# Compositional Verification and Optimization of Interactive Markov Chains[⋆]

Holger Hermanns[1], Jan Krčál[2], and Jan Křetínský[2,3]

[1] Saarland University – Computer Science, Saarbrücken, Germany
[2] Faculty of Informatics, Masaryk University, Czech Republic
[3] Institut für Informatik, Technical University Munich, Germany

**Abstract.** Interactive Markov chains (IMC) are compositional behavioural models extending labelled transition systems and continuous-time Markov chains. We provide a framework and algorithms for compositional verification and optimization of IMC with respect to time-bounded properties. Firstly, we give a specification formalism for IMC. Secondly, given a time-bounded property, an IMC component and the assumption that its unknown environment satisfies a given specification, we synthesize a scheduler for the component optimizing the probability that the property is satisfied in any such environment.

## 1 Introduction

The ever increasing complexity and size of systems together with software reuse strategies naturally enforce the need for component based system development. For the same reasons, checking reliability and optimizing performance of such systems needs to be done in a *compositional* way. The task is to get useful guarantees on the behaviour of a component of a larger system. The key idea is to incorporate assumptions on the rest of the system into the verification process. This *assume-guarantee reasoning* is arguably a successful divide-and-conquer technique in many contexts [MC81, AH96, HMP01].

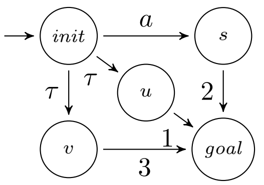
In this work, we consider a continuous-time stochastic model called *interactive Markov chains* (IMC). First, we give a language for expressing assumptions about IMC. Second, given an IMC, an assumption on its environment and a property of interest, we synthesize a controller of the IMC that optimizes the guarantee, and we compute this optimal guarantee, too.

**Interactive Markov chains** are behavioural models of probabilistic systems running in continuous real time appropriate for the component-based approach [HK09]. IMC have a well-understood compositional theory rooted in process algebra, and are in use as semantic backbones for dynamic fault trees, architectural

[⋆] The work has received support from the Czech Science Foundation, project No. P202/12/G061, from the German Science Foundation DFG as part of SFB/TR 14 AVACS, and by the EU FP7 Programme under grant agreement no. 295261 (MEALS) and 318490 (SENSATION).

description languages, generalized stochastic Petri nets and Statemate extensions, see [HK09] for a survey. IMC are applied in a large spectrum of practical applications, ranging from water treatment facilities [HKR+10] to ultra-modern satellite designs [EKN+12].

IMC arise from classical labelled transition systems by incorporating the possibility to change state according to a random *delay* governed by a negative exponential distribution with a given rate, see transitions labelled 1, 2 and 3 in the figure. Apart from delay expirations, state transitions may be triggered by the execution of *internal* ($\tau$) actions or *external* (synchronization) actions. Internal actions are assumed to happen instantaneously and therefore take precedence over delay transitions. External actions are the process algebraic means for interaction with other components, see $a$ in the figure. By dropping the delay transitions, labelled transition systems are regained in their entirety. Dropping action-labelled transitions instead yields continuous-time Markov chains – one of the most used performance and reliability models.

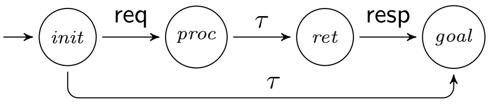


$$\begin{tikzcd} \arrow[r] & init \arrow[rr, "a"] \arrow[d, "\tau"'] \arrow[dr, "\tau"] & & s \arrow[d, "2"] \\ & v \arrow[rr, "1"] \arrow[rr, "3"', bend right] & u \arrow[r] & goal \end{tikzcd}$$

The fundamental problem in the analysis of IMC is that of *time-bounded reachability.* It is the problem to approximate the probability that a given set of states is reached within a given deadline. We illustrate the compositional setting of this problem in the following examples.

**Examples.** In the first example, consider the IMC $\mathcal{C}$ from above and an unknown environment $\mathcal{E}$ with no assumptions. Either $\mathcal{E}$ is initially not ready to synchronize on the external action $a$ and thus one of the internal actions is taken, or $\mathcal{E}$ is willing to synchronize on $a$ at the beginning. In the latter case, whether $\tau$ or $a$ happens is resolved non-deterministically. Since this is out of control of $\mathcal{C}$, we must assume the worst case and let the environment decide which of the two options will happen. For more details on this design choice, see [BHK+12]. If there is synchronization on $a$, the probability to reach *goal* within time $t = 1.5$ is $1 - e^{-2t} \approx 0.95$. Otherwise, $\mathcal{C}$ is given the choice to move to $u$ or $v$. Naturally, $v$ is the choice maximizing the chance to get to *goal* on time as it has a higher rate associated. In this case the probability amounts to $1 - e^{-3t} \approx 0.99$, while if $u$ were chosen, it would be only 0.78. Altogether, the guaranteed probability is 95% and the strategy of $\mathcal{C}$ is to choose $v$ in *init*.

The example depicted on the right illustrates the necessity of assumptions on the environment: As it is, the environment can drive the component to state *ret* and let it get stuck there by not synchronising on resp ever. Hence no better guarantee than 0 can be derived. However, this changes if we know some specifics about the behaviour of the environment: Let us assume that we know that once synchronization on req occurs, the environment must be ready to synchronise on resp within some random time according to, say, an exponential distribution with rate 2. Under this assumption, we are able to derive a guarantee of 95%, just as in the previous example.



$$\begin{tikzcd} \arrow[r] & init \arrow[r, "\mathsf{req}"] \arrow[rrr, bend right, "\tau"'] & proc \arrow[r, "\tau"] & ret \arrow[r, "\mathsf{resp}"] & goal \end{tikzcd}$$

Observe the form of the time constraint we imposed in the last example: "within a random time distributed according to Exp(2)" or symbolically $\Diamond_{\leq Exp(2)} \varphi$. We call this a *continuous time constraint*. If a part of the environment is e.g. a model of a communication network, it is clear we cannot impose hard bounds (discrete time constraints) such as "within 1.5" as in e.g. a formula of MTL $\Diamond_{\leq 1.5}\varphi$. Folklore tells us that messages might get delayed for longer than that. Yet we want to express high assurance that they arrive on time. In this case one might use e.g. a formula of CSL $Pr_{\geq 0.95}(\Diamond_{\leq 1.5}\varphi)$. However, consider now a system with two transitions labelled with resp in a row. Then this CSL formula yields only a zero guarantee. By splitting the time 1.5 in halves, the respective $Pr_{\geq 0.77}(\Diamond_{\leq 0.75}\varphi)$ yields only the guarantee $0.77^2 = 0.60$. The actual guarantee 0.80 is given by the convolution of the two exponential distributions and as such can be exactly obtained from our continuous time constraint $\Diamond_{\leq Exp(2)}\varphi$.

**Our contribution** is the following:

1. We introduce a specification formalism to express assumptions on continuous-time stochastic systems. The novel feature of the formalism are the continuous time constraints, which are vital for getting guarantees with respect to time-bounded reachability in IMC.
2. We incorporate the assume-guarantee reasoning to the IMC framework. We show how to synthesize $\epsilon$-optimal schedulers for IMC in an *unknown environment satisfying a given specification* and approximate the respective guarantee.

In our recent work [BHK+12] we considered a very restricted setting of the second point. Firstly, we considered no assumptions on the environment as the environment of a component might be entirely unknown in many scenarios. Secondly, we were restricted to IMC that never enable internal and external transitions at the same state. This was also a severe limitation as this property is not preserved during the IMC composition process and restricts the expressivity significantly. Both examples above violate this assumption. In this paper, we lift the assumption.

Each of the two extensions shifts the solution methods from complete information stochastic games to (one-sided) *partial observation* stochastic games, where we need to solve the quantitative reachability problem. While this is undecidable in general, we reduce our problem to a game played on an *acyclic graph* and show how to solve our problem in exponential time. (Note that even the qualitative reachability in the acyclic case is PSPACE-hard [CD10].)

**Related Work.** The *synthesis* problem is often stated as a game where the first player controls a component and the second player simulates an environment [RW89]. Model checking of *open* systems, i.e. operating in an unknown environment, has been proposed in [KV96]. There is a body of work on *assume-guarantee* reasoning for parallel composition of *real-time* systems [TAKB96, HMP01]. Lately, games with *stochastic continuous-time* have gained attention, for a very general class see [BF09]. While the second player models possible schedulers of the environment, the structure of the environment is fixed there and the verification is thus not compositional. The same holds for [Spr11, HNP+11], where time is under the control of the components.

A compositional framework requires means for specification of systems. A specification can be also viewed as an *abstraction* of a set of systems. Three valued abstractions stemming from [LT88] have also been applied to the timed setting, namely in [KKLW07] to continuous-time Markov chains (IMC with no non-determinism), or in [KKN09] to IMC. Nevertheless, these abstractions do not allow for constraints on time distributions. Instead they would employ abstractions on transition probabilities. Further, a compositional framework with timed specifications is presented in [DLL+12]. This framework explicitly allows for time constraints. However, since the systems under consideration have non-deterministic flow of time (not stochastic), the natural choice was to only allow for discrete (not continuous) time constraints.

Although IMC support compositional design very well, analysis techniques for IMC proposed so far (e.g. [KZH+11, KKN09, ZN10, GHKN12] are not compositional. They are all bound to the assumption that the analysed IMC is a *closed* system, i.e. it does not depend on interaction with the environment (all actions are internal). Some preliminary steps to develop a framework for synthesis of controllers based on models of hardware and control requirements have been taken in [Mar11]. The first attempt at compositionality is our very recent work [BHK+12] discussed above.

Algorithms for the *time-bounded reachability* problem for closed IMC have been given in [ZN10, BS11, HH13] and compositional abstraction techniques to compute it are developed in [KKN09]. In the closed interpretation, IMC have some similarities with continuous-time Markov decision processes. For this formalism, algorithms for time-bounded reachability are developed in [BHKH05, BS11].

## 2 Interactive Markov Chains

In this section, we introduce the formalism of interactive Markov chains together with the standard way to compose them. We denote by $\mathbb{N}$, $\mathbb{R}_{>0}$, and $\mathbb{R}_{\geq 0}$ the sets of positive integers, positive real numbers and non-negative real numbers, respectively. Further, let $\mathcal{D}(S)$ denote the set of probability distributions over the set $S$.

**Definition 1 (IMC).** *An* interactive Markov chain (IMC) *is a quintuple* $\mathcal{C} = (S, \mathbb{A}\mathrm{ct}^\tau, \hookrightarrow, \rightsquigarrow, s_0)$ *where* $S$ *is a finite set of* states*,* $\mathbb{A}\mathrm{ct}^\tau$ *is a finite set of* actions *containing a designated* internal action $\tau$*,* $s_0 \in S$ *is an* initial *state,*

- $\hookrightarrow \subseteq S \times \mathbb{A}\mathrm{ct}^\tau \times S$ *is an* interactive transition *relation, and*
- $\rightsquigarrow \subseteq S \times \mathbb{R}_{>0} \times S$ *is a* Markovian transition *relation.*

Elements of $\mathbb{A}\mathrm{ct} := \mathbb{A}\mathrm{ct}^\tau \setminus \{\tau\}$ are called *external actions*. We write $s \overset{a}{\hookrightarrow} t$ whenever $(s, a, t) \in \hookrightarrow$, and $s \overset{\lambda}{\rightsquigarrow} t$ whenever $(s, \lambda, t) \in \rightsquigarrow$ where $\lambda$ is called a *rate* of the transition. We say that an external action $a$, or internal $\tau$, or Markovian transition is *available* in $s$, if $s \overset{a}{\hookrightarrow} t$, $s \overset{\tau}{\hookrightarrow} t$ or $s \overset{\lambda}{\rightsquigarrow} t$ for some $t$ (and $\lambda$), respectively.

IMC are well suited for compositional modelling, where systems are built out of smaller ones using standard composition operators. *Parallel composition* $\|_A$ over a *synchronization alphabet* $A$ produces a product of two IMC with transitions given by the rules

**(PC1)** $(s_1, s_2) \overset{a}{\hookrightarrow} (s_1', s_2')$ for each $s_1 \overset{a}{\hookrightarrow} s_1'$ and $s_2 \overset{a}{\hookrightarrow} s_2'$ and $a \in A$,
**(PC2, PC3)** $(s_1, s_2) \overset{a}{\hookrightarrow} (s_1', s_2)$ for each $s_1 \overset{a}{\hookrightarrow} s_1'$ and $a \notin A$, and symmetrically,
**(PC4, PC5)** $(s_1, s_2) \overset{\lambda}{\rightsquigarrow} (s_1', s_2)$ for each $s_1 \overset{\lambda}{\rightsquigarrow} s_1'$, and symmetrically.

Further, *hiding* $\setminus A$ an alphabet $A$, yields a system, where each $s \overset{a}{\hookrightarrow} s'$ with $a \notin A$ is left as it is, and each $s \overset{a}{\hookrightarrow} s'$ with $a \in A$ is replaced by internal $s \overset{\tau}{\hookrightarrow} s'$.

Hiding $\setminus \mathbb{A}\mathrm{ct}$ thus yields a *closed* IMC, where external actions do not appear as transition labels (i.e. $\hookrightarrow \subseteq S \times \{\tau\} \times S$). A closed IMC (under a scheduler $\sigma$, see below) moves from state to state and thus produces a *run* which is an infinite sequence of the form $s_0\, t_1\, s_1\, t_2\, s_2 \cdots$ where $s_n$ is the $n$-th visited state and $t_n$ is the time of arrival to $s_n$. After $n$ steps, the scheduler resolves the non-determinism among internal $\tau$ transitions based on the *path* $\mathfrak{p} = s_0\, t_1 \cdots t_n\, s_n$.

**Definition 2 (Scheduler).** *A* scheduler *of an IMC* $\mathcal{C} = (S, \mathbb{A}\mathrm{ct}^\tau, \hookrightarrow, \rightsquigarrow, s_0)$ *is a measurable function* $\sigma : (S \times \mathbb{R}_{\geq 0})^* \times S \to \mathcal{D}(S)$ *such that for each path* $\mathfrak{p} = s_0\, t_1\, s_1 \cdots t_n\, s_n$ *with* $s_n$ *having* $\tau$ *available,* $\sigma(\mathfrak{p})(s) > 0$ *implies* $s_n \overset{\tau}{\hookrightarrow} s$*. The set of all schedulers for* $\mathcal{C}$ *is denoted by* $\mathfrak{S}(\mathcal{C})$*.*

The decision of the scheduler $\sigma(\mathfrak{p})$ determines $t_{n+1}$ and $s_{n+1}$ as follows. If $s_n$ has available $\tau$, then the run proceeds immediately, i.e. at time $t_{n+1} := t_n$, to a state $s_{n+1}$ randomly chosen according to the distribution $\sigma(\mathfrak{p})$. Otherwise, only Markovian transitions are available in $s_n$. In such a case, after waiting for a random time $t$ chosen according to the exponential distribution with the rate $\mathrm{R}(s_n) = \sum_{s_n \overset{\lambda}{\rightsquigarrow} s'} \lambda$, the run moves at time $t_{n+1} := t_n + t$ to a randomly chosen next state $s_{n+1}$ with probability $\lambda / r$ where $s_n \overset{\lambda}{\rightsquigarrow} s_{n+1}$. This defines a probability space $(\mathbb{R}\mathrm{uns}, \mathcal{F}, \mathcal{P}^\sigma_{\mathcal{C}})$ over the runs in the standard way [ZN10].

## 3 Time-Bounded Reachability

In this section, we introduce the studied problems. One of the fundamental problems in verification and performance analysis of continuous-time stochastic systems is time-bounded reachability. Given a *closed* IMC $\mathcal{C}$, a set of goal states $G \subseteq S$ and a time bound $T \in \mathbb{R}_{\geq 0}$, the *value of time-bounded reachability* is defined as $\sup_{\sigma \in \mathfrak{S}(\mathcal{C})} \mathcal{P}^\sigma_{\mathcal{C}}\left[\Diamond^{\leq T} G\right]$ where $\mathcal{P}^\sigma_{\mathcal{C}}\left[\Diamond^{\leq T} G\right]$ denotes the probability that a run of $\mathcal{C}$ under the scheduler $\sigma$ visits a state of $G$ before time $T$. We have seen an example in the introduction. A standard assumption over all analysis techniques published for IMC [KZH+11, KKN09, ZN10, GHKN12] is that each cycle contains a Markovian transition. It implies that the probability of taking infinitely many transitions in finite time, i.e. of Zeno behaviour, is zero. One can $\varepsilon$-approximate the value and compute the respective scheduler in time $\mathcal{O}(\lambda^2 T^2 / \varepsilon)$ [ZN10] recently improved to $\mathcal{O}(\sqrt{\lambda^3 T^3 / \varepsilon})$ [HH13].

For an *open* IMC to be put in parallel with an unknown environment, the optimal scheduler is computed so that it optimizes the guarantee against all possible environments. Formally, for an IMC $\mathcal{C} = (C, \mathbb{A}\mathrm{ct}^\tau, \hookrightarrow, \rightsquigarrow, c_0)$ and an environment IMC $\mathcal{E}$ with the same action alphabet $\mathbb{A}\mathrm{ct}^\tau$, we introduce a composition $\mathcal{C}|\mathcal{E} = (\mathcal{C} \parallel_{\mathbb{A}\mathrm{ct}} \mathcal{E}) \setminus \mathbb{A}\mathrm{ct}$ where all open actions are hidden, yielding a closed system. In order to compute guarantees on $\mathcal{C}|\mathcal{E}$ provided we use a scheduler $\sigma$ in $\mathcal{C}$, we consider schedulers $\pi$ of $\mathcal{C}|\mathcal{E}$ that *respect* $\sigma$ on the internal actions of $\mathcal{C}$, written $\pi \in \mathfrak{S}_\sigma(\mathcal{C}|\mathcal{E})$; the formal definition is below. The *value of compositional time-bounded reachability* is then defined in [BHK+12] as

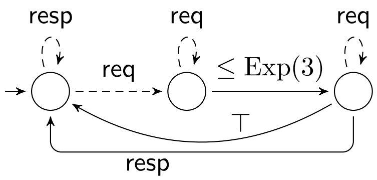
$$\sup_{\sigma \in \mathfrak{S}(\mathcal{C})} \inf_{\substack{\mathcal{E} \in \mathrm{ENV} \\ \pi \in \mathfrak{S}_\sigma(\mathcal{C}|\mathcal{E})}} \mathcal{P}^{\pi}_{\mathcal{C}|\mathcal{E}}\left[\Diamond^{\leq T} G\right]$$

where ENV denotes the set of all IMC with the action alphabet $\mathbb{A}\mathrm{ct}^\tau$ and $\Diamond^{\leq T} G$ is the set of runs that reach $G$ in the first component before $T$. Now $\pi$ *respects* $\sigma$ on internal actions of $\mathcal{C}$ if for every path $\mathfrak{p} = (c_0, e_0)\, t_1 \cdots t_n(c_n, e_n)$ of $\mathcal{C}|\mathcal{E}$ there is $p \in [0,1]$ such that for each internal transition $c_n \overset{\tau}{\hookrightarrow} c$ of $\mathcal{C}$, we have $\pi(\mathfrak{p})(c, e_n) = p \cdot \sigma(\mathfrak{p}_\mathcal{C})(c)$. Here $\mathfrak{p}_\mathcal{C}$ is the projection of $\mathfrak{p}$ where $\sigma$ can only see the path of moves in $\mathcal{C}$ and not in which states $\mathcal{E}$ is. Formally, we define *observation* of a path $\mathfrak{p} = (c_0, e_0)\, t_1 \cdots t_n(c_n, e_n)$ as $\mathfrak{p}_\mathcal{C} = c_0 t_1 \cdots t_n c_n$ where each maximal consecutive sequence $t_i\, c_i \cdots t_j\, c_j$ with $c_k = c_i$ for all $i \leq k \leq j$ is rewritten to $t_i\, c_i$. This way, $\sigma$ ignores precisely the internal steps of $\mathcal{E}$.

## 3.1 Specifications of Environments

In the second example in the introduction, without any assumptions on the environment only zero guarantees could be derived. The component was thus indistinguishable from an entirely useless one. In order to get a better guarantee, we introduce a formalism to specify assumptions on the behaviour of environments.

*Example 1.* In the mentioned example, if we knew that after an occurrence of req the environment is ready to synchronize on resp in time distributed according to Exp(3) or faster, we would be able to derive a guarantee of 0.26. We will depict this assumption as shown below.



The dashed arrows denote *may* transitions, which may or may not be available, whereas the full arrows denote *must* transitions, which the environment is ready to synchronize on. Full arrows are further used for time transitions.

Although such a system resembles a timed automaton, there are several fundamental differences. Firstly, the time constraints are given by probability distributions instead of constants. Secondly, there is only one clock that, moreover, gets reset whenever the state is *changed.* Thirdly, we allow modalities of may and must transitions. Further, as usual with timed or stochastic specifications, we require determinism.

**Definition 3 (MCA syntax).** *A* continuous time constraint *is either* $\top$ *or of the form* $\bowtie d$ *with* $\bowtie \in \{\leq, \geq\}$ *and* $d$ *a continuous distribution. We denote the set of all continuous time constraints by* $\mathcal{CTC}$. *A* modal continuous-time automaton (MCA) *over* $\Sigma$ *is a tuple* $\mathcal{S} = (Q, q_0, \dashrightarrow, \longrightarrow, \rightsquigarrow)$, *where*

- $Q$ *is a non-empty finite set of* locations *and* $q_0 \in Q$ *is an* initial location,
- $\longrightarrow, \dashrightarrow : Q \times \Sigma \to Q$ *are* must *and* may *transition functions, respectively, satisfying* $\longrightarrow \subseteq \dashrightarrow$,
- $\rightsquigarrow : Q \to \mathcal{CTC} \times Q$ *is a* time flow *function.*

We have seen an example of an MCA in the previous example. Note that upon taking req from the first state, the waiting time is chosen and the waiting starts. On the other hand, when req *self-loop* is taken in the middle state, the waiting process is not restarted, but continues on the background independently.(1) We introduce this independence as a useful feature to model properties as "response follows within some time after request" in the setting with concurrently running processes. Further, we have transitions under $\top$ corresponding to "$> 0$", meaning there is no restriction on the time distribution except that the transition takes non-zero time. We formalize this in the following definition. With other respects, the semantics of may and must transitions follows the standards of modal transition systems [LT88].

**Definition 4 (MCA semantics).** *An IMC* $\mathcal{E} = (E, \mathbb{A}\mathrm{ct}^\tau, \hookrightarrow, \rightsquigarrow, e_0)$ conforms *to an MCA specification* $\mathcal{S} = (Q, q_0, \dashrightarrow, \longrightarrow, \rightsquigarrow)$, *written* $\mathcal{E} \models \mathcal{S}$, *if there is a* satisfaction relation $\mathcal{R} \subseteq E \times Q$ *containing* $(e_0, q_0)$ *and satisfying for each* $(e, q) \in \mathcal{R}$ *that whenever*

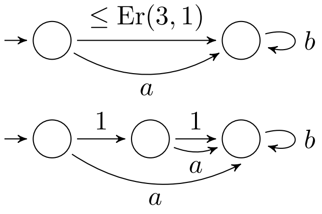
1. $q \xrightarrow{a} q'$ *then there is some* $e \overset{a}{\hookrightarrow} e'$ *and if, moreover,* $q \neq q'$ *then* $e'\mathcal{R}q'$,
2. $e \overset{a}{\hookrightarrow} e'$ *then there is (unique)* $q \overset{a}{\dashrightarrow} q'$ *and if, moreover,* $q \neq q'$ *then* $e'\mathcal{R}q'$,
3. $e \overset{\tau}{\hookrightarrow} e'$ *then* $e'\mathcal{R}q$,
4. $q \overset{ctc}{\rightsquigarrow} q'$ *then for every IMC* $\mathcal{C}$ *and every scheduler* $\pi \in \mathfrak{S}(\mathcal{C}|e)$,(2) *there is a random variable* $Stop : \mathbb{R}\mathrm{uns} \to \mathbb{R}_{>0}$ *on the probability space* $(\mathbb{R}\mathrm{uns}, \mathcal{F}, \mathcal{P}^\pi_{\mathcal{C}|e})$ *such that*
   - *if* $ctc$ *is of the form* $\bowtie d$ *then the cumulative distribution function of* $Stop$ *is point-wise* $\bowtie$ *cumulative distribution function of* $d$ *(there are no constraints when* $ctc = \top$*), and*
   - *for every run* $\rho$ *of* $\mathcal{C}|e$ *under* $\pi$, *either a transition corresponding to synchronization on action* $a$ *with* $q \overset{a}{\dashrightarrow} q' \neq q$ *is taken before time* $Stop(\rho)$, *or*
     - *the state* $(c, e')$ *visited at time* $Stop(\rho)$ *satisfies* $e'\mathcal{R}q'$, *and*
     - *for all states* $(\bar{c}, \bar{e})$ *visited prior to that, whenever*
       *(a)* $q \xrightarrow{a} q'$ *then there is* $e \overset{a}{\hookrightarrow} e'$,
       *(b)* $e \overset{a}{\hookrightarrow} e'$ *then there is* $q \overset{a}{\dashrightarrow} q'$.

*The semantics of* $\mathcal{S}$ *is the set* $[\![\mathcal{S}]\!] = \{\mathcal{E} \in \mathrm{IMC} \mid \mathcal{E} \models \mathcal{S}\}$ *of all conforming IMC.*

(1) This makes no difference for memoryless exponential distributions, but for all other distributions it does.

(2) Here $e$ stands for the IMC $\mathcal{E}$ with the initial state $e$.

*Example 2.* We illustrate this definition. Consider the MCA on the right above specifying that $a$ is ready and $b$ will be ready either immediately after taking $a$ or within the time distributed according to the Erlang distribution $\mathrm{Er}(3,1)$, which is a convolution of three $\mathrm{Exp}(1)$ distributions. The IMC below conforms to this specification (here, $\mathit{Stop} \sim \mathit{Er}(2,1)$ can be chosen). However, observe that it would not conform, if there was no transition under $a$ from the middle to the right state. Satisfying the modalities throughout the waiting is namely required by the last bullet of the previous definition.



## 3.2 Assume-Guarantee Optimization

We can now formally state what guarantees on time-bounded reachability we can derive provided the unknown environment conforms to a specification $\mathcal{S}$. Given an *open* IMC $\mathcal{C}$, a set of goal states $G \subseteq C$ and a time bound $T \in \mathbb{R}_{\geq 0}$, the *value of compositional time-bounded reachability conditioned by an MCA* $\mathcal{S}$ is defined as

$$v_{\mathcal{S}}(\mathcal{C}) \quad := \quad \sup_{\sigma \in \mathfrak{S}(\mathcal{C})} \; \inf_{\substack{\mathcal{E} \in \mathrm{ENV}: \mathcal{E} \models \mathcal{S} \\ \pi \in \mathfrak{S}_{\sigma}(\mathcal{C}|\mathcal{E})}} \mathcal{P}^{\pi}_{\mathcal{C}|\mathcal{E}} \left[ \Diamond^{\leq T} G \right]$$

In this paper, we pose a technical assumption on the set of schedulers of $\mathcal{C}$. For some clock resolution $\delta > 0$, we consider only such schedulers $\sigma$ that take the same decision for any pair of paths $c_0 t_1 \dots t_n c_n$ and $c_0 t'_1 \dots t'_n c_n$ with $t_i$ and $t'_i$ equal when rounded down to a multiple of $\delta$ for all $1 \leq i \leq n$. This is no practical restriction as it is not possible to achieve arbitrary resolution of clocks when implementing the scheduler. Observe this is a safe assumption as it is *not* imposed on the unknown environment.

We consider specifications $\mathcal{S}$ where distributions have differentiable density functions. In the rest of the paper we show how to approximate $v_{\mathcal{S}}(\mathcal{C})$ for such $\mathcal{S}$. Firstly, we make a product of the given IMC and MCA. Secondly, we transform the product to a game. This game is further discretized into a partially observable stochastic game played on a dag where the quantitative reachability is solved. For full proofs, see [HKK13].

## 4 Product of IMC and Specification

In this section, we first translate MCA $\mathcal{S}$ into a sequence of IMC $(\mathcal{S}_i)_{i \in \mathbb{N}}$. Second, we combine the given IMC $\mathcal{C}$ with the sequence $(\mathcal{S}_i)_{i \in \mathbb{N}}$ into a sequence of product IMC $(\mathcal{C} \times \mathcal{S}_i)_{i \in \mathbb{N}}$ that will be further analysed. The goal is to reduce the case where the unknown environment is bound by the specification to a setting where we solve the problem for the product IMC while quantifying over all possible environments (satisfying only a simple technical assumption discussed at the end of the section), denoted $\mathrm{ENV}'$. The reason why we need a sequence of products instead of one product is that we need to approximate arbitrary distributions with more and more precise and detailed hyper-Erlang distributions expressible in IMC. Formally, we want to define the sequence of the products $\mathcal{C} \times \mathcal{S}_i$ so that

$$v_{product}(\mathcal{C} \times \mathcal{S}_i) \quad := \quad \sup_{\sigma \in \mathfrak{S}(\mathcal{C})} \quad \inf_{\substack{\mathcal{E} \in \mathrm{ENV}' \\ \pi \in \mathfrak{S}_\sigma((\mathcal{C} \times \mathcal{S}_i)|\mathcal{E})}} \quad \mathcal{P}^{\pi}_{(\mathcal{C} \times \mathcal{S}_i)|\mathcal{E}}\left[\Diamond^{\leq T} G\right]$$

approximates the compositional value:

**Theorem 1.** *For every IMC $\mathcal{C}$ and MCA $\mathcal{S}$, $v_{\mathcal{S}}(\mathcal{C}) = \lim_{i \to \infty} v_{product}(\mathcal{C} \times \mathcal{S}_i)$.*

Note that in $v_{product}$, $\sigma$ is a scheduler over $\mathcal{C}$, not the whole product $\mathcal{C} \times \mathcal{S}_i$.[3] Constructing a product with the specification intuitively corresponds to adding a known, but uncontrollable and unobservable part of the environment to $\mathcal{C}$. We proceed as follows: We translate the MCA $\mathcal{S}$ into a sequence of IMC $\mathcal{S}_i$ and then the product will be defined as basically a parallel composition of $\mathcal{C}$ and $\mathcal{S}_i$.

There are two steps in the translation of $\mathcal{S}$ to $\mathcal{S}_i$. Firstly, we deal with the modal transitions. A *may* transition under $a$ is translated to a standard external transition under $a$ that has to synchronize with $a$ in both $\mathcal{C}$ and $\mathcal{E}$ simultaneously, so that the environment may or may not let the synchronization occur. Further, each *must* transition under $a$ is replaced by an external transition, that synchronizes with $a$ in $\mathcal{C}$, but is hidden before making product with the environment. This way, we guarantee that $\mathcal{C}$ can take $a$ and make progress no matter if the general environment $\mathcal{E}$ would like to synchronize on $a$ or not.

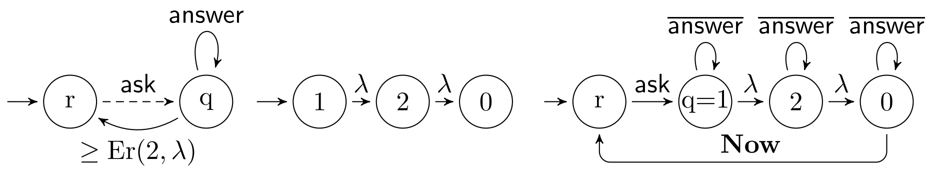
Formally, the must transitions are transformed into special "barred" transitions that will be immediately hidden in the product $\mathcal{C} \times \mathcal{S}_i$ as opposed to transitions arising from may transitions. Let $\overline{\mathbb{A}\mathrm{ct}} = \{\bar{a} \mid a \in \mathbb{A}\mathrm{ct}\}$ denote a fresh copy of the original alphabet. We replace all modal transitions as follows

– whenever $q \overset{a}{\dashrightarrow} r$ set $q \overset{a}{\hookrightarrow} r$,
– whenever $q \xrightarrow{a} r$ set $q \overset{\bar{a}}{\hookrightarrow} r$.

The second step is to deal with the *timed* transitions, especially with the constraints of the form $\bowtie d$. Such a transition is, roughly speaking, replaced by a phase-type approximation of $d$. This is a continuous-time Markov chain (an IMC with only timed transitions) with a sink state such that the time to reach the sink state is distributed with $d'$. For any continuous distribution $d$, we can find such $d'$ arbitrarily close to $d$.

*Example 3.* Consider the following MCA on the left. It specifies that whenever ask is taken, it cannot be taken again for at least the time distributed by $\mathrm{Er}(2, \lambda)$ and during all that time, it is ready to synchronize on answer. This specifies systems that are allowed to ask, but not too often, and whenever they ask, they must be ready to receive (possibly more) answers for at least the specified time.

[3] Here we overload the notation $\mathfrak{S}_\sigma((\mathcal{C} \times \mathcal{S}_i)|\mathcal{E})$ introduced for pairs in a straightforward way to triples, where $\sigma$ ignores both the second and the third components.


answer
ask
r
q
≥ Er(2, λ)
1
2
0
λ
λ
$\overline{\text{answer}}$
$\overline{\text{answer}}$
$\overline{\text{answer}}$
r
ask
q=1
2
0
λ
λ
Now

After performing the first step of replacing the modal transitions as described above, we proceed with the second step as follows. We replace the timed transition with a phase-type, e.g. the one represented by the IMC in the middle. Observe that while the Markovian transitions are taken, answer must still be available. Hence, we duplicate the corresponding self-loops on all the new states. Further, since the time constraint is of the form $\geq$, getting to the state $(q, 0)$ does not guarantee that we already get to the state $r$. It can possibly take longer. To this end, we connect the states $(q, 0)$ and $r$ by a special external action **Now**. Since this action is synchronized with $\mathcal{E} \in \text{ENV}'$, the environment can block the progress for arbitrarily long time. Altogether, we obtain the IMC on the right.

In the case of "$\leq$" condition, we would instead add the **Now** transition from each auxiliary state to the sink, which could instead shorten the waiting time.

When constructing $\mathcal{S}_i$, we replace each distribution $d$ with its hyper-Erlang phase-type approximation $d_i$ with $i$ branches of lengths 1 to $i$ and rates $\sqrt{i}$ in each branch. For formal description, see [HKK13]. Formally, let $\textbf{Now} \notin \mathbb{A}\text{ct} \cup \overline{\mathbb{A}\text{ct}}$ be a fresh action. We replace all timed transitions as follows:

– whenever $q \overset{\top}{\rightsquigarrow} r$ such that $q \neq r$ set $q \overset{\textbf{Now}}{\hookrightarrow} r$,
– whenever $q \overset{\bowtie d}{\rightsquigarrow} r$ where the phase-type $d_i$ corresponds to a continuous-time Markov chain (IMC with only timed transitions) with the set of states $D$, the initial state *1* and the sink state *0*, then
  1. identify the states $q$ and *1*,
  2. for every $u \in D$ and $q \overset{\alpha}{\hookrightarrow} q$, set $u \overset{\alpha}{\hookrightarrow} u$,
  3. for every $u \in D$ and $q \overset{\alpha}{\hookrightarrow} p$ with $p \neq q$, set $u \overset{\alpha}{\hookrightarrow} p$,
  4. if $\bowtie = \leq$, then identify $r$ and *0*, and set $u \overset{\textbf{Now}}{\hookrightarrow} r$ for each $u \in D$,
  5. if $\bowtie = \geq$, then set $0 \overset{\textbf{Now}}{\hookrightarrow} r$.

Intuitively, the new timed transitions model the delays, while in the "$\leq$" case, the action **Now** can be taken to speed up the process of waiting, and in the "$\geq$" case, **Now** can be used to block further progress even after the delay has elapsed.

The product is now the parallel composition of $\mathcal{C}$ and $\mathcal{S}_i$, where each action $\bar{a}$ synchronizes with $a$ and the result is immediately hidden. Formally, the product $\mathcal{C} \times \mathcal{S}$ is defined as $\mathcal{C} \;\|^{\textbf{PC6}}_{\mathbb{A}\text{ct}\cup\overline{\mathbb{A}\text{ct}}}\; \mathcal{S}_i$, where $\|^{\textbf{PC6}}_{\mathbb{A}\text{ct}\cup\overline{\mathbb{A}\text{ct}}}$ is the parallel composition with one additional axiom:

**(PC6)** $s_1 \overset{a}{\hookrightarrow} s_1'$ and $s_2 \overset{\bar{a}}{\hookrightarrow} s_2'$ implies $(s_1, s_2) \overset{\tau}{\hookrightarrow} (s_1', s_2')$,

saying that $a$ synchronizes also with $\bar{a}$ and, in that case, is immediately hidden (and any unused $\bar{a}$ transitions are thrown away).

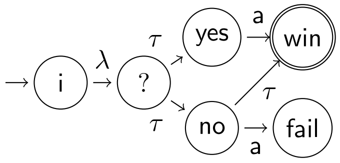
The idea of **Now** is that it can be taken in arbitrarily short, but non-zero time. To this end, we define $\text{ENV}'$ in the definition of $v_{product}(\mathcal{C} \times \mathcal{S}_i)$ to denote all environments where **Now** is only available in states that can be entered by only a Markovian transition. Due to this requirement, each **Now** can only be taken after waiting for some time.

## 5 Controller-Environment Games

So far, we have reduced our problem to computing $\lim_{i\to\infty} v_{product}(\mathcal{C} \times \mathcal{S}_i)$. Note that we are still quantifying over unknown environments. Further, the behaviour of each environment is limited by the uncontrollable *stochastic* flow of time caused by its Markovian transitions. This setting is still too difficult to be solved directly. Therefore, in this section, we reduce this setting to one, where the stochastic flow of time of the environment (limited in an unknown way) is replaced by a free *non-deterministic* choice of the *second player*.

We want to turn the product IMC $\mathcal{C} \times \mathcal{S}_i$ into a two-player *controller–environment game* (CE game) $\mathcal{G}_i$, where player **con** controls the decisions over internal transitions in $\mathcal{C}$; and player **env** simulates the environment including speeding-up/slowing-down $\mathcal{S}$ using **Now** transitions. In essence, **con** chooses in each state with internal transitions one of them, and **env** chooses in each state with external (and hence synchronizing) transitions either which of them should be taken, or a *delay* $d \in \mathbb{R}_{>0}$ during which no synchronization occurs. The internal and external transitions take zero time to be executed if chosen. Otherwise, the game waits until either the delay $d$ elapses or a Markovian transition occurs.

This is the approach taken in [BHK+12] where no specification is considered. However, there is a catch. This construction is only correct under the assumption of [BHK+12] that there are no states of $\mathcal{C}$ with both external and internal transitions available.

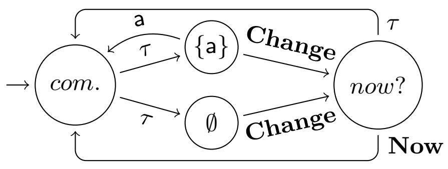


*Example 4.* Consider the IMC $\mathcal{C}$ on the right (for instance with a trivial specification not restricting the environment). Note that there are both internal and external actions available in no.

As $\tau$ transitions take zero time, the environment $\mathcal{E}$ must spend almost all the time in states without $\tau$. Hence, when ? is entered, $\mathcal{E}$ is almost surely in such a state $e$. Now $\tau$ form ? is taken and $\mathcal{E}$ cannot move to another state when yes/no is entered. Since action a either *is* or *is not* available in $e$, the environment cannot choose to synchronize in no and not to synchronize in yes. As a result, the environment *"commits" in advance* to synchronize over a either in both yes and no or in none of them. Therefore, in the game we define, **env** cannot completely freely choose which external transition is/is not taken. Further, note that the scheduler of $\mathcal{C}$ cannot observe whether a is currently available in $\mathcal{E}$, which intrinsically induces imperfect information.

In order to transfer these "commitments" to the game, we again make use of the compositionality of IMC and put the product $\mathcal{C} \times \mathcal{S}_i$ in parallel with an IMC *Commit* and then define the game on the result.

The action alphabet of *Commit* is $\mathbb{A}\mathrm{ct} \cup \{\mathbf{Now}, \mathbf{Change}\}$ and the state space is $2^{\mathbb{A}\mathrm{ct}} \cup \{\mathit{commit}, \mathit{now?}\}$ (in the figure, $\mathbb{A}\mathrm{ct} = \{\mathsf{a}\}$; for formal description, see [HKK13]). State $A \subseteq \mathbb{A}\mathrm{ct}$ corresponds to $\mathcal{E}$ being committed to the set of currently available actions $A$. Thus $A \overset{a}{\hookrightarrow} \mathit{commit}$ for each $a \in A$. This commitment must be respected until the state of $\mathcal{E}$ is changed: either (1) by an external transition from the commitment set (which in *Commit* leads to the state *commit* where a new commitment is immediately chosen); or (2) by a **Change** transition (indicating the environment changed its state due to its Markovian transition).



The game $\mathcal{G}_i$ is played on the arena $\big(\mathcal{C} \times \mathcal{S}_i \parallel_{\mathbb{A}\mathrm{ct}\cup\{\mathbf{Now}\}} \mathit{Commit}\big) \setminus \big(\mathbb{A}\mathrm{ct} \cup \{\mathbf{Now}\}\big)$ with its set of states denoted by $G_i$. Observe that external actions have either been hidden (whenever they were available in the commitment), or discarded (whenever not present in the current commitment). The only external action that remains is **Change**. The game $\mathcal{G}_i$ is played as follows. There are two types of states: *immediate* states with some $\tau$ transitions available and *timed* states with no $\tau$ available. The game starts in $v_0 = (c_0, q_0, \mathit{commit})$.

- In an immediate state $v_n = (c, q, e)$, **con** chooses a probability distribution over transitions corresponding to the internal transitions in $\mathcal{C}$ (if there are any). Then, **env** either approves this choice (chooses $\checkmark$) and $v_{n+1}$ is chosen randomly according to this distribution, or rejects this choice and chooses a $\tau$ transition to some $v_{n+1}$ such that the transition does *not* correspond to any internal transitions of $\mathcal{C}$. Then the game moves at time $t_{n+1} = t_n$ to $v_{n+1}$.
- In a timed state $v_n = (c, q, e)$, **env** chooses a delay $d > 0$. Then Markovian transitions (if available) are resolved by randomly sampling a time $t$ according to the exponential distribution with rate $\mathrm{R}(v_n)$ and randomly choosing a target state $v_{n+1}$ where each $v_n \overset{\lambda}{\rightsquigarrow} v$ is chosen with probability $\lambda/\mathrm{R}(v_n)$.
  - If $t < d$, $\mathcal{G}_i$ moves at time $t_{n+1} = t_n + t$ to $v_{n+1}$, (Markovian transition wins)
  - else $\mathcal{G}_i$ moves at time $t_{n+1} = t_n + d$ to $(c, q, \mathit{now?})$. ($\mathcal{E}$ takes **Change**)

This generates a run $v_0 t_1 v_1 t_1 \cdots$. The set $(G_i \times \mathbb{R}_{\geq 0})^* \times G_i$ of prefixes of runs is denoted $\mathbb{H}\mathrm{istories}(\mathcal{G})$. We formalize the choice of **con** as a *strategy* $\sigma : \mathbb{H}\mathrm{istories}(\mathcal{G}_i) \to \mathcal{D}(G_i)$. We further allow the **env** to randomize and thus his *strategy* is $\pi : \mathbb{H}\mathrm{istories}(\mathcal{G}_i) \to \mathcal{D}(\{\checkmark\} \cup G_i) \cup \mathcal{D}(\mathbb{R}_{>0})$. We denote by $\Sigma$ and $\Pi$ the sets of all strategies of the players **con** and **env**, respectively.

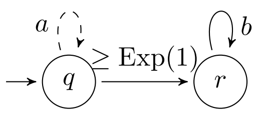
Since **con** is not supposed to observe the state of the specification and the state of *Commit*, we consider in $\Sigma$ only those strategies that satisfy $\sigma(p) = \sigma(p')$, whenever *observations* of $p$ and $p'$ are the same. Like before, the observation of $(c_0, q_0, e_0) t_1 \cdots t_n (c_n, q_n, e_n) \in \mathbb{H}\mathrm{istories}(\mathcal{G})$ is a sequence obtained from $c_0 t_1 \cdots t_n c_n$ by replacing each maximal consecutive sequence $t_i\, c_i \cdots t_j\, c_j$ with all $c_k$ the same, by $t_i\, c_i$. This replacement takes place so that the player cannot observe transitions that do not affect $\mathcal{C}$. Notice that now $\mathfrak{S}(\mathcal{C})$ is in one-to-one correspondence with $\Sigma$. Further, in order to keep CE games out of Zeno behaviour, we consider in $\Pi$ only those strategies for which the induced Zeno runs have zero measure, i.e. the sum of the chosen delays diverges almost surely no matter what **con** is doing. The *value of* $\mathcal{G}_i$ is now defined as

$$v_{\mathcal{G}_i} \quad := \quad \sup_{\sigma\in\Sigma}\inf_{\pi\in\Pi} \mathcal{P}^{\sigma,\pi}_{\mathcal{G}_i}\left[\Diamond^{\leq T} G\right]$$

where $\mathcal{P}^{\sigma,\pi}_{\mathcal{G}_i}\left[\Diamond^{\leq T} G\right]$ is the probability of all runs of $\mathcal{G}_i$ induced by $\sigma$ and $\pi$ and reaching a state with the first component in $G$ before time $T$. We now show that it coincides with the value of the $i$th product:

**Theorem 2.** *For every IMC $\mathcal{C}$, MCA $\mathcal{S}$, $i \in \mathbb{N}$, we have $v_{\mathcal{G}_i} = v_{product}(\mathcal{C}\times\mathcal{S}_i)$.*

This result allows for approximating $v_{\mathcal{S}}(\mathcal{C})$ through computing $v_{\mathcal{G}_i}$'s. However, from the algorithmic point of view, we would prefer approximating $v_{\mathcal{S}}(\mathcal{C})$ by solving a single game $\mathcal{G}$ whose value $v_{\mathcal{G}}$ we could approximate directly. This is indeed possible. But first, we need to clarify, why the approximation sequence $\mathcal{S}_i$ was crucial even in the case where all distributions of $\mathcal{S}$ are already exponential.



Consider the MCA on the right and a conforming environment $\mathcal{E}$, in which $a$ is available iff $b$ becomes available within 0.3 time units. If Player **env** wants to simulate this behaviour, he needs to know how long the transition to $r$ is going to take so that he can plan his behaviour freely, only sticking to satisfying the specification. If we translate Exp(1) directly to a single Markovian transition (with no error incurred), **env** knows nothing about this time as exponential distributions are memoryless. On the other hand, with finer hyper-Erlang, he knows how long the current branch of hyper-Erlang is roughly going to take. In the limit, he knows the precise waiting time right after coming to $q$.

To summarize, **env** is too weak in $\mathcal{G}_i$, because it lacks the information about the precise time progress of the specification. The environment needs to know how much time is left before changing the location of $\mathcal{S}$. Therefore, the game $\mathcal{G}$ is constructed from $\mathcal{G}_1$ by multiplying the state space with $\mathbb{R}_{\geq 0}$ where we store the exact time to be waited. After the product changes the state so that the specification component switches to a state with $\bowtie d$ constraint, this last component is overwritten with a number generated according to $d$. This way, the environment knows precisely how much time is left in the current specification location. This corresponds to the infinitely precise hyper-Erlang, where we at the beginning randomly enter a particular branch, which is left in time with Dirac distribution. For more details, see [HKK13].

Denoting the *value of* $\mathcal{G}$ by $v_{\mathcal{G}} := \sup_{\sigma\in\Sigma}\inf_{\pi\in\Pi} \mathcal{P}^{\sigma,\pi}_{\mathcal{G}}\left[\Diamond^{\leq T} G\right]$, we obtain:

**Theorem 3.** *For every IMC $\mathcal{C}$ and MCA $\mathcal{S}$, we have $v_{\mathcal{G}} = \lim_{i\to\infty} v_{\mathcal{G}_i}$.*

## 6 Approximation Using Discrete-Time PO Games

In this section, we briefly discuss the approximation of $v_{\mathcal{G}}$ by a discrete time turn-based partial-observation stochastic game $\Delta$. The construction is rather standard; hence, we do not treat the technical difficulties in great detail (see [HKK13]). We divide the time bound $T$ into $N$ intervals of length $\kappa = T/N$ such that the clock resolution $\delta$ (see Section 3.2) satisfies $\delta = n\kappa$ for some $n \in \mathbb{N}$.

1. We enhance the state space with a counter $i \in \{0, \dots, N\}$ that tracks that $i \cdot \kappa$ time has already elapsed. Similarly, the $\mathbb{R}_{\geq 0}$-component of the state space is discretized to $\kappa$-multiples. In timed states, time is assumed to pass exactly by $\kappa$. In immediate states, actions are assumed to take zero time.
2. We let at most one Markovian transition occur in one step in a timed state.
3. We unfold the game into a tree until on each branch a timed state with $i = N$ is reached. Thereafter, $\Delta$ stops. We obtain a graph of size bounded by $b^{\leq N \cdot |G|}$ where $b$ is the maximal branching and $G$ is the state space of $\mathcal{G}$.

Let $\Sigma_\Delta$ and $\Pi_\Delta$ denote the set of randomized history-dependent strategies of **con** and **env**, respectively, where player **con** observes in the history only the first components of the states, i.e. the states of $\mathcal{C}$, and the elapsed time $\lfloor i/n \rfloor$ up to the precision $\delta$. Then $v_\Delta := \sup_{\sigma \in \Sigma_\Delta} \inf_{\pi \in \Pi_\Delta} \mathcal{P}_\Delta^{\sigma,\pi}(\Diamond G)$ denotes the value of the game $\Delta$ where $\mathcal{P}_\Delta^{\sigma,\pi}(\Diamond G)$ is the probability of the runs of $\Delta$ induced by $\sigma$ and $\pi$ and reaching a state with first component in $G$. Let $b$ be a constant bounding (a) the sum of outgoing rates for any state of $\mathcal{C}$, and (b) densities and their first derivative for any distribution in $\mathcal{S}$.

**Theorem 4.** *For every IMC $\mathcal{C}$ and MCA $\mathcal{S}$, $v_\mathcal{G}$ is approximated by $v_\Delta$:*

$$|v_\mathcal{G} - v_\Delta| \quad \leq \quad 10\kappa(bT)^2 \ln \tfrac{1}{\kappa}.$$

*A strategy $\sigma^*$ optimal in $\Delta$ defines a strategy $(10\kappa(bT)^2 \ln \frac{1}{\kappa})$-optimal in $\mathcal{G}$. Further, $v_\Delta$ and $\sigma^*$ can be computed in time polynomial in $|\Delta|$, hence in time $2^{\mathcal{O}(|\mathcal{G}|)}$.*

The proof of the error bound extends the technique of the previous bounds of [ZN10] and [BHK$^+$12]. Its technical difficulty stems from partial observation and from semi-Markov behaviour caused by the arbitrary distributions in the specification. The game is unfolded into a tree in order to use the result of [KMvS94]. Without the unfolding, the best known (naive) solution would be a reduction to the theory of reals, yielding an EXPSPACE algorithm.

## 7 Summary

We have introduced an assume-guarantee framework for IMC. We have considered the problem to approximate the guarantee on time-bounded reachability properties in an unknown environment $\mathcal{E}$ that satisfies a given assumption. The assumptions are expressed in a new formalism, which introduces continuous time constraints. The algorithmic solution results from Theorems 1 to 4:

**Corollary 1.** *For every IMC $\mathcal{C}$ and MCA $\mathcal{S}$ and $\varepsilon > 0$, a value $v$ and a scheduler $\sigma$ can be computed in exponential time such that $|v_\mathcal{S}(\mathcal{C}) - v| \leq \varepsilon$ and $\sigma$ is $\varepsilon$-optimal in $v_\mathcal{S}(\mathcal{C})$.*

In future work, we want to focus on identifying structural subclasses of IMC allowing for polynomial analysis.

**Acknowledgement.** We thank Tomáš Brázdil and Vojtěch Řehák for fruitful discussions and for their feedback.

## References

[AH96] Alur, R., Henzinger, T.A.: Reactive modules. In: LICS, pp. 207–218 (1996)

[BF09] Bouyer, P., Forejt, V.: Reachability in stochastic timed games. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S., Thomas, W. (eds.) ICALP 2009, Part II. LNCS, vol. 5556, pp. 103–114. Springer, Heidelberg (2009)

[BHK$^+$12] Brázdil, T., Hermanns, H., Krčál, J., Křetínský, J., Řehák, V.: Verification of open interactive markov chains. In: FSTTCS, pp. 474–485 (2012)

[BHKH05] Baier, C., Hermanns, H., Katoen, J.-P., Haverkort, B.R.: Efficient computation of time-bounded reachability probabilities in uniform continuous-time Markov decision processes. Theor. Comp. Sci. 345(1), 2–26 (2005)

[BS11] Buchholz, P., Schulz, I.: Numerical Analysis of Continuous Time Markov Decision processes over Finite Horizons. Computers and Operations Research 38, 651–659 (2011)

[CD10] Chatterjee, K., Doyen, L.: The complexity of partial-observation parity games. In: Fermüller, C.G., Voronkov, A. (eds.) LPAR-17. LNCS, vol. 6397, pp. 1–14. Springer, Heidelberg (2010)

[DLL$^+$12] David, A., Larsen, K.G., Legay, A., Møller, M.H., Nyman, U., Ravn, A.P., Skou, A., Wasowski, A.: Compositional verification of real-time systems using ECDAR. STTT 14(6), 703–720 (2012)

[EKN$^+$12] Esteve, M.-A., Katoen, J.-P., Nguyen, V.Y., Postma, B., Yushtein, Y.: Formal correctness, safety, dependability and performance analysis of a satellite. In: Proc. of ICSE. ACM and IEEE Press (2012)

[GHKN12] Guck, D., Han, T., Katoen, J.-P., Neuhäußer, M.R.: Quantitative timed analysis of interactive markov chains. In: Goodloe, A.E., Person, S. (eds.) NFM 2012. LNCS, vol. 7226, pp. 8–23. Springer, Heidelberg (2012)

[HH13] Hatefi, H., Hermanns, H.: Improving time bounded reachability computations in interactive Markov chains. In: Arbab, F., Sirjani, M. (eds.) FSEN 2013. LNCS, vol. 8161. Springer, Heidelberg (2013)

[HK09] Hermanns, H., Katoen, J.-P.: The how and why of interactive Markov chains. In: de Boer, F.S., Bonsangue, M.M., Hallerstede, S., Leuschel, M. (eds.) FMCO 2009. LNCS, vol. 6286, pp. 311–338. Springer, Heidelberg (2010)

[HKK13] Hermanns, H., Krčál, J., Křetínský, J.: Compositional verification and optimization of interactive markov chains. CoRR, abs/1305.7332 (2013)

[HKR$^+$10] Haverkort, B.R., Kuntz, M., Remke, A., Roolvink, S., Stoelinga, M.I.A.: Evaluating repair strategies for a water-treatment facility using Arcade. In: Proc. of DSN, pp. 419–424 (2010)

[HMP01] Henzinger, T.A., Minea, M., Prabhu, V.S.: Assume-guarantee reasoning for hierarchical hybrid systems. In: Di Benedetto, M.D., Sangiovanni-Vincentelli, A.L. (eds.) HSCC 2001. LNCS, vol. 2034, pp. 275–290. Springer, Heidelberg (2001)

[HNP$^+$11] Hahn, E.M., Norman, G., Parker, D., Wachter, B., Zhang, L.: Game-based abstraction and controller synthesis for probabilistic hybrid systems. In: QEST, pp. 69–78 (2011)

[KKLW07] Katoen, J.-P., Klink, D., Leucker, M., Wolf, V.: Three-valued abstraction for continuous-time Markov chains. In: Damm, W., Hermanns, H. (eds.) CAV 2007. LNCS, vol. 4590, pp. 311–324. Springer, Heidelberg (2007)

[KKN09] Katoen, J.-P., Klink, D., Neuhäußer, M.R.: Compositional abstraction for stochastic systems. In: Ouaknine, J., Vaandrager, F.W. (eds.) FORMATS 2009. LNCS, vol. 5813, pp. 195–211. Springer, Heidelberg (2009)

[KMvS94] Koller, D., Megiddo, N., von Stengel, B.: Fast algorithms for finding randomized strategies in game trees. In: STOC, pp. 750–759 (1994)

[KV96] Kupferman, O., Vardi, M.: Module checking. In: Alur, R., Henzinger, T.A. (eds.) CAV 1996. LNCS, vol. 1102, pp. 75–86. Springer, Heidelberg (1996)

[KZH$^+$11] Katoen, J.-P., Zapreev, I.S., Hahn, E.M., Hermanns, H., Jansen, D.N.: The ins and outs of the probabilistic model checker MRMC. Performance Evaluation 68(2), 90–104 (2011)

[LT88] Larsen, K.G., Thomsen, B.: A modal process logic. In: LICS, pp. 203–210 (1988)

[Mar11] Markovski, J.: Towards supervisory control of interactive Markov chains: Controllability. In: ACSD, pp. 108–117 (2011)

[MC81] Misra, J., Mani Chandy, K.: Proofs of networks of processes. IEEE Trans. Software Eng. 7(4), 417–426 (1981)

[RW89] Ramadge, P.J.G., Wonham, W.M.: The control of discrete event systems. Proceedings of the IEEE 77(1) (1989)

[Spr11] Sproston, J.: Discrete-time verification and control for probabilistic rectangular hybrid automata. In: QEST, pp. 79–88 (2011)

[TAKB96] Tasiran, S., Alur, R., Kurshan, R.P., Brayton, R.K.: Verifying abstractions of timed systems. In: Sassone, V., Montanari, U. (eds.) CONCUR 1996. LNCS, vol. 1119, pp. 546–562. Springer, Heidelberg (1996)

[ZN10] Zhang, L., Neuhäußer, M.R.: Model checking interactive Markov chains. In: Esparza, J., Majumdar, R. (eds.) TACAS 2010. LNCS, vol. 6015, pp. 53–68. Springer, Heidelberg (2010)