot(2^{2n}\cdot|\Delta_{\phi,r'}|\cdot|\Delta_{\psi,s}|)\right)\\
\leq \mathcal{O}\left(2^{3n\cdot(|\phi|+|\psi|+1)}\cdot n^{3\cdot(|\phi|+|\psi|+1)}\right)\\
\leq \mathcal{O}\left(2^{3n\cdot|f|}\cdot n^{3\cdot|f|}\right).
\end{array}$$

[5]in terms of unary representation

We have thus proved that the size of $\Delta_{f,r}$ is bounded by $\mathcal{O}(2^{3\cdot|R_\mathcal{A}|\cdot|f|}\cdot|R_\mathcal{A}|^{3\cdot|f|})$, and so by $\mathcal{O}(2^{4\cdot|R_\mathcal{A}|\cdot|f|})$.

To construct $\Delta_{f,r}$, we need to take into account the time to decide whether a region s belongs to the set Pr_S. It is explained in the proof of Proposition 5.9 that $s \in \mathrm{Pr}_S$ iff the frying pan automaton $\mathcal{C}'$ has a cycle. The time to construct $\mathcal{C}'$ and to check if it has a cycle is bounded by $\mathcal{O}(|R_\mathcal{A}|^2 \cdot 2^{|R_\mathcal{A}|})$. If follows that the time to construct $\Delta_{f,r}$ is bounded by $\mathcal{O}(2^{4\cdot|R_\mathcal{A}|\cdot|f|})$. □

Corollary 7.3. *The discrete model-checking problem is in* 4ExpTime *in the product of the sizes of $R_\mathcal{A}$ and f, and in* 5ExpTime *in the product of the sizes of $\mathcal{A}$ and f.*

Proof. By Theorem 6.1, the model-checking problem reduces to checking the satisfiability of the PA formula $\Delta_{f,r}$. We recall that the size of $R_\mathcal{A}$ is in $\mathcal{O}(2^{|\mathcal{A}|})$ and that PA has a decidable theory with complexity 3ExpTime in the size of the formula. The claim follows by Proposition 7.2. □

We now turn to the fragment ∃PTCTL of discrete PTCTL logic where the universal quantification over parameters is prohibited (see Definition 3.1). For this fragment, we have a lower complexity.

Proposition 7.4. *Let f be a formula of* ∃PTCTL *and let $r \in R$. Then the discrete model-checking problem for f and r can be solved in* 2ExpTime *in the product of the sizes of $R_\mathcal{A}$ and f, and in* 3ExpTime *in the product of the sizes of $\mathcal{A}$ and f.*

Proof. We are going to show that the PA formula $\Delta_{f,r}$ constructed in the proof of Theorem 6.1 can be transformed into a formula of the existential fragment of PA. The size of this formula will be again bounded by $\mathcal{O}(2^{4\cdot|R_\mathcal{A}|\cdot|f|})$ as in Proposition 7.2. The claim follows since the existential fragment of PA has a decidable theory in NP.

The formula $\Delta_{f,r}$ is in ∃PA, except in one case : when f has a subformula of the form $\psi \exists \mathrm{U}_{\sim\alpha}\, \phi$. In this case, the existential formula $\exists t \sim \alpha\ \lambda^S_{r,r'}(t)$ appears in $\Delta_{\psi \exists \mathrm{U}_{\sim\alpha}\, \phi, r}$ and a negation applied to it transforms $\exists t$ into $\forall t$.

We now proceed to the proof. Since f is a formula of ∃PTCTL, it has the form $\exists\theta_1 \cdots \exists\theta_m\ g$ with g a PTCTL formula without quantifiers. Let us prove by induction on g that $\Delta_{g,r}$ and $\Delta_{\neg g,r}$ can be both constructed as a ∃PA formula and that their sizes $|\Delta_{g,r}|$ and $|\Delta_{\neg g,r}|$ are bounded by $\mathcal{O}(2^{3\cdot|R_\mathcal{A}|\cdot|g|}\cdot|R_\mathcal{A}|^{3\cdot|g|})$. This proof is not fully detailed since it closely follows the lines of the proof of Theorem 6.1 and Proposition 7.2.

To show that $\Delta_{g,r}$ and $\Delta_{\neg g,r}$ are in ∃PA, it is enough to show that the negation of $\exists t \sim \alpha\ \lambda^S_{r,r'}(t)$ is in ∃PA (the rest is treated by induction). Recall that formula $\lambda^S_{r,r'}(t)$ is a disjunction of formulae of the form $t = m$ or $\exists z\ t = m + cz$ where m and c are integer constants. So $\exists t \sim \alpha\ \lambda^S_{r,r'}(t)$ is a disjunction of formulae of the form

$$\exists t\ t \sim \alpha \wedge t = m,$$
$$\exists t\ \exists z\ t \sim \alpha \wedge t = m + cz.$$

In every case except one, existential quantifiers can be eliminated as indicated in Table II. Therefore, it remains to study the negation of formula

Table II. Elimination of existential quantifiers

Formula	Equivalent formula
$\exists t\ t \sim \alpha \wedge t = m$	$m \sim \alpha$
$\exists t\ \exists z\ t > (\geq)\ \alpha\ \wedge\ t = m + cz$	true
$\exists t\ \exists z\ t < (\leq)\ \alpha\ \wedge\ t = m + cz$	$m < (\leq)\ \alpha$

$$\exists t\ \exists z\ t = \alpha \wedge t = m + cz$$

which is equivalent to formula $\exists z\ \alpha = m + cz$. We consider two cases. Suppose first that $m < c$. Then $\neg(\exists z\ \alpha = m + cz)$ is equivalent to $\bigvee_{m'<c, m' \neq m} \alpha = m' \bmod c$ which is expressed in $\exists$PA as $\bigvee_{m'<c, m' \neq m} \exists z\ \alpha = m' + cz$. Suppose now that $m \geq c$. If we consider $m_0 = m \bmod c$, then $\exists z\ \alpha = m + cz$ is equivalent to $\exists z\ \alpha = m_0 + cz \wedge \alpha \geq m$. The latter formula can be treated as explained just before.

Let us now study the sizes of $\Delta_{g,r}$ and $\Delta_{\neg g,r}$. Again we focus on $\exists t \sim \alpha\ \lambda^S_{r,r'}(t)$ and its negation : their sizes are both bounded by $\mathcal{O}(n \cdot 2^{2 \cdot |R_\mathcal{A}|})$. Indeed, we know by Proposition 7.1 that $|\lambda^S_{r,r'}(t)|$ is bounded by $\mathcal{O}(2^{2 \cdot |R_\mathcal{A}|})$. This bound can be improved into $\mathcal{O}(n 2^{|R_\mathcal{A}|})$ if the constants c and m are written in binary representation. We thus obtain the announced bound. Now looking at the proof of Proposition 7.2, we see by induction on g that $|\Delta_{g,r}|$ and $|\Delta_{\neg g,r}|$ are both bounded $\mathcal{O}(2^{3 \cdot |R_\mathcal{A}| \cdot |g|} \cdot |R_\mathcal{A}|^{3 \cdot |g|})$.

This completes the proof. □

7.2 Dense Time

We now show that we obtain identical complexities for dense time. Indeed, we recall that due to the predicate $\mathbb{N}$, RA has a decidable theory with the same complexity 3ExpTime as PA (This predicate appears in the formulae constructed in the proofs of Propositions 5.3, 5.7 and Theorem 6.3).

The proofs are only sketched since they are similar. We begin with a result similar to Corollary 7.3.

Theorem 7.5. *The dense model-checking problem is in* 4ExpTime *in the sizes of $R_\mathcal{A}$ and f, and in* 5ExpTime *in the product of the sizes of $\mathcal{A}$ and f.*

Proof. First, the RA formulae $\lambda^S_{r,r'}(t)$ and $\chi^S_{r,r'}(t)$ have a size, and can be constructed in time, bounded by $\mathcal{O}(2^{2 \cdot |R_\mathcal{A}|})$. Second, we can prove by induction on f that $|\Delta_{f,r}|$ is bounded by $\mathcal{O}(2^{3 \cdot |R_\mathcal{A}| \cdot |f|} \cdot |R_\mathcal{A}|^{4 \cdot |f|})$. Thus $\Delta_{f,r}$ has a size, and can be constructed in time, bounded by $\mathcal{O}(2^{4 \cdot |R_\mathcal{A}| \cdot |f|})$. Third, RA has a decidable theory with complexity 3ExpTime in the size of the formula. The announced result follows. □

The next proposition is the counterpart of Proposition 7.4 for formulae f of dense $\exists$PTCTL logic. Note that the proof will transform the formula $\Delta_{f,r}$ not only into a formula of the existential fragment of RA but also of PA.

Proposition 7.6. *Let f be a $\exists$PTCTL formula and let $r \in R$. Then the dense model-checking problem for f and r can be solved in* 2ExpTime *in the product of the sizes of $R_\mathcal{A}$ and f, and in* 3ExpTime *in the product of the sizes of $\mathcal{A}$ and f.*

Proof. The proof follows the same schema as for Proposition 7.4. The main difference is that formula $\exists t \sim \alpha\ \lambda^S_{r,r'}(t)$ is a disjunction of RA (instead of PA)

Table III. Elimination of existential quantifiers

Formula		Equivalent formula
Formula (3) with	$t = \alpha$	false
	$t < (\leq)\ \alpha$	$m + 1 \leq \alpha$
	$t > (\geq)\ \alpha$	$\alpha \leq m$
Formula (4) with	$t = \alpha$	false
	$t < (\leq)\ \alpha$	$m + 1 \leq \alpha$
	$t > (\geq)\ \alpha$	true

formulae of the form

$$\exists t \quad t \sim \alpha \wedge t = m, \tag{2}$$

$$\exists t\ \exists z\ \ t \sim \alpha \wedge \mathbb{N}(z) \wedge t = m + cz, \tag{3}$$

$$\exists t \quad t \sim \alpha \wedge m < t < m + 1, \tag{4}$$

$$\exists t\ \exists z\ \ t \sim \alpha \wedge \mathbb{N}(z) \wedge m + cz < t < m + cz + 1 \tag{5}$$

where m and c are constants in $\mathbb{N}$.

The first two formulae are treated as for Proposition 7.4. For the last two formulae, existential quantifiers can be eliminated as indicated in Table III.

So, in any case, we obtain a PA (instead of RA) formula. Looking at the proof of Theorem 6.3, we see that the transformed formula $\Delta_{f,r}$ is a formula of the existentiel fragment of PA. This completes the proof. □

Corollary 7.7. *Let f be a* ∃PTCTL *formula and let $r \in R$. Then $\Delta_{f,r}$ can be constructed as a formula of the existential fragment of* PA.

In Sections 7.1 and 7.1, we have given upper bounds for the discrete and dense model-checking problems. Let us make some comments about lower bounds :

—Discrete and dense model-checking for ∃PTCTL logic is at least PSpace-Hard since the model-checking problem defined in [Alur et al. 1990] is a particular case of our problem (PSpace-Hardness is proved in [Alur et al. 1990] for dense time and in [Aceto and Laroussinie 2002] for discrete time).

—Discrete and dense model-checking for PTCTL logic is at least 2NExpTime-Hard since Presburger arithmetic is present in PA and in RA (see [Fischer and Rabin 1974]).

Therefore there is a gap between lower and upper bounds which needs more research effort.

8. RELATED WORKS

The work by Wang et al in [Wang 1995; Wang and Hsiung 1997] is closely related to our work. The logic they consider is a strict subset of the logic considered in this paper: in their works the parameters are all implicitely quantified existentially, so their logic corresponds to our fragment ∃PTCTL. The technique they use to establish the decidability result while ingenious is more complex than the technique that we propose in this paper. So the main contribution of the present work with regard to their work is, we feel, a simpler proof of decidability of a generalization of their logic. For ∃PTCTL logic, we obtain a similar bound on the complexity of the model-checking problem.

Emerson et al have also studied an extension of TCTL with parameters in [Emerson and Trefler 1999]. They make strong assumptions on the timed models used (their timed models are a very limited class of discrete-timed automata). They also impose strong restrictions on the use of parameters in the way they constrain the scope of the temporal operators: parameters can only be used to express upper bounds. The main interest of their work is to have identified a fragment of parametric TCTL that have a polynomial time model-checking problem for the restricted class of timed models that they consider.

Alur et al [Alur et al. 1999] have also studied the extension of real-time logics with parameters but in the context of linear time. In linear time, where the satisfiability problem can be reduced to the model-checking problem, the panorama changes. In particular, they show that, in linear time context, the use of equality leads to undecidability of the model-checking problem.

Alur et al [Alur et al. 1993] have studied the introduction of parameters in timed automata. They show that if only one clock is constrained by a parameter then the emptiness problem for the new class of automata is decidable, but when three clocks are used, the problem becomes undecidable. To solve the problem, they also rely on the use of Presburger arithmetic.

Other researchers have also proposed the use of Presburger arithmetic (or RA arithmetic) in the context of timed automata. In particular, Comon et al [Comon and Jurski 1999] have studied the use of the RA arithmetic to express the reachability relation of timed automata. Their work is more ambitious, since they do not only consider durations between regions but also durations between individual clock valuations. Nevertheless they do not consider properties expressible in temporal logics. Along the same line, it is shown in [Dang et al. 2000] that it is possible to extend the results of Comon et al to analyse the binary reachability relation of discrete pushdown automata.

Finally, we have solved in this article the more general problem of parametric synthesis : the set of valuations is effectively definable by a formula in PA (or RA). It follows that any question formulated in PA (or RA) about the set of valuations is decidable.

ACKNOWLEDGMENTS

The authors thank the three referees, especially one who has given many valuable remarks.

REFERENCES

Aceto, L. and Laroussinie, F. 2002. Is your model-checker on time ? on the complexity of model-checking for timed modal logics. *Journal of Logic and Algebraic Programming 52–53*, 7–51.

Alur, R., Courcoubetis, C., and Dill, D. 1990. Model checking for real-time systems. In *Annual IEEE Symposium on Logic in Computer Science, LICS'90*. IEEE Computer Society Press, 414–425.

Alur, R. and Dill, D. 1994. A theory of timed automata. *Theoretical Computer Science 126*, 183–235.

Alur, R., Etessami, K., Torre, S. L., and Peled, D. 1999. Parametric temporal logic for "model measuring". In *International Colloquium of Automata, languages and Programming, ICALP'99*. Lecture Notes in Computer Science, vol. 1644. Springer, 159–168.

Alur, R., Feder, T., and Henzinger, T. 1996. The benefits of relaxing punctuality. *Journal of the ACM 43,* 1, 116–146.

Alur, R., Henzinger, T., and Vardi, M. 1993. Parametric real-time reasoning. In *Annual Symposium on Theory of Computing, STOC'93*. ACM Press, 592–601.

Champarnaud, J.-M. 2001. Subset construction complexity for homogeneous automata, position automata and zpc-structures. *Theoret. Comput. Sci. 267*, 17–34.

Clarke, E., Emerson, E., and Sistla, A. 1986. Automatic verification of finite-state concurrent systems using temporal-logic specifications. *ACM Transactions on Programming Languages and Systems 8,* 2, 244–263.

Comon, H. and Jurski, Y. 1999. Timed automata and the theory of real numbers. In *International Conference on Concurrency Theory, CONCUR'99*. Lecture Notes in Computer Science, vol. 1664. Springer, 242–257.

Dang, Z., Ibarra, O. H., Bultan, T., Kemmerer, R. A., and Su, J. 2000. Binary reachability analysis of discrete pushdown timed automata. In *International Conference on Computer Aided Verification, CAV'00*. Lecture Notes in Computer Science, vol. 1855. Springer, 69–84.

Eilenberg, S. 1974. *Automata, Languages and Machines*. Vol. A. Academic Press.

Emerson, E. A. and Trefler, R. J. 1999. Parametric quantitative temporal reasoning. In *14th Annual IEEE Symposium on Logic in Computer Science, LICS'99*. IEEE Computer Society. 336–343.

Ferrante, J. and Rackoff, C. 1975. A decision procedure for the first-order theory of real addition with order. *SIAM J. Computing 4*, 69–76.

Fischer, M. J. and Rabin, M. O. 1974. Super-exponential complexity of Presburger arithmetic. In *Symposium in Applied Mathematics, SIAM-AMS*. Vol. VII. Amer. Math. Soc., 27–41.

Henzinger, T. 1998. It's about time: Real-time logics reviewed. In *International Conference on Concurrency Theory, CONCUR'98*. Lecture Notes in Computer Science, vol. 1466. Springer, 439–454.

Lewis, H. and Papadimitriou, C. 1998. *Elements of the theory of computation*. Prentice Hall.

Oppen, D. 1978. A $2^{2^{2^{pn}}}$ upper bound on the complexity of Presburger arithmetic. *Journal of Computer and System Sciences 16,* 3, 323–332.

Pnueli, A. 1977. The temporal logic of programs. In *Annual Symposium on Foundations of Computer Science, FOCS'77*. IEEE Computer Society Press, 46–57.

Wang, F. 1995. Timing behavior analysis for real-time systems. In *10th Annual IEEE Symposium on Logic in Computer Science, LICS'95*. 112–122.

Wang, F. and Hsiung, P.-A. 1997. Parametric analysis of computer systems. In *International Conference on Algebraic Methodology and Software Technology, AMAST'97*. Lecture Notes in Computer Science, vol. 1349. Springer, 539–553.

Weispfenning, V. 1999. Mixed real-integer linear quantifier elimination. In *International Symposium on Symbolic and Algebraic Computation, ISSAC'99*. ACM, 129–136.

Received May 2004; revised September 2006; accepted October 2006