Real-Time Logics: Fictitious Clock as an Abstraction of Dense Time

Jean-François Raskin and Pierre-Yves Schobbens
{jfr,pys}@info.fundp.ac.be

Computer Science Institute, University of Namur
Rue Grandgagnage 21, 5000 Namur, Belgium
Tel: +32 81 72 4990 Fax: +32 81 72 4967

Abstract. In this paper we study two possible semantics for the real-time logic MTL (Metric Temporal Logic). In the first semantics, called dense time semantics, time is modeled by the positive real numbers. In the second one, called fictitious clock semantics, real-time information is delivered by a global fictitious clock. We show that the fictitious clock semantics can be viewed as an abstraction of the dense time semantics. This abstraction relation is formalized by a parametric conservative connection. This formalization can be used to partially decide undecidable problems in the dense time semantics by reasoning on the fictitious clock semantics.

1 Introduction

It is now widely recognized that the use of formal methods is very useful (and often necessary) for developing correct concurrent and reactive systems. This observation is still clearer when dealing with real-time and hybrid systems. One of the favorite formalisms to specify and verify concurrent systems are temporal logics. Temporal logics [10] are modal logics that enable the expression of properties about a.o. the ordering of events in executions of concurrent programs [14]. These logics are more and more used as tools for the verification of concurrent finite state programs [11, 6].

The properties that can be expressed in temporal logics are *qualitative* constraints about the ordering of events; *quantitative* timing constraints cannot be expressed. Logics that are able to express quantitative requirements are called *real-time logics* [5, 4]. Real-time logics have received a lot of attention from the research community since the early 90s [13, 4, 5, 2, 3]. The formulae of a linear real-time logic are evaluated in timed sequences of states. There are two main ways to introduce real-time information in sequences of states, giving two different semantics for real-time:

- *dense time semantics*: in this framework an interval of the positive real numbers is associated to each state of the sequence;
- *fictitious clock semantics*: a global fictitious clock generates special events, called ticks, at a fixed rate. The granularity of the real-time information depends upon the rate of the clock.

Intuitively the fictitious clock semantics is a kind of abstraction of the dense time semantics: roughly, the dense real-time information is rounded by the clock rate. This paper is devoted to the study and the formalization of this relation between the two semantics.

An appropriate mathematical framework to relate semantic domains is abstract interpretation [7, 8]. In abstract interpretation an *abstract domain* is related to a *concrete domain* by a Galois connection [9]. Properties of Galois connections are used to conduct analysis in the concrete domain by computing in the abstract domain. To formalize the relation between the dense time and the fictitious clock semantics, we use a variant of a Galois connection. By similar principles we show that it is possible to partially decide the model-checking and satisfiability problems in dense time semantics by computing in the fictitious clock semantics. Dense time logic is an adequate formalism at the specification stage, while fictitious clock logic deals easily with implementations. The formalization of the relation between the two semantics is thus also useful to ensure a proper transition.

The rest of this paper is organized as follows: section 2 introduces the two semantics of real-time and their respective logics: $\mathsf{MTL}_{\mathsf{DT}}$ for the dense time and $\mathsf{MTL}_{\mathsf{FC}}$ for the fictitious clock. Section 3 develops some definitions of the theory of abstract interpretation and applies it to formalize the relation between the two semantics. Section 4 shows how to partially decide whether a given dense time formula f is satisfiable by deciding the satisfiability of a related fictitious clock formulae $\downarrow^{\delta} f$ or $\uparrow^{\delta} f$. Section 5 extends the approach to partially decide the model-checking in the dense time semantics. Due to the lack of space, the proofs are not presented in this paper, the interested reader can find them in [16].

2 The two semantics and their logics

2.1 The dense time semantics

In the dense time semantics, time is modeled by the real numbers [1]. State evolves with time: we represent this by a function from instants, i.e. real numbers, to states. As we are interested in discrete event systems, i.e. systems that evolve from state to state by non continuous changes, we can pair each state s of the system with the interval of time during which the system remains in this state s. A model of a system execution is an infinite sequence of couples (s_i, I_i) where s_i is a description of the system state and I_i is the interval of real time during which the system remains in the i^{th} state of the system. It is clear that this numbering is only conventional, hence repetition (stuttering) of states is irrelevant.

Notation. Let I be an interval, i.e. a non-empty convex subset of the positive real numbers $\mathbb{R}^+$: $l(I)$, respectively $r(I)$, denotes the left, respectively the right, end bound of the interval I.

[1] Note that the rational numbers can also be used; the property needed is the denseness of the domain, i.e. between two time points there is always a third point.

Definition 1. $I_0^m, I_1^m, \ldots, I_n^m, \ldots$ is a **sequence of intervals** that partitions $\mathbb{R}^+$ iff:

1. $\forall i \geq 0 \cdot r(I_i) = l(I_{i+1})$ and $I_i \bigcap I_{i+1} = \emptyset$;
2. $\bigcup_i I_i = \mathbb{R}^+$.

The second point of the definition ensures that the time has no bound. It excludes executions where the system executes an infinite number of changes in a finite amount of time. This is called "non-Zenoness" property [1].

Definition 2. A **dense time model** m is an infinite timed sequence of states

$$m = (s_0^m, I_0^m), (s_1^m, I_1^m), \ldots, (s_n^m, I_n^m), \ldots$$

where s_i^m is a subset of the propositions ($\mathcal{P}$), i.e. $s_i^m \subseteq \mathcal{P}$, that are true in the i^{th} state of the model m and I_i^m the interval of real time during which the model m stays in the i^{th} state. Furthermore, $I_0^m, I_1^m, \ldots, I_n^m, \ldots$ forms a sequence of intervals that partition $\mathbb{R}^+$. Let us note that a dense time model can also be viewed as a function from $\mathbb{R}^+$ to $2^{\mathcal{P}}$.

We have defined the models of the dense time semantics. We can now define the logic $\mathsf{MTL}_{\mathsf{DT}}$ [5, 13].

Definition 3 $\mathsf{MTL}_{\mathsf{DT}}$ **syntax.** A formula of $\mathsf{MTL}_{\mathsf{DT}}$ is composed of propositional symbols $p, p_1, p_2, \ldots, q, \ldots$, the usual boolean connectives $\vee$ and $\neg$, the qualitative temporal operator "Until" ($\mathcal{U}$) and the quantitative temporal operators "bounded Until" ($\mathcal{U}_{\sim c}$). A well-formed formula of $\mathsf{MTL}_{\mathsf{DT}}$ satisfies the following syntactical rule:

$$Formula_{\mathsf{DT}} ::= p \mid f_1 \vee f_2 \mid \neg f | f_1 \mathcal{U} f_2 \mid f_1 \mathcal{U}_{\sim c} f_2$$
$$\text{Where } \sim \in \{<, \leq, =, \geq, >\}, p \in \mathcal{P},$$
$$f, f_1, f_2 \text{ are well-formed formulae of } \mathsf{MTL}_{\mathsf{DT}} \text{ and } c \in \mathbb{Q}^+$$

To define the semantics of the $\mathsf{MTL}_{\mathsf{DT}}$ logic elegantly, we define the notion of ${}^f Fine$ model, adapted from [1]. In the following definitions, we note $m \models_{\mathsf{DT}} f$ to express that m is a model of the SC formula f, $(m, i) \models_{\mathsf{DT}} f$ to express that f is verified in all points of interval I_i^m, $(m, i, t) \models_{\mathsf{DT}} f$ to express that f is verified at time $t \in I_i^m$ of m.

Definition 4 ^{f}**Fine models.** For a $\mathsf{MTL}_{\mathsf{DT}}$ formula f, a timed state sequence $m = (s, I)$ is called ${}^f Fine$ iff for all $i \geq 0$, for all sub-formulae f_1 of f, for all $t, t' \in I_i^m : (m, i, t) \models_{\mathsf{DT}} f_1$ iff $(m, i, t') \models_{\mathsf{DT}} f_1$.

It is important to note that every model m can be refined in a ${}^f Fine$ model. It can be shown that a $\mathsf{MTL}_{\mathsf{DT}}$ formula is satisfiable iff it has a ${}^f Fine$ model. The model m in the following definition, without loss of generality, is considered to be ${}^{f_1} Fine$ and ${}^{f_2} Fine$[2]. The semantics that we give to $\mathsf{MTL}_{\mathsf{DT}}$ presents some

[2] Let us note that definition 4 and 5 are not cyclic since every timed sequence of state is propositions-Fine.

subtle differences from the usual semantics as given in [5]. Our semantics of the $\mathcal{U}$ operator is invariant to the fact that intervals are open or closed, see [16] for details. Here is our semantics:

Definition 5 $\mathsf{MTL}_{\mathsf{DT}}$ **semantics.** A $\mathsf{MTL}_{\mathsf{DT}}$ formula f holds in a dense time model m at real-time $t \in I_i^m$, noted $(m,i,t) \models_{\mathsf{DT}} f$, iff (by recursion) :

- $(m,i,t) \models_{\mathsf{DT}} p$ iff $p \in s_i^m$ for $p \in \mathcal{P}$;
- $(m,i,t) \models_{\mathsf{DT}} f_1 \vee f_2$ iff $(m,i,t) \models_{\mathsf{DT}} f_1$ or $(m,i,t) \models_{\mathsf{DT}} f_2$;
- $(m,i,t) \models_{\mathsf{DT}} \neg f$ iff $(m,i,t) \not\models_{\mathsf{DT}} f$;
- $(m,i,t) \models_{\mathsf{DT}} f_1 \mathcal{U} f_2$ iff $\exists j \geq i$ such that
  1. $(m,j) \models_{\mathsf{DT}} f_2$
  2. $\forall k : i \leq k < j : (m,k) \models_{\mathsf{DT}} f_1$
- $(m,i,t) \models_{\mathsf{DT}} f_1 \mathcal{U}_{\sim c} f_2$, with $\sim \in \{<, \leq\}$, iff $\exists j \geq i$ such that
  1. $(m,j) \models_{\mathsf{DT}} f_2$
  2. $\forall k : i \leq k < j : (m,k) \models_{\mathsf{DT}} f_1$
  3. $\exists t' \in I_j^m : t' - t \sim c$;
- $(m,i,t) \models_{\mathsf{DT}} f_1 \mathcal{U}_{=c} f_2$, iff $\exists j \geq i$ such that
  1. $(m,j) \models_{\mathsf{DT}} f_2$
  2. $\forall k : i \leq k < j : (m,k) \models_{\mathsf{DT}} f_1$
  3. either :
     - $\exists t' \in I_j^m : t' - t = c$ and $(m,j) \models_{\mathsf{DT}} f_1$;
     - **or** $l(I_j^m) - t = c$ and $l(I_j^m) \in I_j^m$.
- $(m,i,t) \models_{\mathsf{DT}} f_1 \mathcal{U}_{\sim c} f_2$, with $\sim \in \{>, \geq\}$, iff $\exists j \geq i$ such that
  1. $(m,j) \models_{\mathsf{DT}} f_2$
  2. $\forall k : i \leq k < j : (m,k) \models_{\mathsf{DT}} f_1$
  3. either :
     - $\exists t' \in I_j^m : t' - t \sim c$ and $(m,j) \models_{\mathsf{DT}} f_1$;
     - **or** $\forall t' \in I_j^m : t' - t \sim c$.

A formula f holds in a model m, noted $m \models_{\mathsf{DT}} f$ iff $(m,0,0) \models_{\mathsf{DT}} f$; this is the *anchored interpretation* [1].

As usual we can define abbreviations: $f_1 \wedge f_2 \equiv \neg(\neg f_1 \vee \neg f_2)$, $\top \equiv \neg f \vee f$, $\bot \equiv \neg\top$, $\Diamond f \equiv \top \mathcal{U} f$, $\Box f \equiv \neg\Diamond\neg f$, $\Diamond_{\sim c} f \equiv \top \mathcal{U}_{\sim c} f$, $\Box_{\sim c} f \equiv \neg\Diamond_{\sim c}\neg f$.

Example 1 Some $\mathsf{MTL}_{\mathsf{DT}}$ *formulae.* In the intuitive readings below, we consider that p and q are state formulae and that the time unit is seconds :

- $\Box(p \rightarrow \Diamond_{\leq 2} q)$. A p state is always followed by a q state within two seconds (bounded response time).
- $\Box(\neg\Box_{\leq 2} q)$. q is never true during two seconds consecutively.
- $\Box(p \rightarrow \Diamond_{=3} q)$. A p state is always followed by a q state exactly 3 seconds later (exact response time).

We will now study "stuttering". It allows a simplification of the formalization of the relation between the dense time and the fictitious clock semantics. A step is a **stuttering step** if it links two successive identical states. Two models are stuttering equivalent, noted $\mathsf{Stutt}^*_{\mathsf{DT}}(m_1, m_2)$, if m_1 can be obtained from m_2 by deleting and adding stuttering steps. $\mathsf{Stutt}^*_{\mathsf{DT}}$ is an equivalence relation and defines classes of equivalent models.

Theorem 6. $\mathsf{MTL}_{\mathsf{DT}}$ *is invariant to stuttering:*

$$\mathsf{Stutt}^*_{\mathsf{DT}}(m_1, m_2) \Rightarrow ((m_1, 0) \models_{\mathsf{DT}} f \Leftrightarrow (m_2, 0) \models_{\mathsf{DT}} f)$$

$\mathsf{MTL}_{\mathsf{DT}}$ is undecidable [4].

2.2 The fictitious clock semantics

Fictitious clock models are infinite sequences of states, where each state has a time-stamp which is a natural number. A global clock generates special periodic events called ticks. Those ticks indicate that time has increased of one time unit.

Example 2. To illustrate the notion of fictitious clock model, let consider the following sequence of states:

$$\cdots \rightarrow \{p\}_3 \rightarrow \{p,q\}_3 \rightarrow^{\mathsf{tick}} \{p,q\}_4 \rightarrow \{p\}_4 \rightarrow^{\mathsf{tick}} \{p,q\}_5 \rightarrow \cdots$$

In this example the first two states are labeled by the time-stamp 3. When the system is in the second, a tick takes place, time has increased of 1 time unit i.e. the time-stamp of the third state is 4.

We avoid the introduction of events in the models for simplicity and adopt the following convention: if the time stamp of two adjacent states are different, then the special proposition tick is true in the first state. With our convention, the sequence of example 2 is represented by:

$$\cdots \rightarrow \{p\} \rightarrow \{p,q,\mathsf{tick}\} \rightarrow \{p,q\} \rightarrow \{p,\mathsf{tick}\} \rightarrow \{p,q\} \rightarrow \cdots$$

Definition 7 Fictitious-clock model. A fictitious clock model h is a infinite sequence of states $h = h_0, h_1, \ldots, h_n, \ldots$ where h_i is the subset of the propositions that are true in the i^{th} state of h and contains the special proposition tick if the global clock produces the special event tick just after the i^{th} state. Furthermore there are infinitely many h_i such that $\mathsf{tick} \in h_i$. (non-Zenoness).

Definition 8. The **time difference function** between the i^{th} state and the j^{th} state of a fictitious clock model h is:$\mathsf{TD}(h,i,j) = \sharp\{k \mid i \leq k < j : \mathsf{tick} \in h_k\}$.

The syntax of $\mathsf{MTL}_{\mathsf{FC}}$ is as for $\mathsf{MTL}_{\mathsf{DT}}$. Nevertheless, to show the intended meaning, we use in $\mathsf{MTL}_{\mathsf{FC}}$ temporal operators decorated with $\sim$: $\widetilde{\mathcal{U}}, \widetilde{\mathcal{U}}_{\sim c}, \widetilde{\Box}, \widetilde{\Diamond}$.

Definition 9 $\mathsf{MTL}_{\mathsf{FC}}$ **semantics.** A $\mathsf{MTL}_{\mathsf{FC}}$ formula f holds in a fictitious-clock model h at state $i \in \mathbb{N}$, noted $(h,i) \models_{\mathsf{FC}} f$, iff (by recursion) :

- $(h,i) \models_{\mathsf{FC}} p$ iff $p \in h_i$ for $p \in \mathcal{P}$;
- $(h,i) \models_{\mathsf{FC}} f_1 \vee f_2$ iff $(h,i) \models_{\mathsf{FC}} f_1$ or $(h,i) \models_{\mathsf{FC}} f_2$;
- $(h,i) \models_{\mathsf{FC}} \neg f$ iff $(h,i) \not\models_{\mathsf{FC}} f$;
- $(h,i) \models_{\mathsf{FC}} f_1 \widetilde{\mathcal{U}} f_2$ iff $\exists j \geq i$ such that:
  1. $(h,j) \models_{\mathsf{FC}} f_2$
  2. $\forall k \cdot i \leq k < j : (h,k) \models_{\mathsf{FC}} f_1$;
- $(h,i) \models_{\mathsf{FC}} f_1 \widetilde{\mathcal{U}}_{\sim c} f_2$ iff $\exists j \geq i$ such that:
  1. $(h,j) \models_{\mathsf{FC}} f_2$
  2. $\forall k \cdot i \leq k < j : (h,k) \models_{\mathsf{FC}} f_1$
  3. $\mathsf{TD}(h,i,j) \sim c$;

A formula f holds in a model h, noted $h \models_{\mathsf{FC}} f$, iff $(h,0) \models_{\mathsf{FC}} f$; this is the "anchored interpretation".

Definition 10. A step h_i, h_{i+1} in a fictitious clock model h is a **stuttering step** iff $\mathsf{tick} \notin h_i$ and $h_i = h_{i+1} \setminus \{\mathsf{tick}\}$.

Two fictitious clock models are stuttering equivalent, noted $\mathsf{Stutt}^*_{\mathsf{FC}}(h^1, h^2)$, if they only differ by stuttering steps.

Theorem 11. $\mathsf{MTL}_{\mathsf{FC}}$ *is invariant to stuttering:*

$$\mathsf{Stutt}^*_{\mathsf{FC}}(h^1, h^2) \Rightarrow ((h^1, 0) \models_{\mathsf{FC}} f \Leftrightarrow (h^2, 0) \models_{\mathsf{FC}} f)$$

Example 3 Some $\mathsf{MTL}_{\mathsf{FC}}$ *formulae.* The formulae of example 1 have now a different meaning:

- $\widetilde{\Box}(p \rightarrow \widetilde{\Diamond}_{\leq 2} q)$. A p state is always followed by a q state before the clock ticks 3 times. Thus this constraint is weaker than the corresponding dense time constraint.
- $\widetilde{\Box}(\neg \widetilde{\Box}_{\leq 2} q)$. At every instant, q will become false before the clock ticks 3 times.
- $\widetilde{\Box}(p \rightarrow \widetilde{\Diamond}_{=3} q)$. A p state is always followed by a q state between the 3^{rd} and the 4^{th} tick of the clock following the p state.

The satisfiability and model-checking problems of $\mathsf{MTL}_{\mathsf{FC}}$ are decidable.

3 Fictitious clock as an abstraction of dense time

3.1 Introduction

A general and elegant framework to relate semantic domains is *abstract interpretation.* In abstract interpretation terminology, the values of a concrete domain $\mathbb{C}$ are related to the values of an abstract domain $\mathbb{A}$. This relation is formalized by two functions:

- the *abstraction function* α that associates to each concrete value c its best approximation $\alpha(c)$ in the abstract domain;

– the *concretization function* γ which maps each abstract value a to its concretization $\gamma(a)$ in the concrete domain.

Here the concrete values will be sets of dense time models and the abstract values will be sets of fictitious clock models. We concentrate now on the mathematical background underlying abstract interpretation. To be useful, the functions α and γ must have particular properties:

Definition 12 Conservative Connection. $R = \langle \mathbb{C}, \sqsubseteq_{\mathbb{C}}, \mathbb{A}, \sqsubseteq_{\mathbb{A}}, \alpha, \gamma \rangle$ is called a conservative connection if the following conditions are satisfied:

1. The pairs $\langle \mathbb{C}, \sqsubseteq_{\mathbb{C}} \rangle$ and $\langle \mathbb{A}, \sqsubseteq_{\mathbb{A}} \rangle$ are partially ordered sets;
2. The functions $\alpha : \mathbb{C} \to \mathbb{A}$ and $\gamma : \mathbb{A} \to \mathbb{C}$ are monotone;
3. For every $c \in \mathbb{C}$, $c \sqsubseteq_{\mathbb{C}} \gamma(\alpha(c))$;

In abstract interpretation *Galois connections* are usually used instead of conservative connections. Galois connections have the additional property that $\alpha(\gamma(a)) \sqsubseteq a$. We do not have this property in our framework because the abstract domain is imposed.

Theorem 13. *If* $R = \langle \mathbb{C}, \sqsubseteq_{\mathbb{C}}, \mathbb{A}, \sqsubseteq_{\mathbb{A}}, \alpha, \gamma \rangle$ *is a conservative connection then*

$$\alpha(c) \sqsubseteq_{\mathbb{A}} a \Rightarrow c \sqsubseteq_{\mathbb{C}} \gamma(a)$$

Usually, the concrete and abstract domains are sets of sets of concrete and abstract values. In this context, each element of $2^{\mathbb{C}}$ represents a concrete property and each element of $2^{\mathbb{A}}$ an abstract property.

3.2 Formalizing the abstraction relation

In this subsection we define formally the relation between the dense time and the fictitious clock semantics. We use an abstraction and a concretization function.

Definition 14. The **concrete domain** is the set of sets of dense time models $(2^{M_{\mathsf{DT}}})$; on this domain the relation $\sqsubseteq_{\mathsf{DT}}$ is defined as follows:

$$C_1 \sqsubseteq_{\mathsf{DT}} C_2 \Leftrightarrow \forall c_1 \in C_1 \cdot \exists c_2 \in C_2 : \mathsf{Stutt}^*_{\mathsf{DT}}(c_1, c_2)$$

Definition 15. the **abstract domain** is the set of sets of fictitious clock models $(2^{M_{\mathsf{FC}}})$; on this domain the relation $\sqsubseteq_{\mathsf{FC}}$ is defined as follows:

$$A_1 \sqsubseteq_{\mathsf{FC}} A_2 \Leftrightarrow \forall a_1 \in A_1 \cdot \exists a_2 \in A_2 : \mathsf{Stutt}^*_{\mathsf{FC}}(a_1, a_2)$$

Before going into details, let us precise the intuition behind this relation. In the fictitious clock semantics, a global clock delivers information through ticks. The rate of this clock, noted δ, is the real amount of time that separates two ticks. δ gives the granularity of time in fictitious clock models: the difference between two instants is rounded by the rate of the clock. Between two ticks we only know the ordering of states in the sequence and that the difference between two instants in that interval is inferior to the rate of the clock.

Example 4. Let us consider this fragment of a fictitious clock model

$$h = \cdots \rightarrow^{\mathsf{tick}} \{p\} \rightarrow \{q\} \rightarrow^{\mathsf{tick}} \cdots$$

between the two ticks we have no information about the real-time. We do not know how long the system remains in state $\{p\}$, we only know that it evolves to state $\{q\}$ before the following tick of the global clock.

Furthermore, when the system is in a given state, we do not know the exact amount of time it will take for the next tick to be produced. It is only known that this delay, noted d in the sequel, is strictly inferior to the rate of the clock ($0 \leq d < \delta$). To relate the two semantics, we will define the behavior of a fictitious clock in the dense time semantics. If we consider a model m at time t the behavior of a particular clock is determined by the delay that separates t of the following tick of the clock and the rate of the clock. The behavior of the clock is determined in the sense that with d and δ we can predict the exact time of all the following ticks. We can now define a function which, given a dense time model m (concrete model), returns the fictitious model that is its abstraction for a given global clock of delay d and rate δ. To define the function in an elegant way, we first put the model in "sliced" form:

Definition 16 $\mathsf{S}^{\mathsf{d},\delta}$**.** A dense time model m respects the slicing induced by the clock of parameters (d, δ), noted $\mathsf{S}^{\mathsf{d},\delta}(m)$ iff the following is verified:

$$\forall n \geq 0 \cdot \exists i \geq 0 : r(I_i^m) = n \cdot \delta + d$$

Theorem 17. $\forall m \in M_{\mathsf{DT}} \cdot \forall d \cdot 0 \leq d < \delta \cdot \exists m' \in M_{\mathsf{DT}} : \mathsf{S}^{\mathsf{d},\delta}(m') \wedge \mathsf{Stutt}^{*}_{\mathsf{DT}}(m, m')$

As stated by the theorem 17 every dense time model m has a stuttering equivalent model m' which respects the special form imposed by a particular clock (d, δ). We can now define the function that abstracts a given dense time model for a particular clock:

Definition 18 $\mathsf{TickIn}^{\mathsf{d},\delta}$**.** The predicate $\mathsf{TickIn}^{\mathsf{d},\delta}$ has the following definition:

$$\mathsf{TickIn}^{\mathsf{d},\delta}(I_i^m) \Leftrightarrow \exists n \geq 0 \cdot r(I_i^m) = n \cdot \delta + d$$

Definition 19 Function $\mathsf{f}^{\mathsf{d},\delta}_{\mathsf{DT}\rightarrow\mathsf{FC}}$**.** The function $\mathsf{f}^{\mathsf{d},\delta}_{\mathsf{DT}\rightarrow\mathsf{FC}} : M_{\mathsf{DT}} \rightarrowtail M_{\mathsf{FC}}$ is a partial function which maps a dense time model m to its abstraction, a fictitious model h. The partial function is defined as follows:

$$\begin{gathered}\forall m \in M_{\mathsf{DT}} \cdot \mathsf{S}^{\mathsf{d},\delta}(m) : \\ h = \mathsf{f}^{\mathsf{d},\delta}_{\mathsf{DT}\rightarrow\mathsf{FC}}(m) \\ \Leftrightarrow \\ \forall i \cdot i \geq 0 : (\mathsf{TickIn}^{\mathsf{d},\delta}(I_i^m) \rightarrow h_i = s_i^m \cup \{\mathsf{tick}\} \wedge \neg\mathsf{TickIn}^{\mathsf{d},\delta}(I_i^m) \rightarrow h_i = s_i^m)\end{gathered}$$

We can also define the relation which exists between a fictitious clock model h and its possible concretizations for a particular clock:

Definition 20 Relation $\mathsf{R}^{\mathsf{d},\delta}_{\mathsf{FC}\to\mathsf{DT}}$. The relation $\mathsf{R}^{\mathsf{d},\delta}_{\mathsf{FC}\to\mathsf{DT}} : M_{\mathsf{FC}} \times M_{\mathsf{DT}}$ relates a fictitious clock model h to its corresponding dense time models for a given clock (d, δ):

$$\forall m \in M_{\mathsf{DT}} \cdot \mathsf{S}^{\mathsf{d},\delta}(m) \cdot \forall h \in M_{\mathsf{FC}} : \mathsf{R}^{\mathsf{d},\delta}_{\mathsf{FC}\to\mathsf{DT}}(h, m) \Leftrightarrow h = \mathsf{f}^{\mathsf{d},\delta}_{\mathsf{DT}\to\mathsf{FC}}(m)$$

We can now define the abstraction and concretization functions. Those functions are extensions of $\mathsf{f}^{\mathsf{d},\delta}_{\mathsf{DT}\to\mathsf{FC}}$ and $\mathsf{R}^{\mathsf{d},\delta}_{\mathsf{FC}\to\mathsf{DT}}$ to sets of models. They will be parameterized by the clock rate δ, but the parameter d of the clock will be considered as unknown to fit with the semantics of formulae, we only can state that $0 \leq d < \delta$:

Definition 21. The **abstraction function** α^{δ} is a function which maps a set of dense time models C to its best approximation A, a set of fictitious clock models, for a given clock rate δ. $a \in \alpha^{\delta}(C)$ iff

$$\exists c \in C \cdot \exists d \cdot 0 \leq d < \delta \cdot \exists c' \in M_{\mathsf{DT}} \cdot \mathsf{S}^{\mathsf{d},\delta}(c') \wedge \mathsf{Stutt}^{*}_{\mathsf{DT}}(c, c') \wedge a = \mathsf{f}^{\mathsf{d},\delta}_{\mathsf{DT}\to\mathsf{FC}}(c')$$

Definition 22. The **concretization function** γ^{δ} is a function which maps a set of fictitious clock models A to its corresponding set of dense time models C for a given clock rate δ. More formally : $\gamma^{\delta} : 2^{M_{\mathsf{FC}}} \to 2^{M_{\mathsf{DT}}}$ and

$$c \in \gamma^{\delta}(A) \Leftrightarrow \exists a \in A \cdot \exists d \cdot 0 \leq d < \delta : \mathsf{R}^{\mathsf{d},\delta}_{\mathsf{FC}\to\mathsf{DT}}(a, c)$$

Theorem 23. *The concrete domain $2^{M_{\mathsf{DT}}}$ is pre-ordered by $\sqsubseteq_{\mathsf{DT}}$ and the abstract domain $2^{M_{\mathsf{FC}}}$ is pre-ordered by $\sqsubseteq_{\mathsf{FC}}$.*

$\langle 2^{M_{\mathsf{DT}}}, \sqsubseteq_{\mathsf{DT}} \rangle$ and $\langle 2^{M_{\mathsf{FC}}}, \sqsubseteq_{\mathsf{FC}} \rangle$ are pre-ordered sets, they are not partially ordered as it is required by the theorem 3 to obtain a *conservative connection.* However the preorders $\sqsubseteq_{\mathsf{DT}}$ and $\sqsubseteq_{\mathsf{FC}}$ are constructed with equivalence relations $\mathsf{Stutt}^{*}_{\mathsf{DT}}$ and $\mathsf{Stutt}^{*}_{\mathsf{FC}}$. Those two relations induce the definition of equivalent classes of models in the two semantics. Sets of those classes are partially ordered by the set inclusion relation. It can easily be shown that monotonicity of γ^{δ} and α^{δ} on those preorders induce monotonicity of γ^{δ} and α^{δ} on the partially ordered sets $(\mathbb{A}, \mathbb{C})$ of sets of classes induced by the relations $\mathsf{Stutt}^{*}_{\mathsf{DT}}$ and $\mathsf{Stutt}^{*}_{\mathsf{FC}}$. In the sequel, for simplicity, we will work directly on the preorder instead of on the partially ordered sets of classes. This is equivalent as the two logics are invariant to stuttering.

Theorem 24. *The structure $R = \langle 2^{M_{\mathsf{DT}}}, \sqsubseteq_{\mathsf{DT}}, 2^{M_{\mathsf{FC}}}, \sqsubseteq_{\mathsf{FC}}, \alpha^{\delta}, \gamma^{\delta} \rangle$ is a conservative connection.*

Corollary 25. *As the structure $R = \langle 2^{M_{\mathsf{DT}}}, \sqsubseteq_{\mathsf{DT}}, 2^{M_{\mathsf{FC}}}, \sqsubseteq_{\mathsf{FC}}, \alpha^{\delta}, \gamma^{\delta} \rangle$ is a conservative connection, we have that:*

$$\forall \delta > 0 \cdot \forall C \in 2^{M_{\mathsf{DT}}}, A \in 2^{M_{\mathsf{FC}}} :$$
$$\alpha^{\delta}(C) \sqsubseteq_{\mathsf{FC}} A \Rightarrow C \sqsubseteq_{\mathsf{DT}} \gamma^{\delta}(A)$$

3.3 Order of approximations

In abstract interpretation, a value a of an abstract domain A is an approximation of a value c of an concrete domain $\mathbb{C}$ iff $c \sqsubseteq_{\mathbb{C}} \gamma(a)$. A notion of order of approximation can be defined as follows:

Definition 26 Order of approximations. An abstract value a_1 is a better approximation for the value c that the abstract value a_2 in the connection induced by (α, γ) iff:

1. $c \sqsubseteq_{\mathbb{C}} \gamma(a_1)$ and $c \sqsubseteq_{\mathbb{C}} \gamma(a_2)$
2. $\gamma(a_1) \sqsubseteq_{\mathbb{C}} \gamma(a_2)$

This order can be directly applied to our parameterized connection $(\alpha^\delta, \gamma^\delta)$. More interesting is the notion of precision induced by the parameter δ of the connection. It seems intuitive to assert that if an abstraction is obtained by a more precise clock ($\delta_1 < \delta_2$) then this abstraction is more accurate. Formally, this is stated by $\delta_1 < \delta_2 \Rightarrow \gamma^{\delta_1}(\alpha^{\delta_1}(c)) \sqsubseteq_{\mathsf{DT}} \gamma^{\delta_2}(\alpha^{\delta_2}(c))$. But this assertion is not valid. Let us consider the following counter example:

Example 5. Let m be the following model:

$$m = (\{\}, [0,8))(\{p\}, [8,8])(\{\}, (8,16))(\{p\}, [16,16]) \cdots$$

In m, the proposition p is true at each multiple of 8. Let C be the singleton $\{m\}$. If we choose $\delta_1 = 7$ and $\delta_2 = 9$ ($\delta_1 < \delta_2$) we have that:

- $\alpha^{\delta_2}(C) \models_{\mathsf{FC}} \widetilde{\Box}\widetilde{\Diamond}_{\leq 1} p$. In fact, as in m there is a p at each multiple of 8, two consecutive p states are at most separated by 1 tick. This implies that $\gamma^{\delta_2}(\alpha^{\delta_2}(C)) \models_{\mathsf{DT}} \Box\Diamond_{<18} p$.
- $\alpha^{\delta_1}(C) \models_{\mathsf{FC}} \widetilde{\Box}\widetilde{\Diamond}_{\leq 2} p$. Furthermore, as in m p is true at each multiple of 8, there is a clock (d_1, δ) such that $h = \mathsf{f}^{\mathsf{d},\delta}_{\mathsf{DT}\to\mathsf{FC}}(m)$, with:

  $$h = \{\} \to^{\mathsf{tick}} \{\} \to^{\mathsf{tick}} \{p\} \to \{\} \to^{\mathsf{tick}} \{\} \to \{p\} \to \{\} \to^{\mathsf{tick}} \cdots$$

  this model is obtained by the clock $d = 0$ and $\delta = 7$. Now we can easily see that the model m'

  $$m' = (\{\}, [0,6))(\{\}, [6,13])(\{\}, (13,20))(\{p\}, [20,20])(\{\}, (20,27]) \cdots$$

  can be obtained by concretizing the model h with a clock $d = 6$, $\delta = 7$. Thus m' belongs to $\gamma^{\delta_1}(\alpha^{\delta_1}(C))$. Clearly there is no model m'' equivalent to m' modulo stuttering steps that belongs to $\gamma^{\delta_2}(\alpha^{\delta_2}(C))$, as in this set all models respect $\Box\Diamond_{<18} p$, which is not the case of m'. Thus, we have that $\gamma^{\delta_1}(\alpha^{\delta_1}(C)) \not\sqsubseteq_{\mathsf{DT}} \gamma^{\delta_2}(\alpha^{\delta_2}(C))$.

To refine the approximation we must strengthen the requirement. The following theorem establishes that we obtain a refinement if $\delta_2 = \frac{\delta_1}{n}$, for $n > 0$.

Theorem 27. *If we have that $\delta_1 = \frac{\delta_2}{n}$, with $n \in \mathbb{N}^+$ then we have that:*

$$\gamma^{\delta_1}(\alpha^{\delta_1}(C)) \sqsubseteq_{\mathsf{DT}} \gamma^{\delta_2}(\alpha^{\delta_2}(C))$$

4 The satisfiability problem

In this section we apply the development of previous sections. We show that it is possible to reason on the satisfiability of $\mathsf{MTL}_{\mathsf{DT}}$ formulae by deciding the satisfiability of related $\mathsf{MTL}_{\mathsf{FC}}$ formulae. The satisfiability problem of a formula f is to decide whether there exists a model that satisfies f. The satisfiability problem is undecidable for $\mathsf{MTL}_{\mathsf{DT}}$ but decidable for $\mathsf{MTL}_{\mathsf{FC}}$ [4]. As we have defined a formal link between the two semantics, we can use this link to infer satisfiability of $\mathsf{MTL}_{\mathsf{DT}}$ formulae by computing satisfiability of related $\mathsf{MTL}_{\mathsf{FC}}$ formulae:

Theorem 28. *Let f_1 be an $\mathsf{MTL}_{\mathsf{DT}}$ formula and f_2 an $\mathsf{MTL}_{\mathsf{FC}}$ formula:*

1. *if $\gamma^\delta(Mod_{\mathsf{FC}}(f_2)) \sqsubseteq_{\mathsf{DT}} Mod_{\mathsf{DT}}(f_1)$ [3] and $Mod_{\mathsf{FC}}(f_2) \neq \emptyset$ then $Mod_{\mathsf{DT}}(f_1) \neq \emptyset$ and f_1 is satisfiable.*
2. *if $\alpha^\delta(Mod_{\mathsf{DT}}(f_1)) \sqsubseteq_{\mathsf{FC}} Mod_{FC}(f_2)$ and $Mod_{\mathsf{FC}}(f_2) = \emptyset$ then $Mod_{\mathsf{DT}}(f_1) = \emptyset$ and f_1 is not satisfiable.*

The theorem 28 gives us the basis for a partial automatic treatment of the satisfiability problem in $\mathsf{MTL}_{\mathsf{DT}}$, provided we define which formulae of $\mathsf{MTL}_{\mathsf{FC}}$ must be used for a given $\mathsf{MTL}_{\mathsf{DT}}$ formulae. In the point 1 of theorem 28 the formulae f_2 is a kind of under-approximation of f_1. On the other hand, in the point 2, f_2 is an over-approximation of f_1. We define now two constructive functions: one noted $\uparrow^\delta$, which maps a $\mathsf{MTL}_{\mathsf{DT}}$ formula to its *over-approximation* in $\mathsf{MTL}_{\mathsf{FC}}$, the other noted $\downarrow^\delta$, which maps a $\mathsf{MTL}_{\mathsf{DT}}$ formula to its *under-approximation.*

Definition 29. The **over-approximation** $\uparrow^\delta$ and the **under-approximation** $\downarrow^\delta$ are defined inductively as follows:

- $\uparrow^\delta (p) = p$ if $p \in \mathcal{P}$;
- $\uparrow^\delta (f_1 \vee f_2) = \uparrow^\delta f_1 \vee \uparrow^\delta f_2$
- $\uparrow^\delta (\neg f_1) = \neg \downarrow^\delta f_1$
- $\uparrow^\delta (f_1 \mathcal{U} f_2) = \uparrow^\delta f_1 \widetilde{\mathcal{U}} \uparrow^\delta f_2$
- $\uparrow^\delta (f_1 \mathcal{U}_{\leq c} f_2) = \uparrow^\delta f_1 \widetilde{\mathcal{U}}_{\leq \lceil \frac{c}{\delta} \rceil} \uparrow^\delta f_2$
- $\uparrow^\delta (f_1 \mathcal{U}_{< c} f_2) = \uparrow^\delta f_1 \widetilde{\mathcal{U}}_{\leq \lceil \frac{c}{\delta} \rceil} \uparrow^\delta f_2$
- $\uparrow^\delta (f_1 \mathcal{U}_{= c} f_2) = \uparrow^\delta f_1 \widetilde{\mathcal{U}}_{= \lceil \frac{c}{\delta} \rceil} \uparrow^\delta f_2 \vee \uparrow^\delta f_1 \widetilde{\mathcal{U}}_{= \lfloor \frac{c}{\delta} \rfloor} \uparrow^\delta f_2$
- $\uparrow^\delta (f_1 \mathcal{U}_{\geq c} f_2) = \uparrow^\delta f_1 \widetilde{\mathcal{U}}_{\geq \lfloor \frac{c}{\delta} \rfloor} \uparrow^\delta f_2$
- $\uparrow^\delta (f_1 \mathcal{U}_{> c} f_2) = \uparrow^\delta f_1 \widetilde{\mathcal{U}}_{\geq \lfloor \frac{c}{\delta} \rfloor} \uparrow^\delta f_2$

- $\downarrow^\delta (p) = p$ if $p \in \mathcal{P}$;
- $\downarrow^\delta (f_1 \vee f_2) = \downarrow^\delta f_1 \vee \downarrow^\delta f_2$
- $\downarrow^\delta (\neg f_1) = \neg \uparrow^\delta f_1$
- $\downarrow^\delta (f_1 \mathcal{U} f_2) = \downarrow^\delta f_1 \widetilde{\mathcal{U}} \downarrow^\delta f_2$
- $\downarrow^\delta (f_1 \mathcal{U}_{\leq c} f_2) = \downarrow^\delta f_1 \widetilde{\mathcal{U}}_{\leq \lfloor \frac{c}{\delta} \rfloor - 1} \downarrow^\delta f_2$

[3] $Mod_{\mathsf{FC}}(f) = \{m \mid (m, 0) \models_{\mathsf{FC}} f\}$ and $Mod_{\mathsf{DT}}(f) = \{m \mid (m, 0, 0) \models_{\mathsf{DT}} f\}$.

- $\downarrow^\delta (f_1 \mathcal{U}_{<c} f_2) = \downarrow^\delta f_1 \widetilde{\mathcal{U}}_{\leq \lfloor \frac{c}{\delta} \rfloor - 1} \downarrow^\delta f_2$
- $\downarrow^\delta (f_1 \mathcal{U}_{=c} f_2) = \widetilde{\Box}_{\leq \lceil \frac{c}{\delta} \rceil} (\downarrow^\delta f_1) \wedge \widetilde{\Box}_{= \lceil \frac{c}{\delta} \rceil} (\downarrow^\delta f_2) \wedge \widetilde{\Box}_{= \lfloor \frac{c}{\delta} \rfloor} (\downarrow^\delta f_2)$
- $\downarrow^\delta (f_1 \mathcal{U}_{\geq c} f_2) = \downarrow^\delta f_1 \widetilde{\mathcal{U}}_{\geq \lceil \frac{c}{\delta} \rceil + 1} \downarrow^\delta f_2$
- $\downarrow^\delta (f_1 \mathcal{U}_{>c} f_2) = \downarrow^\delta f_1 \widetilde{\mathcal{U}}_{\geq \lceil \frac{c}{\delta} \rceil + 1} \downarrow^\delta f_2$

Theorem 30. *For all dense time formula f, we have:*

- $\gamma^\delta(Mod_{\mathsf{FC}}(\downarrow^\delta f)) \sqsubseteq_{\mathsf{DT}} Mod_{\mathsf{DT}}(f)$
- $\alpha^\delta(Mod_{\mathsf{DT}}(f)) \sqsubseteq_{\mathsf{FC}} Mod_{FC}(\uparrow^\delta f)$

Corollary 31. *For all dense time formulae f, we have:*

- *if $\uparrow^\delta f$ is not satisfiable then f is also not satisfiable*
- *if $\downarrow^\delta f$ is satisfiable then f is also satisfiable*

4.1 An example

To illustrate the applicability of the theory developed so far, we treat here an non-trivial problem which is part of the CSMA/CD protocol. This protocol is used to share an unique communication medium among several computers. The principle of the protocol is illustrated by the following scenario:

Example 6 CSMA/CD. When a computer wants to send some message, it tests if the line is busy. If not, it begins to send; on the contrary, if the line is busy, the computer waits. A **collision** occurs when more than one computer are transmitting at the same time. The delay of propagation plays an important role in the protocol: If one station begins to send, the other stations see that the station is sending, not directly but at most α time units after, where α is the propagation delay. Consequently, a collision may occur between 0 and α time units after a computer has begun to send. The noise of the collision can also take α time units to reach the sending computer. A computer is only sure that no collision will occur after 2α time units. A sending which is interrupted by a collision is lost. The sender of such a message must know that the message was lost.

We now formalize a part of the protocol in $\mathsf{MTL}_{\mathsf{DT}}$. This set of axioms, noted S, describes the behavior of a computer taking part in the network:

1. $\neg Sending$
   Initially the computer is not sending.
2. $\Box(C \rightarrow \Diamond_{\leq 2\alpha} SeeC)$
   A collision is always perceived within 2α, 2 times the maximal propagation delay.
3. $\Box(SeeC \rightarrow \neg EndSend)$
   If a collision is detected then the sending is lost.
4. $\Box(SeeC \vee EndSend \rightarrow \neg Sending)$
   If a computer ends sending or detects a collision then it does not send anymore.
5. $\Box(BeginToSend \rightarrow Sending)$
   If a computer begins to send something, *Sending* becomes true.
6. $\Box(Sending \rightarrow Sending \mathcal{U}(EndToSend \vee SeeC))$
   If a computer is sending, it continues to send until it has sent the entire message or it has detected a collision.

7. $\Box(\neg Sending \rightarrow \neg Sending \mathcal{U} BeginToSend)$
   A computer evolves to state *Sending* only if it begins the sending of a message.
8. $\Box(BeginToSend \rightarrow \Box_{\leq \lambda} \neg EndToSend)$
   A sending lasts at least λ.
9. $\Box(\Box_{\leq \alpha}(Sending \wedge \neg C) \rightarrow \neg C \mathcal{U} EndSend)$
   If there is no collision after α, then no collision occur during this sending.

This set of axioms defines a solution for the following requirement R:

$$\Box\neg((Sending \wedge Sending \mathcal{U} EndToSend) \wedge C)$$

expressing that a computer cannot succeed in sending if a collision has occurred during this sending. This requires that the computer is always aware of the lost of one of its messages due to a collision. In the ethernet protocol the parameters are $\alpha = 25.6\mu s$ and $\lambda = 782\mu s$. To show that the solution proposed ensures R, we must prove that $S \rightarrow R$ is valid. Which is equivalent to show that $\neg(S \rightarrow R)$ is not satisfiable. There does not exist an algorithm to check this in $\mathsf{MTL}_{\mathsf{DT}}$ but by the corollary 31 if we can establish that $\uparrow^{\delta} \neg(S \rightarrow R)$ is not satisfiable for some δ then $\neg(S \rightarrow R)$ is also not satisfiable. We have tested the satisfiability of $\uparrow^{\delta} \neg(S \rightarrow R)$ with the implementation of [15]. For the parameters $\delta = 50\mu s$ and also for $\delta = 150\mu s$ the automatic procedure can establish the non-satisfiability of $\uparrow^{\delta} \neg(S \rightarrow R)$ and thus, by corollary 31, that $S \rightarrow R$ is valid in the dense time semantics. On the other hand, the validity of $S \rightarrow R$ can not be established with a clock rate $\delta = 500\mu s$: in this case, the granularity of time is not fine enough. Thus if the non-satisfiability or satisfiability of f can not be established by corollary 31 with clock rate δ_1, we can retry with a more precise clock rate $\delta_2 < \delta_1$.

5 The model-checking problem

The model-checking problem is to decide whether a program P satisfies a given property expressed in a temporal logic. Here we are interested in verifying that a concurrent program exhibiting real-time behaviors verifies some property expressed as a real-time logic formula. A real-time program P can be modeled by a timed automaton which defines a set of timed sequences of states: the models of its possible executions. As for the logic MTL, we can define the semantics of timed automata in the dense time framework or in the fictitious clock framework.

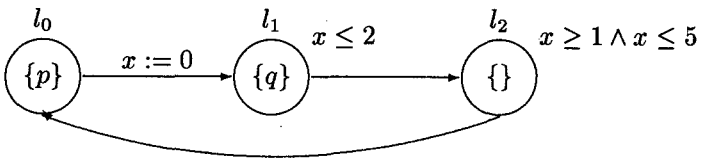

Fig. 1. Timed automaton A.

A timed automata is a finite state machine augmented with clocks and clock constraints. A clock can be reset simultaneously with any transition. For instance, in the timed automaton of figure 1 the clock x is reset each time the automaton evolves from location l_0 to location l_1. In the dense time semantics, at any time $t \in \mathbb{R}^+$ the value of a clock is the time elapsed since the last time it was reset. In the fictitious clock semantics, the value of the clock is the number of ticks since the clock was reset. In an timed automaton, locations are decorated by:

- *propositions*: a location l is labelled by the propositions that are true when the automaton stays in l. For instance, in location l_1 of the automaton of figure 1, the proposition q is true.
- *clock constraints*: those constraints impose real-time requirements on the behavior of the automaton. In our example, the automaton can stay in location l_2 only if it has crossed the edge from l_0 to l_1 at most 5 time units and at least 1 time unit earlier. In the fictitious clock semantics, the interpretation is different: the automaton can stay in location l_2 only if the clock has produced at least 1 tick and at most 5 ticks since the automaton has crossed the edge from l_0 to l_1 for the last time.

Having the intuition of what a timed automaton is, we can define it formally:

Definition 32 Clock constraints. For a set of clocks C, the set of timing constraints is defined inductively as follows: $\rho ::= x \sim c \mid \rho_1 \vee \rho_2 \mid \rho_1 \wedge \rho_2$, $\sim \in \{<, \leq, =, \geq, >\}$, ρ_1 and ρ_2 are well-formed timing constraints, c is a rational number. The set of constraints is noted $\Delta^{\mathbb{Q}^+}(C)$.

Definition 33. A **Timed Automata** A is a tuple $\langle L, L_0, C, E, \mathcal{L}_\mathcal{P}, \mathcal{L}_\mathcal{C}, \mathcal{F} \rangle$ where:

- L is a finite set of locations;
- $L_0 \subseteq L$ is the subset of initial locations;
- C is a finite set of clocks;
- $E \subseteq L \times 2^C \times L$ a set of edges. An edge (l_1, λ, l_2) represents a transition from location l_1 to location l_2, λ is the subset of clocks that are reset when crossing the edge;
- $\mathcal{L}_\mathcal{P} : L \to 2^\mathcal{P}$ is a labelling function which labels a location with the set of atomic propositions that are true in that location;
- $\mathcal{L}_\mathcal{C} : L \to \Delta^{\mathbb{Q}^+}$ is a labelling function which assigns to each location a constraint of $\Delta^{\mathbb{Q}^+}$ on the value of clocks that should be verified when staying in that location;
- $\mathcal{F}$ is a set of sets of accepting locations.

Timed automata also admit two semantics: in the dense time semantics clocks are of type *real* whereas in the fictitious clock semantics they are of type *natural*. The evolution of their value along time in a run of a timed automaton is formally

described by the two functions ${}^{\mathsf{DT}}\eta$ and ${}^{\mathsf{FC}}\eta$, one for each semantics. [4] But before defining them let us precise the notion of run:

Definition 34. A Dense Time Run is an infinite sequence

$$r = (l_0^r, I_0^r) \rightarrow^{\lambda_0} (l_1^r, I_1^r) \rightarrow^{\lambda_1} \cdots (l_n^r, I_n^r) \rightarrow^{\lambda_n} \cdots$$

- l_i are locations;
- $I_0 I_1 \cdots I_n \cdots$ is a sequence of intervals that partition $\mathbb{R}^+$;
- λ_i are sets of clocks to reset.

Definition 35. A Fictitious Clock Run is an infinite sequence

$$r = l_0 \rightarrow_{t_0}^{\lambda_0} l_1 \rightarrow_{t_1}^{\lambda_1} \cdots l_n \rightarrow_{t_n}^{\lambda_n} \cdots$$

- l_i are locations;
- t_i = tick if there is a tick during the transition from l_i to l_{i+1} otherwise $l_i = \neg$tick;
- λ_i are sets of clocks to be reset.

Definition 36 Clock Interpretation. The clock interpretation is defined as follows:

- in the **dense time semantics**: the clock interpretation in time $t \in I_i^r$, noted ${}^{\mathsf{DT}}\eta_t^r$, along a run r respects the following definition:

$$\forall x \in C :^{\mathsf{DT}} \eta_t^r(x) = \begin{cases} t - r(I_j^r) & \text{if } x \in \lambda_j \text{ and } \forall k \cdot j < k < i : x \notin \lambda_k (t \in I_i^r) \\ t & \text{if } \forall j : 0 \leq j < i : x \notin \lambda_j \end{cases}$$

- in the **fictitious clock semantics**: the clock interpretation in state r_i of a fictitious clock run r respects the following definition:

$$\forall x \in C :^{\mathsf{FC}} \eta_i^r(x) = \mathsf{TD}(r, j, i) \text{ where}$$
$$j = \begin{cases} 0 & \text{if } \forall k \cdot 0 \leq k < i : x \notin \lambda_k \\ l+1 & \text{if } x \in \lambda_l \wedge \forall k \cdot l < k < i : x \notin \lambda_k \end{cases}$$

Definition 37. A dense time run r is accepted by A iff it respects the following requirements:

- **Initiality**: $l_0^r \in L_0$;
- **Consecution**: $\forall i \geq 0 : (l_i^r, \lambda_i, l_{i+1}^r) \in E$ or $l_i^r = l_{i+1}^r$ and $\lambda_i = \emptyset$ (stuttering);
- **Timing constraints**: $\forall i \geq 0 \cdot \forall t \in I_i : {}^{\mathsf{DT}}\eta_t^r \models \mathcal{L}_C(l_i^r)$;
- **Acceptance**: for all $F_i \in \mathcal{F}$, there exists a location $l_i \in F_i$ which appears infinitely often along r (Büchi condition).

The timed sequence of states (dense time model) m corresponding to an run r is defined as follows:

[4] In the following we use $\mathsf{f}_{\mathsf{DT}\rightarrow\mathsf{FC}}^{\mathsf{d},\delta}$, $\mathsf{S}^{\mathsf{d},\delta}$, ... on runs instead of models, their extension to runs is obvious.

- **Interval**: $\forall i \geq 0 : I_i^m = I_i^r$;
- **Adequation**: $\forall i \geq 0 : s_i^m = \mathcal{L}_\mathcal{P}(l_i^r)$;

The set of dense time runs of A is noted $Run_{\mathsf{DT}}(A)$ and the set of corresponding dense time models $Exec_{\mathsf{DT}}(A)$.

Definition 38. A fictitious clock run r is accepted by A if it respects the following requirements:

- **Timing constraints**: $\forall i \geq 0 :^{\mathsf{FC}} \eta_i^r \models \mathcal{L}_\mathcal{C}(l_i)$;
- Initiality, consecution and acceptance are the same as in definition 37.

The fictitious clock model $h = h_0, h_1, \cdots, h_n, \cdots$ corresponding to an run r is defined as follows:

- **Timing**: $\forall i \geq 0 : \mathsf{tick} \in h_i \Leftrightarrow t_i = \mathsf{tick}$;
- **Adequation**: $\forall i \geq 0 : h_i \backslash \{\mathsf{tick}\} = \mathcal{L}_\mathcal{P}(l_i)$;

The set of fictitious clock runs of an automaton A is noted $Run_{\mathsf{FC}}(A)$ and the set of corresponding models $Exec_{\mathsf{FC}}(A)$.

As we already said, the model-checking problem is undecidable in the dense time semantics. But as for the satisfiability problem of $\mathsf{MTL}_{\mathsf{DT}}$, we claim that the dense time model-checking problem can often be decided by deciding a related problem in the fictitious clock semantics. The following theorem is the basis for a *partial* decision procedure for the model-checking problem in the dense time semantics:

Theorem 39. *The automaton A verifies the dense time property f if:*

1. $Exec_{\mathsf{DT}}(A) \sqsubseteq_{\mathsf{DT}} \gamma^\delta(Exec_{\mathsf{FC}}(\Uparrow^\delta A))$
2. $\gamma^\delta(Mod_{\mathsf{FC}}(\downarrow^\delta f)) \sqsubseteq_{\mathsf{DT}} Mod_{\mathsf{DT}}(f)$
3. $Exec_{\mathsf{FC}}(\Uparrow^\delta A) \sqsubseteq_{\mathsf{FC}} Mod_{\mathsf{FC}}(\downarrow^\delta f)$

This theorem is of practical interest because exact real-time information is often not necessary to prove interesting properties of real-time systems. It remains us to define formally the function $\Uparrow^\delta$ which must return a fictitious clock *over-approximation* of a given dense time automaton:

Definition 40. The function $\Uparrow^\delta$ maps a dense time automata A to its fictitious clock **over-approximation** A':

- $L = L', L_0 = L'_0, C = C', E = E', \mathcal{L}_\mathcal{P} = \mathcal{L}'_\mathcal{P}, \mathcal{F} = \mathcal{F}'$;
- For all the locations $l \in L$, we define the function $\nearrow^\delta$ as follows:
  - induction cases:
    - $\mathcal{L}_\mathcal{C}(l) = \rho_1 \vee \rho_2 \Rightarrow \mathcal{L}'_\mathcal{C}(l) = \nearrow^\delta \rho_1 \vee \nearrow^\delta \rho_2$;
    - $\mathcal{L}_\mathcal{C}(l) = \rho_1 \wedge \rho_2 \Rightarrow \mathcal{L}'_\mathcal{C}(l) = \nearrow^\delta \rho_1 \wedge \nearrow^\delta \rho_2$;
  - base cases:
    - $\nearrow^\delta (x \leq c) = (x \leq \lceil \frac{c}{\delta} \rceil)$

* $\nearrow^{\delta}(x < c) = (x \leq \lceil \frac{c}{\delta} \rceil)$
* $\nearrow^{\delta}(x = c) = (x = \lceil \frac{c}{\delta} \rceil \vee x = \lfloor \frac{c}{\delta} \rfloor)$
* $\nearrow^{\delta}(x \geq c) = (x \geq \lfloor \frac{c}{\delta} \rfloor)$
* $\nearrow^{\delta}(x > c) = (x \geq \lfloor \frac{c}{\delta} \rfloor)$

This function relaxes the constraints in the fictitious clock automaton.

Theorem 41. $Exec_{\mathsf{DT}}(A) \sqsubseteq_{\mathsf{DT}} \gamma^{\delta}(Exec_{\mathsf{FC}}(\Uparrow^{\delta} A))$

In [12] Henzinger et al have studied how to use digital techniques to decide problems in the dense time semantics. Their work is not easily comparable to our because of the rather different semantical models used in the two works. Nevertheless we can assert that we propose a more general approach while in [12] they propose more preservation for restricted properties, i.e. digitizable properties, see the paper for details.

6 Conclusion

The formalization presented in this paper clarifies the common intuition according to which the fictitious clock semantics can be seen as an abstraction of the dense time semantics. A conservative connection between the two semantics has been defined. We have presented a method to partially decide the logic $\mathsf{MTL}_{\mathsf{DT}}$. Another way to obtain an algorithmic approach to dense time is to define a decidable subset of $\mathsf{MTL}_{\mathsf{DT}}$, for instance: MITL [1]. The postulate on which our approach is based, which is also a postulate of abstract interpretation in general, is that the exact real-time information is seldom necessary to prove interesting properties of systems. On the other hand the approach of MITL is to restrict the logic but all the exact real-time information is preserved during verification. It would be interesting to compare pragmatically the applicability of the two approaches. To our knowledge there is no implementation of satisfiability and model-checking of MITL formulae. An important problem to implement efficiently decision procedures in dense time is the absence of a good data structure which would allow combining discrete information (system state) and continuous information (dense time). In our proposal this problem disappears and symbolic data structures as Sharing Trees or BDDs can be used. This has already given good results for a satisfiability checking procedure [15].

Some designs or implementations use a global clock to rule their real-time behavior. The fictitious clock semantics is well-suited to study those systems. On the other hand, the dense time semantics is natural in early phases of program development and should be preferred to fictitious clock in this context. If the two semantics are used in the development of a same program, it is important to have a formal link between the two semantics. Our parameterized connection is such a formal link.

References

1. R. Alur. *Techniques for Automatic Verification of Real-Time Systems.* PhD thesis, Stanford University, 1991.

2. R. Alur, C. Courcoubetis, and D. Dill. Model-checking for real-time systems. In *Proceedings of the 5th Symposium on Logic in Computer Science*, pages 414–425, Philadelphia, June 1990.
3. R. Alur and T.A. Henzinger. Logics and models of real time: a survey. In J.W. de Bakker, K. Huizing, W.-P. de Roever, and G. Rozenberg, editors, *Real Time: Theory in Practice*, Lecture Notes in Computer Science 600, pages 74–106. Springer-Verlag, 1992.
4. R. Alur and T.A. Henzinger. Real-time logics: complexity and expressiveness. *Information and Computation*, 104(1):35–77, 1993. Preliminary version appears in the Proc. of 5th LICS, 1990.
5. R. Alur and T.A. Henzinger. A really temporal logic. *Journal of the ACM*, 41(1):181–204, 1994. Preliminary version appears in Proc. 30th FOCS, 1989.
6. J.R. Burch, E.M. Clarke, K.L. McMillan, D.L. Dill, and L.J. Hwang. Symbolic model checking: 10^{20} states and beyond. In *Proceedings of the 5th Symposium on Logic in Computer Science*, pages 428–439, Philadelphia, June 1990.
7. P. Cousot and R. Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Conference Record of Fourth ACM Symposium on Programming Languages (POPL'77)*, pages 238–252, Los Angeles, California, January 1977.
8. P. Cousot and R. Cousot. Abstract interpretation frameworks. *Journal of Logic and Computation*, 2(4):511–547, 1992.
9. P. Cousot and R. Cousot. Comparison of the Galois Connection and Widening/Narrowing Approaches to Abstract Interpretation (Invited Paper). In M. Bruynooghe and M. Wirsing, editors, *Proceedings of the Fourth International Workshop on Programming Language Implementation and Logic Programming (PLILP'92)*, Lecture Notes in Computer Science, Leuven, August 1992. Springer-Verlag.
10. E.A. Emerson. *Handbook in Theoretical Computer Science, Formal Models and Semantics*, chapter Temporal and Modal Logic, pages 995–1072. Elsevier, 1990.
11. R. Gerth, D. Peled, M. Y. Vardi, and P. Wolper. Simple on-the-fly automatic verification of linear temporal logic. In *Proc. 15th Work. Protocol Specification, Testing, and Verification*, Warsaw, June 1995. North-Holland.
12. T.A. Henzinger, Z. Manna, and A. Pnueli. What good are digital clocks? In W. Kuich, editor, *ICALP 92: Automata, Languages, and Programming*, Lecture Notes in Computer Science 623, pages 545–558. Springer-Verlag, 1992.
13. Ron Koymans. *Specifying message passing and time-critical systems with temporal logic*. LNCS 651, Springer-Verlag, 1992.
14. A. Pnueli. The temporal logic of programs. In *Proc. 18th IEEE Symposium on Foundation of Computer Science*, pages 46–57, 1977.
15. J.-F. Raskin. Model-Generation of a Fictitious Clock Real-Time Logic: A Symbolic Decision Procedure Using Sharing-Trees. Research Paper RP-09-96, Computer Science Department, FUNDP, Namur (Belgium), March 1996.
16. J.-F. Raskin and P.-Y. Schobbens. Real-Time Logics: Fictitious Clock as an Abstraction of Dense Time. Research Paper RP-17-96, Computer Science Department, FUNDP, Namur (Belgium), September 1996.