Science of Computer Programming 77 (2012) 1212–1234

Contents lists available at SciVerse ScienceDirect

Science of Computer Programming

journal homepage: www.elsevier.com/locate/scico

Modal event-clock specifications for timed component-based design[☆]

Nathalie Bertrand [a], Axel Legay [a], Sophie Pinchinat [a], Jean-Baptiste Raclet [b,*]

[a] IRISA/INRIA Rennes Bretagne Atlantique, Campus Universitaire de Beaulieu, 35042 Rennes Cedex, France
[b] IRIT/CNRS, 118 Route de Narbonne, 31062 Toulouse, France

ARTICLE INFO

Article history:
Available online 3 March 2011

Keywords:
Component-based systems
Interface-based design
Timed modal specification
Conjunction
Product
Quotient

ABSTRACT

Modal specifications are classic, convenient, and expressive mathematical objects to represent interfaces of component-based systems. However, time is a crucial aspect of systems for practical applications, e.g. in the area of embedded systems. And yet, only few results exist on the design of timed component-based systems. In this paper, we propose a timed extension of modal specifications, together with fundamental operations (conjunction, product, and quotient) that enable reasoning in a compositional way about timed system. The specifications are given as modal event-clock automata, where clock resets are easy to handle. We develop an entire theory that promotes efficient incremental design techniques.

1. Introduction

Nowadays, systems are tremendously large and complex, resulting from the assembling of several components. These many components are in general designed by teams, working independently but with a common agreement on what the interface of each component should be. As a consequence, the development of mathematical foundations that allow one to reason at the abstract level of interfaces, in order to infer properties of the global implementation, and to design or to advisedly (re)use components, is a very active research area, known as *compositional reasoning* [1,2]. In a logical interpretation, interfaces are specifications and components that implement an interface are understood as models. Aiming at practical applications as the final goal, the software engineering point of view naturally leads to the following requirements for a good theory of interfaces.

1. *Satisfiability/Consistency and Satisfaction.* It should be decidable whether a specification admits a model, and whether a given component implements a given interface. Moreover, for the synthesis of components to be effective, satisfiable interfaces should always have finitely presentable models.
2. *Refinement and shared refinement. Refinement* of specifications [3,4] expresses inclusion of sets of models, and therefore allows us to compare interfaces. Related to this implication-like concept, the intersection, or *greatest lower bound*, is an optimal interface refining two given interfaces.
3. *Compositionality of the abstraction.* The interface theory should also provide combination operators on interfaces, reflecting the standard compositions of models by, *e.g.* parallel product.
4. *Quotient.* Last but not least, a quotienting operation, dual to composition is crucial to perform incremental design. Intuitively, the quotient enables us to describe a part of a global specification assuming another part is already realized by

[☆] This work was funded by the European project COMBEST, IST-STREP 215543.

[*] Corresponding author.
E-mail addresses: nathalie.bertrand@inria.fr (N. Bertrand), axel.legay@inria.fr (A. Legay), sophie.pinchinat@irisa.fr (S. Pinchinat), raclet@irit.fr, jean-baptiste.raclet@irit.fr (J.-B. Raclet).

0167-6423/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.scico.2011.01.007

some component. Together with the composition $\otimes$, the quotient operator $\oslash$ enjoys the following fundamental property at the component level:

$$\mathcal{C}_2 \models \mathcal{S} \oslash \mathcal{S}_1 \Leftrightarrow \forall \mathcal{C}_1 \, [\mathcal{C}_1 \models \mathcal{S}_1 \Rightarrow \mathcal{C}_1 \otimes \mathcal{C}_2 \models \mathcal{S}] \qquad (\star)$$

where $\mathcal{S}$, $\mathcal{S}_1$ are interfaces, $\mathcal{C}_i$ components, and $\models$ is the satisfaction relation.

Building good interface theories is the subject of intensive studies which have led to theories based on models such as interface automata [5,6], modal automata or specifications [7–11], and their respective timed extension [12,13]. Modal specifications are deterministic automata equipped with transitions of the following two types: *may* and *must*. The components that implement such interfaces are deterministic automata; an alternative language-based semantics can therefore be considered, as presented in [8,9]. Informally, a must-transition is available in every component that implements the modal specification, while a may-transition need not be. Modal specifications are interpreted as logical specifications matching the conjunctive nu-calculus fragment of the mu-calculus [14]. As a corollary, but also proved directly in [8], satisfaction and consistency of modal specifications are decidable, and the finite model property holds. Refinement between modal specifications coincides with a standard notion of alternating simulation. Since components can be seen as specifications where all transitions are typed must (all possible implementation choices have been made), satisfaction is also expressed via alternating simulation. Shared refinement is effectively computed via a product-like construction. Combination of modal specifications, handling synchronization products *à la* Arnold and Nivat [15], and the dual quotient combinators can be efficiently handled in this setting [9].

Recently, a timed extension of the theory of modal specifications has been introduced [13], motivated by the fact that time can be a crucial parameter in practice, *e.g.* in embedded-system applications. In this piece of work, components are timed automata as defined in [16], and naturally, an effective and expressive region-based semantics enables the combination of modalities and timing constraints.

In [17], we build on this preliminary paper and develop a complete compositional approach for modal specifications of timed systems. This framework favors methodologies for an incremental design process and proposes low complexity algorithms for computing product and quotient, as well as for the satisfiability decision procedure.

The synchronous product of timed objects requires a tight control on clocks [16], and so should its dual quotient. Actually, developing the theory in the general framework where components can reset their clocks in an arbitrary manner is a difficult question. Indeed, computing the resets of clocks of a product or of a quotient depends on how the control of clocks is distributed among the components. This information has to be provided *a priori*, which requires an extra formalism. We therefore restrict in [17] the presentation to the class of components definable by event-clock automata [18]: in these timed automata, resets are fully determined by the actions. Interfaces whose models are event-clock automata are called *modal event-clock specifications* (MECS).

Inheriting from the region-based semantics of timed modal specifications [13], we study in [17] the satisfiability as well as the consistency problems for MECS. Satisfiability is PSPACE-complete, hence no harder than traditional decision problems in the class of timed automata. Refinement serves as a theoretical basis to develop the product and the quotient of MECS. We propose two equivalent characterizations of these operations. Not surprisingly according to the semantics, inefficient EXPTIME constructions via the region graphs of the MECS (seen as untimed specifications) are provided. More interestingly, we present an alternative direct and efficient PTIME constructions.

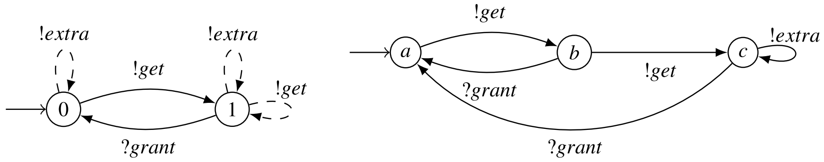
The paper is organized as follows. In Section 2, we introduce the timed modal specification setting, with preliminaries on untimed modal specifications and the definition of modal event-clock specifications. Section 3 focuses on MECS and presents effective techniques to compute the binary operations of greatest lower bound, product, and quotient; it corresponds to the heart of our contribution in [17]. In Section 4, we discuss two new extensions in order to lift some limitations of our previous work. First, we assume in [17] that all interfaces and all implementations are defined over the same alphabet of actions. This is not realistic as large systems are often composed of many subsystems possessing their own local alphabet. As a result in Section 4.3 we extend the results of [17] to interfaces and implementations over dissimilar alphabets. These results rely on alphabet equalization operations in which modalities play a central role. Secondly we discuss in Section 4.4 how to go beyond MECS and how to extend our results in [17] to interfaces with arbitrary resets. In Section 5, we compare our framework with the existing literature; this section has been enriched and updated since [17]. Section 6 concludes the paper.

In Sections 2 and 3, we fix Σ a finite set of actions.

2. Timed modal specifications

In this section we recall the framework of *modal specifications* defined in [19,20] and its timed extension, recently proposed in [13]: We discuss the semantics, the preorder refinement and the satisfiability problem for untimed and timed modal specifications.

The results introduced in the next subsection mostly come from [9]. Nevertheless they are revisited here introducing a uniform way to handle both consistent and inconsistent states (as opposed to the original definition where the so-called *pseudo-specifications* needed to be considered). This explains why proofs are detailed.



(a) Client *Cl*.

(b) Automaton $\mathcal{M}$.

Fig. 1. The modal specification *Cl* accepts the automaton $\mathcal{M}$.

2.1. Preliminaries on untimed specifications

A modal specification is an automaton equipped with two types of transitions: *must*-transitions, that are required and *may*-transitions, that are allowed.

Definition 1 (*Modal Specification*)**.** A modal specification (MS) is a tuple $\mathcal{R} = (P^{\bot\!\!\!\bot}, \lambda^0, \Delta^m, \Delta^M)$ where $P^{\bot\!\!\!\bot} = P \cup \bot\!\!\!\bot$ is a finite set of states with $\bot\!\!\!\bot \cap P = \emptyset$, $\lambda^0 \in P^{\bot\!\!\!\bot}$ is the unique initial state, and $\Delta^M \subseteq \Delta^m \subseteq P \times \Sigma \times P^{\bot\!\!\!\bot}$. Δ^M and Δ^m correspond respectively to *must*-transitions and *may*-transitions. We additionally assume that Δ^m is *deterministic* (hence so is Δ^M) and *complete*, that is, for every state $p \in P$ and every action $a \in \Sigma$, there is exactly one state $\lambda \in P^{\bot\!\!\!\bot}$ such that $(p, a, \lambda) \in \Delta^m$.

We use p (resp. λ) as typical element of P (resp. $P^{\bot\!\!\!\bot}$). The condition $\Delta^M \subseteq \Delta^m$ naturally imposes that every required transition is also allowed. The set of states $\bot\!\!\!\bot$ denotes the "bad states" which correspond to local inconsistency in the specification. Elements of $\bot\!\!\!\bot$ are *sink states* with no outgoing transition since both Δ^M and Δ^m are subsets of $P \times \Sigma \times P^{\bot\!\!\!\bot}$. *Global inconsistency* can be derived as follows: we let $\mathcal{I}$ be the set of *inconsistent* states that must lead (that is via a sequence of *must*-transitions) to a local inconsistency; states in $P^{\bot\!\!\!\bot} \setminus \mathcal{I}$ are *consistent*. Formally $\mathcal{I} = \{\lambda_0 \mid \exists n \geq 0,\ \exists \lambda_1 \cdots \lambda_n \in P^{\bot\!\!\!\bot}\ \exists a_1 \cdots a_n \in \Sigma \text{ s.t. } \lambda_n \in \bot\!\!\!\bot \text{ and } (\lambda_i, a_{i+1}, \lambda_{i+1}) \in \Delta^M\}$. Notice that in particular $\bot\!\!\!\bot \subseteq \mathcal{I}$. We say that the modal specification $\mathcal{R}$ is *consistent* whenever its initial state is consistent, i.e. $\lambda^0 \notin \mathcal{I}$; otherwise $\mathcal{R}$ is *inconsistent*.

Note that completeness is not a restriction since from any incomplete specification, one can derive a complete one by adding may-transitions to a possibly new state $\bot \in \bot\!\!\!\bot$, while preserving consistency. Intuitively, in state $p \in P$ a may-transition to some state $\lambda \in \bot\!\!\!\bot$ labeled by action a means that action a is forbidden in p. This interpretation will become clearer when we define the set of models of a modal specification.

In the following, we write or draw $p \xrightarrow{a} \lambda$ (resp. $p \overset{a}{\dashrightarrow} \lambda$) to mean $(p, a, \lambda) \in \Delta^M$ (resp. $(p, a, \lambda) \in \Delta^m \setminus \Delta^M$); in other words, solid arrows denote *required* transitions, whereas dashed arrows represent *allowed* but not required transitions. Finally we will write or draw $p \overset{a}{\rightsquigarrow} \lambda$ to indicate either $p \overset{a}{\dashrightarrow} \lambda$ or $p \xrightarrow{a} \lambda$.

Example 1. Consider a client for a given resource available in a system. The alphabet of actions includes: "*get*" when the resource is requested; "*grant*" in case of access to the resource; and, "*extra*" which occurs when a privileged access with extended time is requested.

In order to simplify the figures, states in $\bot\!\!\!\bot$ are not represented and transitions of the form $q \overset{a}{\dashrightarrow} \bot \in \bot\!\!\!\bot$ are not depicted. Action names may be preceded by some "!" or "?" when the occurrence of the actions respectively stems from the designed component or from its environment; this will enable us to underline the desired intent of a specification.

The modal specification *Cl* for the client in Fig. 1(a) specifies that a "*get*" request may be sent again. Moreover every "*get*" request *must* be granted. Additionally the client may request extended time at any moment.

Models of MS are *deterministic automata*, with possibly infinitely many states, which we abbreviate to *automata* in what follows. An automaton is a structure of the form $\mathcal{M} = (M, m^0, \Delta)$ where M is a (possibly infinite) set of states, $m^0 \in M$ is a unique initial state, and $\Delta \subseteq M \times \Sigma \to M$ is a *partial* transition function. The model relation $\models$ defined below is a particular case of alternating simulation [4] between the model and the consistent part, if any, of the specification.

Definition 2 (*Model Relation*)**.** Let $\mathcal{R} = (P^{\bot\!\!\!\bot}, \lambda^0, \Delta^m, \Delta^M)$ be a MS. An automaton $\mathcal{M} = (M, m^0, \Delta)$ is a model of $\mathcal{R}$, written $\mathcal{M} \models \mathcal{R}$, if there exists a binary relation $\rho \subseteq M \times (P \setminus \mathcal{I})$ such that $(m^0, \lambda^0) \in \rho$, and for all $(m, p) \in \rho$, the following hold: (1) for every $(p, a, \lambda) \in \Delta^M$ there is a transition $(m, a, m') \in \Delta$ with $(m', \lambda) \in \rho$, and (2) for every $(m, a, m') \in \Delta$ there is a transition $(p, a, \lambda) \in \Delta^m$ with $(m', \lambda) \in \rho$.

We denote by $\mathsf{Mod}(\mathcal{R})$, the set of models of an MS $\mathcal{R}$. Remark in Definition 2 that inconsistent states of the specification cannot appear in the relation ρ. Consequently, a transition of the form $(p, a, \lambda) \in \Delta^m$ where $\lambda \in \mathcal{I}$ is inconsistent is interpreted as: in any model, no a-transition from a state in relation with p is allowed. Moreover, for $\lambda^0 \in \mathcal{I}$ no ρ can exist and actually we have:

Lemma 1. *Let $\mathcal{R}$ be a* MS. $\mathsf{Mod}(\mathcal{R}) \neq \emptyset$ *if, and only if,* $\mathcal{R}$ *is consistent.*

Proof. ($\Rightarrow$) Assume $\mathcal{R}$ is inconsistent, *i.e.* $\lambda^0 \in \mathcal{I}$. For every automaton $\mathcal{M} = (M, m^0, \Delta)$, there cannot be any binary relation $\rho \subseteq M \times (P \setminus \mathcal{I})$ with $(m^0, \lambda^0) \in \rho$, since $\lambda^0 \in \mathcal{I}$. Hence $\mathcal{R}$ has no model.

($\Leftarrow$) Assume $\mathcal{R}$ is consistent. Intuitively, a finite model is obtained by mimicking the must-transitions of the specification. Let $p^0 = \lambda^0 \in P$, and consider the automaton $\mathcal{M}$ obtained as follows. We let m^0 be the initial state of $\mathcal{M}$, and we let m^0

be related to p^0 by a binary relation $\rho \subseteq M \times (P \setminus \mathcal{I})$ which we incrementally construct: ρ is the least relation such that for every $(m, p) \in \rho$, if $(p, a, p') \in \Delta^M$ for some $p' \in P$, then there is a target state m' in $\mathcal{M}$ of a transition (m, a, m') with $(m', p') \in \rho$. It is not difficult to verify that by construction $\mathcal{M} \models \mathcal{R}$ via the simulation ρ, which entails $\mathsf{Mod}(\mathcal{R}) \neq \emptyset$. □

Example 2. The automaton $\mathcal{M}$ in Fig. 1(b) is a model of the MS *Cl* in Fig. 1(a) as the binary relation $\rho = \{(a, 0), (b, 1), (c, 1)\}$ witnesses.

The semantic preorder between MS relies on an extension of Definition 2.

Definition 3 (*Modal Refinement Preorder*)**.** Given two MS, $\mathcal{R}_1 = (P_1^{\Perp}, \lambda_1^0, \Delta_1^m, \Delta_1^M)$ and $\mathcal{R}_2 = (P_2^{\Perp}, \lambda_2^0, \Delta_2^m, \Delta_2^M)$, $\mathcal{R}_1$ is *a refinement* of $\mathcal{R}_2$, written $\mathcal{R}_1 \preceq \mathcal{R}_2$, whenever there exists a binary relation $\rho \subseteq (\mathcal{I}_1 \times \mathcal{I}_2) \cup (P_1^{\Perp} \times (P_2 \setminus \mathcal{I}_2))$ such that $(\lambda_1^0, \lambda_2^0) \in \rho$, and for all $(\lambda_1, \lambda_2) \in \rho \cap ((P_1 \setminus \mathcal{I}_1) \times (P_2 \setminus \mathcal{I}_2))$:

(1) for every $(\lambda_2, a, \lambda_2') \in \Delta_2^M$ there exists $(\lambda_1, a, \lambda_1') \in \Delta_1^M$ with $(\lambda_1', \lambda_2') \in \rho$
(2) for every $(\lambda_1, a, \lambda_1') \in \Delta_1^m$ there exists $(\lambda_2, a, \lambda_2') \in \Delta_2^m$ with $(\lambda_1', \lambda_2') \in \rho$.

Observe that this definition extends the notion of modal refinement first introduced in [19,20] as here possible inconsistent states are taken into account.

Definition 3 requires some explanations. First, by definition of the domain of ρ, an inconsistent state of $\mathcal{R}_2$ can only be refined as an inconsistent state in $\mathcal{R}_1$ whereas a consistent state in $\mathcal{R}_2$ can either be linked to a consistent or inconsistent state in $\mathcal{R}_1$. Moreover, for pairs of consistent states, Condition (1) ensures that all required transition in $\mathcal{R}_2$ are also required in $\mathcal{R}_1$, and Condition (2) guarantees that each possible transition in $\mathcal{R}_1$ is also allowed in $\mathcal{R}_2$.

Under our assumption that MS are deterministic, we can show that the preorder $\preceq$ between MS matches the model inclusion preorder (this does not hold is MS can be nondeterministic, see Remark 1). We first establish an intermediate result that exploits the embedding of automata into modal specifications.

Definition 4 (*Embedding in* MS)**.** An automaton $\mathcal{M} = (M, m^0, \Delta)$ can be interpreted as a modal specification $\mathcal{M}^* = (M \cup \{\bot_*\}, m^0, \Delta_*^m, \Delta_*^M)$ where $\Delta = \Delta_*^M \subseteq \Delta_*^m$, and $(m, a, \bot_*) \in \Delta_*^m \setminus \Delta_*^M$ when $\Delta(m, a)$ is undefined in $\mathcal{M}$.

Lemma 2. *Given an automaton $\mathcal{M}$ and a* MS *$\mathcal{R}$, $\mathcal{M} \models \mathcal{R}$ iff $\mathcal{M}^* \preceq \mathcal{R}$.*

Proof. ($\Rightarrow$) Observe first that $\bot_*$ is the unique inconsistent state in $\mathcal{M}^*$. Let ρ be the simulation relation stating that $\mathcal{M} \models \mathcal{R}$. For $(m, p) \in \rho$ and every $(p, a, \lambda) \in \Delta^m$ with $\lambda \in \mathcal{I}$, $\mathcal{M}$ has no transition from m labeled by a. In $\mathcal{M}^*$, in this situation, there is by construction a transition from m to $\bot_*$ labeled by a. We then add $(\bot_*, \lambda)$ in ρ. The obtained simulation relation allows us to establish that $\mathcal{M}^* \preceq \mathcal{R}$.

($\Leftarrow$) For the converse direction, the pairs $(\bot_*, \lambda)$ with $\lambda \in \mathcal{I}$ characterized above are removed from the simulation relation stating that $\mathcal{M}^* \preceq \mathcal{R}$ in order to obtain the simulation relation for $\mathcal{M} \models \mathcal{R}$. □

Proposition 1. *Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be two* MS, *then:*

$$\mathcal{R}_1 \preceq \mathcal{R}_2 \quad \textit{if, and only if,} \quad \mathsf{Mod}(\mathcal{R}_1) \subseteq \mathsf{Mod}(\mathcal{R}_2).$$

Proof. ($\Rightarrow$) Let $\mathcal{R}_1 \preceq \mathcal{R}_2$ and $\mathcal{M} \models \mathcal{R}_1$. Then, by Lemma 2, $\mathcal{M}^* \preceq \mathcal{R}_1$. By transitivity of the refinement preorder, $\mathcal{M}^* \preceq \mathcal{R}_2$, and hence $\mathcal{M} \models \mathcal{R}_2$.

($\Leftarrow$) Suppose $\mathsf{Mod}(\mathcal{R}_1) \subseteq \mathsf{Mod}(\mathcal{R}_2)$. If $\mathcal{R}_1$ is inconsistent, trivially $\mathcal{R}_1 \preceq \mathcal{R}_2$. Assume now that $\mathcal{R}_1$ is consistent. Then so must be $\mathcal{R}_2$. We can write p_1^0 (resp. p_2^0) for the initial state of $\mathcal{R}_1$ (resp. $\mathcal{R}_2$). As $\mathcal{R}_1$ and $\mathcal{R}_2$ are deterministic, a simulation relation ρ stating that $\mathcal{R}_1$ is a refinement of $\mathcal{R}_2$, if it exists, is unique. We consider the binary relation ρ as the least relation such that with $(p_1^0, p_2^0) \in \rho$ and for every $(p_1, p_2) \in \rho \cap ((P_1 \setminus \mathcal{I}_1) \times (P_2 \setminus \mathcal{I}_2))$, we let $(\lambda_1, \lambda_2) \in \rho$ whenever $(p_2, a, \lambda_2) \in \Delta_2^M$ and $(p_1, a, \lambda_1) \in \Delta_1^M$, or $(p_1, a, \lambda_1) \in \Delta_1^m$ and $(p_2, a, \lambda_2) \in \Delta_2^m$.

We show that $\rho \subseteq (\mathcal{I}_1 \times \mathcal{I}_2) \cup (P_1^{\Perp} \times (P_2 \setminus \mathcal{I}_2))$, which entails that ρ is a witness for $\mathcal{R}_1 \preceq \mathcal{R}_2$.

- if $(p_2, a, \lambda_2) \in \Delta_2^M$ then $\lambda_2 \in P_2 \setminus \mathcal{I}_2$ otherwise we would have $p_2 \in \mathcal{I}_2$. Moreover every model $\mathcal{M}$ which has a state m related to the state p_2 of $\mathcal{R}_2$ necessarily has an a-transition leaving m. A weaker claim for p_1 is not possible, otherwise we would not have $\mathsf{Mod}(\mathcal{R}_1) \subseteq \mathsf{Mod}(\mathcal{R}_2)$. As a result, $(p_1, a, \lambda_1) \in \Delta_1^M$ and $(\lambda_1, \lambda_2) \in P_1^{\Perp} \times (P_2 \setminus \mathcal{I}_2)$.
- if $(p_1, a, \lambda_1) \in \Delta_1^m$ then $(p_2, a, \lambda_2) \in \Delta_2^m$ as $\mathcal{R}_2$ is complete. We now prove that $(\lambda_1, \lambda_2) \in (\mathcal{I}_1 \times \mathcal{I}_2) \cup (P_1^{\Perp} \times (P_2 \setminus \mathcal{I}_2))$:
  - if $\lambda_1 \in P_1 \setminus \mathcal{I}_1$, we have to prove that $\lambda_2 \in P_2 \setminus \mathcal{I}_2$. As $\lambda_1 \in P_1 \setminus \mathcal{I}_1$, there exists $\mathcal{M} \models \mathcal{R}_1$ having a transition from p_1 labeled by a. As $\mathsf{Mod}(\mathcal{R}_1) \subseteq \mathsf{Mod}(\mathcal{R}_2)$ then $\mathcal{M}$ should also be a model of $\mathcal{R}_2$ and thus a transition a should be allowed in p_2. As a result, $\lambda_2 \in P_2 \setminus \mathcal{I}_2$;
  - if $\lambda_1 \in \mathcal{I}_1$ then for $\lambda_2 \in P_2^{\Perp}$ we have $(\lambda_1, \lambda_2) \in \rho$. □

Remark 1. The determinism of modal specifications is crucial for the Proposition 1. In the nondeterministic case, modal refinement is not complete [3]: $\mathsf{Mod}(\mathcal{R}_1) \subseteq \mathsf{Mod}(\mathcal{R}_2)$ does not necessarily imply $\mathcal{R}_1 \preceq \mathcal{R}_2$.

As a consequence of Definition 3, inconsistent MS refine any MS, and consistent MS can only refine consistent MS. In the following, we write $\mathcal{R}_1 \equiv \mathcal{R}_2$, and say that $\mathcal{R}_1$ and $\mathcal{R}_2$ are *equivalent*, whenever $\mathcal{R}_1 \preceq \mathcal{R}_2$ and $\mathcal{R}_2 \preceq \mathcal{R}_1$. Remark that by merging all states of $\mathcal{I}$, every MS is equivalent to a MS where the set of inconsistent states is at most a singleton.

2.2. Modal event-clock specifications

Let $\mathcal{X}$ be a finite set of *clocks* and let $R_{\geq 0}$ denote the set of non-negative reals. A *clock valuation* over $\mathcal{X}$ is a mapping $\nu : \mathcal{X} \to R_{\geq 0}$. The set of clock valuations over $\mathcal{X}$ is denoted $\mathcal{V}$; in particular, $\overline{0} \in \mathcal{V}$ is the clock valuation such that $\overline{0}(x) = 0$ for all $x \in \mathcal{X}$. Given $\nu \in \mathcal{V}$ and $t \in R_{\geq 0}$, we let $(\nu + t) \in \mathcal{V}$ be the clock valuation obtained by letting t time units elapse after ν, formally, $(\nu + t)(x) = \nu(x) + t$ for every $x \in \mathcal{X}$. Moreover, given $x_a \in \mathcal{X}$ and $\nu \in \mathcal{V}$, the valuation $\nu[0/x_a]$ is defined by $\nu[0/x_a](x_a) = 0$ and $\nu[0/x_a](y) = \nu(y)$ for all $y \neq x_a$.

A *guard* over $\mathcal{X}$ is a finite conjunction of expressions of the form $x \sim c$ where $x \in \mathcal{X}$, $c \in \boldsymbol{N}$ is a *constant*, and $\sim \in \{<, \leq, =, \geq, >\}$. We then denote by $\xi[\mathcal{X}]$ the set of all guards over $\mathcal{X}$. For some fixed $N \in \boldsymbol{N}$, $\xi_N[\mathcal{X}]$ represents the set of guards involving only constants equal to or smaller than N. The *satisfaction relation* $\models \subseteq (\mathcal{V} \times \xi[\mathcal{X}])$ between clock valuations and guards is defined in a natural way and we write $\nu \models g$ whenever ν satisfies g; we denote by **false** the unsatisfiable formula (modulo logical equivalence). In the following, we will often abuse notation and write g to denote the guard g as well as the set of valuations which satisfy g. Note that we only consider here diagonal-free guards as only comparisons between clocks and constants are allowed.

Also, for any $g \in \xi[\mathcal{X}]$, we denote by $g_{\downarrow a}$ the formula which semantics is the set $\{\nu[0/x_a] \mid \nu \models g\}$ and by $g^{\uparrow a}$ the formula whose semantics is the set of valuations ν such that $\nu[0/x_a] \models g$. Note that given a guard g, formulas for $g_{\downarrow a}$ and $g^{\uparrow a}$ can be computed easily. We explain how to compute $g^{\uparrow a}$: assume g is in disjunctive normal form, that is $g = \bigvee_i \bigwedge_j g_{ij}$ where each g_{ij} is of the form $x \sim c$. Then consider the formula $g' = \bigvee_i (\bigwedge_j g_{ij} \wedge (x_a = 0))$ where each unsatisfiable conjunct $\bigwedge_j g_{ij} \wedge (x_a = 0)$ has been replaced by **false**. Obtain the formula $g^{\uparrow a}$ by removing all constraints on x_a in any satisfiable conjunct of g'; notice that since we do not have diagonal constraints, this removal affects only the constraints on x_a.

Event-clock automata [18], form a subclass of timed automata where clock resets are not arbitrary: each action a comes with a clock x_a which is reset exactly when action a occurs. We consider event-clock automata with possibly infinitely many locations.

Definition 5 (*Event-Clock Automata*). An *event-clock automaton* (ECA) over Σ is a tuple $\mathcal{C} = (C, c^0, \delta)$ where C is a set of states, $c^0 \in C$ is the initial state, and $\delta \subseteq C \times \xi_N[\mathcal{X}_\Sigma] \times \Sigma \times C$ is the transition relation (for some $N \in \mathbb{N}$). The pair (Σ, N) is the *signature* of $\mathcal{C}$.

The semantics of an ECA is similar to the one of a timed automaton [16], except that the set of clocks that are reset by a transition is determined by the action of that transition: while firing a transition labeled by a, precisely clock x_a is reset. Event-clock automata do form a strict subclass of timed automata, but they enjoy good properties: they are closed under union and intersection, and more interestingly they can be made deterministic (as opposed to the class of arbitrary timed automata). The ability for event-clock automata to be made deterministic comes from the way clocks are reset and this property significantly eases the definition of binary operators (such as lower bound, product and quotient) on modal variants of event-clock automata.

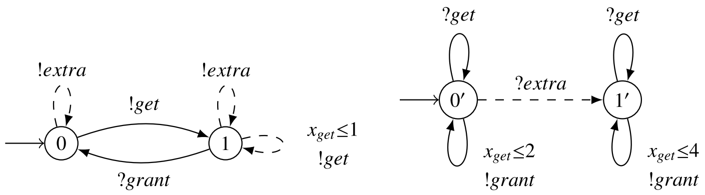
For a fixed signature (Σ, N), a *region* is an equivalence class θ of clock valuations that satisfy the same guards in $\xi_N[\mathcal{X}_\Sigma]$. We denote by Θ_N, or simply Θ, the set of all regions. Given a region $\theta \in \Theta$, we write $\tau(\theta)$ for the set of all regions that can be obtained from θ by letting time elapse: $\tau(\theta) = \{\theta'' \mid \exists \nu'' \in \theta'' \, \exists \nu \in \theta \, \exists t \in R_{\geq 0} \text{ s.t. } \nu'' = \nu + t\}$. The reset and coreset operations are extended from guard to regions in the expected way; given a region θ, we thus write $\theta_{\downarrow a}$ and $\theta^{\uparrow a}$ for respectively the region obtained by resetting clock x_a or, respectively the union of regions obtained via the inverse operation.

Definition 6 (*Region Automaton [16]*). The *region automaton* associated to an ECA $\mathcal{C} = (C, c^0, \delta)$ is the automaton $R(\mathcal{C}) = (C \times \Theta, (c^0, \overline{0}), \Delta)$ over the alphabet $\Theta \times \Sigma$, where the set Δ of transitions is defined as follows: for each $c, c' \in C$, $\theta, \theta', \theta'' \in \Theta$, and $a \in \Sigma$, $((c, \theta), \theta'', a, (c', \theta')) \in \Delta$ whenever there exists $(c, g, a, c') \in \delta$ with $\theta'' \subseteq \tau(\theta) \cap g$ and $\theta' = \theta''_{\downarrow a}$ (recall this is the region obtained from θ'' by resetting the clock x_a).

Remark 2 (*Embedding of Region Automata into Event-Clock Automata*). Note that the region automata we consider extend the ones introduced in [16] since their transition labels keep track of the intermediate region where the action is fired. As a consequence, any automaton over the alphabet $\Theta \times \Sigma$ uniquely defines an ECA whose signature is of the form (Σ, N_Θ), with N_Θ determined by the set of regions Θ. We denote by T the natural injection of region automata into ECA; this mapping enables us to distinguish between the two interpretations of the same syntactic object: $R(\mathcal{C})$ is an automaton whereas $T(R(\mathcal{C}))$ is an ECA. The mappings T and R form a Galois connection (see [21] for an introduction on this particular correspondence).

Definition 7 (*Modal Event-Clock Specification*). A *modal event-clock specification (*MECS*)* over the finite alphabet Σ is a tuple $\mathcal{S} = (Q^{\perp\!\!\!\perp}, \lambda^0, \delta^m, \delta^M)$ where

- $Q^{\perp\!\!\!\perp} := Q \cup \perp\!\!\!\perp$ is a finite set of *locations*, with $\perp\!\!\!\perp \cap Q = \emptyset$, and the initial state is $\lambda^0 \in Q^{\perp\!\!\!\perp}$.
- $\delta^M \subseteq \delta^m \subseteq Q \times \xi[\mathcal{X}_\Sigma] \times \Sigma \times Q^{\perp\!\!\!\perp}$ are finite sets of respectively *must*- and *may*-transitions. Given a may-transition $(q, g, a, \lambda) \in \delta^m$, q is the source state, λ is the destination state, $g \in \xi[\mathcal{X}_\Sigma]$ is the guard that specifies the valuations for which the transition can be taken, $a \in \Sigma$ is the action labeling the transition—recall that the only clock that is then reset is x_a.



(a) Client *Cl*.

(b) Access controller *Acc*.

Fig. 2. Client *Cl* and access controller *Acc*.

Moreover we require that δ^m is *deterministic* (hence, so is δ^M) and *complete*: for any state $q \in Q$, any action $a \in \Sigma$, and any clock valuation $\nu \in \mathcal{V}$, there is exactly one transition $(q, g, a, \lambda) \in \delta^m$ such that $\nu \models g$.

Example 3. As an example of a MECS, we consider in Fig. 2(a) a timed variant of the client *Cl* introduced earlier. The clock corresponding to the action "*get*" is x_{get}.

In this example again, for simplification purposes, transitions of the form $q \overset{g,a}{\dashrightarrow} \bot$ are not depicted. As MECS are complete, these transitions can easily be recovered by taking $g = \bigwedge_i(\tilde{g}_i)$ where the g_i's are the guards appearing in the transitions of the form $q \overset{g_i,a}{\dashrightarrow} \lambda$ or $q \xrightarrow{g_i,a} \lambda$, and each $\tilde{g}_i$ is the semantic negation of g_i. When the guard of a transition is not indicated, it is implicitly true.

The MECS *Cl* for the client in Fig. 2(a) specifies that a "*get*" request may be sent again at most one time unit after the last request.

In what follows, we generalize the graphical conventions already used for untimed objects by writing $q \overset{g,a}{\dashrightarrow} \lambda'$ whenever $(q, g, a, \lambda') \in (\delta^m \setminus \delta^M)$, $q \xrightarrow{g,a} \lambda'$ whenever $(q, g, a, \lambda') \in \delta^M$ and $q \overset{g,a}{\rightsquigarrow} \lambda'$ whenever either $q \overset{g,a}{\dashrightarrow} \lambda'$ or $q \xrightarrow{g,a} \lambda'$.

Remark that a natural untimed object associated to a MECS $\mathcal{S}$ is its region modal automaton, obtained by generalizing Definition 6 from event-clock automata to their modal extension. More precisely, $R(\mathcal{S})$ reflects the modalities of $\mathcal{S} = (Q^{\perp}, \lambda^0, \delta^m, \delta^M)$ as done in [13], the initial state is $(\lambda^0, \overline{0})$ and the set of locally inconsistent states in $R(\mathcal{S})$ is $\perp\!\!\!\perp_{\mathcal{S}} \times \Theta$. Nevertheless, global inconsistency in MECS is not trivial, as detailed in the next paragraph.

Global inconsistency in MECS. Obviously, we expect this notion to coincide with global inconsistency in untimed models when the region automaton of the MECS is considered. Henceforth, inconsistency is not defined on locations but rather on regions: using the inconsistent states (λ, θ) of $R(\mathcal{S})$, we can gather the regions θ for a fixed λ and define it as $\mathcal{I}(\lambda) \in \xi[\mathcal{X}_\Sigma]$.

Definition 8. Let $\mathcal{S} = (Q^{\perp}, \lambda^0, \delta^m, \delta^M)$ be a MECS. For every $\lambda \in Q^{\perp}$, let $\mathcal{I}(\lambda) \in \xi[\mathcal{X}_\Sigma]$ be such that for any region θ,

$$\theta \subseteq \mathcal{I}(\lambda) \quad \text{if, and only if,} \quad (\lambda, \theta) \text{ is globally inconsistent in } R(\mathcal{S}).$$

Lemma 3. *The sets $I(\lambda)$ are closed under timed predecessors: let $\theta' \in \tau(\theta)$, then $\theta' \in I(\lambda)$ implies $\theta \in I(\lambda)$.*

Proof. If $\theta' \in I(\lambda)$ then (λ, θ') is globally inconsistent in $R(\mathcal{S})$ that is, there exists a must-transition from (λ, θ') to a state (λ', θ'') labeled by (θ''', a) such that there is a sequence of must-transitions in $R(\mathcal{S})$ from (λ', θ'') to a local inconsistency. By construction of the modal region automata, there is a transition $\lambda \xrightarrow{g,a} \lambda'$ in $\mathcal{S}$ and $\theta''' \subseteq \tau(\theta') \cap g$ and $\theta'' = \theta'''_{\downarrow a}$. As, by assumption, $\theta' \in \tau(\theta)$, we also have $\theta''' \in \tau(\theta)$ and thus $\theta''' \subseteq \tau(\theta) \cap g$. As a result, there is a must-transition in $R(\mathcal{S})$ from (λ, θ) to (λ', θ'') labeled (θ''', a) and (λ, θ) is globally inconsistent that is, $\theta \in I(\lambda)$. □

Notice that in general the sets $I(\lambda)$ are not closed under timed successors, as letting time elapse may invalidate the guards responsible for the original inconsistency.

Definition 9. A MECS $\mathcal{S}$ is *I-stable* if for any state λ, the set $I(\lambda)$ is closed under timed successors; formally,

$$\tau(I(\lambda)) \subseteq I(\lambda)$$

Otherwise said, I-stability means that inconsistency cannot arise from letting time elapse. This is a mandatory requirement to ensure optimal constructions, as stated in Theorem 4 in Section 3.

Example 4. The MECS $\mathcal{S}_1$ in Fig. 7(b) is not I-stable. Indeed $I(\lambda_1)$ is restricted to the valuation $x = 0$ and is not closed under timed successors.

Interestingly, the computation of $\mathcal{I}(\lambda)$ does not require the computation of the region automaton, but can be done following the lines of a construction in [22]: a backward analysis in the MECS from locations in $\perp\!\!\!\perp$ enables us to synthesize the clock formula $\mathcal{I}(\lambda)$. We proceed as follows.

Let E be the set of must-transitions in $\mathcal{S}$ that lead to a location in $\perp\!\!\!\perp$, and let $e = (\lambda, g, a, \lambda') \in E$ (then $\lambda' \in \perp\!\!\!\perp$). Clearly $g \subseteq \mathcal{I}(\lambda)$ because in any state (λ, θ) of $R(\mathcal{S})$ with $\theta \subseteq g$, a discrete must-transition (labeled by a) is enabled that leads to the inconsistent state $(\lambda', \theta[0/x_a])$. Also, time elapsing needs to be considered: from (λ, θ) in $R(\mathcal{S})$ where $\theta \subseteq (\exists t \in R_{\geq 0})(g[x_b + t/x_b]_{b \in \Sigma})$, it is possible to let time elapse to reach a state (λ, θ'') where $\theta'' \subseteq g$ which shows a

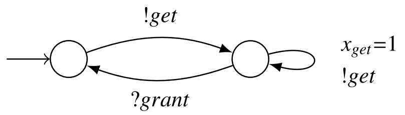


Fig. 3. An example of model for the MECS *Cl* of Fig. 2(a).

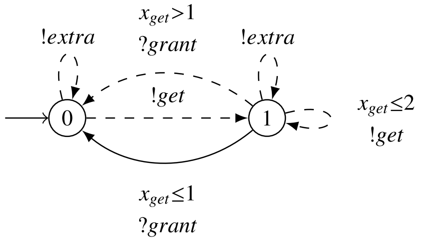


Fig. 4. Another client specification Cl' refined by *Cl* of Fig. 2(a).

must-transition from (λ, θ) to (λ', θ') (labeled by $(a, \theta''))$ in $R(\mathcal{S})$, yielding (λ, θ) is globally inconsistent in $R(\mathcal{S})$. It is easy to generalize the argument for arbitrary target locations λ' that do not necessarily belong to $\perp\!\!\!\perp$.

Clearly, discrete transitions and time elapsing ones as considered above cover all contexts (regions) in which a given location becomes inconsistent, and they are the only ones. The formulas $\{\mathcal{I}(\lambda)\}_{\lambda\in Q^{\perp\!\!\!\perp}}$ that characterize the regions in which a given location is inconsistent are the least solutions (in the lattice $(\xi[\mathcal{X}_\Sigma], \subseteq)$) of the following finite system of equations:

$$\begin{cases} \mathcal{I}(\lambda) = \textbf{true}, & \forall\lambda \in\perp\!\!\!\perp \\ \mathcal{I}(\lambda) = \bigvee\limits_{(\lambda,g,a,\lambda')\in\delta^M} (\exists t \in R_{\geq 0})\Big(g[x_b + t/x_b]_{b\in\Sigma} \wedge \mathcal{I}(\lambda')[0/x_a, \{x_b + t/x_b\}_{b\neq a}]\Big), & \forall\lambda \in Q \end{cases} \qquad (1)$$

This least fixed point can be computed iteratively starting from $\mathcal{I}(\lambda) := \textbf{true}$, for all $\lambda \in\perp\!\!\!\perp$, and $\mathcal{I}(\lambda) := \textbf{false}$, for all $\lambda \in Q$, and by applying the formula monotonic transformer underlying the equation system, in a standard way, until stabilization. Proofs of both correctness and stabilization of this computation can be found in [22,23].

A note on consistency. According to Lemma 1, checking whether an untimed specification has a model amounts to checking its consistency, namely whether the set of states $\perp\!\!\!\perp$ is unreachable from the initial state by a sequence of must-transitions. The consistency problem is thus NLOGSPACE-complete for (untimed) modal specifications, since it can be rephrased as a reachability problem. Consequently, it becomes PSPACE-complete in the timed case. Notice that the consistency of a MECS reduces to deciding whether $\mathcal{I}(\lambda^0) \wedge (\bigwedge_{b\in\Sigma} x_b = 0) \equiv \textbf{false}$.

Semantics of MECS. We now turn to the set of models denoted by a MECS. Recall that given a modal event-clock specification $\mathcal{S}$ over signature (Σ, N), $R(\mathcal{S})$ is a modal specification over the extended alphabet $\Sigma \times \Theta_N$; similarly, given an event-clock automaton $\mathcal{C}$, $R(\mathcal{C})$ is an automaton over alphabet $\Sigma \times \Theta_N$. Having this in mind, the model relation in the timed case is inherited from the one in the untimed case via the region construction:

Definition 10 (*Model Relation*)**.** An event-clock automaton $\mathcal{C}$ is a *model* of MECS $\mathcal{S}$, written $\mathcal{C} \models \mathcal{S}$, if $R(\mathcal{C}) \models R(\mathcal{S})$.

Example 5. An example of model for the client specification of Fig. 2(a) is depicted in Fig. 3. Observe that must-transitions of the specification are reflected in the model and the guards of may-transitions of the specification can be strengthened in the model.

The set of models of a MECS $\mathcal{S}$, is defined by $\mathsf{Mod}(\mathcal{S}) := \{\mathcal{C} \mid \mathcal{C} \models \mathcal{S}\}$. Observing that given a MECS $\mathcal{S}$, $R(T(R(\mathcal{S})))$ and $R(\mathcal{S})$ are isomorphic, we obtain the following:

Lemma 4. *Let $\mathcal{S}$ be a* MECS. *Then,* $\mathsf{Mod}(T(R(\mathcal{S}))) = \mathsf{Mod}(\mathcal{S})$.

In the spirit of Definition 10 for the model relation, the *modal refinement preorder* between MECS also relies on a region-based construction:

Definition 11 (*Modal Refinement Preorder*)**.** Given two MECS $\mathcal{S}_1$ and $\mathcal{S}_2$, $\mathcal{S}_1$ *refines* $\mathcal{S}_2$, written $\mathcal{S}_1 \preceq \mathcal{S}_2$, whenever $R(\mathcal{S}_1) \preceq R(\mathcal{S}_2)$.

Example 6. The client specification *Cl* refines the specification Cl' represented in Fig. 4. Observe that in Cl' a "*get*" request may be resent at most two units of time after the last request whereas in the refined version *Cl* the deadline is at most one. Similarly, in Cl' the "*grant*" action is not guaranteed after more than one unit of time whereas in *Cl* every request is eventually granted.

As a corollary of the analogous results in the untimed setting on MS, it is decidable whether a MECS refines another one. Moreover, refinement and inclusion of models match:

Corollary 1. *Let $\mathcal{S}$, $\mathcal{S}_1$ and $\mathcal{S}_2$ be* MECS. *Then,*

- $\mathsf{Mod}(\mathcal{S}) \neq \emptyset$ *if, and only if $\mathcal{S}$ is consistent;*
- $\mathcal{S}_1 \preceq \mathcal{S}_2$ *if, and only if* $\mathsf{Mod}(\mathcal{S}_1) \subseteq \mathsf{Mod}(\mathcal{S}_2)$.

Proof. The first item is a consequence of the similar result for untimed specifications (see Lemma 1), as well as the immediate observation that given an automaton $\mathcal{M}$, $\mathcal{M} \models R(\mathcal{S})$ implies $T(\mathcal{M}) \models \mathcal{S}$. The second item is a trivial consequence of Proposition 1 and Definition 11. □

The class of deterministic ECA can be embedded into the one of MECS; let $\mathcal{C}$ be an ECA, we denote by $\mathcal{C}^*$ the MECS obtained by typing with must every existing transition in $\mathcal{C}$ and by completing it by adding may-transitions to a state $\perp$ in $\perp\!\!\!\perp$.

Definition 12 (*Embedding in* MECS). An ECA $\mathcal{C} = (C, c^0, \delta)$ can be interpreted as a MECS $\mathcal{C}^* = (C \cup \{\perp_*\}, c^0, \delta^m_*, \delta^M_*)$ where $\delta = \delta^M_* \subseteq \delta^m_*$, and $(\lambda, g, a, \perp_*) \in \Delta^m_* \setminus \Delta^M_*$ with $g = \bigwedge_i \tilde{g_i}$ and where the $\tilde{g_i}$'s are the negation of the guards g_i's appearing in the transitions of the form $(\lambda, g_i, a, \lambda')$ in δ.

Since event-clock are determinizable [18], assuming their determinism should not be regarded as a restriction. We then have:

Corollary 2. *Let $\mathcal{C}$ be an* ECA *and $\mathcal{S}$ a* MECS, *$\mathcal{C} \models \mathcal{S}$ if and only if $\mathcal{C}^* \preceq \mathcal{S}$.*

Proof. This follows from Definition 10 which tells that $\mathcal{C} \models \mathcal{S}$ whenever $R(\mathcal{C}) \models R(\mathcal{S})$. Moreover, by Definition 11, $\mathcal{C}^* \preceq \mathcal{S}$ if and only if, $R(\mathcal{C}^*) \preceq R(\mathcal{S})$. To conclude, it suffices to consider Corollary 1. □

3. Operations on specifications

In this section, we introduce operations on modal event-clock specifications, which enable compositional reasoning. More precisely, we define the greatest lower bound, the product, and the quotient over MECS. For each of these operations, we establish important theoretical properties.

3.1. Greatest lower bound of MECS

We study the concept of *greatest lower bound*, which corresponds to the conjunction of two modal specifications and equivalently to their best shared refinement [24,10]. Greatest lower bound is worth computing to ensure that given several specifications, each of them describing a particular requirement, we are able to check their compatibility.

We first recall the definition of the greatest lower bound in the untimed case. Let $\mathcal{R}_1 = (P_1^{\perp\!\!\!\perp}, \lambda_1^0, \Delta_1^m, \Delta_1^M)$ and $\mathcal{R}_2 = (P_2^{\perp\!\!\!\perp}, \lambda_2^0, \Delta_2^m, \Delta_2^M)$ be two MS. The *greatest lower bound* of $\mathcal{R}_1$ and $\mathcal{R}_2$ is $\mathcal{R}_1 \wedge \mathcal{R}_2 = (P^{\perp\!\!\!\perp}, (\lambda_1^0, \lambda_2^0), \Delta^m_\wedge, \Delta^M_\wedge)$ with $P := P_1 \times P_2$ and $\perp\!\!\!\perp := (\perp\!\!\!\perp_1 \times P_2^{\perp\!\!\!\perp}) \cup (P_1^{\perp\!\!\!\perp} \times \perp\!\!\!\perp_2)$; notice that local inconsistency of a compound state (membership of (λ_1, λ_2) in $\perp\!\!\!\perp$) is inherited from the local inconsistency of the components (membership of λ_1 in $\perp\!\!\!\perp_1$ or membership of λ_2 in $\perp\!\!\!\perp_2$). The transition relations are derived from the following rules.

$$\frac{\lambda_1 \overset{a}{\dashrightarrow} \lambda_1' \text{ and } \lambda_2 \overset{a}{\dashrightarrow} \lambda_2'}{(\lambda_1, \lambda_2) \overset{a}{\dashrightarrow} (\lambda_1', \lambda_2')} \ (Glb1) \qquad \frac{\lambda_1 \overset{a}{\longrightarrow} \lambda_1' \text{ and } \lambda_2 \overset{a}{\rightsquigarrow} \lambda_2'}{(\lambda_1, \lambda_2) \overset{a}{\longrightarrow} (\lambda_1', \lambda_2')} \ (Glb2) \qquad \frac{\lambda_1 \overset{a}{\rightsquigarrow} \lambda_1' \text{ and } \lambda_2 \overset{a}{\longrightarrow} \lambda_2'}{(\lambda_1, \lambda_2) \overset{a}{\longrightarrow} (\lambda_1', \lambda_2')} \ (Glb3)$$

Remark in particular, that if in a state $\lambda = (\lambda_1, \lambda_2)$, we have the contradictory requirements that a is required ($\lambda_1 \overset{a}{\longrightarrow} \lambda_1' \in P_1$) and a should not happen ($\lambda_2 \overset{a}{\dashrightarrow} \lambda_2' \in \perp\!\!\!\perp_2$), then λ is inconsistent. This is indeed guaranteed by the definition of $\mathcal{R}_1 \wedge \mathcal{R}_2$ which imposes $P_1 \times \perp\!\!\!\perp_2 \subseteq \perp\!\!\!\perp$.

Also, since the Rules ($Glb1$)–($Glb3$) uniformly consider consistent and inconsistent states, global inconsistency in $\mathcal{R}_1 \wedge \mathcal{R}_2$ (membership of (λ_1, λ_2) in $\mathcal{I}$) is inherited from local inconsistency of the components (membership of λ_1 in $\mathcal{I}_1$ or membership of λ_2 in $\mathcal{I}_2$).

Greatest lower bound of MECS. The notion of greatest lower bound easily extends to MECS. Let $\mathcal{S}_1$, $\mathcal{S}_2$ be two MECS. The modalities for the transitions in $\mathcal{S}_1 \wedge \mathcal{S}_2$ are derived from those induced in the untimed case (Rules ($Glb1$)–($Glb3$)), and the labels of the transitions are obtained by intersecting the guards for common actions. We therefore get the following Rules ($tGlb1$), ($tGlb2$) and ($tGlb3$).

$$\frac{\lambda_1 \overset{g_1,a}{\dashrightarrow} \lambda_1' \text{ and } \lambda_2 \overset{g_2,a}{\dashrightarrow} \lambda_2'}{(\lambda_1, \lambda_2) \overset{g_1 \wedge g_2,a}{\dashrightarrow} (\lambda_1', \lambda_2')} \ (tGlb1) \qquad \frac{\lambda_1 \overset{g_1,a}{\longrightarrow} \lambda_1' \text{ and } \lambda_2 \overset{g_2,a}{\rightsquigarrow} \lambda_2'}{(\lambda_1, \lambda_2) \overset{g_1 \wedge g_2,a}{\longrightarrow} (\lambda_1', \lambda_2')} \ (tGlb2) \qquad \frac{\lambda_1 \overset{g_1,a}{\rightsquigarrow} \lambda_1' \text{ and } \lambda_2 \overset{g_2,a}{\longrightarrow} \lambda_2'}{(\lambda_1, \lambda_2) \overset{g_1 \wedge g_2,a}{\longrightarrow} (\lambda_1', \lambda_2')} \ (tGlb3)$$

Thanks to Lemma 4, the set of models of a MECS $\mathcal{S}$ matches the set of models of its region version $T(R(\mathcal{S}))$. The following proposition characterizes the greatest lower bound of two MECS via the region graphs.

Proposition 2. *For any two* MECS *$\mathcal{S}_1$ and $\mathcal{S}_2$, $R(\mathcal{S}_1 \wedge \mathcal{S}_2) \equiv R(\mathcal{S}_1) \wedge R(\mathcal{S}_2)$.*

Proof. Consider the binary relation $\mathfrak{R}$ between states of $R(\mathcal{S}_1 \wedge \mathcal{S}_2)$ and of $R(\mathcal{S}_1) \wedge R(\mathcal{S}_2)$ defined by:

$$\mathfrak{R} = \left\{ \Big(\big((\lambda_1, \theta), (\lambda_2, \theta)\big), \big((\lambda_1, \lambda_2), \theta\big) \Big) \mid \lambda_1 \in Q_1^{\perp\!\!\!\perp}, \lambda_2 \in Q_2^{\perp\!\!\!\perp} \right\}.$$

Notice that any reachable state in $R(\mathcal{S}_1) \wedge R(\mathcal{S}_2)$ is of the form $((\lambda_1, \theta_1), (\lambda_2, \theta_2))$ with $\theta_1 = \theta_2$. This can be easily proved inductively since the greatest lower bound for $R(\mathcal{S}_1)$ and $R(\mathcal{S}_2)$ is computed on the extended alphabet $\Sigma \times \Theta$, where the target region is completely determined by the later (since it contains the region when the transition is fired and the clock to be reset).

Let $\lambda_1 \in Q_1^{\perp\!\!\!\perp}$, $\lambda_2 \in Q_2^{\perp\!\!\!\perp}$ and $\theta \in \Theta$. We show that any required transition from $((\lambda_1, \theta), (\lambda_2, \theta))$ in $R(\mathcal{S}_1) \wedge R(\mathcal{S}_2)$ is also mandatory from $((\lambda_1, \lambda_2), \theta\big)$ in $R(\mathcal{S}_1 \wedge \mathcal{S}_2)$, and that any allowed transition from $((\lambda_1, \lambda_2), \theta\big)$ in $R(\mathcal{S}_1 \wedge \mathcal{S}_2)$ is possible from $((\lambda_1, \theta), (\lambda_2, \theta))$ in $R(\mathcal{S}_1) \wedge R(\mathcal{S}_2)$.

Let $((\lambda_1, \lambda_2), \theta)) \overset{\theta'',a}{\dashrightarrow} ((\lambda_1', \lambda_2'), \theta'))$ be a may-transition in $R(\mathcal{S}_1 \wedge \mathcal{S}_2)$. By construction of the region graph, there exists a may-transition $(\lambda_1, \lambda_2) \overset{g,a}{\dashrightarrow} (\lambda_1', \lambda_2')$ with $\theta'' \subseteq \tau(\theta) \cap g$ in $\mathcal{S}_1 \wedge \mathcal{S}_2$. This transition can only be obtained by applying Rule (*tGlb*1); hence there exist $\lambda_1 \overset{g_1,a}{\dashrightarrow} \lambda_1'$ in $\mathcal{S}_1$ and $\lambda_2 \overset{g_2,a}{\dashrightarrow} \lambda_2'$ in $\mathcal{S}_2$ with $g = g_1 \wedge g_2$. Since $\theta'' \subseteq g_1 \cap g_2$ these transitions give rise in $R(\mathcal{S}_1)$ and $R(\mathcal{S}_2)$ respectively to transitions $(\lambda_1, \theta) \overset{\theta'',a}{\dashrightarrow} (\lambda_1', \theta')$ and $(\lambda_2, \theta) \overset{\theta'',a}{\dashrightarrow} (\lambda_2', \theta')$. Hence, in the greatest lower bound $R(\mathcal{S}_1) \wedge R(\mathcal{S}_2)$, thanks to Rule (*Glb*1), there is a may-transition $((\lambda_1, \theta), (\lambda_2, \theta)) \overset{\theta'',a}{\dashrightarrow} ((\lambda_1', \theta'), (\lambda_2', \theta'))$.

Assume now $((\lambda_1, \theta), (\lambda_2, \theta)) \xrightarrow{\theta'',a} ((\lambda_1', \theta'), (\lambda_2', \theta'))$ is a must-transition in $R(\mathcal{S}_1) \wedge R(\mathcal{S}_2)$. According to the rules (*Glb*2) and (*Glb*3) this transition comes from transitions in $R(\mathcal{S}_1)$ and $R(\mathcal{S}_2)$, one of which being a must-transition. W.l.o.g assume $(\lambda_1, \theta) \xrightarrow{\theta'',a} (\lambda_1', \theta')$ and $(\lambda_2, \theta) \overset{\theta'',a}{\dashrightarrow} (\lambda_2', \theta')$ (the latter transition could also be a must). By construction of the region graph, there are transitions $\lambda_1 \xrightarrow{g_1,a} \lambda_1'$ and $\lambda_2 \overset{g_2,a}{\dashrightarrow} \lambda_2'$ in $\mathcal{S}_1$ and $\mathcal{S}_2$ respectively, with $\theta'' \subseteq \tau(\theta) \cap g_1$ and also $\theta'' \subseteq \tau\theta \cap g_2$. In $\mathcal{S}_1 \wedge \mathcal{S}_2$ there is thus a transition $(\lambda_1, \lambda_2) \xrightarrow{g_1 \cap g_2,a} (\lambda_1', \lambda_2')$; this yields a transition $((\lambda_1, \lambda_2), \theta)) \xrightarrow{\theta'',a} ((\lambda_1', \lambda_2'), \theta'))$.

To prove that relation $\mathfrak{R}$ is a witness for $R(\mathcal{S}_1 \wedge \mathcal{S}_2) \preceq R(\mathcal{S}_1) \wedge R(\mathcal{S}_2)$, it now suffices to observe that inconsistent states in $R(\mathcal{S}_1) \wedge R(\mathcal{S}_2)$ can only be linked in $\mathfrak{R}$ to inconsistent states in $R(\mathcal{S}_1 \wedge \mathcal{S}_2)$. This however is a consequence of the fact that must-transition in $R(\mathcal{S}_1) \wedge R(\mathcal{S}_2)$ are also required in $R(\mathcal{S}_1 \wedge \mathcal{S}_2)$, together with the observation that bad states (states in $\perp\!\!\!\perp$ on each side) are linked through $\mathfrak{R}$.

This ends the proof that $R(\mathcal{S}_1 \wedge \mathcal{S}_2)$ refines $R(\mathcal{S}_1) \wedge R(\mathcal{S}_2)$ through $\mathfrak{R}$. Following exactly the same lines, one can prove the reverse refinement, namely: $R(\mathcal{S}_1) \wedge R(\mathcal{S}_2) \preceq R(\mathcal{S}_1 \wedge \mathcal{S}_2)$. Hence the desired result: $R(\mathcal{S}_1 \wedge \mathcal{S}_2) \equiv R(\mathcal{S}_1) \wedge R(\mathcal{S}_2)$. Note that relation $\mathfrak{R}$ establishes moreover an isomorphism between $R(\mathcal{S}_1 \wedge \mathcal{S}_2)$ and $R(\mathcal{S}_1) \wedge R(\mathcal{S}_2)$. □

Computing the conjunction of two MS via rules (*Glb*1)–(*Glb*3) is polynomial in the size of the arguments. Due to the construction of the region graphs, starting from two MECS $\mathcal{S}_1$ and $\mathcal{S}_2$ computing $R(\mathcal{S}_1) \wedge R(\mathcal{S}_2)$ is exponential. The direct construction of the greatest lower bound by using the timed variants of (*Glb*1) to (*Glb*3) is polynomial and therefore worth adopting for effective methods.

Corollary 3. *For any two* MECS $\mathcal{S}_1$ *and* $\mathcal{S}_2$, $\mathcal{S}_1 \wedge \mathcal{S}_2$ *is the* $\preceq$*-greatest lower bound of* $\mathcal{S}_1$ *and* $\mathcal{S}_2$.

Proof. From the untimed case [9], we deduce: $R(\mathcal{S}_1) \wedge R(\mathcal{S}_2) \preceq R(\mathcal{S}_i)$, for $i = 1, 2$. Thus by Proposition 2, we have: $R(\mathcal{S}_1 \wedge \mathcal{S}_2) \preceq R(\mathcal{S}_i)$. Finally as T is monotonic and because $T(R(\mathcal{S})) \equiv \mathcal{S}$ (Lemma 4 and Corollary 1): $\mathcal{S}_1 \wedge \mathcal{S}_2 \preceq \mathcal{S}_i$.

We now show it is the greatest element under $\mathcal{S}_1$ and $\mathcal{S}_2$. Assume that there exists $\mathcal{S}$ such that $\mathcal{S} \preceq \mathcal{S}_i$. Therefore, by definition of $\preceq$, $R(\mathcal{S}) \preceq R(\mathcal{S}_i)$ which entails $R(\mathcal{S}) \preceq R(\mathcal{S}_1) \wedge R(\mathcal{S}_2)$. Now, we have $\mathcal{S} \equiv T(R(\mathcal{S})) \preceq T(R(\mathcal{S}_1 \wedge \mathcal{S}_2))$ since T is monotonic and by Proposition 2; We then conclude that $\mathcal{S} \preceq \mathcal{S}_1 \wedge \mathcal{S}_2$. □

Finally, according to the above, one can establish that the greatest lower bound yields the intersection of the models.

Theorem 1. *For any two* MECS $\mathcal{S}_1$ *and* $\mathcal{S}_2$, $\mathsf{Mod}(\mathcal{S}_1 \wedge \mathcal{S}_2) = \mathsf{Mod}(\mathcal{S}_1) \cap \mathsf{Mod}(\mathcal{S}_2)$.

Proof. From Corollary 3 we have $\mathcal{S}_1 \wedge \mathcal{S}_2 \preceq \mathcal{S}_i$. Then Corollary 1 entails, $\mathsf{Mod}(\mathcal{S}_1 \wedge \mathcal{S}_2) \subseteq \mathsf{Mod}(\mathcal{S}_i)$. Thus $\mathsf{Mod}(\mathcal{S}_1 \wedge \mathcal{S}_2) \subseteq \mathsf{Mod}(\mathcal{S}_1) \cap \mathsf{Mod}(\mathcal{S}_2)$.

Let $\mathcal{C}$ be a ECA such that $\mathcal{C} \in \mathsf{Mod}(\mathcal{S}_1) \cap \mathsf{Mod}(\mathcal{S}_2)$. By Corollary 2, $\mathcal{C}^* \preceq \mathcal{S}_1$ and $\mathcal{C}^* \preceq \mathcal{S}_2$. By Corollary 3, $\mathcal{C}^* \preceq \mathcal{S}_1 \wedge \mathcal{S}_2$ and by Corollary 2, $\mathcal{C} \models \mathcal{S}_1 \wedge \mathcal{S}_2$. As a result $\mathsf{Mod}(\mathcal{S}_1) \cap \mathsf{Mod}(\mathcal{S}_2) \subseteq \mathsf{Mod}(\mathcal{S}_1 \wedge \mathcal{S}_2)$. □

3.2. Product of MECS

The product of MECS relates to the synchronous parallel composition of models, in the spirit of the synchronization mechanism described in the CSP Process Algebra [25]. For MS, it generalizes the synchronized product of automata $\mathcal{M}_1 \otimes \mathcal{M}_2$ that denotes the intersection of their behaviors (languages).

We first recall the product of MS: Let $\mathcal{R}_1 = (P_1^{\perp\!\!\!\perp}, \lambda_1^0, \Delta_1^m, \Delta_1^M)$ and $\mathcal{R}_2 = (P_2^{\perp\!\!\!\perp}, \lambda_2^0, \Delta_2^m, \Delta_2^M)$ be two MS. The product of $\mathcal{R}_1$ and $\mathcal{R}_2$, denoted by $\mathcal{R}_1 \otimes \mathcal{R}_2$, is the MS $(P^{\perp\!\!\!\perp}, (\lambda_1^0, \lambda_2^0), \Delta_{\otimes}^m, \Delta_{\otimes}^M)$, where $P := P_1 \times P_2$ and $\perp\!\!\!\perp := (\perp\!\!\!\perp_1 \times P_2^{\perp\!\!\!\perp}) \cup (P_1^{\perp\!\!\!\perp} \times \perp\!\!\!\perp_2)$;

as for the greatest lower bound, local inconsistency is inherited from local inconsistency of the components. The transitions are derived from the following rules.

$$\frac{\lambda_1 \overset{a}{\dashrightarrow} \lambda_1' \text{ and } \lambda_2 \overset{a}{\rightsquigarrow} \lambda_2'}{(\lambda_1, \lambda_2) \overset{a}{\dashrightarrow} (\lambda_1', \lambda_2')}\ (Prod1) \qquad \frac{\lambda_1 \overset{a}{\rightsquigarrow} \lambda_1' \text{ and } \lambda_2 \overset{a}{\dashrightarrow} \lambda_2'}{(\lambda_1, \lambda_2) \overset{a}{\dashrightarrow} (\lambda_1', \lambda_2')}\ (Prod2) \qquad \frac{\lambda_1 \xrightarrow{a} \lambda_1' \text{ and } \lambda_2 \xrightarrow{a} \lambda_2'}{(\lambda_1, \lambda_2) \xrightarrow{a} (\lambda_1', \lambda_2')}\ (Prod3)$$

Notice that Rules (*Prod*1)–(*Prod*3) uniformly consider consistent and inconsistent states. As for the greatest lower bound operation, global inconsistency depends only on global inconsistency of the components.

Product of MECS. The product of MECS extends the synchronized product of ECA which consists in synchronizing transitions on action names and in taking the conjunction of the guards of the combined transitions.

Let $\mathcal{S}_1, \mathcal{S}_2$ be two MECS. The modalities for the transitions in $\mathcal{S}_1 \otimes \mathcal{S}_2$ are derived from those proposed in the untimed case, and the labels of the transitions are composed of the intersection of the guards together with the common action. We therefore get the following Rules (*tProd*1), (*tProd*2) and (*tProd*3).

$$\frac{\lambda_1 \overset{g_1,a}{\dashrightarrow} \lambda_1' \text{ and } \lambda_2 \overset{g_2,a}{\rightsquigarrow} \lambda_2'}{(\lambda_1, \lambda_2) \overset{g_1 \wedge g_2,a}{\dashrightarrow} (\lambda_1', \lambda_2')}\ (tProd1) \qquad \frac{\lambda_1 \overset{g_1,a}{\rightsquigarrow} \lambda_1' \text{ and } \lambda_2 \overset{g_2,a}{\dashrightarrow} \lambda_2'}{(\lambda_1, \lambda_2) \overset{g_1 \wedge g_2,a}{\dashrightarrow} (\lambda_1', \lambda_2')}\ (tProd2) \qquad \frac{\lambda_1 \xrightarrow{g_1,a} \lambda_1' \text{ and } \lambda_2 \xrightarrow{g_2,a} \lambda_2'}{(\lambda_1, \lambda_2) \xrightarrow{g_1 \wedge g_2,a} (\lambda_1', \lambda_2')}\ (tProd3)$$

Similarly to Proposition 2 for the greatest lower bound, the product of MECS can be alternatively computed by building the product of the region graphs. This construction however causes an exponential blow-up whereas the direct construction is polynomial.

Proposition 3. *For any two* MECS $\mathcal{S}_1$ *and* $\mathcal{S}_2$, $R(\mathcal{S}_1 \otimes \mathcal{S}_2) \equiv R(\mathcal{S}_1) \otimes R(\mathcal{S}_2)$.

Proof. Similarly to the proof of Proposition 2 the binary relation $\mathfrak{R}$ defined as:

$$\mathfrak{R} = \left\{ \Big(\big((\lambda_1, \theta), (\lambda_2, \theta)\big), \big((\lambda_1, \lambda_2), \theta\big) \Big) \mid \lambda_1 \in Q_1^{\perp\!\!\!\perp}, \lambda_2 \in Q_2^{\perp\!\!\!\perp} \right\}$$

is a witness for $R(\mathcal{S}_1 \otimes \mathcal{S}_2) \equiv R(\mathcal{S}_1) \otimes R(\mathcal{S}_2)$. □

In the untimed setting, it is known [9] that the product is monotonic with respect to refinement, and that a product of models is a model of the product. Those properties extend to the timed case as stated in the following theorem.

Theorem 2 (*Properties of the Product*). *For any* MECS $\mathcal{S}_1, \mathcal{S}_1', \mathcal{S}_2, \mathcal{S}_2'$, *and any* ECA $\mathcal{C}_1, \mathcal{C}_2$,

$$(\mathcal{S}_1 \preceq \mathcal{S}_2 \text{ and } \mathcal{S}_1' \preceq \mathcal{S}_2') \implies \mathcal{S}_1 \otimes \mathcal{S}_1' \preceq \mathcal{S}_2 \otimes \mathcal{S}_2'; \quad \text{and}$$

$$(\mathcal{C}_1 \models \mathcal{S}_1 \text{ and } \mathcal{C}_2 \models \mathcal{S}_2) \implies \mathcal{C}_1 \otimes \mathcal{C}_2 \models \mathcal{S}_1 \otimes \mathcal{S}_2.$$

Proof. Given $\mathcal{S}_1$, $\mathcal{S}_2$, $\mathcal{S}_1'$ and $\mathcal{S}_2'$ MECS, such that: $\mathcal{S}_1 \preceq \mathcal{S}_2$ and $\mathcal{S}_1' \preceq \mathcal{S}_2'$. By Definition 11 this is equivalent to: $R(\mathcal{S}_1) \preceq R(\mathcal{S}_2)$ and $R(\mathcal{S}_1') \preceq R(\mathcal{S}_2')$. As the product of simple modal specifications is monotonic for the modal refinement relation (see [9] for a proof), we have: $R(\mathcal{S}_1) \otimes R(\mathcal{S}_1') \preceq R(\mathcal{S}_2) \otimes R(\mathcal{S}_2')$. According to Proposition 3, this is equivalent to: $R(\mathcal{S}_1 \otimes \mathcal{S}_1') \preceq R(\mathcal{S}_2 \otimes \mathcal{S}_2')$. Thus, by Definition 11: $\mathcal{S}_1 \otimes \mathcal{S}_1' \preceq \mathcal{S}_2 \otimes \mathcal{S}_2'$.

Let us now prove that given two ECA $\mathcal{C}_1, \mathcal{C}_2$, we have: $(\mathcal{C}_1 \models \mathcal{S}_1$ and $\mathcal{C}_2 \models \mathcal{S}_2)$ implies $\mathcal{C}_1 \otimes \mathcal{C}_2 \models \mathcal{S}_1 \otimes \mathcal{S}_2$. Suppose that $\mathcal{C}_1 \models \mathcal{S}_1$ and $\mathcal{C}_2 \models \mathcal{S}_2$ then $\mathcal{C}_1^* \preceq \mathcal{S}_1$ and $\mathcal{C}_2^* \preceq \mathcal{S}_2$. By the first part of the theorem, we have $\mathcal{C}_1^* \otimes \mathcal{C}_2^* \preceq \mathcal{S}_1 \otimes \mathcal{S}_2$. Since $\mathcal{C}_1^* \otimes \mathcal{C}_2^*$ and $(\mathcal{C}_1 \otimes \mathcal{C}_2)^*$ are isomorphic, we have $\mathcal{C}_1^* \otimes \mathcal{C}_2^* \equiv (\mathcal{C}_1 \otimes \mathcal{C}_2)^*$, and we conclude that: $\mathcal{C}_1 \otimes \mathcal{C}_2 \models \mathcal{S}_1 \otimes \mathcal{S}_2$. □

As a consequence, the product operation satisfies the property of *independent implementability*, in the sense of [5]: an implementation of a specification of the form $\mathcal{S}_1 \otimes \mathcal{S}_2$ can be obtained by composing any two implementations of $\mathcal{S}_1$ and $\mathcal{S}_2$ respectively.

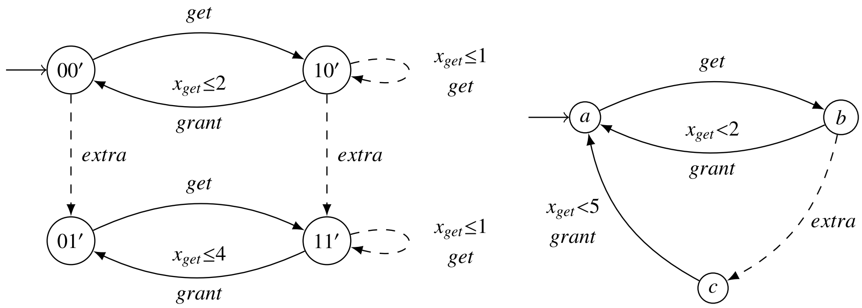
Example 7. The MECS *Acc* in Fig. 2(b) on page 1217 specifies the behavior of an access controller; the access to the resource will be granted for 2 time units after the reception of a "*get*" request. In case of a privileged access providing time, this duration will be extended to 4 time units.

The product *Cl* ⊗ *Acc* is depicted in Fig. 5(a). In the resulting specification, "*extra*" can now only occur *after* a "*get*" request. Timing constraints on the "*grant*" action issued from the access controller are also propagated.

3.3. *Quotient of* MECS

In this section, we define the *quotient operation*. Intuitively, the quotient describes a part of a global specification assuming another part will be realized by some component. We thus consider quotients of *specifications* which is different from the constructions studied in [26] where at least one of the operands is a system.

Inconsistency occurrences in quotients are more complex than in the greatest lower bound and the product cases. Inconsistency can arise from consistent components due to incompatible transition modalities: for example, if in state λ of $\mathcal{R}$ an a-must-transition is required, but if at the same time in the corresponding state λ_1 of $\mathcal{R}_1$ the a-transition is not



(a) The product $Cl \otimes Acc$.

(b) A desired global behavior G.

Fig. 5. The global model $Cl \otimes Acc$ and its specified behavior G.

allowed, there is no reasonable way to build a model whose product with a model of $\mathcal{R}_1$ (in current state λ_1) will enforce an a-transition. This is reflected by Rule (*inconsistency*) below that we explain in detail later.

We start by recalling the quotient operation on untimed modal specifications, then extend it to MECs. Notice that for the untimed case, we revise the definition originally proposed in [9] which indirectly handles inconsistency; originally, the so-called *pseudo-specifications* needed being considered.

Formally, the quotient of the MS $\mathcal{R} = (P^{\perp\!\!\!\perp}, \lambda^0, \Delta^m, \Delta^M)$ by the MS $\mathcal{R}_1 = (P_1^{\perp\!\!\!\perp}, \lambda_1^0, \Delta_1^m, \Delta_1^M)$ is the MS $\mathcal{R} \oslash \mathcal{R}_1 = (P'^{\perp\!\!\!\perp}, (\lambda^0, \lambda_1^0), \Delta_\oslash^m, \Delta_\oslash^M)$, with $P' \subseteq (P \times P_1) \cup \{\top\}$, where $\top$ is a fresh element, and the set $\perp\!\!\!\perp'$ of locally inconsistent states of $\mathcal{R} \oslash \mathcal{R}_1$ contains at least a fresh element $\perp'$.

The rules below achieve the description of the MS $\mathcal{R} \oslash \mathcal{R}_1$. We use $\mathcal{I}$ and $\mathcal{I}_1$ for the set of globally inconsistent states of $\mathcal{R}$ and $\mathcal{R}_1$ respectively. Notation like $\lambda \overset{a}{\dashrightarrow} \mathcal{I}$ therefore means that the a-may-transition from λ leads to an inconsistent state of $\mathcal{R}$. We also use notations $\lambda \overset{a}{\dashrightarrow} P \setminus \mathcal{I}$, $\lambda \xrightarrow{a} \mathcal{I}$, and $\lambda \xrightarrow{a} P \setminus \mathcal{I}$ with the expected meanings, and $\lambda \overset{a}{\dashrightarrow}$ to express that only an a-may-transition is specified from this λ, but no a-must-transition.

We first start with Rules $(I \wedge \neg I1)$-(*top*) which deal with cases where at least one of the local states is inconsistent.

$$\frac{\lambda \in \mathcal{I} \text{ and } \lambda_1 \notin \mathcal{I}_1}{(\lambda, \lambda_1) \in \perp\!\!\!\perp'}\ (I\wedge\neg I1) \qquad \frac{\lambda \in \mathcal{I} \text{ and } \lambda_1 \in \mathcal{I}_1}{(\lambda, \lambda_1) \overset{a}{\dashrightarrow} \top}\ (I\wedge I1)$$

$$\frac{\mathcal{I} \not\ni \lambda \xrightarrow{a} \text{ and } \lambda_1 \in \mathcal{I}_1}{(\lambda, \lambda_1) \overset{a}{\dashrightarrow} \top}\ (\neg Imust\wedge I1) \qquad \frac{\mathcal{I} \not\ni \lambda \overset{a}{\dashrightarrow} \text{ and } \lambda_1 \in \mathcal{I}_1}{(\lambda, \lambda_1) \overset{a}{\dashrightarrow} \top}\ (\neg Imay\wedge I1)$$

$$\frac{}{\top \overset{a}{\dashrightarrow} \top}\ (top)$$

In Rules (*inconsistency*)-(*must*) below, we now assume that both λ and λ_1 are consistent, *i.e.*, $\lambda \notin \mathcal{I}$ and $\lambda_1 \notin \mathcal{I}_1$:

$$\frac{\lambda \xrightarrow{a} \lambda' \text{ and } \lambda_1 \overset{a}{\dashrightarrow} \lambda_1'}{(\lambda, \lambda_1) \in \perp\!\!\!\perp'}\ (inconsistency) \qquad \frac{\lambda \overset{a}{\dashrightarrow} \mathcal{I} \text{ and } \lambda_1 \overset{a}{\rightsquigarrow} P_1 \setminus \mathcal{I}_1}{(\lambda, \lambda_1) \overset{a}{\dashrightarrow} \perp'}\ (mustnot)$$

$$\frac{\lambda \overset{a}{\dashrightarrow} \text{ and } \lambda_1 \overset{a}{\dashrightarrow} \mathcal{I}_1}{(\lambda, \lambda_1) \overset{a}{\dashrightarrow} \top}\ (may1) \qquad \frac{\lambda \overset{a}{\dashrightarrow} \lambda' \notin \mathcal{I} \text{ and } \lambda_1 \overset{a}{\rightsquigarrow} \lambda_1' \notin \mathcal{I}_1}{(\lambda, \lambda_1) \overset{a}{\dashrightarrow} (\lambda', \lambda_1')}\ (may2) \qquad \frac{\lambda \xrightarrow{a} \lambda' \text{ and } \lambda_1 \xrightarrow{a} \lambda_1'}{(\lambda, \lambda_1) \xrightarrow{a} (\lambda', \lambda_1')}\ (must)$$

Proposition 4. *Let $\mathcal{R}$ and $\mathcal{R}1$ be two* MS.

$$\mathcal{R}_1 \otimes \mathcal{R}_2 \preceq \mathcal{R} \Longleftrightarrow \mathcal{R}_2 \preceq \mathcal{R} \oslash \mathcal{R}_1 \tag{2}$$

Proof. $(\Rightarrow)$ Consider the binary relation $\mathfrak{R}$ between the states of $\mathcal{R}_2$ and of the states of $\mathcal{R} \oslash \mathcal{R}_1$ defined by:

$$\mathfrak{R} = \quad \{(\lambda_2, (\lambda, \lambda_1)) \mid ((\lambda_1, \lambda_2), \lambda) \in \rho\} \cup \{(\lambda_2, \top) \mid \lambda_2 \in (P_2 \setminus \mathcal{I}_2)\}$$

where ρ denote the simulation relation allowing to establish that $\mathcal{R}_1 \otimes \mathcal{R}_2 \preceq \mathcal{R}$. We show that $\mathfrak{R}$ is a modal refinement of MS, in the sense of Definition 3. We first show that inconsistent states in $\mathcal{R} \oslash \mathcal{R}_1$ are only related to inconsistent states in $\mathcal{R}_2$.

If (λ, λ_1) is inconsistent in $\mathcal{R} \oslash \mathcal{R}_1$ then there is a must-path to $\perp\!\!\!\perp'$, of length $n \geq 0$.

If $n = 0$ then $(\lambda, \lambda_1) \in \perp\!\!\!\perp'$. We now analyze the rules that may have been applied.

Rule $(I \wedge \neg I_1)$ Suppose that $\lambda \in \mathcal{I}$ and $\lambda_1 \notin \mathcal{I}_1$. Since $\lambda \in \mathcal{I}$, (λ_1, λ_2) is inconsistent in $\mathcal{R}_1 \otimes \mathcal{R}_2$ as $\mathcal{R}_1 \otimes \mathcal{R}_2 \preceq \mathcal{R}$. Moreover, as $\lambda_1 \notin \mathcal{I}_1$, we necessarily have λ_2 inconsistent to have (λ_1, λ_2) inconsistent, by definition of the product operation.

Rule (*inconsistency*) Suppose that $\lambda \notin \mathcal{I}$, $\lambda_1 \notin \mathcal{I}_1$, $\lambda \xrightarrow{a} \lambda'$ and $\lambda_1 \overset{a}{\dashrightarrow} \lambda_1'$. We reason by contradiction; suppose moreover that λ_2 is a consistent state. Then, as $\mathcal{R}_1 \otimes \mathcal{R}_2 \preceq \mathcal{R}$, necessarily we have a must-transition from (λ_1, λ_2). However, by definition of the product, this is impossible if $\lambda_1 \overset{a}{\dashrightarrow} \lambda_1'$. As a result, λ_2 is inconsistent.

Now if $n > 0$, we show that every must-transition leading to a locally inconsistent state in $\mathcal{R} \oslash \mathcal{R}_1$ can be simulated in $\mathcal{R}_2$. This must-transition in $\mathcal{R} \oslash \mathcal{R}_1$ was obtained by Rule (*must*) which indicates us that there is a must-transition in $\mathcal{R}$ and thus also in $\mathcal{R}_1 \otimes \mathcal{R}_2$ as $\mathcal{R}_1 \otimes \mathcal{R}_2 \preceq \mathcal{R}$. By construction, there is also a must-transition in $\mathcal{R}_2$. Hence, there is a must-path of length n from λ_2 in $\mathcal{R}_2$.

Next, the fact that, for consistent states related by $\mathfrak{R}$, every may-transition in $\mathcal{R}_2$ can be simulated by a may-transition of $\mathcal{R} \oslash \mathcal{R}_1$ and that every must-transition in $\mathcal{R} \oslash \mathcal{R}_1$ can also be simulated by a must-transition of $\mathcal{R}_2$ are directly inherited from the proof of a similar proposition in [9] for untimed modal specifications without inconsistent states.

($\Leftarrow$) Consider now the binary relation $\mathfrak{R}$ between the states of $\mathcal{R}_1 \otimes \mathcal{R}_2$ and of the states of $\mathcal{R}$ defined by:

$$\mathfrak{R} = \{((\lambda_1, \lambda_2), \lambda) \mid (\lambda_2, (\lambda, \lambda_1)) \in \rho\}$$

where ρ denote the simulation relation allowing to establish that $\mathcal{R}_2 \preceq \mathcal{R} \oslash \mathcal{R}_1$. We show that $\mathfrak{R}$ is a modal refinement of MS, in the sense of Definition 3. We first show that inconsistent states in $\mathcal{R}$ are only related to inconsistent states in $\mathcal{R}_1 \otimes calR_2$.

Suppose that λ is inconsistent in $\mathcal{R}$, we prove that (λ_1, λ_2) is inconsistent, that is, by definition of the product operation, λ_1 or λ_2 is inconsistent. Suppose that λ_1 is consistent then by Rule $(I \wedge \neg I1)$, $(\lambda, \lambda_1) \in \perp\!\!\!\perp'$ in $\mathcal{R} \oslash \mathcal{R}_1$. Thus, as $\mathcal{R}_2 \preceq \mathcal{R} \oslash \mathcal{R}_1$, λ_2 is inconsistent in $\mathcal{R}_2$.

Next, the fact that, for consistent states related by $\mathfrak{R}$, every may-transition in $\mathcal{R}_1 \otimes \mathcal{R}_2$ can be simulated by a may-transition of $\mathcal{R}$ and that every must-transition in $\mathcal{R}$ can also be simulated by a must-transition of $\mathcal{R}_1 \otimes \mathcal{R}_2$ are directly inherited from the proof of a similar proposition in [9] for untimed modal specifications without inconsistent states. □

Corollary 4. *Let $\mathcal{R}$ and $\mathcal{R}1$ be two* MS*, then for every automaton $\mathcal{M}_2$,*

$$\mathcal{M}_2 \models \mathcal{R} \oslash \mathcal{R}_1 \iff \forall \mathcal{M}_1.\, [\mathcal{M}_1 \models \mathcal{R}_1 \Rightarrow \mathcal{M}_1 \otimes \mathcal{M}_2 \models \mathcal{R}] \tag{3}$$

We now give intuitive explanations for the rules above in particular with respect to Proposition 4. To do so, let $\mathcal{R}^\lambda$ be the MS informally defined as the sub-specification of $\mathcal{R}$ with initial state λ. When explaining a rule involving transitions outgoing from λ in $\mathcal{R}$ and λ_1 in $\mathcal{R}_1$ we will thus speak about models in $\mathcal{R}^\lambda$, $\mathcal{R}_1^{\lambda 1}$ and $\mathcal{R}^\lambda \oslash \mathcal{R}_1^{\lambda 1}$. $\mathcal{R}^\lambda$ and $\mathcal{R}_1^{\lambda 1}$ are just introduced in order to be able to view the local models $\mathcal{R}$ and $\mathcal{R}_1$ from states λ and λ_1. When, say $\lambda \in \mathcal{I}$, we have $\mathsf{Mod}(\mathcal{R}^\lambda) = \emptyset$. Rule $(I \wedge \neg I_1)$ ensures that since there are no models for $\mathcal{R}^\lambda$ and there are models for $\mathcal{R}_1^{\lambda 1}$, there should not be models of $\mathcal{R}^\lambda \oslash \mathcal{R}_1^{\lambda 1}$, otherwise we would not have the right to left implication of Eq. (2) in Proposition 4. For Rules $(\neg Imust \wedge I1)$ and $(\neg Imay \wedge I1)$ (together with Rule (*top*)), since $\mathsf{Mod}(\mathcal{R}_1^{\lambda 1}) = \emptyset$, the right hand side of Eq. (3) is trivially satisfied. Moreover the left hand side of Eq. (3) is also guaranteed whatever the quotient is; we thus set the quotient to be universal, *i.e.* it accepts every model. Rule $(I \wedge I_1)$ together with Rule (*top*), is the case where both $\mathsf{Mod}(\mathcal{R}^\lambda) = \emptyset$ and $\mathsf{Mod}(\mathcal{R}_1^{\lambda 1}) = \emptyset$. In this case, the universal MS that accepts every model can be in the quotient, and this is what is chosen in order to get the greatest such MS, as required by Eq. (2).

We now come to the set of rules where both λ and λ_1 are consistent ($\lambda \notin \mathcal{I}$ and $\lambda_1 \notin \mathcal{I}_1$), which by Lemma 1 amounts to saying that $\mathsf{Mod}(\mathcal{R}^\lambda) \neq \emptyset$ and $\mathsf{Mod}(\mathcal{R}_1^{\lambda 1}) \neq \emptyset$. Rule (*inconsistency*) corresponds to the inability of guaranteeing the a-transition required in $\mathcal{R}^\lambda$ since it may not exist in some models of $\mathcal{R}^{\lambda_1}$. Hence, only an inconsistent MS can be considered so that Eq. (3) holds. Rule (*mustnot*) deals with the case where a is forbidden in $\mathcal{R}^\lambda$, but is authorized or even mandatory in $\mathcal{R}_1^{\lambda 1}$: it should be forbidden in the quotient. In Rule (*may*1), a is not possible from λ_1, and a is not mandatory from λ: it can therefore safely be authorized in the quotient. Rule (*may*2) is very straightforward, as models of the quotient may have an a-transition irrespectively of what is required in $\mathcal{R}_1^{\lambda 1}$. Finally, Rule (*must*) is the simple case of must requirements; notice that we implicitly have $\lambda_1' \notin \mathcal{I}_1$, since by assumption $\lambda_1 \notin \mathcal{I}_1$.

One can easily verify that the conditions of the premises of Rules $(I \wedge \neg I_1)$-(*must*) are exclusive, hence the quotient construction yields a deterministic object. Note also that it is complete.

Quotient of MECS. We first consider the following example illustrating the use of quotient in the interface-based development of a component.

Example 8. A desired global behavior G is depicted in Fig. 5(b) on page 1222. It specifies that any "*get*" request must be fulfilled; the access to the resource is granted for 2 time units and 5 time units in the privileged mode. A model of $G/(Cl \otimes Acc)$ will act as a protocol converter between Cl and the access controller Acc ; the overall system thus obtained will satisfy G.

We fully make use of the quotient of MS by defining the quotient of two MECS as being the quotient of their modal region automata, seen as a MECS (see Remark 2 on page 1216), that is $T(R(\mathcal{S}) \oslash R(\mathcal{S}_1))$.

Proposition 5. *For any* MECS *$\mathcal{S}$, $\mathcal{S}_1$ and $\mathcal{S}_2$,*

$$\mathcal{S}_1 \otimes \mathcal{S}_2 \preceq \mathcal{S} \iff \mathcal{S}_2 \preceq T(R(\mathcal{S}) \oslash R(\mathcal{S}_1)) \tag{4}$$

Proof. ($\Rightarrow$) Suppose that $\mathcal{S}_1 \otimes \mathcal{S}_2 \preceq \mathcal{S}$; then by definition of $\preceq$, $R(\mathcal{S}_1 \otimes \mathcal{S}_2) \preceq R(\mathcal{S})$. By Proposition 3 this is equivalent to $R(\mathcal{S}_1) \otimes R(\mathcal{S}_2) \preceq R(\mathcal{S})$. Thus by Proposition 4, $R(\mathcal{S}_2) \preceq R(\mathcal{S}) \oslash R(\mathcal{S}_1)$. As T is monotonic with respect to $\preceq$, we have: $T(R(\mathcal{S}_2)) \preceq T(R(\mathcal{S}) \oslash R(\mathcal{S}_1))$. Thus, $\mathcal{S}_2 \preceq T(R(\mathcal{S}) \oslash R(\mathcal{S}_1))$.

($\Leftarrow$) Suppose now that $\mathcal{S}_2 \preceq T(R(\mathcal{S}) \oslash R(\mathcal{S}_1))$. By Theorem 2, we have $\mathcal{S}_1 \otimes \mathcal{S}_2 \preceq \mathcal{S}_1 \otimes T(R(\mathcal{S}) \oslash R(\mathcal{S}_1))$, and by Lemma 4, this is equivalent to $\mathcal{S}_1 \otimes \mathcal{S}_2 \preceq T(R(\mathcal{S}_1)) \otimes T(R(\mathcal{S}) \oslash R(\mathcal{S}_1))$.

Additionally, one can easily show that $T(R(\mathcal{S}_1)) \otimes T(R(\mathcal{S}) \oslash R(\mathcal{S}_1)) = T(R(\mathcal{S}_1) \otimes \big(R(\mathcal{S}) \oslash R(\mathcal{S}_1)\big))$, which entails $\mathcal{S}_1 \otimes \mathcal{S}_2 \preceq T(R(\mathcal{S}_1) \otimes \big(R(\mathcal{S}) \oslash R(\mathcal{S}_1)\big))$. Finally, as $R(\mathcal{S}) \oslash R(\mathcal{S}_1) \preceq R(\mathcal{S}) \oslash R(\mathcal{S}_1)$, we have by Proposition 4, $R(\mathcal{S}_1) \otimes \big(R(\mathcal{S}) \oslash R(\mathcal{S}_1)\big) \preceq R(\mathcal{S})$. As a result, $\mathcal{S}_1 \otimes \mathcal{S}_2 \preceq T(R(\mathcal{S}))$, that is $\mathcal{S}_1 \otimes \mathcal{S}_2 \preceq \mathcal{S}$. □

Corollary 5. *For any* MECS $\mathcal{S}$, $\mathcal{S}_1$, $\mathcal{S}_2$, *and any* ECA $\mathcal{C}_2$,

$$\mathcal{C}_2 \models T(R(\mathcal{S}) \oslash R(\mathcal{S}_1)) \iff \forall \mathcal{C}_1.\, [\mathcal{C}_1 \models \mathcal{S}_1, \mathcal{C}_1 \otimes \mathcal{C}_2 \models \mathcal{S}] \tag{5}$$

Proof. The proof here is similar to the one for Corollary 4 and relies on Proposition 5 (see [9]). □

From a practical point of view, the quotient operation enables incremental design: consider a desired global specification $\mathcal{S}$, and the specification $\mathcal{S}_1$ of a preexisting component. By computing $T(R(\mathcal{S}) \oslash R(\mathcal{S}_1))$ and by checking its consistency, one can test whether a component implementing $\mathcal{S}_1$ can be reused in order to realize $\mathcal{S}$, or not. Note that by (5) the specification $T(R(\mathcal{S}) \oslash R(\mathcal{S}_1))$ is maximally permissive in the sense that it characterizes *all* the components $\mathcal{C}_2$ such that for any $\mathcal{C}_1$ implementing $\mathcal{S}_1$, the composed system $\mathcal{C}_1 \otimes \mathcal{C}_2$ implements $\mathcal{S}$.

However defining the quotient of MECS at the level of their modal region automata yields an exponential time complexity caused by the region automaton construction. We now propose an alternative polynomial construction for the quotient of MECS which will provide a correct solution for incremental design but not always the maximally permissive one.

The quotient of a MECS $\mathcal{S} = (Q^{\perp}, \lambda^0, \delta^m, \delta^M)$ by a MECS $\mathcal{S}_1 = (Q_1^{\perp}, \lambda_1^0, \delta_1^m, \delta_1^M)$ is the MECS $\mathcal{S} \oslash \mathcal{S}_1 = ((Q \times Q_1) \cup \{\top\} \cup \{\perp'\}, (\lambda^0, \lambda_1^0), \delta_{\oslash}^m, \delta_{\oslash}^M)$, where transitions follow Rules ($tI \wedge \neg I_1$)-($tmust$) inspired from Rules ($I \wedge \neg I_1$)-($must$). The transitions in the quotient carry the guards inherited from the components, but also additional guards that carefully reflect consistency or inconsistency assumptions on their local source/target states.

Basically, we rephrase the rules for the untimed setting where premises expressing consistency/inconsistency of the local states are transferred to the conclusion as guards like $\mathcal{I}(\lambda)$, $\mathcal{I}(\lambda')$, or their negation. Typically, this is how Rules ($I \wedge \neg I1$) and ($tI \wedge \neg I1$) compare with each other: indeed, since for any region $\theta \subseteq \mathcal{I}(\lambda) \wedge \neg\mathcal{I}(\lambda')$ (if any), the states (λ, θ) and (λ', θ) of $R(\mathcal{S})$ are respectively inconsistent and consistent, Rule ($I \wedge \neg I1$) applies, as reflected by Rule ($tI \wedge \neg I1$).

We apply the principle to all remaining rules and as announced, we get Rules ($tI \wedge \neg I_1$)-($tmust$) for the quotient of MECS.

$$\frac{}{(\lambda, \lambda_1) \xrightarrow{\mathcal{I}(\lambda)\wedge\neg\mathcal{I}(\lambda_1),a} \perp'}\;(tI\wedge\neg I1) \qquad \frac{}{(\lambda, \lambda_1) \overset{\mathcal{I}(\lambda)\wedge\mathcal{I}(\lambda_1),a}{\dashrightarrow} \top}\;(tI\wedge I1)$$

$$\frac{\lambda \xrightarrow{g,a}}{(\lambda, \lambda_1) \overset{\neg\mathcal{I}(\lambda)\wedge\mathcal{I}(\lambda_1)\wedge g,a}{\dashrightarrow} \top}\;(t\neg Imust\wedge I1) \qquad \frac{\lambda \overset{g,a}{\dashrightarrow}}{(\lambda, \lambda_1) \overset{\neg\mathcal{I}(\lambda)\wedge\mathcal{I}(\lambda_1)\wedge g,a}{\dashrightarrow} \top}\;(t\neg Imay\wedge I1)$$

$$\frac{}{\top \overset{\mathbf{true},a}{\dashrightarrow} \top}\;(ttop)$$

We now consider rules for consistent situations. To lighten notation we write $H = \neg\mathcal{I}(\lambda)\wedge\neg\mathcal{I}(\lambda_1)$ for the healthy situations where only consistent regions are involved in both components of the quotient. This guard is systematically added to the transition of the conclusion in every rule below. Also, in order to express that the target of a transition in a premise is consistent, as for the state λ_1' Rule ($tmay1$), we add the guard $\neg\mathcal{I}(\lambda_1')^{\uparrow a}$ (see on page 1216) in the conclusion to ensure that the reached region does not generate inconsistency, in the same spirit as Eq. (1).

$$\frac{\lambda \xrightarrow{g,a} \text{ and } \lambda_1 \overset{g_1,a}{\dashrightarrow}}{(\lambda, \lambda_1) \xrightarrow{H\wedge g\wedge g_1,a} \perp'}\;(tinconsistency) \qquad \frac{\lambda \overset{g,a}{\dashrightarrow} \lambda' \text{ and } \lambda_1 \overset{g_1,a}{\rightsquigarrow} \lambda_1'}{(\lambda, \lambda_1) \overset{H\wedge g\wedge g_1\wedge\mathcal{I}(\lambda')^{\uparrow a}\wedge\neg\mathcal{I}(\lambda_1')^{\uparrow a},a}{\dashrightarrow} \perp'}\;(tmustnot)$$

$$\frac{\lambda \overset{g,a}{\dashrightarrow} \text{ and } \lambda_1 \overset{g_1,a}{\dashrightarrow} \lambda_1'}{(\lambda, \lambda_1) \overset{H\wedge g\wedge g_1\wedge\mathcal{I}(\lambda_1')^{\uparrow a},a}{\dashrightarrow} \top}\;(tmay1) \qquad \frac{\lambda \overset{g,a}{\dashrightarrow} \lambda' \text{ and } \lambda_1 \overset{g_1,a}{\rightsquigarrow} \lambda_1'}{(\lambda, \lambda_1) \overset{H\wedge g\wedge g_1\wedge\neg\mathcal{I}(\lambda')^{\uparrow a}\wedge\neg\mathcal{I}(\lambda_1')^{\uparrow a},a}{\dashrightarrow} (\lambda', \lambda_1')}\;(tmay2)$$

$$\frac{\lambda \xrightarrow{g,a} \lambda' \text{ and } \lambda_1 \xrightarrow{g_1,a} \lambda_1'}{(\lambda, \lambda_1) \xrightarrow{H\wedge g\wedge g_1\wedge\neg\mathcal{I}(\lambda')^{\uparrow a}\wedge\neg\mathcal{I}(\lambda_1')^{\uparrow a},a} (\lambda', \lambda_1')}\;(tmust)$$

This quotient operation for MECS can be used on ECA as the class of *deterministic* ECA can be embedded into the one of MECS; it suffices to type with must every existing transition in the ECA, and to complete it by adding may-transitions to state $\perp'$. Assuming determinism of event-clock automata is not restrictive, since they are known to be determinizable [18]. Observe that the quotient of two event-clock automata embedded into MECS is not an event-clock automaton since *e.g.* Rule ($tmay1$) introduces a may-transition for the top state.

Finally, the quotienting operation yields a deterministic and complete specification. Hence:

Lemma 5. *Modal event-clock specifications are closed under quotient.*

Proof. We prove that the may-transition relation δ^m of $\mathcal{S} \oslash \mathcal{S}_1$ is deterministic and complete, that is, for every state of the quotient, every action a and every clock valuation ν, there is exactly one transition with the guard G such that $\nu \models G$.

This is immediate for state $\top$ of the quotient as by construction the guard associated to each of its outgoing transitions is **true**.

Now consider a state of the form (λ, λ_1) in the quotient. Suppose first that $\nu \models H$. As $\mathcal{S}$ and $\mathcal{S}_1$ are deterministic and complete, there is exactly one transition $(\lambda, g, a, \lambda')$ with $\nu \models g$ in $\mathcal{S}$, and one transition $(\lambda_1, g, a, \lambda_1')$ with $\nu \models g_1$ in $\mathcal{S}_1$. Thus $\nu \models g \wedge g_1$ and guards of the form $g \wedge g_1$ are mutually exclusive. In conclusion, $\nu \models H \wedge g \wedge g_1$ and ν satisfies exactly one of the guards in the conclusion of the Rules ($tinconsistency$), ($tmustnot$), ($tmay_1$), ($tmay_2$) or ($tmust$).

Now if $\nu \models \neg H$ then ν satisfies one of the guards G in the conclusion of Rules ($tI \wedge \neg I1$), ($tI \wedge I1$), ($t\neg Imust \wedge I1$) or ($t\neg Imay \wedge I1$) as $\mathcal{I}(\lambda) \wedge \neg \mathcal{I}(\lambda')$, $\mathcal{I}(\lambda) \wedge \mathcal{I}(\lambda')$ and $\neg \mathcal{I}(\lambda) \wedge \mathcal{I}(\lambda')$ form a partition of $\neg H$. □

As for the product operation, the quotient operations in the timed and untimed settings relate via the region construction as follows.

Proposition 6. *For any two* MECS $\mathcal{S}$ *and* $\mathcal{S}_1$, $R(\mathcal{S} \oslash \mathcal{S}_1) \preceq R(\mathcal{S}) \oslash R(\mathcal{S}_1)$.

Proof. To ease notations, we simply write $\lambda\lambda_1\theta$ (resp. $\lambda\theta, \lambda\theta\lambda_1\theta$) instead of $((\lambda, \lambda_1), \theta\big)$ (resp. $(\lambda, \theta), ((\lambda, \theta), (\lambda_1, \theta))$). when it is clear from the context. Also, we adopt the convention of writing $\mathcal{I}$ and $\mathcal{I}_1$ for the inconsistent states of $R(\mathcal{S})$ and $R(\mathcal{S}_1)$ respectively.

Consider the binary relation $\mathfrak{R}$ between the states of $R(\mathcal{S} \oslash \mathcal{S}_1)$ and of the states $R(\mathcal{S}) \oslash R(\mathcal{S}_1)$ defined by:

$$\begin{aligned}\mathfrak{R} = {} & \{(\lambda\lambda_1\theta, \lambda\theta\lambda_1\theta) \mid \lambda \in Q, \lambda_1 \in Q_1, \theta \in \Theta\} \cup \{(\lambda\lambda_1\theta, \top) \mid \lambda \in Q, \lambda_1 \in Q_1, \theta \in \Theta\} \\ & \cup \{(\perp'\theta, \top) \mid \theta \in \Theta\} \cup \{(\top\theta, \top) \mid \theta \in \Theta\} \cup \{(\perp'\theta, \perp') \mid \theta \in \Theta\}\end{aligned}$$

Note that in the definition of $\mathfrak{R}$, we have used the same symbols $\perp'$ and $\top$ to denote the particular locations of $\mathcal{S} \oslash \mathcal{S}_1$ and the particular states of $R(\mathcal{S}) \oslash R(\mathcal{S}_1)$.

We show that $\mathfrak{R}$ is a modal refinement of MS, in the sense of Definition 3 by proceeding in the following order.

1. We show that inconsistent states in $R(\mathcal{S}) \oslash R(\mathcal{S}_1)$ are only related to inconsistent states in $R(\mathcal{S} \oslash \mathcal{S}_1)$.
2. We show that must-transitions in $R(\mathcal{S}) \oslash R(\mathcal{S}_1)$ are simulated in $R(\mathcal{S} \oslash \mathcal{S}_1)$.
3. We show that may-transitions in $R(\mathcal{S} \oslash \mathcal{S}_1)$ are simulated in $R(\mathcal{S}) \oslash R(\mathcal{S}_1)$.

We now detail the proofs.

1. If $\lambda\theta\lambda_1\theta$ is inconsistent, then there is a must-path to $\perp\!\!\perp'$, of length $n \geq 0$. We reason by induction over n to show that this path is simulated by a must-path from $\lambda\lambda_1\theta$ to $\perp'\theta'$ (for some θ') of length $n + 1$.

   If $n = 0$, it is then sufficient to show that for any $\lambda\theta\lambda_1\theta \in \perp\!\!\perp'$ there is a transition $\lambda\lambda_1\theta \longrightarrow \perp'\theta'$ (for some region θ'). We now analyze the rules that may have been applied.

   Rule ($I \wedge \neg I_1$) Then $\lambda\theta \in I$ and $\lambda_1\theta \notin I_1$, which can be rephrased as $\theta \in I(\lambda) \wedge \neg I(\lambda_1)$. Applying Rule ($tI \wedge \neg I_1$) yields $\lambda\lambda_1\theta \xrightarrow{\theta,a} \perp'\theta_{\downarrow a}$, which concludes the proof.

   Rule (*inconsistency*) Then $\lambda\theta \xrightarrow{\theta'',a} \lambda'\theta''_{\downarrow a}$, $\lambda_1\theta \overset{\theta'',a}{\dashrightarrow} \lambda_1'\theta''_{\downarrow a}$, $\lambda\theta \notin I$, and $\lambda_1\theta \notin I_1$; all in all $\theta \subseteq H$. These transitions are justified by some timed transitions $\lambda \xrightarrow{g,a} \lambda'$ and $\lambda_1 \overset{g_1,a}{\dashrightarrow} \lambda_1'$ in $\mathcal{S} \oslash \mathcal{S}_1$ with $\theta'' \subseteq g \wedge g_1$. By Lemma 3, since by assumption $\theta \subseteq H$, we also have $\theta'' \subseteq H$. Rule ($tinconsistency$) does apply and yields $\lambda\lambda_1\theta \xrightarrow{\theta'',a} \perp'\theta''_{\downarrow a}$, which concludes the proof.

   If $n > 0$, then Rule ($must$) is applied at the first step yielding $\lambda\theta\lambda_1\theta \xrightarrow{\theta'',a} \lambda'\theta''_{\downarrow a}\lambda_1'\theta''_{\downarrow a}$, with the hypothesis that $\lambda\theta \notin I$, and $\lambda_1\theta \notin I_1$, i.e. $\theta \in H$. Moreover, there are transitions $\lambda\theta \xrightarrow{\theta'',a} \lambda'\theta''_{\downarrow a}$ and $\lambda_1\theta \xrightarrow{\theta'',a} \lambda_1'\theta''_{\downarrow a}$, which arise from two timed transitions $\lambda \xrightarrow{g,a} \lambda'$ and $\lambda_1 \xrightarrow{g_1,a} \lambda_1'$; then $\theta'' \subseteq g \wedge g_1$. By Lemma 3, $\theta'' \subseteq H$. We can then apply Rule ($tmust$) to $\lambda\lambda_1\theta$ to get $\lambda\lambda_1\theta \xrightarrow{\theta'',a} \lambda'\lambda_1'\theta''_{\downarrow a}$, where $(\lambda'\lambda_1'\theta''_{\downarrow a}, \lambda'\theta''_{\downarrow a}\lambda_1'\theta''_{\downarrow a}) \in \mathfrak{R}$. Since $\lambda'\theta''_{\downarrow a}\lambda_1'\theta''_{\downarrow a}$ has a path to $\perp\!\!\perp'$ of length $n - 1$, by the induction hypothesis, $\lambda'\lambda_1'\theta''_{\downarrow a}$ has a path of length n to $\perp'\theta'$ for some θ', and we are done.
2. Without loss of generality, we can assume that $\lambda\theta\lambda_1\theta$ is consistent (by the above), and since we consider a must-transition from $\lambda\theta\lambda_1\theta$, only Rule ($must$) can apply in which case we necessarily assume $\lambda\theta \notin I$ and $\lambda_1\theta \notin I_1$, that is $\theta \subseteq H$. Moreover, the must-transition from $\lambda\theta\lambda_1\theta$ is of the form $\lambda\theta\lambda_1\theta \xrightarrow{\theta'',a} \lambda'\theta'\lambda_1'\theta_1'$, with $\lambda'\theta'\lambda_1'\theta_1'$ consistent. Therefore $\lambda\theta \xrightarrow{\theta'',a} \lambda'\theta'$ and $\lambda_1\theta \xrightarrow{\theta'',a} \lambda_1'\theta_1'$, which shows that $\theta' = \theta_1' = \theta''_{\downarrow a}$. There must exist $\lambda \xrightarrow{g,a} \lambda'$ in $\mathcal{S}$ and $\lambda_1 \xrightarrow{g_1,a} \lambda_1'$ in $\mathcal{S}_1$, such that $\theta'' \subseteq g \wedge g_1$.

   Since $\theta \subseteq H$, by Lemma 3, so is θ'' which shows that Rule ($tmust$) applies to $\lambda\lambda_1\theta$ to obtain $\lambda\lambda_1\theta \xrightarrow{\theta'',a} \lambda'\lambda_1'\theta''_{\downarrow a}$, and we are done.

3. We now consider may-transitions from $\lambda\lambda_1\theta$. They can arise from Rules $(tI \wedge I_1)$, $(t\neg Imust \wedge I_1)$, $(t\neg Imay \wedge I_1)$, $(tmustnot)$, $(tmay1)$, $(tmay2)$, or $(tmust)$ (as it is also a may-transition). We analyze each case.

   Rule $(tI \wedge I_1)$ Then the may-transition is $\lambda\lambda_1\theta \overset{\theta'',a}{\dashrightarrow} \top\theta''_{\downarrow a}$ where $\theta'' \subseteq I(\lambda) \wedge I(\lambda_1)$. By Lemma 3, we also have $\theta \subseteq I(\lambda) \wedge I(\lambda_1)$ so that Rule $(I \wedge I_1)$ applies to $\lambda\theta\lambda_1\theta$, yielding $\lambda\theta\lambda_1\theta \overset{\theta'',a}{\dashrightarrow} \top$. This concludes the proof.

   Rule $(t\neg Imust \wedge I_1)$ Then the may-transition is $\lambda\lambda_1\theta \overset{\theta'',a}{\dashrightarrow} \top\theta''_{\downarrow a}$ where $\theta'' \subseteq \neg I(\lambda) \wedge I(\lambda_1)$. By Lemma 3, we also have $\theta \subseteq I_1(\lambda)$. If $\theta \subseteq \neg I(\lambda)$, then Rule $(\neg Imust \wedge I_1)$ applies to $\lambda\theta\lambda_1\theta$ and we are done. Otherwise $\theta \subseteq I(\lambda)$, and Rule $(I \wedge I_1)$ applies, which concludes as well since $\top$ in $R(\mathcal{S}) \oslash R(\mathcal{S}_1)$ simulates everything.

   Rule $(t\neg Imay \wedge I_1)$ The reasoning is similar to the previous case.

   Rule $(tmustnot)$ Then the may-transition is $\lambda\lambda_1\theta \overset{\theta'',a}{\dashrightarrow} \bot'\theta''_{\downarrow a}$ with some $\lambda \overset{g,a}{\dashrightarrow} \lambda'$ in $\mathcal{S}$ and $\lambda_1 \overset{g_1,a}{\rightsquigarrow} \lambda'_1$ in $\mathcal{S}_1$, and $\theta'' \subseteq H \wedge g \wedge g_1 \wedge I(\lambda')^{\uparrow a} \wedge \neg I(\lambda'_1)^{\uparrow a}$. Therefore $\lambda\theta \overset{\theta'',a}{\dashrightarrow} \lambda'\theta''_{\downarrow a}$, $\lambda_1\theta \overset{\theta'',a}{\rightsquigarrow} \lambda'_1\theta''_{\downarrow a}$, $\lambda'\theta''_{\downarrow a} \in I$, and $\lambda'_1\theta''_{\downarrow a} \notin I_1$. If $\theta \notin I(\lambda)$ and $\theta \notin I(\lambda_1)$, that is $\lambda\theta \notin I$ and $\lambda_1\theta \notin I_1$, we can apply Rule $(mustnot)$ to $\lambda\theta\lambda_1\theta$ and obtain $\lambda\theta\lambda_1\theta \overset{\theta'',a}{\dashrightarrow} \bot'$ to conclude. Otherwise one of the rules $(I \wedge I_1)$, $(\neg Imust \wedge I_1)$, or $(\neg Imay \wedge I_1)$ applies, leading $\lambda\theta\lambda_1\theta$ to $\top$ which also concludes the proof.

   Rule $(tmay1)$ Then the may-transition is $\lambda\lambda_1\theta \overset{\theta'',a}{\dashrightarrow} \top\theta''_{\downarrow a}$ with $\lambda \overset{g,a}{\dashrightarrow}$ in $\mathcal{S}$, $\lambda_1 \overset{g_1,a}{\dashrightarrow} \lambda'_1$ in $\mathcal{S}_1$, and $\theta'' \subseteq H \wedge g \wedge g_1 \wedge I(\lambda'_1)^{\uparrow a}$. In particular, $\theta''_{\downarrow a} \notin I(\lambda_1)$. If $\theta \notin I(\lambda)$ and $\theta \notin I(\lambda_1)$, we can apply Rule $(may1)$ to $\lambda\theta\lambda_1\theta$, and we conclude. Otherwise one of the rules $(I \wedge I_1)$, $(\neg Imust \wedge I_1)$, or $(\neg Imay \wedge I_1)$ applies, leading $\lambda\theta\lambda_1\theta$ to $\top$ which also concludes the proof.

   Rule $(tmay2)$ Then the transition is $\lambda\lambda_1\theta \overset{\theta'',a}{\dashrightarrow} \lambda'\lambda'_1\theta''_{\downarrow a}$. A reasoning very close to the case $(tmay1)$ applies. We leave it to the reader.

   Rule $(tmust)$ Then the transition is $\lambda\lambda_1\theta \xrightarrow{\theta'',a} \lambda'\lambda'_1\theta''_{\downarrow a}$ with $\lambda \xrightarrow{g,a} \lambda'$, $\lambda_1 \xrightarrow{g_1,a} \lambda'_1$, and $\theta'' \subseteq H \wedge g \wedge g_1$. Again, if $\theta \notin I(\lambda)$ and $\theta \notin I(\lambda_1)$, we can apply Rule $(must)$ to $\lambda\theta\lambda_1\theta$, yielding $\lambda\theta\lambda_1\theta \xrightarrow{\theta'',a} \lambda'\theta''_{\downarrow a}\lambda'_1\theta''_{\downarrow a}$ and which concludes. Otherwise, one of the rules $(I \wedge I_1)$, $(\neg Imust \wedge I_1)$, or $(\neg Imay \wedge I_1)$ applies, leading $\lambda\theta\lambda_1\theta$ to $\top$ which also concludes the proof. □

The correctness of the quotient construction is stated by the following.

Theorem 3 (*Correctness of the Quotient*). *For any* MECS $\mathcal{S}$ *and* $\mathcal{S}_1$,

$$\mathcal{S}_1 \otimes (\mathcal{S} \oslash \mathcal{S}_1) \preceq \mathcal{S}. \tag{6}$$

Proof. From Proposition 6, we have $R(\mathcal{S} \oslash \mathcal{S}_1) \preceq R(\mathcal{S}) \oslash R(\mathcal{S}_1)$. By Proposition 4, we then deduce $R(\mathcal{S}_1) \otimes R(\mathcal{S} \oslash \mathcal{S}_1) \preceq R(\mathcal{S})$. Then by Proposition 3, this implies that $R(\mathcal{S}_1 \otimes (\mathcal{S} \oslash \mathcal{S}_1)) \preceq R(\mathcal{S})$. Thus, by definition of $\preceq$, we have $\mathcal{S}_1 \otimes (\mathcal{S} \oslash \mathcal{S}_1) \preceq \mathcal{S}$. □

Corollary 6 (*Properties of the Quotient*). *For any* MECS $\mathcal{S}$, $\mathcal{S}_1$ *and* $\mathcal{S}_2$,

$$\mathcal{S}_2 \preceq \mathcal{S} \oslash \mathcal{S}_1 \Longrightarrow \mathcal{S}_1 \otimes \mathcal{S}_2 \preceq \mathcal{S}. \tag{7}$$

For any MECS $\mathcal{S}$, $\mathcal{S}_1$, $\mathcal{S}_2$, *and any* ECA $\mathcal{C}_2$,

$$\mathcal{C}_2 \models \mathcal{S} \oslash \mathcal{S}_1 \Longrightarrow \forall \mathcal{C}_1.\,[\mathcal{C}_1 \models \mathcal{S}_1, \mathcal{C}_1 \otimes \mathcal{C}_2 \models \mathcal{S}] \tag{8}$$

Proof. By Definition 11, $\mathcal{S}_2 \preceq \mathcal{S} \oslash \mathcal{S}_1$ if and only if $R(\mathcal{S}_2) \preceq R(\mathcal{S} \oslash \mathcal{S}_1)$. By Proposition 6, this implies that $R(\mathcal{S}_2) \preceq R(\mathcal{S}) \oslash R(\mathcal{S}_1)$. The latter is equivalent to $R(\mathcal{S}_1) \otimes R(\mathcal{S}_2) \preceq R(\mathcal{S})$ by Eq. (2), or equivalently $R(\mathcal{S}_1 \otimes \mathcal{S}_2) \preceq R(\mathcal{S})$ by Proposition 3, which finally is equivalent to $\mathcal{S}_1 \otimes \mathcal{S}_2 \preceq \mathcal{S}$ by Definition 11.

Suppose that $\mathcal{C}_2 \models \mathcal{S} \oslash \mathcal{S}_1$, by Corollary 2, this is equivalent to $\mathcal{C}_2^* \preceq \mathcal{S} \oslash \mathcal{S}_1$. Then by Proposition 6, we have $\mathcal{C}_2^* \otimes \mathcal{S}_1 \preceq \mathcal{S}$. Similarly if $\mathcal{C}_1 \models \mathcal{S}_1$ then $\mathcal{C}_1^* \preceq \mathcal{S}_1$. By Theorem 2 we then have $\mathcal{C}_1^* \otimes \mathcal{C}_2^* \preceq \mathcal{S}_1 \otimes \mathcal{C}_2^*$. By transitivity of $\preceq$ we deduce that $\mathcal{C}_1^* \otimes \mathcal{C}_2^* \preceq \mathcal{S}$, that is $\mathcal{C}_1 \otimes \mathcal{C}_2 \models \mathcal{S}$. □

Example 9. We now reconsider Example 8. The MECS $G \oslash (Cl \otimes Acc)$ is represented in Fig. 6. Not surprisingly, the state $c/11'$ is inconsistent. This is because, in the state $11'$ in Fig. 5(a), the resource is granted for 4 units of time whereas in the state c of the desired behavior G in Fig. 5(b), it must be granted for 5 units of time. To avoid this inconsistency, the transition "*extra*" from state $b/10'$ to $c/11'$ will not be implemented in any model of $G \oslash (Cl \otimes Acc)$. Thus, the protocol converter will disallow the privileged mode.

Consider the MECS in Fig. 7, this counter-example shows that the inverse refinement of Proposition 6 is not true in general: for the two MECS $\mathcal{S}$ and $\mathcal{S}_1$ of Fig. 7, we have $R(\mathcal{S} \oslash \mathcal{S}_1) \not\preceq R(\mathcal{S}) \oslash R(\mathcal{S}_1)$. As a consequence, $\mathcal{S} \oslash \mathcal{S}_1$ is not the maximal solution $\mathcal{S}_X$ of the equation $\mathcal{S}_1 \otimes \mathcal{S}_X \preceq \mathcal{S}$.

Observe that $\mathcal{S}_1$ in Fig. 7(b) has a very special form: by letting time elapse from λ_1, the transition leading to a locally inconsistent state may not be firable. By imposing I-stability (see page 1217), our quotient construction is the $\preceq$-maximal solution of the equation $\mathcal{S}_1 \otimes X \preceq \mathcal{S}$, hence it is complete in the following sense.

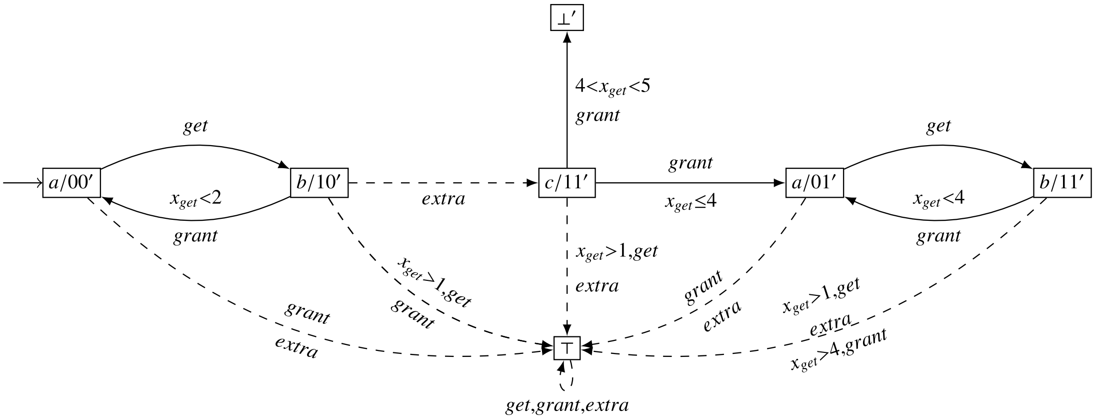
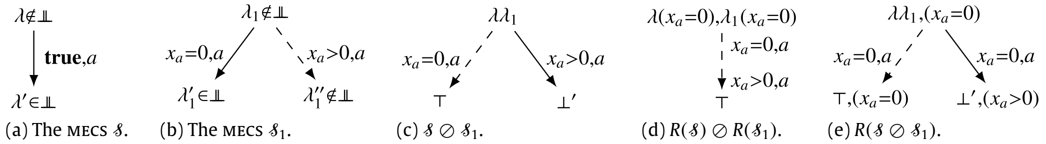


Fig. 6. The quotient $G \oslash (Cl \otimes Acc)$.



(a) The MECS $\mathcal{S}$. (b) The MECS $\mathcal{S}_1$. (c) $\mathcal{S} \oslash \mathcal{S}_1$. (d) $R(\mathcal{S}) \oslash R(\mathcal{S}_1)$. (e) $R(\mathcal{S} \oslash \mathcal{S}_1)$.

Fig. 7. Example showing that $R(\mathcal{S} \oslash \mathcal{S}_1) \not\preceq R(\mathcal{S}) \oslash R(\mathcal{S}_1)$ in general.

Theorem 4 (*Completeness of the Quotient Construction*)**.** *For any* MECS $\mathcal{S}$*,* $\mathcal{S}_1$ *and* $\mathcal{S}_2$ *such that* $\mathcal{S}$ *and* $\mathcal{S}_1$ *are I-stable,*

$$\mathcal{S}_1 \otimes \mathcal{S}_2 \preceq \mathcal{S} \Longrightarrow \mathcal{S}_2 \preceq \mathcal{S} \oslash \mathcal{S}_1 \tag{9}$$

The rest of the section is dedicated to the proof of Theorem 4. We fix two I-stable MECS $\mathcal{S}$ and $\mathcal{S}_1$.

We first need the following lemma that expresses what is gained by assuming I-stable MECS; notations are inherited from Proposition 6.

Lemma 6. *For any region* $\theta \in \Theta$*,* $\theta \in I(\lambda\lambda_1)$ *if, and only if,* $\lambda\theta\lambda_1\theta$ *is globally inconsistent in* $R(\mathcal{S} \oslash \mathcal{S}_1)$*.*

Proof. Notice that the direction $\Leftarrow$) always holds and is already proved in Point 1 of Proposition 6. To prove the reverse direction $\Rightarrow$) we need the I-stability assumption. Let $\theta \in I(\lambda\lambda_1)$. Then there exists a must-path from $\lambda\lambda_1\theta$ to $\perp'\theta'$ for some θ' and this must-path has length at least 1 according to our way of constructing the quotient. We show by induction over the length $k \geq 1$ of this path that there exists a must-path of length $k - 1$ from $\lambda\theta\lambda_1\theta$ to $\perp\!\!\!\perp'$.

If $k = 1$, that is $\lambda\lambda_1\theta \xrightarrow{\theta'',a} \perp'\theta''_{\downarrow a}$ for some $\theta'' \in \tau(\theta)$, we must consider two cases depending on which of the Rules ($tI \wedge \neg I_1$) and (*tinconsistency*) has been applied.

Rule ($tI \wedge \neg I_1$) Then $\theta'' \in I(\lambda) \wedge \neg I(\lambda_1) \wedge \tau(\theta)$. By definition of $I(\lambda)$, $\theta'' \in I(\lambda)$ implies $\theta \in I(\lambda)$. Moreover, because $\mathcal{S}_1$ is I-stable, $\theta'' \notin I(\lambda_1)$ implies $\theta \notin I(\lambda_1)$. Therefore $\lambda\theta \in I$ and $\lambda_1\theta \notin I_1$ and we can apply Rule ($I \wedge \neg I_1$), yielding $\lambda\theta\lambda_1\theta \in \perp\!\!\!\perp'$. Which concludes the proof.

Rule (*tinconsistency*) Then $\theta'' \in H \wedge g \wedge g_1$. By I-stability of both $\mathcal{S}$ and $\mathcal{S}_1$, $\theta'' \in H$ guarantees that $\theta \in H$ as well; therefore $\lambda\theta \notin I$ and $\lambda_1\theta \notin I_1$. Now, one easily show that Rule (*inconsistency*) applies to $\lambda\theta\lambda_1\theta$ so that $\lambda\theta\lambda_1\theta \in \perp\!\!\!\perp$, which concludes the proof.

Otherwise the must-path from $\lambda\lambda_1\theta$ to $\perp'\theta'$ is $k > 1$. Necessarily this path is justified by the application of Rule (*tmust*) in its first step: $\lambda\lambda_1\theta \xrightarrow{\theta'',a} \lambda'\lambda'_1\theta''_{\downarrow a}$ for some $\theta'' \in \tau(\theta) \wedge H \wedge g \wedge g_1 \wedge \neg\mathcal{I}(\lambda')^{\uparrow a} \wedge \neg\mathcal{I}(\lambda'_1)^{\uparrow a}$. Again, by I-stability of $\mathcal{S}$ and $\mathcal{S}_1$, we infer that $\theta \in H$ so that $\lambda\theta \notin I$ and $\lambda_1\theta \notin I_1$, and because $\theta'' \in \neg\mathcal{I}(\lambda')^{\uparrow a} \wedge \neg\mathcal{I}(\lambda'_1)^{\uparrow a}$, the reached states $\lambda'\theta''_{\downarrow a}$ and $\lambda'_1\theta''_{\downarrow a}$ are also consistent. Rule (*mustapplies*) and yields $\lambda\theta\lambda_1\theta \xrightarrow{\theta'',a} \lambda'\theta''_{\downarrow a}\lambda'_1\theta''_{\downarrow a}$. Because by assumption there is a must-path of length $k - 1$ from $\lambda'\lambda'_1\theta''_{\downarrow a}$ to $\perp'\theta'$, we use the induction hypothesis to exhibit a must-path of length $k - 2$ from $\lambda'\theta''_{\downarrow a}\lambda'_1\theta''_{\downarrow a}$ to $\perp\!\!\!\perp'$ which together with the step $\lambda\theta\lambda_1\theta \xrightarrow{\theta'',a} \lambda'\theta''_{\downarrow a}\lambda'_1\theta''_{\downarrow a}$ shows a must-path of length $(k - 1)$ from $\lambda\theta\lambda_1\theta$ to $\perp\!\!\!\perp'$. This concludes the proof of the induction step. □

The exact characterization of inconsistent states of the quotient given by Lemma 6 is one of the key points to prove the following.

Proposition 7. *For any two I-stable* MECS $\mathcal{S}$ *and* $\mathcal{S}_1$*,* $R(\mathcal{S}) \oslash R(\mathcal{S}_1) \preceq R(\mathcal{S} \oslash \mathcal{S}_1)$*.*

Before establishing Proposition 7, we explain how to exploit it to derive the result of Theorem 4: Assume $\mathcal{S}_1 \otimes \mathcal{S}_2 \preceq \mathcal{S}$. Then by Definition 11, $R(\mathcal{S}_1 \otimes \mathcal{S}_2) \preceq R(\mathcal{S})$. According to Proposition 3, this is equivalent to $R(\mathcal{S}_1) \otimes R(\mathcal{S}_2) \preceq R(\mathcal{S})$. By Proposition 4 Eq. (2), this is equivalent to $R(\mathcal{S}_2) \preceq R(\mathcal{S}) \oslash R(\mathcal{S}_1)$. Thanks to the to-be-proved Proposition 7 above and Proposition 6, we know that $R(\mathcal{S}) \oslash R(\mathcal{S}_1) \equiv R(\mathcal{S} \oslash \mathcal{S}_1)$. By Definition 11, from $R(\mathcal{S}_2) \preceq R(\mathcal{S} \oslash \mathcal{S}_1)$, we conclude that $\mathcal{S}_2 \preceq \mathcal{S} \oslash \mathcal{S}_1$.

We now show Proposition 7 by exhibiting a modal refinement of the MS $R(\mathcal{S} \oslash \mathcal{S}_1)$ by the MS $R(\mathcal{S}) \oslash R(\mathcal{S}_1)$. Consider the following binary relation $\mathfrak{R}$ between the states of $R(\mathcal{S}) \oslash R(\mathcal{S}_1)$ and the states of $R(\mathcal{S} \oslash \mathcal{S}_1)$:

$$\mathfrak{R} = \{(\lambda\theta\lambda_1\theta, \lambda\lambda_1\theta) \mid \lambda \in Q, \lambda_1 \in Q_1, \theta \in \Theta\} \cup \{(\top, \top\theta) \mid \theta \in \Theta\} \cup \{(\bot', \bot'\theta) \mid \theta \in \Theta\}$$

According to Definition 3, $\mathfrak{R}$ is a modal refinement whenever the following properties hold.

1. Inconsistent states in $R(\mathcal{S} \oslash \mathcal{S}_1)$ are only related to inconsistent states in $R(\mathcal{S}) \oslash R(\mathcal{S}_1)$.
2. Must-transitions in $R(\mathcal{S} \oslash \mathcal{S}_1)$ are simulated $R(\mathcal{S}) \oslash R(\mathcal{S}_1)$.
3. May-transitions in $R(\mathcal{S}) \oslash R(\mathcal{S}_1)$ are simulated in $R(\mathcal{S} \oslash \mathcal{S}_1)$.

Point 1 is immediate by Lemma 6.

Regarding Point 2, assume a must-transition $\lambda\lambda_1\theta \xrightarrow{\theta'',a} \lambda'\lambda'_1\theta''_{\downarrow a}$ in $R(\mathcal{S} \oslash \mathcal{S}_1)$, where $\theta'' \in \tau(\theta)$. As in the proof of Proposition 6, we can assume without loss of generality that $\lambda\lambda_1\theta$ is consistent – otherwise apply Point 1. This must-transition arises from applying Rule (*tmust*) which requires that $\theta'' \in H \wedge g \wedge g_1 \wedge \neg\mathcal{I}(\lambda')^{\uparrow a} \wedge \neg\mathcal{I}(\lambda'_1)^{\uparrow a}$. By I-stability, we also have $\theta \in H$ and henceforth, $\lambda\theta$ and $\lambda_1\theta$ are consistent, as the states $\lambda'\theta''_{\downarrow a}$ and $\lambda'_1\theta''_{\downarrow a}$ since $\theta'' \in \neg\mathcal{I}(\lambda')^{\uparrow a} \wedge \neg\mathcal{I}(\lambda'_1)^{\uparrow a}$. We can therefore apply Rule (*must*) to $\lambda\theta\lambda_1\theta$, to obtain the good candidate must-transition $\lambda\theta\lambda_1\theta \xrightarrow{\theta'',a} \lambda'\theta''_{\downarrow a}\lambda'_1\theta''_{\downarrow a}$.

We finally focus on may-transitions in $R(\mathcal{S}) \oslash R(\mathcal{S}_1)$ and simulate them in $R(\mathcal{S} \oslash \mathcal{S}_1)$. The proof is routine with a recurrent use of the I-stability assumption to ensure that if a time-successor θ'' of θ is in H, so is θ; we omit this tedious but easy proof. This concludes the proof of Proposition 7.

Notice that our direct construction for the quotient has nice properties: it is in essence based on a cartesian product, hence it yields a polynomial-time algorithm, as opposed to the exponential blow-up caused by the original definition relying on an indirect construction via the region graphs. Additionally, remark that quotienting MECS, while abstracting from a particular choice of implementations, amounts to quotienting logical statements denoted by the very MECS. In the untimed setting, the quotient operation is a particular case of the exponential construction introduced in [27] for arbitrary mu-calculus statements. However, in our setting we can take advantage of the restricted fragment of the logic captured by the modal specifications, namely the conjunctive nu-calculus as shown in [14], in order to design an ad hoc polynomial-time construction. The present contribution suggests a similar situation for a timed extension of the mu-calculus.

4. Discussion

In this section we address several directions for short and medium term future work. We first complete a discussion about the complexity of the constructions we have considered. Then we analyze the impact of the I-stability assumption and we consider two extensions of the presented work that overcome some of the current limitations: we first consider the support of local alphabets; modal event-clock specifications are no longer supposed to be defined over a unique alphabet global to the system, but on local alphabets. Secondly, we consider specifications that are more expressive than modal event-clock specifications and investigate how to generalize the theory to specifications with arbitrary resets.

4.1. Complexity issues

For each of the three operations we have considered, namely the greatest lower bound (Section 3.1), the product (Section 3.2), and the quotient (Section 3.3), we established polytime constructions in the size of the state space. Actually, a closer look at this result exhibits a worst-case exponential blow-up in the number of transitions: the subtlety lies in the way the formulas of the form $\mathcal{I}(\lambda)$, which denote the sets of inconsistent regions, are handled. In particular, the quotient construction raises up the need to compute guards of the form $\mathcal{I}(\lambda')^{\uparrow a}$, as in, *e.g.* Rule (*tmustnot*). This computation is polytime whenever the set $\mathcal{I}(\lambda)$ is given in a disjunctive normal form (*dnf*). Maintaining formulas $\mathcal{I}(\lambda)$ in dnf has a cost as we explain in detail here.

The direct computation of the formulas $\mathcal{I}(-)$'s yields formulas in dnf by construction (see Eq. (1)), provided the guards of the MECS are conjuncts of atomic formulas. This is an issue for the quotient construction as discussed below. Notice that for the cases of the greatest lower bound and the product, computing the formulas $\mathcal{I}(-)$ from scratch can be avoided: indeed, one easily verifies that for every states λ_1 and λ_2 of some MECS $\mathcal{S}_1$ and $\mathcal{S}_2$ respectively, the formula $\mathcal{I}((\lambda_1, \lambda_2))$ is equal to $\mathcal{I}(\lambda_1) \vee \mathcal{I}(\lambda_2)$, where (λ_1, λ_2) is the compound state in either $\mathcal{S}_1 \wedge \mathcal{S}_2$ or $\mathcal{S}_1 \otimes \mathcal{S}_2$. The dnf assumption therefore propagates for free in these cases.

Regarding the quotient construction, as stated by Lemma 6, there is no obvious and cheap way to obtain the new formulas $\mathcal{I}(-)$ from the local formulas; they therefore need being computed from scratch from Eq. (1). However, the guards of the resulting MECS, as described by Rules ($tI \wedge I_1$)-(*tmust*), are not conjuncts of atomic formulas anymore (as opposed to the cases of the greatest lower bound and the product). In order to run the algorithm yielded by Eq. (1), one must first convert the guards into dnf, and this is where worst-case exponential blow-up may occur.

4.2. About the I-stability assumption

The efficient quotient construction described by the Rules $(tI \wedge I_1)$-$(tmust)$ is a key aspect of the theory. As explained by Theorem 4, this correct construction (Theorem 3) is exact as long as the involved MECS are I-stable. Ensuring I-stability is therefore an issue, and it is important to determine its closure properties with respect to operators we use over MECS. Concerning the binary operations of greatest lower bound (Section 3.1) and product (Section 3.2), it is immediate to see that I-stability is preserved. Indeed, whenever $\mathcal{S}_1$ and $\mathcal{S}_2$ are I-stable, so are $\mathcal{S}_1 \wedge \mathcal{S}_2$ and $\mathcal{S}_1 \otimes \mathcal{S}_2$. Indeed, since the set of inconsistent regions of a compound state (λ_1, λ_2) is the union of the sets of the inconsistent regions for each the local states, assuming $\mathcal{I}(\lambda_1)$ and $\mathcal{I}(\lambda_2)$ closed under time elapsing, entails that it is also the case for $\mathcal{I}((\lambda_1, \lambda_2))$. Regarding the quotient, the situation is different as exemplified by the I-stable MECS $Cl \otimes Acc$ and G of Fig. 5(b), whose quotient in Fig. 6 is not I-stable because $\mathcal{I}(c/11') = (4 < x_{get} < 5)$ is not closed under time elapsing.

This is unfortunate in our attempt to developing an adequate theory for the incremental design of component-based systems, as iterative quotienting is central. An interesting perspective would be to find a way to "I-stabilize" MECS by under-approximation, say by defining and computing efficiently, if it exists, the greatest I-stable refinement. Importantly, this under-approximation would need being compared to the under-approximation obtained by quotienting non-I-stable MECS (see Theorem 3).

4.3. Dissimilar alphabets

In Section 2 we fixed a finite set Σ of actions and then supposed that every modal event-clock specification was defined over Σ. This assumption can be unrealistic when designing a system. Indeed large systems are composed of many subsystems, each of them described via an interface possessing its own local alphabet of actions. Moreover it is usual to enrich a refined specification by enlarging its alphabet, entailing ultimately implementations defined over a superset of the actions of their initial interface. In order to handle these two scenarios we develop an approach relying on an alphabet equalization operation in which modalities play a central role.

Two natural kinds of alphabet equalization operations for MECS can be defined: weak and strong extensions which consist in adding to every state may- or must-self loops, respectively, labeled by the extra actions.

Definition 13 (*Weak and Strong Extensions*)**.** Let $\mathcal{S} = (Q^{\perp\!\!\perp}, \lambda^0, \delta^m, \delta^M)$ be a MECS over the alphabet Σ and let $\Sigma' \supseteq \Sigma$:

1. The weak extension of $\mathcal{S}$ to Σ' is the MECS $\mathcal{S}_{\Uparrow\Sigma'} = (Q^{\perp\!\!\perp}, \lambda^0, \delta^{m\prime}, \delta^M)$ such that $\delta^m \subseteq \delta^{m\prime}$ and for all $a \in \Sigma' \setminus \Sigma$ and $q \in Q^{\perp\!\!\perp}$, $(q, true, a, q) \in \delta^{m\prime}$;
2. The strong extension of $\mathcal{S}$ to Σ' is the MECS $\mathcal{S}_{\uparrow\Sigma'} = (Q^{\perp\!\!\perp}, \lambda^0, \delta^{m\prime}, \delta^{M\prime})$ such that $\delta^m \subseteq \delta^{m\prime}$, $\delta^M \subseteq \delta^{M\prime}$ and for all $a \in \Sigma' \setminus \Sigma$ and $q \in Q^{\perp\!\!\perp}$, $(q, true, a, q) \in \delta^{m\prime} \cap \delta^{M\prime}$.

Weak and strong model/refinement relations, allowing for alphabet enlargement through refinement, can now be defined:

Definition 14 (*Weak and Strong Model/Refinement Relations*)**.** Let $\mathcal{S}$ be a MECS over Σ and $\mathcal{C}$ an ECA over Σ' with $\Sigma' \supseteq \Sigma$:

1. $\mathcal{C}$ is a weak implementation of $\mathcal{S}$, written $\mathcal{C} \models_w \mathcal{S}$, if $R(\mathcal{C}) \models R(\mathcal{S}_{\Uparrow\Sigma'})$;
2. $\mathcal{C}$ is a strong implementation of $\mathcal{S}$, written $\mathcal{C} \models_s \mathcal{S}$, if $R(\mathcal{C}) \models R(\mathcal{S}_{\uparrow\Sigma'})$.

Now let $\mathcal{S}_1$ and $\mathcal{S}_2$ be MECS over Σ_1 and Σ_2 respectively with $\Sigma_1 \supseteq \Sigma_2$:

1. $\mathcal{S}_1$ is a weak refinement of $\mathcal{S}_2$, written $\mathcal{S}_1 \preceq_w \mathcal{S}_2$, if $R(\mathcal{S}_1) \preceq R(\mathcal{S}_{\Uparrow\Sigma_1})$;
2. $\mathcal{S}_1$ is a strong refinement of $\mathcal{S}_2$, written $\mathcal{S}_1 \preceq_s \mathcal{S}_2$, if $R(\mathcal{S}_1) \preceq R(\mathcal{S}_{\uparrow\Sigma_1})$.

Next we define conjunction, product and quotient for MECS over dissimilar alphabets. These operations are performed by, first, equalizing alphabets and then, applying the operators already defined for the case of equal alphabets. The first step requires to use either weak or strong extension depending on the operation. Indeed alphabet equalization should be *neutral*; it should not constrain what other specifications may want to require regarding these extra actions. For conjunction, observe that, by Section 3.1:

$$q_1 \overset{g,a}{\leadsto} \lambda' \;\wedge\; q_2 \overset{true,a}{\dashrightarrow} q_2 \Rightarrow (q_1, q_2) \overset{g,a}{\leadsto} (\lambda', q_2)$$

For product, following Section 3.2:

$$q_1 \overset{g,a}{\leadsto} \lambda' \;\otimes\; q_2 \xrightarrow{true,a} q_2 \Rightarrow (q_1, q_2) \overset{g,a}{\leadsto} (\lambda', q_2)$$

These observations intuitively reveal our solution: for conjunction, weak extension must be used for alphabet equalization; whereas strong extension is needed for product.

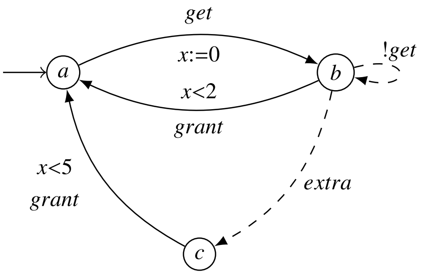


Fig. 8. A desired global behavior not expressible by MECS.

Definition 15 (*Operations*). Let $\mathcal{S}$, $\mathcal{S}_1$ and $\mathcal{S}_2$ be MECS respectively defined over Σ, Σ_1 and Σ_2:

$$\begin{aligned}
&\mathcal{S}_1 \wedge \mathcal{S}_2 = \mathcal{S}_{1\Uparrow(\Sigma_1\cup\Sigma_2)} \wedge \mathcal{S}_{2\Uparrow(\Sigma_1\cup\Sigma_2)}\\
&\mathcal{S}_1 \otimes \mathcal{S}_2 = \mathcal{S}_{1\uparrow(\Sigma_1\cup\Sigma_2)} \otimes \mathcal{S}_{2\uparrow(\Sigma_1\cup\Sigma_2)}\\
&\mathcal{S} \oslash \mathcal{S}_1 = \mathcal{S}_{\Uparrow(\Sigma\cup\Sigma_1)} \oslash \mathcal{S}_{1\uparrow(\Sigma\cup\Sigma_1)}
\end{aligned}$$

The operations enjoy the following properties which generalize the results presented in Sections 2 and 3:

Proposition 8. 1. *Weak and strong implementation/refinement relations are related as follows:*

$$\models_s \subseteq \models_w \quad \text{and} \quad \preceq_s \subseteq \preceq_w$$

2. *Weak and strong modal refinements are both sound and complete:*

$$\begin{aligned}
\mathcal{S}_1 \preceq_w \mathcal{S}_2 \Leftrightarrow \{\mathcal{C} \mid \mathcal{C} \models_w \mathcal{S}_1\} \subseteq \{\mathcal{C} \mid \mathcal{C} \models_w \mathcal{S}_2\}\\
\mathcal{S}_1 \preceq_s \mathcal{S}_2 \Leftrightarrow \{\mathcal{C} \mid \mathcal{C} \models_s \mathcal{S}_1\} \subseteq \{\mathcal{C} \mid \mathcal{C} \models_s \mathcal{S}_2\}
\end{aligned}$$

3. *For any* MECS $\mathcal{S}_1$, $\mathcal{S}_2$ *respectively defined over* Σ_1 *and* Σ_2 *and any* ECA $\mathcal{C}$ *defined over* $\Sigma \supseteq \Sigma_1 \cup \Sigma_2$*:*

$$\mathcal{C} \models_w \mathcal{S}_1 \wedge \mathcal{S}_2 \quad \Leftrightarrow \quad \mathcal{C} \models_w \mathcal{S}_1 \text{ and } \mathcal{C} \models_w \mathcal{S}_2$$

4. *For any* MECS $\mathcal{S}'_1$, $\mathcal{S}'_2$ *respectively defined over* Σ'_1 *and* Σ'_2*, any* $\mathcal{S}_1$ *and any* ECA $\mathcal{C}_1$ *defined over* $\Sigma_1 \supseteq \Sigma'_1$ *and any* $\mathcal{S}_2$ *and any* ECA $\mathcal{C}_2$ *defined over* $\Sigma_2 \supseteq \Sigma'_2$

$$\begin{aligned}
(\mathcal{S}_1 \preceq_s \mathcal{S}'_1 \text{ and } \mathcal{S}_2 \preceq_s \mathcal{S}'_2) &\implies \mathcal{S}_1 \otimes \mathcal{S}_2 \preceq_s \mathcal{S}'_1 \otimes \mathcal{S}'_2; \quad \text{and}\\
(\mathcal{C}_1 \models_s \mathcal{S}'_1 \text{ and } \mathcal{C}_2 \models_s \mathcal{S}'_2) &\implies \mathcal{C}_1 \otimes \mathcal{C}_2 \models_s \mathcal{S}'_1 \otimes \mathcal{S}'_2.
\end{aligned}$$

These implications are false if weak refinement or implementation are used instead of the strong forms.

5. *Let* $\mathcal{S}$, $\mathcal{S}_1$ *and* $\mathcal{S}_2$ *be* MECS *respectively defined over* Σ, Σ_1 *and* Σ_2 *such that* $\Sigma_2 \supseteq \Sigma \supseteq \Sigma_1$*:*

$$\mathcal{S}_2 \preceq_s \mathcal{S} \oslash \mathcal{S}_1 \Longrightarrow \mathcal{S}_1 \otimes \mathcal{S}_2 \preceq_s \mathcal{S}.$$

Now let $\mathcal{C}_2$ *be an* ECA *defined over* Σ_2*:*

$$\mathcal{C}_2 \models_s \mathcal{S} \oslash \mathcal{S}_1 \Longrightarrow \forall \mathcal{C}_1.[\mathcal{C}_1 \models_s \mathcal{S}_1 \Rightarrow \mathcal{C}_1 \otimes \mathcal{C}_2 \models_s \mathcal{S}].$$

4.4. Beyond event-clock automata

One may wonder why we restricted our framework to event-clock timed automata, and did not deal with general timed automata. Surprisingly, we do not exploit the determinizability of event-clock automata, even if we require modal specifications to be deterministic: indeed starting from deterministic modal event-clock specifications, one can show that all operations preserve determinacy. We are rather interested in the very specific treatment of resets for event-clock automata.

Example 10. A modified version of the desired global behavior for our running example is represented on Fig. 8. This specification ensures that the grant will come at most 2 time units after the first request to the resource. Several requests can be made because of the may-loop on state *b*, and thus modeling this behavior by a MECS where clocks are always resets on their corresponding action is not possible.

In order to go beyond the class of event-clock timed automata, we need to clarify what the meaning of timed modal specifications should be. In particular when we build a quotient of two specifications, we should decide on what this operation means, and hence what we allow for the specifications. In the context of event-recording modal specifications, the alphabet determines the clock set, hence there is no doubt that all specifications share a common set of clocks. Outside of this class, it is no longer clear what the requirements for the sets of clocks should be. When building the product of two specifications, it is natural to consider the disjoint union of the sets of clocks (of both specifications). This could be simplified in the case of event-clock timed modal specifications since the action determines the clock to be reset. Hence, when synchronizing two specifications on an action, exactly the same clock is reset in both specifications.

We choose here to decide that when quotienting a timed modal specification $\mathcal{S}$ by some other $\mathcal{S}_1$, the obtained specification also has as its set of clocks the disjoint union of sets of clocks of the two components. An alternative would be to require inclusion of the set of clocks of $\mathcal{S}_1$ in the set of clocks of $\mathcal{S}$.

Let us present the generalization of the theory presented so far for MECS. We detail the operations of greatest lower bound, product and quotient for timed modal specification. Let $\mathcal{S}_1 = (Q_1^{\perp\!\!\!\perp}, \lambda_1^0, \mathcal{X}_1, \delta_1^m, \delta_1^M)$ and $\mathcal{S}_2 = (Q_2^{\perp\!\!\!\perp}, \lambda_2^0, \mathcal{X}_2, \delta_2^m, \delta_2^M)$ be two modal timed automata over the same alphabet[1] Σ. The conjunction $\mathcal{S}_1 \wedge \mathcal{S}_2$, the product $\mathcal{S}_1 \otimes \mathcal{S}_2$ and the quotient $\mathcal{S}_1 \oslash \mathcal{S}_2$, are timed modal specifications over the same alphabet Σ and with set of clocks $\mathcal{X}_1 \cup \mathcal{X}_2$.

Definition 16 (*Timed Modal Specification*)**.** A timed modal specification (TMS) $\mathcal{S}$ is a tuple $(Q^{\perp\!\!\!\perp}, \lambda^0, \mathcal{X}, \delta^m, \delta^M)$ where

- $Q^{\perp\!\!\!\perp} := Q \cup \perp\!\!\!\perp$ is a finite set of *locations*, with $\perp\!\!\!\perp \cap Q = \emptyset$, and the initial state is $\lambda^0 \in Q^{\perp\!\!\!\perp}$.
- $\mathcal{X}$ is a finite set of clocks.
- $\delta^M \subseteq \delta^m \subseteq Q \times \xi[\mathcal{X}] \times \Sigma \times 2^{\mathcal{X}} \times Q^{\perp\!\!\!\perp}$ are finite sets of respectively *must*- and *may*-transitions, δ^m being *deterministic*.

Refinement. As we did in the case of MECS, we define the refinement relation for timed modal specifications only if they share the same set of clocks. Thus, in order to be able to compare two arbitrary TMS $\mathcal{S}_1$ and $\mathcal{S}_2$ over $\mathcal{X}_1$ and $\mathcal{X}_2$ respectively, we first need to augment their sets of clocks to $\mathcal{X}_1 \cup \mathcal{X}_2$. Given $\mathcal{S}$ a timed modal specification over $\mathcal{X}$, and $\mathcal{X}'$ a set of clocks, we denote by $\mathcal{S}^{\mathcal{X}'}$ the specification obtained by setting the set of clocks of $\mathcal{S}$ to $\mathcal{X} \cup \mathcal{X}'$. For $\mathcal{S}_1$ and $\mathcal{S}_2$ two timed modal specifications, over $\mathcal{X}_1$ and $\mathcal{X}_2$ respectively, we say that $\mathcal{S}_1$ refines $\mathcal{S}_2$, denoted $\mathcal{S}_1 \preceq \mathcal{S}_2$ if and only if $R(\mathcal{S}_1^{\mathcal{X}_2}) \preceq R(\mathcal{S}_2^{\mathcal{X}_1})$.

Operations. As for modal specifications, we explain here how to compute the greatest lower bound, the product and the quotient of two timed modal specifications. In all constructions, the resulting specification is over the same alphabet and with set of clocks the disjoint union of the sets of clocks of the two components. The rules are derived from the ones for modal event-clock specifications, the only difference resides in the resets (which were implicit for MECS) where the resulting reset is the union of reset sets.

As an example, we detail below one rule for each operation, for the greatest lower bound, the product, and the quotient, respectively.

$$\frac{\lambda_1 \xrightarrow{g_1,a,Y_1} \lambda_1' \text{ and } \lambda_2 \overset{g_2,a,Y_2}{\dashrightarrow} \lambda_2'}{(\lambda_1,\lambda_2) \xrightarrow{g_1\wedge g_2,a,Y_1\cup Y_2} (\lambda_1',\lambda_2')}\ (TGlb2) \quad \frac{\lambda_1 \xrightarrow{g_1,a,Y_1} \lambda_1' \text{ and } \lambda_2 \overset{g_2,a,Y_2}{\dashrightarrow} \lambda_2'}{(\lambda_1,\lambda_2) \overset{g_1\wedge g_2,a,Y_1\cup Y_2}{\dashrightarrow} (\lambda_1',\lambda_2')}\ (TProd2) \quad \frac{\lambda \overset{g,a,Y}{\dashrightarrow} \text{ and } \lambda_1 \overset{g_1,a,Y_1}{\dashrightarrow} \lambda_1'}{(\lambda,\lambda_1) \overset{H\wedge g\wedge g_1\wedge \mathcal{I}(\lambda_1')[0/x_a],a,Y\cup Y_1}{\dashrightarrow} \top}\ (Tmay1)$$

As MECS equipped with their operations, general deterministic timed modal specifications enjoy a series of nice properties making them a good candidate for a theory of interfaces for real-time systems.

Proposition 9. *For any timed modal specification $\mathcal{S}$, $\mathcal{S}_1$ and $\mathcal{S}_2$ over sets of clocks $\mathcal{X}$, $\mathcal{X}_1$ and $\mathcal{X}_2$ respectively:*

1. *via the region construction, operators are equivalent to their untimed version*
$R(\mathcal{S}_1 \wedge \mathcal{S}_2) \equiv R(\mathcal{S}_1^{\mathcal{X}_2}) \wedge R(\mathcal{S}_2^{\mathcal{X}_1}) \qquad R(\mathcal{S}_1 \otimes \mathcal{S}_2) \equiv R(\mathcal{S}_1^{\mathcal{X}_2}) \otimes R(\mathcal{S}_2^{\mathcal{X}_1}) \qquad R(\mathcal{S} \oslash \mathcal{S}_1) \preceq R(\mathcal{S}^{\mathcal{X}_1}) \oslash R(\mathcal{S}_1^{\mathcal{X}})$
2. *property of the greatest lower bound: $\mathcal{S}_1 \wedge \mathcal{S}_2$ is the greatest lower bound of $\mathcal{S}_1^{\mathcal{X}_2}$ and $\mathcal{S}_2^{\mathcal{X}_1}$*
3. *property of the product: $\mathcal{S}_1 \preceq \mathcal{S}_2$ and $\mathcal{S}_1' \preceq \mathcal{S}_2'$ implies $\mathcal{S}_1 \otimes \mathcal{S}_1' \preceq \mathcal{S}_2 \otimes \mathcal{S}_2'$*
4. *property of the quotient: $\mathcal{S}_2^{\mathcal{X}\cup\mathcal{X}_1} \preceq (\mathcal{S} \oslash \mathcal{S}_1)^{\mathcal{X}_2} \Longrightarrow (\mathcal{S}_1 \otimes \mathcal{S}_2)^{\mathcal{X}} \preceq \mathcal{S}^{\mathcal{X}_1\cup\mathcal{X}_2}$*

Comparison with modal event-clock specifications. At a first glance, the way we propose to deal with general timed modal specifications is not a conservative extension of the theory we developed for the particular case of modal event-clock specifications. Still, we explain here how MECS can be naturally embedded into TMS, and how the operations on MECS can be seen as operations on general TMS as just defined above. Trivially, any modal event-clock specification $\mathcal{S}$ on alphabet Σ can be seen as a timed modal specification $\tilde{\mathcal{S}}$ with set of clocks $\mathcal{X}_\Sigma$ (formed of one clock x_a for each action a) such that each transition $\overset{g,a}{\rightsquigarrow}$ in $\mathcal{S}$ is transformed into a transition $\overset{g,a,\{x_a\}}{\rightsquigarrow}$ in $\tilde{\mathcal{S}}$. Of course the reverse transformation, from TMS to MECS, is only possible if each action is paired with a clock (or possibly a set of clocks—as we will use later) which is reset exactly when the action takes place.

Let us now detail the embedding of the operations on MECS into operations on timed modal specifications. The apparent difference between MECS and TMS for all operations is that for TMS we require the set of clocks to be disjoint, which is not the case for MECS. However computing the greatest lower bound, the product or the quotient of two MECS can be seen as a particular case of the same operations on TMS by letting: in $\mathcal{S}_i$ the clocks associated with actions are annotated by superscript i. When building the greatest lower bound, the product or the quotient as in TMS, one obtains a TMS over set of clocks $\mathcal{X}_\Sigma^1 \cup \mathcal{X}_\Sigma^2$. However, this resulting TMS enjoys the property that whenever action a labels a transition, the reset set is exactly $\{x_a^1, x_a^2\}$. Renaming these two clocks into a common clock x_a we recover a TMS which can be seen as a MECS. Note that this series of operations is equivalent to taking the greatest lower bound, the product or the quotient directly on MECS. In particular, when equating clocks x_a^1 and x_a^2 to x_a, regarding guards, it corresponds to taking the conjunction of guards on the same clock, which is what is done for MECS.

[1] We assume here that all timed modal specifications are defined over the same global alphabet Σ. The treatment of dissimilar alphabets follows the same line as for MECS in Section 4.3.

Further possible improvements. A different point of view would consist in favoring the semantics of our models rather than the syntax. It would thus make sense to consider the refinement of timed modal specifications as a timed alternating simulation between the two specifications, without requiring that they share a common set of clocks. This view point has been explored for input/output timed automata by David *et al.* [28] and we plan to see how far the methods they develop for their model extend to timed modal automata. Doing this, we would be able to quotient any pair of timed modal specifications, getting rid of requirements over their respective sets of clocks.

5. Related work

Regarding a theory of interfaces, we compare our approach with the following settings: Interface automata of [5], timed interfaces of [12], and a timed extension of modal specifications of [29].

Interface automata. In interface automata [5], an interface is represented by an input/output automaton [30], *i.e.*, an automaton whose transitions are typed with *input* and *output* rather than must and may modalities. The semantics of such an automaton is given by a two-player game: the *input* player represents the environment, and the *output* player represents the component itself. As explained in [10], interfaces and modalities are in essence orthogonal to each other. Moreover, interface automata do not encompass any notion of model, and thus neither the model relation nor the consistency, because one cannot distinguish between interfaces and components. Alternatively, properties of interfaces are described in game-based logics, *e.g.*, ATL [31], with a high-cost complexity. Refinement between interface automata corresponds to the alternating refinement relation between games [4], *i.e.*, an interface refines another one if its environment is more permissive whereas its component is more restrictive. As for modal automata, the quotient operator is defined for the deterministic case only [32]. Moreover, shared refinement is defined in an *ad hoc* manner [6] for the very particular and restricted class of synchronous interfaces [33]. Composition of interface automata differs from the one over modal specifications. Indeed, in interface automata, the game-based approach offers an optimistic treatment of composition: two interfaces can be composed if there exists at least one environment in which they can interact together in a safe way. In [7], Larsen et al. proposed *modal interfaces* that are modal specifications composed in a game-based manner. This work suggests that modal interfaces subsume interface automata.

There are many other works that study interface theories and component-based design. Among them, one find a series of very practical works that do not study quotient and conjunction, but rather focus on richer composition operations and specific models of computation for interconnection and software design [34–36]. While our theory is certainly more general, it would be of interest to learn from those models in order to generalize our composition operation.

In a series of [37,38], Cattani and Winskel have proposed a categorial axiomatic for bisimulation that is a congruence for general process languages such as CCS. Since our notion of refinement share similarities with classical notions of simulation and bisimulation, it could be of interest to see whether modal automata can be captured by Cattani's framework. This may enrich the refinement relation with new preservation properties.

Timed interfaces. In [12], de Alfaro et al. proposed *timed interfaces* which extend timed automata just as interface automata extend automata. The syntax of a timed interface is thus similar to the one of a *timed input/output automaton* [39], but the semantics is given by a game. The composition operator defined for timed interfaces allows to capture the timing dimension between interfaces: "what are the temporal ordering constraints on communication events between components?". Compared to timed modal specifications, the work in [12] lacks a notion of implementation and of refinement; Moreover, neither shared refinement nor quotient are studied.

In a very recent work [28], David et al. proposed a new version of timed interfaces. The major differences in comparison with the results in [12] are (1) a clear definition of the concept of implementation, (2) the definition of shared refinement and quotient operations, (3) the definition of a game-based refinement operator, which extends the one proposed in [40], and (4) an implementation within the UPPAAL-TIGA toolset [41]. In [28], timed interfaces are assumed to be deterministic and input-enabled. This makes it impossible to reach an immediate error state, where a component proposes an output that cannot be captured by the other component. Input-enabledness shall not be seen as a way to avoid error states. Indeed, such error states can be designated by the designer as states which do not warrant desirable temporal properties. It is worth mentioning that it is much easier to define a notion of implementation for timed interfaces than for untimed interfaces. Indeed, one can simply distinguish implementations from specifications by adding constraints on their timing behaviors. As an example, in [28], the authors assume that an implementation is a specification where outputs are urgent and where the environment should not be responsible for the progress of time.

Timed modal specifications differ from timed input/output automata already in nature since they consider orthogonal features: input/output or may/must modalities. In comparison to [28], our work (1) allows one to combine and compare specifications that share clock variables, (2) is based on region and not on the continuous-time semantics and (3) encompasses the possibility of dissimilar alphabets.

Timed extension of modal specifications. A timed extension of modal specifications appeared in [29] in a process algebra style. The formalism proposed is a variant of CCS whose semantics relies on the configuration graph rather than on the region graph, as done here. No logical characterization is developed, nor any notion of model relation (satisfaction) or consistency (satisfiability). Moreover, the quotient has not been considered at all.

6. Conclusion and future work

Modal specifications offer a well-adapted algebraic framework for compositional reasoning on component-based systems, that enables incremental design as well as reuse of components. In this paper, we have presented a timed extension of modal specifications using event-clock timed automata. All essential features expected from a theory of interface (such as refinement, conjunction, satisfiability, product, and quotient) are fully addressed. They are all efficiently treated in this framework except for refinement which relies on the region graph construction. A symbolic definition of this relation that is sound, albeit possibly not complete, can be noted as an open issue.

In addition to implementation, several research directions still need to be investigated in the future. We aim at continuing the work initiated in Section 4.4 and study timed modal specifications in a broader framework than the one of MECS, since event-clock automata are strictly less expressive than timed automata. Another topic concerns a logical characterization of modal event-clock specifications (or even more general timed modal specifications), in the spirit of [14] who established the correspondence between simple modal specifications and conjunctive nu-calculus. Such a characterization brings insight into the expressiveness of the specification formalism. One should also introduce stochastic aspects in the model. Finally, in [42,3], one has proposed a model which combines advantages of both interface automata and modal specifications. One should follow a similar direction and combine our model with the one of timed interfaces proposed in [28].

References

[1] T.A. Henzinger, J. Sifakis, The embedded systems design challenge, in: Proceedings of the 14th International Symposium on Formal Methods, FM'06, in: Lecture Notes in Computer Science, vol. 4085, Springer, 2006, pp. 1–15.

[2] T.A. Henzinger, J. Sifakis, The discipline of embedded systems design, IEEE Computer 40 (10) (2007) 32–40.

[3] K.G. Larsen, U. Nyman, A. Wasowski, On modal refinement and consistency, in: Proceedings of the 18th International Conference on Concurrency Theory, CONCUR'07, in: Lecture Notes in Computer Science, vol. 4703, Springer, 2007, pp. 105–119.

[4] R. Alur, T.A. Henzinger, O. Kupferman, M.Y. Vardi, Alternating refinement relations, in: Proceedings of the 9th International Conference on Concurrency Theory, CONCUR'98, in: Lecture Notes in Computer Science, vol. 1466, Springer, 1998, pp. 163–178.

[5] L. de Alfaro, T.A. Henzinger, Interface automata, in: Proceedings of the 9th ACM SIGSOFT International Symposium on Foundations of Software Engineering, FSE'01, 2001, pp. 109–120.

[6] L. Doyen, T.A. Henzinger, B. Jobstmann, T. Petrov, Interface theories with component reuse, in: Proceedings of the 8th International Conference on Embedded Software, EMSOFT'08, ACM Press, 2008, pp. 79–88.

[7] K.G. Larsen, U. Nyman, A. Wasowski, Modal I/O automata for interface and product line theories, in: Proceedings of the 16th European Symposium on Programming, ESOP'07, in: Lecture Notes in Computer Science, vol. 4421, Springer, 2007, pp. 64–79.

[8] J.-B. Raclet, Quotient de spécifications pour la réutilisation de composants, Ph.D. Thesis, Université de Rennes I, December 2007 (in French).

[9] J.-B. Raclet, Residual for component specifications, in: Proc. of the 4th International Workshop on Formal Aspects of Component Software, FACS'07, in: Electr. Notes Theor. Comput. Sci., vol. 215, 2008, pp. 93–110.

[10] J.-B. Raclet, E. Badouel, A. Benveniste, B. Caillaud, R. Passerone, Why are modalities good for interface theories?, in: Proceedings of the 9th International Conference on Application of Concurrency to System Design, ACSD'09, IEEE Computer Society Press, 2009, pp. 119–127.

[11] A. Antonik, M. Huth, K.G. Larsen, U. Nyman, A. Wasowski, 20 years of modal and mixed specifications, Bulletin of European Association of Theoretical Computer Science 1 (94).

[12] L. de Alfaro, T.A. Henzinger, M. Stoelinga, Timed interfaces, in: Proceedings of the 2nd International Workshop on Embedded Software, EMSOFT'02, in: Lecture Notes in Computer Science, vol. 2491, Springer, 2002, pp. 108–122.

[13] N. Bertrand, S. Pinchinat, J.-B. Raclet, Refinement and consistency of timed modal specifications, in: Proceedings of the 3rd International Conference on Language and Automata Theory and Applications, LATA'09, in: Lecture Notes in Computer Science, vol. 5457, Springer, 2009, pp. 152–163.

[14] G. Feuillade, S. Pinchinat, Modal specifications for the control theory of discrete-event systems, Discrete Event Dynamic Systems 17 (2) (2007) 181–205.

[15] A. Arnold, M. Nivat, Metric interpretations of infinite trees and semantics of non deterministic recursive programs, Theoretical Computer Science 11 (1980) 181–205.

[16] R. Alur, D.L. Dill, A theory of timed automata, Theoretical Computer Science 126 (2) (1994) 183–235.

[17] N. Bertrand, A. Legay, S. Pinchinat, J.-B. Raclet, A compositional approach on modal specifications for timed systems, in: Proc. of the 11th International Conference on Formal Engineering Methods, ICFEM'09, in: Lecture Notes in Computer Science, vol. 5885, Springer, 2009, pp. 679–697.

[18] R. Alur, L. Fix, T.A. Henzinger, Event-clock automata: a determinizable class of timed automata, Theoretical Computer Science 211 (1999) 1–13.

[19] K.G. Larsen, B. Thomsen, A modal process logic, in: Proceedings of the Third Annual Symposium on Logic in Computer Science, LICS'88, IEEE, 1988, pp. 203–210.

[20] K.G. Larsen, Modal specifications, in: Proceedings of the International Workshop on Automatic Verification Methods for Finite State Systems, in: Lecture Notes in Computer Science, vol. 407, Springer, 1989, pp. 232–246.

[21] B.A. Davey, H.A. Priestley, Introduction to Lattices and Order, Cambridge University Press, 2002.

[22] T.A. Henzinger, X. Nicollin, J. Sifakis, S. Yovine, Symbolic model checking for real-time systems, Information Computation 111 (2) (1994) 193–244.

[23] P. Bouyer, From qualitative to quantitative analysis of timed systems, "mémoire d'habilitation" Report, Université Paris 7, Paris, France, Jan. 2009.

[24] K.G. Larsen, B. Steffen, C. Weise, A constraint oriented proof methodology based on modal transition systems, in: Proc. of the 1st International Workshop on Tools and Algorithms for Construction and Analysis of Systems, TACAS'95, in: Lecture Notes in Computer Science, vol. 1019, Springer, 1995, pp. 17–40.

[25] C.A.R. Hoare, Communicating sequential processes, Communications of the ACM 21 (8) (1978) 666–677.

[26] B. Jonsson, K.G. Larsen, On the complexity of equation solving in process algebra, in: Proceedings of the International Joint Conference on Theory and Practice of Software Development, TAPSOFT'91, Springer, 1991, pp. 381–396.

[27] A. Arnold, A. Vincent, I. Walukiewicz, Games for synthesis of controllers with partial observation, Theoretical Computer Science 303 (1) (2003) 7–34.

[28] A. David, K.G. Larsen, A. Legay, U. Nyman, A. Wasowski, Timed I/O automata: a complete specification theory for real-time systems, in: Proceedings of the 13th ACM International Conference on Hybrid Systems: Computation and Control, HSCC(10), ACM, 2010, pp. 91–100.

[29] K. Čerāns, J.C. Godskesen, K.G. Larsen, Timed modal specification — theory and tools, in: Proceedings of the 5th International Conference on Computer Aided Verification, CAV'93, in: Lecture Notes in Computer Science, vol. 697, Springer, 1993, pp. 253–267.

[30] N. Lynch, M.R. Tuttle, An introduction to Input/Output automata, CWI-quarterly 2 (3) (1989) 219–246.

[31] R. Alur, T.A. Henzinger, O. Kupferman, Alternating-time temporal logic, Journal of the ACM 49 (5) (2002) 672–713.

[32] P. Bhaduri, Synthesis of interface automata, in: Automated Technology for Verification and Analysis, Third International Symposium, ATVA 2005, Taipei, Taiwan, October 4–7, 2005, Proceedings, in: Lecture Notes in Computer Science, vol. 3707, Springer, 2005, pp. 338–353.

[33] A. Chakrabarti, L. de Alfaro, T.A. Henzinger, F.Y.C. Mang, Synchronous and bidirectional component interfaces, in: Computer Aided Verification, 14th International Conference, CAV 2002,Copenhagen, Denmark, July 27–31, 2002, Proceedings, in: Lecture Notes in Computer Science, vol. 2404, Springer, 2002, pp. 414–427.

[34] G. Gößler, J. Sifakis, Composition for component-based modeling, Science of Computer Programming 55 (1–3) (2005) 161–183.

[35] J.L. Fiadeiro, L.F. Andrade, Interconnecting objects via contracts, in: Proc. of the 38th International Conference on Technology of Object-Oriented Languages and Systems, Components for Mobile Computing, TOOLS'38, IEEE Computer Society, 2001, pp. 182–183.

[36] J.L. Fiadeiro, T.S.E. Maibaum, Interconnecting formalisms: supporting modularity, reuse and incrementality, in: Proc. of the 3rd ACM SIGSOFT Symposium on Foundations of Software Engineering, SIGSOFT FSE'95, ACM, 1995, pp. 72–80.

[37] G.L. Cattani, G. Winskel, Presheaf models for CCS-like languages, Theoretical Computer Science 300 (1–3) (2003) 47–89.

[38] G.L. Cattani, G. Winskel, Profunctors, open maps and bisimulation, Mathematical Structures in Computer Science 15 (3) (2005) 553–614.

[39] D.K. Kaynar, N.A. Lynch, R. Segala, F.W. Vaandrager, The Theory of Timed I/O Automata, Synthesis Lectures on Computer Science, second ed., Morgan & Claypool Publishers, 2010.

[40] P. Bulychev, T. Chatain, A. David, K.G. Larsen, Efficient on-the-fly algorithm for checking alternating timed simulation, in: Formal Modeling and Analysis of Timed Systems, 7th International Conference, FORMATS 2009, Budapest, Hungary, September 14–16, 2009, Proceedings, in: Lecture Notes in Computer Science, vol. 5813, Springer, 2009, pp. 73–87.

[41] G. Behrmann, A. Cougnard, A. David, E. Fleury, K.G. Larsen, D. Lime, UPPAAL-TIGA: time for playing games!, in: Computer Aided Verification, 19th International Conference, CAV 2007, Berlin, Germany, July 3–7, 2007, Proceedings, in: Lecture Notes in Computer Science, vol. 4590, Springer, 2007, pp. 121–125.

[42] J.-B. Raclet, E. Badouel, A. Benveniste, B. Caillaud, A. Legay, R. Passerone, Modal interfaces: unifying interface automata and modal specifications, in: Proceedings of the 9th ACM & IEEE International conference on Embedded software, EMSOFT 2009, Grenoble, France, October 12–16, 2009, ACM, 2009, pp. 87–96.