Uniformity by Construction in the Analysis of Nondeterministic Stochastic Systems

Holger Hermanns Sven Johr

Universität des Saarlandes, FR 6.2 Informatik
D-66123 Saarbrücken, Germany

Abstract

Continuous-time Markov decision processes (CTMDPs) are behavioral models with continuous-time, nondeterminism and memoryless stochastics. Recently, an efficient timed reachability algorithm for CTMDPs has been presented [2], allowing one to quantify, e. g., the worst-case probability to hit an unsafe system state within a safety critical mission time. This algorithm works only for uniform CTMDPs – CTMDPs in which the sojourn time distribution is unique across all states. In this paper we develop a compositional theory for generating CTMDPs which are uniform by construction. To analyze the scalability of the method, this theory is applied to the construction of a fault-tolerant workstation cluster example, and experimentally evaluated using an innovative implementation of the timed reachability algorithm. All previous attempts to model-check this seemingly well-studied example needed to ignore the presence of nondeterminism, because of lacking support for modelling and analysis.

1 Motivation

Nondeterministic stochastic systems combine the behavior of classical labeled transition systems (LTSs), with classical stochastic processes, Markov chains in particular. The most widely known model is that of Markov decision processes in discrete time (DTMDPs) which combines LTS and discrete-time Markov chains (DTMCs). This model originates from the operations research (OR) and planning context. In the setting of concurrency theory, that model is often called probabilistic automata [27, 28], and has been the subject of many studies [24, 22]. Model checking algorithms for DTMDPs are available [3, 7], have been implemented and applied [18].

Continuous-time Markov chains (CTMCs) extend DTMCs with memoryless continuous time. These models have a very rich spectrum of applications, ranging from disk storage dimensioning [26] to signalling transduction networks [6]. In the context of concurrency theory, the extension of CTMCs with nondeterminism has lead to different models, including stochastic transition systems [8] and interactive Markov chains (IMC) [14]. The latter is an orthogonal superposition of LTSs and CTMCs with convenient compositional properties.

On the other hand, continuous-time Markov decision processes (CTMDPs) have been proposed in OR [25], but not many results are available, and model checking of CTMDP has not been addressed successfully thus far. A core ingredient to model check timed safety and liveness properties has been developed in [2] by an algorithm addressing timed-bounded reachability probabilities in a CTMDP. The algorithm computes the maximal probability to reach a set of goal states within a given time bound, among all CTMCs induced by time-abstract schedulers. Unfortunately, the algorithm is of limited generality, since it is restricted to so-called uniform CTMDPs (uCTMDPs), CTMDPs in which the sojourn time distribution is unique across all states.

Nevertheless we have recently made use of this algorithm in the verification of large STATEMATE [4] models of train control systems. The algorithm allowed us to verify properties like: "*The probability to hit a safety-critical system configuration within a mission time of 3 hours is at most 0.01*." for very large systems with about 10^6 states and 10^7 transitions.

The main goal in [4] has been to show how we can generate models (LTSs) from STATEMATE, incorporate timing behavior in a compositional way into them, and finally obtain a uCTMDP, amenable to timed reachability analysis. The LTSs have *huge* intermediate state spaces which required us to develop special symbolic data structures and minimizers to make this trajectory possible. Our tool chain therefore consists of a series of symbolic steps, followed by another series of explicit steps. The main contribution of the present paper is the theoretical foundation for the explicit part of that trajectory, (Section IV(b) and IV(c1) of [4]), namely *uniformity preservation* along the explicit part of the tool chain. Additionally, we describe the construction and transformation steps formally, and prove that uniformity is indeed preserved and that timed reachability properties are preserved as well. A second contribution which goes beyond the work reported in [4] is that we provide insight into the construction and timed reachability analysis algorithm, using a more academic but easily scalable case study. This case study allows us to provide a detailed analysis of the time and memory behavior of our implementation of the algorithm in [2]. It turns out that the implementation is quite efficient, albeit being only prototypical.

As a case study, we focus on the *fault-tolerant workstation cluster* (FTWC). This modeling problem has earlier been studied [13, 18] using CTMCs, although the system is nondeter-

ministic in nature. As an interesting side result, our analysis results are compared to the results obtained in the past using that less faithful modelling style.

The paper is organized as follows. Section 2 introduces the necessary background. Section 3 discusses our compositional uIMC construction approach, while Section 4 describes the transformation to uCTMDPs and the implementation details for the timed reachability algorithm. In Section 5 we apply the discussed steps to the FTWC. Finally, Section 6 concludes the paper.

2 Background

This section introduces the necessary background concerning CTMDPs and IMCs. Disjoint union is denoted $A \dot\cup B$. Distr(Ω) denotes the set of all probability distributions over Ω, and given $\mu \in$ Distr(Ω) the *support* of μ is the smallest closed measurable set A such that $\mu(\Omega \backslash A) = 0$.

Continuous-time Markov decision processes. We introduce a mild variation of continuous-time Markov decision processes [25] where states may have multiple transitions labeled with the same action. As will be evident later, the transformation procedure yields this mild variation of continuous-time Markov decision processes.

Definition 1 (CTMDP) *A* continuous-time Markov decision process *(CTMDP) is a tuple* $(S, L, \mathbf{R}, s_0)$ *where* S *is a finite non-empty set of* states, L *is a finite non-empty set of* transition labels *also called* actions, $\mathbf{R} \subseteq S \times L \times (S \longrightarrow \mathbb{R}^+)$ *is the transition relation,* $s_0 \in S$ *is the* initial state.

Given a transition $(s, a, R) \in \mathbf{R}$ and set $Q \subseteq S$ we denote by $R(Q) := \sum_{s' \in Q} R(s')$ the cumulative rate to reach set Q from state s under transition (s, a, R), R is the *rate function* of (s, a, R). In $\mathbf{R}(s) := \{(s, a, R) \in \{s\} \times L \times (S \longrightarrow \mathbb{R}^+) \mid (s, a, R) \in \mathbf{R}\}$ we collect all transitions emanating from s.

Behavior. The behavior of a CTMDP is as follows. Suppose $(s, a, R) \in \mathbf{R}$ for some $R : S \longrightarrow \mathbb{R}^+$. Then $R(s') > 0$ indicates the existence of an a-transition emanating from s and leading to s'. If there are distinct outgoing transitions of s, one of them is selected nondeterministically. This nondeterminism is resolved by a *scheduler*, see below. Given that a transition labeled a has been selected, the probability of triggering that a-transition from s to s' within t time units equals the outcome of a negative exponential distribution with rate $R(s')$, i. e., $1 - e^{-R(s') \cdot t}$. If $R(s') > 0$ for more than one s' the transition (s, a, R) can be viewed as a hyperedge [10] labeled with a, connecting s with target states s', and possibly different rates for each of the s'. In this case there is a race between the relevant transitions. The probability to reach state s' from state s within t time units given that $(s, a, R) \in \mathbf{R}$ is taken is given by $\Pr_R(s, s') \cdot \left(1 - e^{-E_R \cdot t}\right)$ where $E_R = \sum_{s \in S} R(s)$ denotes the total rate of leaving state s under transition (s, a, R) and $\Pr_R(s, s') := \frac{R(s')}{E_R}$. The first factor of the product denotes the discrete branching probability to reach s' from s under transition (s, a, R), the second factor denotes the sojourn time in s, i. e., a negative exponential distribution with rate E_R. Note, that the sojourn time only depends on the rate function R. We extend the notation for discrete branching probabilities to sets of states by standard summation, i. e., $\Pr_R(s, Q) = \sum_{q \in Q} \Pr_R(s, q)$.

Whenever all of the E_R are equal to each other we call the CTMDP *uniform*, for short uCTMDP. Intuitively, this means that the sojourn time distribution in each state s is the same, independently of the transition chosen.

Paths. A *(timed) path* σ in CTMDP $\mathcal{C} = (S, L, \mathbf{R}, s_0)$ is a possibly infinite sequence of states, labels and time points, i. e.,

$\sigma \in \left(S \times (L \times \mathbb{R}^+ \times S)^*\right) \dot\cup \left(S \times (L \times \mathbb{R}^+ \times S)^\omega\right)$.

By $first(\sigma) = s_0$ we denote the first state of path σ. A finite path $\sigma' = s_0 a_1 t_1 s_1 \ldots a_k t_k s_k$ has *length* k, denoted as $|\sigma'| = k$ and its last state equals $last(\sigma') = s_k$.

Scheduler. A scheduler resolves the nondeterminism inherent in a CTMDP. Most generally, schedulers can be considered as functions from finite paths to probability distributions over transitions. We can distinguish essentially three dimensions: randomization, time dependence and history dependence. Motivated by the availability of the timed reachability algorithm presented in [2], we restrict ourselves to randomized time-abstract history-dependent schedulers.

Definition 2 (CTMDP scheduler) *Let* $\mathcal{C}$ *be a CTMDP with transition set* $\mathbf{R}$. *A* scheduler D over $\mathcal{C}$ *is a function* $D : (S \times L)^* \times S \to$ *Distr*$(\mathbf{R})$ *such that each support of* $D(\sigma)$ *is a subset of* $\mathbf{R}(last(\sigma))$.

The support condition ensures that the scheduler distributes its whole probability mass among the outgoing transitions of $last(\sigma)$.

σ-algebra and probability measure for CTMDP. The σ-algebra over infinite paths and the probability measure $Pr^{\mathcal{C}}_D$ over infinite paths induced by some scheduler D is given by a standard construction [1]. We need to omit it due to space constraints. We refer to [31] for more details concerning measurability of the induced timed path probabilities.

Interactive Markov chains. We recall the model of interactive Markov chains from [14] and give here only the most relevant definitions.

Definition 3 (Interactive Markov Chain) *An* interactive Markov chain, *IMC, is a tuple* $(S, Act, \longrightarrow, \dashrightarrow, s_0)$ *where* S *is a nonempty, finite set of* states, Act *is a finite set of* actions, $\longrightarrow \subseteq S \times Act \times S$ *is the set of* interactive transitions, $\dashrightarrow \subseteq S \times \mathbb{R}^+ \times S$ *is the set of* Markov transitions *and* $s_0 \in S$ *is the* initial state.

As usual, we assume a distinguished internal action τ. We assume $\tau \in Act$, and let $Act_{\backslash \tau}$ denote $Act \backslash \{\tau\}$. All

other actions are called visible. $\mathbf{Rate}(s, s')$ denotes the cumulative[1] transition rate from s to s'. We let $\mathbf{r}(s, C) := \sum_{s' \in C} \mathbf{Rate}(s, s')$ and let $E_s = \mathbf{r}(s, S)$ denote the exit rate of state s. We call state s *stable*, written $s \not\xrightarrow{\tau}$, whenever no $s' \in S$ exists such that $s \xrightarrow{\tau} s'$. Otherwise it is called *unstable*.

Behavior. An IMC can be viewed as a usual labeled transition system that additionally supports stochastic behavior. The usual behavior of labeled transition systems is present via interactive transitions leading from state s to s' via some action $a \in Act$. Stochastic behavior is included via Markov transitions. A Markov transition leads from state s to s' with a particular rate $\lambda \in \mathbb{R}^+$. The delay of this transition is governed by a negative exponential distribution with rate λ, i. e., the probability of triggering this transition within t time units is given by $1 - e^{-\lambda \cdot t}$. When all transitions emanating from state s are Markov transitions, the next state is selected according to the *race condition* between these transitions. The probability to move from state s to state s' within t time units is then given as $\Pr(s, s', t) = \mathbf{P}_s(s') \cdot \left(1 - e^{-E_s \cdot t}\right)$, where $\mathbf{P}_s(s') := \frac{\mathbf{Rate}(s,s')}{E_s}$ is the discrete branching probability to reach s' from s.

Definition 4 (Uniform IMC) *We call IMC* $\mathcal{M} = (S, Act, \longrightarrow, \dashrightarrow, s_0)$ uniform *iff there exists an* $E \in \mathbb{R}^+$ *such that for all* $s \in S$ *it holds that if* $s \not\xrightarrow{\tau}$ *then* $E_s = E$. *The class of uniform IMCs is denoted by* uIMC.

Due to the maximal progress assumption, rates of Markov transitions at unstable states do not play a role in this definition. This will become important later.

Special cases. LTSs and CTMCs are special cases of IMCs. For LTS the set of Markov transitions is empty. By definition, LTSs are uniform with rate $E = 0$. If, on the other hand the interactive transition relation is empty, the IMC reduces to a CTMC. The definition of uniform IMCs is a conservative extension of uniformity in CTMCs: A CTMC is uniform if there exists a rate $E \in \mathbb{R}^+$ such that $E_s = E$ for all states s. Intuitively, this means that the probability of taking a transition within t time units is the same regardless of the state currently occupied.

Nonuniform CTMCs can be *uniformized* [19], without affecting the probabilistic behavior (in terms of state probabilities). Uniformization is often considered as a mapping from a CTMC to a DTMC, but it is better to view it as a twist on the CTMC level [25]. To uniformize a non-uniform CTMC, one chooses a rate E at least as large as the largest exit rate of any of the states. Suppose in state s the cumulative rate E_s in the CTMC is smaller than E, then s is equipped with an additional self-loop with rate $E - E_s$. Intuitively, this ensures that all state changes occur with the same (average) frequency in the resulting uniform CTMC. The probability that n state changes occur within t time units in a uCTMC is given by a Poisson distribution with parameter $E \cdot t$, $\psi(n, E \cdot t)$, denoted by $\psi(n)$ whenever parameter $E \cdot t$ is clear.

State partitioning. It will be instrumental to partition the states of an IMC according to the outgoing transitions of each state. We use this partitioning when discussing the transformation from IMCs to CTMDPs later on. A distinction is made between *Markov states* (S_M) with at least one emanating Markov and no interactive transition; *interactive states* (S_I) with at least one emanating interactive transition and no Markov transitions; *hybrid states* (S_H) with both Markov and interactive outgoing transitions; *absorbing states* (S_A) without outgoing transitions. S can thus be written as $S = S_M \dot\cup S_I \dot\cup S_H \dot\cup S_A$. As an abbreviation we use $S_{IH} := S_I \dot\cup S_H$.

Open vs. closed system view. We distinguish two different views on a given IMC $\mathcal{M}$. Usually, we consider $\mathcal{M}$ as being *open* which means that $\mathcal{M}$ can interact with the environment. In particular, it can be composed with other IMCs (e. g. via parallel composition). In this case we impose the *maximal progress* assumption [14] which embodies the precedence of τ-actions (but not of visible actions) over Markov transitions. The intuition is that visible actions can be subject to composition, and are hence delayable, while internal actions are not. The maximal progress condition is central to the compositional theory of IMC.

In contrast the *closed system view* only exhibits its actions to the environment but closes the IMC for interaction. We assume that models we are going to analyze are closed, and impose an *urgency assumption* which means that any action (visible or not) has precedence over Markov transitions, since it can no longer be delayed by composition. This urgency assumption, is not compatible with composition of IMCs [14]. But this does not influence our modeling trajectory, since the closed system view is applied only on *complete* models, which are no longer subject to composition.

Paths. A *path label* is a pair comprising an action $a \in Act \dot\cup \{\top\}$ and a time point $t \in \mathbb{R}^+$. The distinguished action $\top$ is used to uniquely identify a path label which belongs to a Markov transition. For given IMC $\mathcal{M}$ the set of all path labels is given by $L_{Path_\mathcal{M}} = (Act \dot\cup \{\top\}) \times \mathbb{R}^+$. Intuitively, whenever in a given path σ a state change from s to s' is possible via (a, t) this means that in s a transition labeled a and leading to s' was taken at time t. Whenever a state change occurs w.r.t. a Markov transition at time t this is indicated by $(\top, t)$.

A *timed path* in IMC $\mathcal{M}$ is a possibly infinite alternating sequence $\sigma = v_0(a_0, t_0)v_1(a_1, t_1)\ldots$ of states and path labels. We use the same notation as introduced for CTMDPs. The subscript $\mathcal{M}$ indicates that we refer to paths in IMC $\mathcal{M}$.

Scheduler. The inherent nondeterminism in a closed IMC is, as for CTMDPs, resolved by a scheduler. In its most general form schedulers are functions from decision paths to probability distributions over interactive transitions. Note, that so far, schedulers over IMCs have not been considered in the literature. We restrict to the class randomized time-abstract history dependent schedulers, which matches the class of schedulers we use for CTMDPs.

Definition 5 (IMC scheduler) *Let* $\mathcal{M}$ *be an IMC with inter-*

[1] Since Markov transitions form a relation they may have multiple outgoing Markov transitions from s to s' with possibly different rates.

active transition relation $\rightarrow$*. A scheduler over* $\mathcal{M}$ *is a function* $D : (S \times Act)^* \times S_{IH} \longrightarrow Distr(\rightarrow)$ *such that each support of* $D(\sigma)$ *is a subset of* $(\{s\} \times Act \times S) \cap \longrightarrow$.

Note, that there is a difference in the domain on which a scheduler in an IMC and a CTMDP bases its decision for the next step. Since finite paths in IMCs can end in Markov states, in which a scheduler decision is simply not possible, these paths are not included in the domain.

σ-algebra and probability measure for IMC. The σ-algebra over infinite paths follows a standard cylinder set construction as, e. g., given in [31] for CTMDPs. The scheduler dependent probability measure $Pr_D^{\mathcal{M}}$ over infinite paths can easily be defined but is not in the scope of this paper. For details we refer to [16].

Phase-type distributions. A phase-type distribution is usually defined as the distribution of the time until absorption in a finite and absorbing[2] CTMC [23]. In principle, any probability distribution on $[0, \infty)$ can be approximated arbitrarily closely by a phase-type distribution given enough phases, i. e., states. In order to derive uniformity by construction it is important that the absorbing CTMC of a phase-type distribution can be uniformized, just like any CTMC. The result will no longer have an absorbing state, but a state which is reentered from itself according to a Poisson distribution. It is, after all, a special IMC.

3 Uniformity in composition and minimization of uIMCs

In [4] we constructed uniform IMCs by composition out of smaller uIMCs. The strategy is based on the nowadays classical compositional minimization principle (see, e.g., [12]) which has its roots in process algebra, and has been implemented in an exemplary way in the CADP toolkit [5] together with convenient scripting support [29]. We make use of this toolkit in our studies.

In this section we first present the formal properties of our specific construction, focussing on hiding, parallel composition and minimization of uIMC. In particular, we prove for each of these concepts that it preserves uniformity. Thus we can ensure that constructing larger IMCs out of smaller uniform IMCs yields again uniform IMCs.

Hiding. Hiding, also called abstraction, is used to internalize particular actions of a given IMC. Hiding of action a in IMC $\mathcal{M}$ turns this particular action a into the unobservable action τ. The structural operational semantic rules for hiding in IMCs are as follows [14]:

$$\frac{s \xrightarrow{b} s' \qquad a \neq b}{\mathsf{hide}\ a\ \mathsf{in}\ (s) \xrightarrow{b} \mathsf{hide}\ a\ \mathsf{in}\ (s')} \qquad \frac{s \xrightarrow{a} s'}{\mathsf{hide}\ a\ \mathsf{in}\ (s) \xrightarrow{\tau} \mathsf{hide}\ a\ \mathsf{in}\ (s')}$$

$$\frac{s \xrightarrow{\lambda} s'}{\mathsf{hide}\ a\ \mathsf{in}\ (s) \xrightarrow{\lambda} \mathsf{hide}\ a\ \mathsf{in}\ (s')}$$

[2] An absorbing CTMC has a single absorbing state.

IMC hide a in $(\mathcal{M})$ is obtained by applying these rules to the initial state of $\mathcal{M}$. The first two rules are as expected, the third rule gives the definition of hide for Markov transitions, which remain untouched by the operator semantics.

Lemma 1 hide a in $(\mathcal{M})$ *is uniform whenever* $\mathcal{M}$ *is uniform.*

The proof is based on the observation that the uniformity definition manifests itself in conditions for stable states, and that hiding does not introduce more stable states, but introduces instable ones. The converse is not necessarily true: A non-uniform IMC $\mathcal{N}$ might give rise to a uniform IMC hide a in $(\mathcal{N})$.

Parallel composition. The structural operational rules of parallel composition $||[A]||$ [14], with *synchronization set* $A \subseteq Act_{\setminus\tau}$, are defined on the state-level of IMCs.

$$\frac{s \xrightarrow{a} s' \quad a \notin \{a_1,\ldots,a_n\}}{s||[\{a_1,\ldots,a_n\}]|v \xrightarrow{a} s'||[\{a_1,\ldots,a_n\}]||v} \qquad \frac{v \xrightarrow{a} v' \quad a \notin \{a_1,\ldots,a_n\}}{s||[\{a_1,\ldots,a_n\}]|v \xrightarrow{a} s||[\{a_1,\ldots,a_n\}]||v'}$$

$$\frac{s \xrightarrow{a} s' \quad v \xrightarrow{a} v' \quad a \in \{a_1,\ldots,a_n\}}{s||[\{a_1,\ldots,a_n\}]|v \xrightarrow{a} s'||[\{a_1,\ldots,a_n\}]||v'}$$

$$\frac{s \xrightarrow{\lambda} s'}{s||[\{a_1,\ldots,a_n\}]|v \xrightarrow{\lambda} s'||[\{a_1,\ldots,a_n\}]||v} \qquad \frac{v \xrightarrow{\lambda} v'}{s||[a_1,\ldots,a_n]|v \xrightarrow{\lambda} s||[\{a_1,\ldots,a_n\}]||v'}$$

The upper rules are common for CSP- (or LOTOS-) style calculi. The two rules at the bottom leave the Markov behavior in both states untouched: The transitions are just interleaved (which is justified by the memoryless property of exponential distributions). The parallel composition of two IMCs $\mathcal{M}$ and $\mathcal{N}$ on synchronization set A is defined as $s_0^{\mathcal{M}}|[A]|s_0^{\mathcal{N}}$, where $s_0^{\mathcal{M}}$ denotes the initial state of $\mathcal{M}$ and $s_0^{\mathcal{N}}$ is the initial state of $\mathcal{N}$. The resulting IMC is referred to as $\mathcal{M}|[A]|\mathcal{N}$. Given the structural operational semantics, we can establish the following lemma.

Lemma 2 $\mathcal{M}|[A]|\mathcal{N}$ *is uniform whenever* $\mathcal{M}$ *and* $\mathcal{N}$ *are uniform.*

The proof uses that Markov transitions are just interleaved by the operational rules, which implies that the uniform rates of $\mathcal{N}$ and $\mathcal{M}$ just add up in $\mathcal{M}|[A]|\mathcal{N}$. Again the converse direction is not valid, i. e., whenever a uIMC is the result of a parallel composition of two IMCs $\mathcal{M}$ and $\mathcal{N}$ this does not imply that $\mathcal{M}$ and $\mathcal{N}$ are uniform.

Minimization. We will follow a compositional minimization strategy, where we minimize intermediate graphs according to an equivalence relation which (1) abstracts from internal computation, similar to weak or branching bisimulation, (2) employs lumping [21] of Markov transitions, and (3) leaves the branching structure otherwise untouched. There are many candidates for such an equivalence notion. The equivalence relation we focus on here is *stochastic branching bisimulation* which is a variant of branching bisimulation [30] and stochastic weak bisimulation [14]. Among various possible notions, branching bisimulation and its stochastic counterpart have been proven useful, and are implemented efficiently in the

CADP toolkit [11]. The uniformity preservation result we establish below can also be established for other variations (such as weak bisimulation [14]). In the sequel, when we talk about uniform IMCs, we mean uniform with respect to its reachable states. The restriction to reachable states allows us to handle models with unreachable states having arbitrary rates. This contradicts the original uniformity condition, but is irrelevant for the behavior of the IMC.

Definition 6 (Stochastic branching bisimulation) *For a given IMC $\mathcal{M} = (S, Act, \longrightarrow, \dashrightarrow, s_0)$, an equivalence relation $B \subseteq S \times S$ is a* stochastic branching bisimulation *iff for all $s_1, s_2, t_1 \in S$ the following holds: If $(s_1, t_1) \in B$ then*

1. *$s_1 \xrightarrow{a} s_2$ implies*
   either $a = \tau$ and $(s_2, t_1) \in B$, or
   $\exists t_1', t_2 \in S : t_1 \xrightarrow{\tau^*} t_1' \xrightarrow{a} t_2 \wedge (s_1, t_1') \in B \wedge (s_2, t_2) \in B$, and
2. *$s_1 \not\xrightarrow{\mathcal{I}}$ implies $\exists t_1' : t_1 \xrightarrow{\tau^*} t_1' \not\xrightarrow{\mathcal{I}} : \forall C \in S/B :$ **Rate**$(s_1, C) =$ **Rate**(t_1', C).*

Two states are stochastic branching bisimilar, *iff they are contained in some stochastic branching bisimulation B.*

Given IMC $\mathcal{M}$, $StoBraBi(\mathcal{M})$ denotes the stochastic branching bisimilarity quotient IMC of $\mathcal{M}$. For two IMCs $\mathcal{M}$ and $\mathcal{N}$ we say that $\mathcal{M}$ and $\mathcal{N}$ are stochastic branching bisimilar iff $s_0^{\mathcal{M}}$ and $s_0^{\mathcal{N}}$ are branching bisimilar. We can prove the following lemma, using that the uniformity condition for IMCs is imposed for stable states only.

Lemma 3 *Given IMCs $\mathcal{M}$ and $\mathcal{N}$, it holds that if $\mathcal{M}$ and $\mathcal{N}$ are stochastic branching bisimilar then: $\mathcal{M}$ is uniform iff $\mathcal{N}$ is uniform.*

We can directly derive the following corollary from the above lemma.

Corollary 1 *An IMC $\mathcal{M}$ is uniform iff $StoBraBi(\mathcal{M})$ is uniform.*

Elapse. We use phase-type distributions to specifiy the timed probabilistic behavior of a system under study. Recall that, structurally, a phase-type distribution Ph is a CTMC (S, A, R) with a distinguished initial state i and absorbing state a. In the composition context, the distribution Ph can be viewed as describing the time up to which the occurrence of an event f has to be delayed, since the occurrence of some other event r. This interpretation is a special case of what is called a *time constraint* in [15], where an *elapse* operator is introduced. This operator enriches Ph with "synchronization potential" needed to effectively incorporate the Markov chain of Ph into the behavior described by some LTS or IMC. Since Ph is a CTMC we can assume that it is uniformized without making any restrictions [19].

We will use this operator below, so we give its semantics here, simplified to what is actually used later. We denote the operator by $El(Ph, f, r)$. It takes as parameters a distribution Ph (given as a uCTMC (S, A, R) with distinguished states i and a and uniform rate E), an action f which should only occur once the Ph distributed delay is over, and an action r governing when that delay should start. With these parameters, $El(Ph, f, r)$ generates IMC $(S \dot\cup \{s'\}, A, \left\{a \xrightarrow{f} s', s' \xrightarrow{r} i\right\}, R \dot\cup \left\{s' \overset{E}{\dashrightarrow} s'\right\}, s')$. Here s' is a fresh initial state of the resulting IMC where we have added two interactive transitions. The freshly inserted state s' gets a Markov self-loop with rate E assigned. This assures uniformity of the resulting IMC. We will also use a variant of this operator, referred to as El', where the initial state remains unchanged at i, but otherwise the construction is the same.

In $El(Ph, f, r)$, between the occurrences of r and f, there must be a delay which is given by Ph. To enforce this also for the LTS of our system under study, we incorporate this IMC into the system model using parallel composition with appropriate synchronization sets.

4 Transformation and uCTMDP analysis

In this section we describe a transformation procedure from closed uIMCs to uCTMDPs and discuss in what sense it preserves the properties we are interested in. We then recall from [2] how to analyze the resulting CTMDP, explain how we implemented this algorithm, and apply both steps to the FTWC case study.

4.1 From uIMC to uCTMDP

In the following description we assume that uIMC $\mathcal{M} = (S, Act, \longrightarrow, \dashrightarrow, s_0)$ is given and that we aim to transform $\mathcal{M}$ into a CTMDP, say $\mathcal{C}_{\mathcal{M}}$. We present a transformation from a (u)IMC under consideration to a *strictly alternating* (u)IMC. The latter is a (u)IMC in which interactive and Markov states occur strictly alternating, and where hybrid states are not present anymore. Each strictly alternating IMC corresponds to a CTMDP such that the timed behavior of both systems is the same.

We do not allow for Zeno-behavior in the model under consideration, Zenoness manifests itself as cycles of interactive transitions, which owed to the closed system view can happen in zero time. The model may in general contain absorbing states, but for the sake of simplicity, we will not consider them, and thus assume $S_A = \emptyset$. In the uniform setting absorbing states are absent (for non-zero uniform rates).

An IMC $\mathcal{M}$ with state space S is called *alternating* iff for all $s \in S$, $(\{s\} \times \mathbb{R}^+ \times S) \cap \dashrightarrow \neq \emptyset$ implies $(\{s\} \times Act \times S) \cap \longrightarrow = \emptyset$. If in addition *(i.)* $(S_M \times \mathbb{R}^+ \times S_M) \cap \dashrightarrow = \emptyset$ and *(ii.)* $(S_I \times Act \times S_I) \cap \longrightarrow = \emptyset$, we call $\mathcal{M}$ *strictly alternating*. An IMC satisfying the condition *(i.)* (respectively *(ii.)*) is called *Markov (interactive) alternating* IMC.

Technically, the transformation procedure comprises the following steps, which will be shown to preserve the probabilistic behavior of the model, and which result in a strictly alternating IMC: (1) ensure that S contains interactive and

Markov states only (alternating IMC); (2) make the target state of each Markov transition an interactive state (Markov alternating IMC); (3) make the target state of each interactive transition a Markov state (interactive alternating IMC). While (1) has to be done first, steps (2) and (3) can also be interchanged.

Step (1): Alternating. An alternating IMC does not posses hybrid states while the original IMC may do. Thus let $s \in S_H$. Owed to the *urgency* assumption which is due to our closed system view, all Markov transitions are cut off and s is added to S_I. For the corresponding *alternating* IMC $\mathbf{a}(\mathcal{M})$ we change the Markov transition relation to $(S_M \times \mathbb{R}^+ \times S) \cap \dashrightarrow$.

Step (2): Markov alternating. We need to split all occurring sequences of Markov transitions and states. In summary this ensures that each Markov transition always ends in an interactive state. This is achieved as follows. Suppose $s \in S_M$ with $s \overset{\lambda}{\dashrightarrow} s'$ and $s' \in S_M$. In order to break this sequence of Markov states we introduce a fresh interactive state (s,s') which is connected to s via $s \overset{\lambda}{\dashrightarrow} (s,s')$. State (s,s') in turn is connected via $(s,s') \xrightarrow{\tau} s'$ to s'. This yields the following transformation step. For a given alternating IMC $\mathcal{M}$, we define the *Markov alternating* IMC:
$\mathbf{mA}(\mathcal{M}) := (S', Act, \longrightarrow_{\mathbf{mA}(\mathcal{M})}, \dashrightarrow_{\mathbf{mA}(\mathcal{M})}, s_0)$ with

- $S' = S \,\dot\cup\, \{(s,s') \in S_M \times S_M \mid \exists \lambda \in \mathbb{R}^+ : s \overset{\lambda}{\dashrightarrow} s'\}$,
- $\longrightarrow_{\mathbf{mA}(\mathcal{M})} = \longrightarrow \dot\cup\, \{((s,s'),\tau,s') \in S' \times \{\tau\} \times S_M \mid \exists \lambda \in \mathbb{R}^+ : s \overset{\lambda}{\dashrightarrow} s'\}$,
- $\dashrightarrow_{\mathbf{mA}(\mathcal{M})} = \dashrightarrow \cap (S_M \times \mathbb{R}^+ \times S_I) \;\dot\cup\; \{(s,\lambda,(s,s')) \in S_M \times \mathbb{R}^+ \times S' \mid s \overset{\lambda}{\dashrightarrow} s'\}$.

Step (3): Interactive alternating. We now handle sequences of interactive transitions ending in a Markov state. To compress these sequences, we calculate the transitive closure of interactive transitions for each interactive state s that (is either the initial state of the uIMC or) has at least one Markov predecessor. The computation is carried out in a way such that we get all Markov successors of s that terminate these sequences. We label the resulting compressed transitions with words over the alphabet $Act^+_{\setminus\tau} \,\dot\cup\, \{\tau\}$ (also denoted $Words$). Note, that interactive states that do not have any Markov state as predecessor will not be contained in the resulting interactive (or strictly) alternating uIMC any more. These states violate the strict alternation of interactive and Markov states and therefore will not be contained in the CTMDP.

For a Markov alternating IMC $\mathcal{M}$ we define $\mathbf{iA}(\mathcal{M}) := (S', Words, \longrightarrow_{\mathbf{iA}(\cdot)}, \dashrightarrow, s_0)$ by

- $S' = S_M \,\dot\cup\, S'_I$ where $S'_I = \{s \in S_I \mid \exists t \in S_M : t \overset{\lambda}{\dashrightarrow} s$, for some $\lambda \in \mathbb{R}^+\} \cup \{s_0\}$,
- $\longrightarrow_{\mathbf{iA}(\cdot)} := \{(s,W,t) \in S'_I \times Words \times S_M \mid s \overset{W}{\Longrightarrow} t\}$.

This yields a strictly alternating IMC. So, after applying steps (1)–(3) to uIMC $\mathcal{M}$ we obtain uIMC $\mathcal{M}'$ which is strictly alternating and where interactive transitions are labeled by words over the alphabet $Act^+_{\setminus\tau} \,\dot\cup\, \{\tau\}$. The resulting uIMC $\mathcal{M}' = (S = S_I \,\dot\cup\, S_M, Words, \longrightarrow, \dashrightarrow, s_0)$ can now be interpreted as CTMDP $\mathcal{C}_{\mathcal{M}} = (S_I, Words, \mathbf{R}, s_0)$ where $\mathbf{R} := \{(s,A,R) | R(s') = \sum_{i=1}^{n} \lambda_i$ iff $\exists u \in S_M, \lambda_i \in \mathbb{R}_{\geq 0}$ such that $s \xrightarrow{A} u \wedge u \overset{\lambda_i}{\dashrightarrow} s', i = 1,2\ldots,n\}$.

Property preservation. Given a path σ in $\mathcal{M}$ we know that there exists a path σ' in CTMDP $\mathcal{C}_{\mathcal{M}}$ that corresponds to σ due to the transformation. On the other hand, for each path σ in $\mathcal{C}_{\mathcal{M}}$ there exists a set of paths in $\mathcal{M}$ that corresponds to σ. For a given σ in $\mathcal{M}$ let $\Psi(\sigma)$ denote the corresponding path in $\mathcal{C}_{\mathcal{M}}$ and, for given σ in $\mathcal{C}_{\mathcal{M}}$ let $\Phi(\sigma)$ be the set of corresponding paths in $\mathcal{M}$. We extend this notation to sets of paths.

Theorem 1 *Given IMC $\mathcal{M} = (S_{\mathcal{M}}, Act, \longrightarrow, \dashrightarrow, s_0)$ and CTMDP $\mathcal{C}_{\mathcal{M}} = (S_{\mathcal{C}}, Words, \mathbf{R}, s_0)$ as result of the transformation. (1) For each scheduler D in $\mathcal{M}$ there exists a scheduler D' in $\mathcal{C}_{\mathcal{M}}$ such that for all measurable sets A of paths (in $\mathcal{M}$)*
$$Pr^{\mathcal{M}}_{D}(A) = Pr^{\mathcal{C}_{\mathcal{M}}}_{D'}(\Psi(A)) .$$
(2) For each scheduler D in $\mathcal{C}_{\mathcal{M}}$ there exists a scheduler D' in $\mathcal{M}$ such that for all measurable sets A of paths (in $\mathcal{C}_{\mathcal{M}}$)
$$Pr^{\mathcal{C}_{\mathcal{M}}}_{D}(A) = Pr^{\mathcal{M}}_{D'}(\Phi(A)) .$$

Note that this theorem does not require uniformity. The proof of this theorem uses the path measures induced by a scheduler in the original system (uIMC or uCTMDP) to construct the scheduler in the goal system (uCTMDP or uIMC). For (1) it can be shown that there exists a set of paths Λ in $\mathcal{M}$ for each path σ in $\mathcal{C}_{\mathcal{M}}$ such that Λ comprises all paths (of $\mathcal{M}$) that share the same *trace* information as σ and the same stochastic behavior. Assume that D is the scheduler in $\mathcal{M}$ and D' denotes the scheduler we intend to construct for $\mathcal{C}_{\mathcal{M}}$. Now, $D'(\sigma)(s,a,R)$ is then defined as the sum over $D(\sigma')(\sigma'')$ for all $\sigma' \in \Lambda$ and σ'' that correspond to (s,a,R) (in $\mathcal{C}_{\mathcal{M}}$). Each of the summands has to be weighted by the probability of its occurrence in $\mathcal{M}$. To show (2) we need to reconstruct scheduler D' in $\mathcal{M}$ from scheduler D in $\mathcal{C}_{\mathcal{M}}$. This is the more difficult case. For a σ in $\mathcal{M}$ there is not necessarily a path σ' in $\mathcal{C}_{\mathcal{M}}$ that corresponds to σ, because $last(\sigma)$ may end in an interactive state without Markov predecessors. Thus, for the construction of D' we have to take prefixes of σ into account for which a corresponding path in $\mathcal{C}_{\mathcal{M}}$ exists. For details we refer to [16].

4.2 Timed reachability in uCTMDPs

In this section we briefly recall the algorithm for timed reachability analysis in uCTMDPs [2]. Note, that the slight variant considered here (in which we allow for nondeterministic choices among the same action $a \in L$) implies only a slight change in the algorithm given in [2] where we have to range over all emanating transitions (instead of all actions) of a given state s.

For a uCTMDP $\mathcal{C}$ with uniform rate E we aim to calculate the maximal probability to reach a given set of states B within t time units from a particular state s in $\mathcal{C}$ w.r.t. all schedulers D according to Definition 2. We denote this by

Algorithm 1 (c.f. [2]) Computing $\sup_D Pr_D^{\mathcal{C}}(s, \overset{\leq t}{\rightsquigarrow} B)$

$k := k(\varepsilon, E, t)$; $\forall s \in S : q_{k+1} := 0$;
for all $i = k, k-1, \ldots, 1$ **do**
  for all $s \in S \backslash B$ **do** m := -1;
    for all $(s, a, R) \in \mathbf{R}(s)$ **do**
      $m := \max\{m, \psi(i) \cdot \Pr_R(s, B) + \sum_{s' \in S} \Pr_R(s, s') \cdot q_{i+1}(s')\}$;
    $q_i(s) = m$;
  for all $s \in B$ **do** $q_i(s) := \psi(i) + q_{i+1}(s)$;
for all $s \in S$ **do**
  if $s \notin B$ **then** $q(s) := q_1(s)$; **else** $q(s) := 1$;
return q;

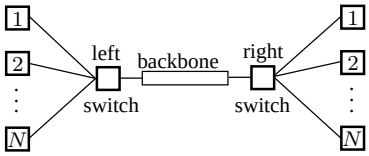
$\sup_D Pr_D^{\mathcal{C}}(s, \overset{\leq t}{\rightsquigarrow} B)$, where $Pr_D^{\mathcal{C}}$ is the scheduler dependent probability measure on $\mathcal{C}$. $Pr_D^{\mathcal{C}}$ is defined on infinite paths, and thus we have to map the timed reachability probability $s, \overset{\leq t}{\rightsquigarrow} B$ to the set of all infinite paths in which it is fulfilled.

In [2] the problem of approximating this probability is tackled. It it is shown that it is sufficient to consider non-randomized step-dependent schedulers $D : S \times \{0, \ldots, k\} \mapsto L$ and that in order to derive the maximal value of $Pr_D^{\mathcal{C}}(s, \overset{\leq t}{\rightsquigarrow} B)$, the transitions to be selected by a scheduler D can be computed by a greedy backward strategy up to a specific depth k which can be precomputed on the basis of E, t and the accuracy ε of the approximation. In particular, k is the smallest index such that $\sum_{n=k+1}^{\infty} \psi(n) \leq \varepsilon$. Note, that this is the *right truncation point* as defined in [9].

Algorithm 1 shows the pseudo-code of the timed reachability algorithm. For $i = k$ the transition that has to be scheduled is chosen such that the one-step probability to reach B is maximized. Since this is the k^{th} step in evolution of the CTMDP, this probability is weighted with the Poisson probability for realizing k steps within t time units ($\psi(i)$). In [2] it is shown that $q_i(s) = \sum_{n=i}^{k} \psi(n) \cdot \Pr_{R_i}(s, B) \cdot \Pr_{R_{i+1}}(s, B) \cdot \cdots \cdot \Pr_{R_n}(s, B)$ has to be computed such that it is optimal under all $\sum_{n=i}^{k} \psi(n) \cdot \Pr_{R_*}(s, B) \cdot \Pr_{R_{i+1}}(s, B) \cdot \cdots \cdot \Pr_{R_n}(s, B)$. Where R_j is the rate function that belongs to the transition selected by scheduler D_0 at step j and R_* ranges over all rate functions that could be selected at step i. Scheduler D_0 is the scheduler that is constructed to maximize the reachability probability.

The algorithm returns for each state the maximal probability to reach a state $s \in B$ within time t.

We have implemented Algorithm 1 in JAVA and integrated it into the ETMCC model checker [17]. The transition relation is stored as sparse matrices storing action and rate information separately. This means that we partitioned the state space in interactive and Markov states (as for strictly alternating IMCs). The Markov states are in one-to-one correspondence to the rate functions of the CTMDP transitions. The implementation is prototypical and so far we have not put efforts in implementing more advanced techniques, e. g., more efficient state space representations of the CTMDP. However, the performance of this prototypical implementation is quite promising as can be seen in the next paragraph.



	fail	*repair*
Workstation	500 h	0.5 h
Switch	4000 h	4 h
Backbone	5000 h	8 h

Figure 1. FTWC with mean fail and repair times.

5 Fault-tolerant work station cluster

We are now in the position to construct a model of the fault-tolerant workstation cluster (FTWC) example. This example has first been studied in [13], and is also one of the PRISM benchmarks. Remarkably, these studies treated the model as a CTMC, which is not easy to justify, as we will argue below. The general design of the workstation cluster is depicted on the top of Figure 1. The overall system consists of two sub-clusters which are connected via a backbone. There are N workstations in each sub-cluster which are connected together in a star topology with a switch as central node that additionally provides the interface to the backbone.

Each of the components in the fault-tolerant workstation cluster can break down (fail) and then needs to be repaired before becoming available again. The mean time to failure and the mean repair time for each component in isolation are depicted on the bottom of Figure 1. They correspond to mean durations of exponential distributions.

There is a single repairunit for the entire cluster, not depicted in Figure 1, which is only capable of repairing one failed component at a time. Essentially, this means that when multiple components are failed, they must be handled in sequence, and there is a decision to be taken which of the failed components the repairunit is assigned first (or next). Notably, this nondeterminism which is inherent in the specification has been ignored in the original model [13] and subsequent work. It has been approximated by using a very fast, but probabilistic decision, encoded via the use of very high rates (of exponential distributions) assigned to the decisive transitions. These high rates are absent in the original problem statement where the repairunit is assigned nondeterministically.

Labeled transition systems. We now construct a uIMC modeling the behavior of the FTWC example. We do so in a compositional manner. There are six basic ingredients, namely the workstations (left and right), the switches (left and right), the backbone and the repairunit. Their behavior is modeled as simple LTSs. In Figure 2 we depict two different LTSs, where a white circle is used to represent the initial state of the respective LTS. The LTS of the repairunit (RU) is shown on the left hand side, where we depicted for sake of readability only two transitions. In fact, there are five parallel transitions emanating the initial state labeled with, e. g., *g_wsL*, and five transitions

ending in the initial state labeled with, e. g., *r_wsL*. The LTSs for workstations (WS), switches (SW) and the backbone (BA) are very similar in nature, their general structure is given on the right hand side of the figure. Each of them can *fail* and has to grab the repairunit afterwards. Only when the repairunit is assigned to that particular component, a *repair* can be performed. Once the component is repaired, the repairunit will be released and can be assigned to another failed component. For a particular component, e. g., the left workstations, the actions *g* and *r* in Figure 2 have to be replaced by the according actions, e. g., by *g_wsL* and *r_wsL*, respectively (this is an instance of process algebraic relabelling).

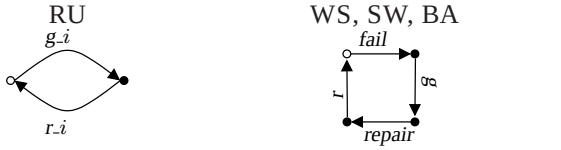
Time constraints. For each of the components, except for the repairunit, the occurence of *fail* and *repair* is governed by delays. These delays have to be incorporated in the model by composition.

We exemplify the construction of time constraints and the incorporation of these time constraints into the LTS for the (left) workstations as follows: The LTS of a workstation is shown rightmost in Figure 2. The belonging time constraints are depicted on the left hand side of Figure 3. They are obtained by applying $El'(\circ \xrightarrow{0.002} \circ \xrightarrow{0.002}, \mathit{fail}, \mathit{r\_wsL})$ and $El(\circ \xrightarrow{2} \circ \xrightarrow{2}, \mathit{repair}, \mathit{g\_wsL})$, respectively. The *elapse* operator preserves uniformity and thus these time constraints are uIMCs.

On the right hand side of Figure 3 we have depicted the IMC describing the timed behavior of a (left) workstation. This IMC is obtained by fully interleaving the time constraints and subsequently synchronizing with *WS* on synchronization set $\{\mathit{fail}, \mathit{repair}, \mathit{g\_wsL}, \mathit{r\_wsL}\}$. Parallel composition of uIMCs preserves uniformity and thus the resulting IMCs are uniform, too.

System model. The FTWC is composed out of the components for the workstations, switches, the backbone and the repairunit. First, the workstations, switches and the backbone are constructed as exemplified above and, after hiding actions *fail, repair* they are composed in parallel (with empty synchronization set).

The resulting uIMCs are then minimized, and composed with the repairunit on the synchronization alphabet $\{g\_i, r\_i\}$, for $i \in \{\mathit{wsL}, \mathit{wsR}, \mathit{swL}, \mathit{swR}, \mathit{bb}\}$, which yields the overall system description of the FTWC (modulo another hiding and minimization step) as an IMC, which, *by construction*, is still uniform.



$i \in \{\text{wsL, wsR, swL, swR, bb}\}$

Figure 2. LTSs for the FTWC.

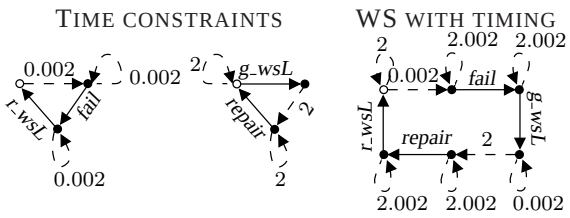


Figure 3. Time constraints and Workstation.

Technicalities. The compositional construction of the FTWC has been carried out using the CADP toolbox [5], and here especially the SVL scripting language and the BCG_MIN tool. For $N = 14$ we obtained an intermediate state space with $5 \cdot 10^6$ states and $6 \cdot 10^7$ transitions. This model then reduces to a uIMC with $6 \cdot 10^4$ states and $5 \cdot 10^5$ transitions. For $N = 16$ we were not able to construct the FTWC in a compositional way. The intermediate state space generation stopped with an incomplete system description that already took 2 GB of hard disk memory.

Larger models were generated with PRISM [18]. PRISM generates a CTMC of the FTWC example in which the nondeterminism is replaced by uniform probability distributions using a fixed large rate Γ. In order to retain the original nondeterminism, we replaced this particular Γ by an interactive transition and applied afterwards the transformation. We made sure that for $N \leq 14$ equivalent models were obtained via CADP and PRISM – up to uniformity.

Results. In this section we report on the results and statistics we obtained when analyzing the FTWC. We focus on the performance of the transformation and of the analysis algorithm.

As in [13] we say that our system operates in *premium* quality when there are at least N workstations operational. These workstations have to be connected to each other via operational switches. When the number of operational workstations in one sub-cluster is below N the premium quality can be ensured by an operational backbone under the condition that there are N operational workstations in total. We are interested in the following property: *"What is the worst case probability to hit a state in which premium service is not guaranteed within t time units?"* for which we report results and statistics.

In Table 1 we have collected different statistics of the transformation and reachability analysis for different N. Columns 2-6 display the number of states, number of transitions and the memory usage of the CTMDP representation of the FTWC. The depicted numbers are given for the strictly alternating IMCs (which comprises precisely what needs to be stored for the corresponding CTMDP), and thus are differentiated in interactive states/transitions and Markov states/transitions. In column 7 the time for the transformation (from uIMC to uCTMDP) is shown. We highlight, that the prototypical implementation of the transformation procedure works quite efficiently also for large systems ($N = 128$). In columns 8 and 9 we collect statistics of the implementation of the reachability algorithm. The runtime results for a time bound of $100h$ are given in column 8 and the results for a time bound of $30000h$

are shown in the column 9. In the last two columns we show the required number of iterations taken by the reachability algorithm in order to have a precision of 0.000001. Readers familiar with CTMC analysis time and space requirements will appreciate the efficiency apparent in these figures, given that we report about a JAVA prototype, and need to deal with nondeterminism.

Finally, we show in Figure 4 graphs for $N = 4$ and $N = 128$ in which we compare the worst case probabilities obtained by the CTMDP algorithm with the probabilities obtained from CTMC analysis. As evident in the plot, the CTMC analysis consistently overestimates the true probabilities (computed with ETMCC, confirmed with CADP). This is quite remarkable, because the CTMDP algorithm accounts for the worst-case. Nothing worse is possible in the model, and we would thus expect, that this probability will be higher than in a corresponding CTMC model of the system. This overestimation, which indicates a modeling flaw in the CTMC approach, can be explained as follows. When replacing a nondeterministic selection by high rates, certain paths become possible (though with low probability), that in a nondeterministic interpretation would be absent, and thus not contribute to the reachability probability. Concretely, in the CTMC implementation, there are sometimes races between very high rates and ordinary rates, which are entirely artificial and do not exist in the more faithful interpretation we use.

6 Conclusion

This paper has introduced a compositional approach to the generation of nondeterministic stochastic systems, and their analysis. More precisely, the paper has made the following contributions: (1) we have devised a sound compositional trajectory to construct *uniform* interactive Markov Chains. This model is transformed into a *uniform* continuous-time Markov decision process, and this transformation is shown to preserve path probability measures. (2) We have presented convincing experimental results for the first implementation of the uniform CTMDP analysis algorithm of [2]. (3) We have discussed that nondeterminism lets certain systems look different, in particular the fault-tolerant workstation cluster example. Surprisingly, the previous studies of this model overestimated the worst-case reachability probabilities which we, for the first time, were able to compute accurately.

The experiments show that the transformation and the analysis algorithm scale very well. Especially compared to the simpler CTMC case, the time and space requirements are of similar order (for models of similar size). Even though our prototypical JAVA implementation is performing remarkably well on this example, we are currently porting the algorithm to C++ in order to integrate it with the MRMC tool [20].

References

[1] R. B. Ash and C. A. Doléans-Dade. *Probability & Measure Theory*. Academic Press, second edition, 2000.

[2] C. Baier, B. R. Haverkort, H. Hermanns, and J.-P. Katoen. Efficient Computation of Time-Bounded Reachability Probabilities in Uniform Continuous-Time Markov Decision Processes. *Theor. Comput. Sci.*, 345(1):2–26, 2005.

[3] A. Bianco and L. de Alfaro. Model Checking of Probabilistic and Nondeterministic Systems. *FSTTCS*, 15, 1995.

[4] E. Böde, M. Herbstritt, H. Hermanns S. Johr, T. Peikenkamp, R. Pulungan R. Wimmer, and B. Becker. Compositional Performability Evaluation for Statemate. In *International Conference on Quantitative Evaluation of Systems (QEST)*, pages 167–176. IEEE Computer Society, 2006.

[5] CADP. Project Website, Aug 2006. `http://www.inrialpes.fr/vasy/cadp/demos.html`.

[6] M. Calder, V. Vyshemirsky, D. Gilbert, and R. Orton. Analysis of signalling pathways using continuous time Markov chains. *Transactions on Computational Systems Biology*, 2006. To appear.

[7] R. Cleaveland, S. P. Iyer, and M. Narasimha. Probabilistic temporal logics via the modal mu-calculus. *Theor. Comput. Sci.*, 342(2-3):316–350, 2005.

[8] L. de Alfaro. Stochastic Transition Systems. In *International Conference on Concurrency Theory (CONCUR)*, pages 423–438, 1998.

[9] B. L. Fox and P. W. Glynn. Computing Poisson probabilities. *Communications of the ACM*, 31(4):440–445, 1988.

[10] G. Gallo, G. Longo, S. Pallottino, and S. Nguyen. Directed hypergraphs and applications. *Discrete Appl. Math.*, 42(2-3):177–201, 1993.

[11] H. Garavel and H. Hermanns. On Combining Functional Verification and Performance Evaluation Using CADP. In *Formal Methods Europe (FME)*, pages 410–429, 2002.

[12] S. Graf, B. Steffen, and G. Lüttgen. Compositional Minimisation of Finite State Systems Using Interface Specifications. *Formal Asp. Comput.*, 8(5):607–616, 1996.

[13] B.R. Haverkort, H. Hermanns, and J.-P. Katoen. On the Use of Model Checking Techniques for Dependability Evaluation. In *Symposium on Reliable Distributed Systems (SRDS'00)*, pages 228–237. IEEE Computer Society, 2000.

[14] H. Hermanns. *Interactive Markov Chains and the Quest for Quantified Quality*, volume 2428 of *LNCS*. Springer, 2002.

[15] H. Hermanns and J.-P.Katoen. Automated compositional Markov chain generation for a plain-old telephone system. *Science of Comp. Programming*, 36:97–127, 2000.

[16] H. Hermanns and S. Johr. Uniformity by Construction in the Analysis of Nondeterministic Stochastic Systems, 2006. Long version of submission. Available at `http://depend.cs.uni-sb.de/~johr/PDS/longV.pdf`.

N	# States		# Transitions		Mem	Transf. time (s)	Runtime (s)		# Iterations	
	Inter.	Markov	Inter.	Markov			100 h	30000 h	100 h	30000 h
1	110	81	155	324	14.2 KB	5.37	0.01	6.04	372	62161
2	274	205	403	920	38.6 KB	4.32	0.01	12.33	372	62284
4	818	621	1235	3000	122 KB	5.25	0.04	37.28	373	62528
8	2770	2125	4243	10712	428 KB	5.83	0.13	47.77	375	63016
16	10130	7821	15635	40344	1.56 MB	6.61	0.52	294.97	378	63993
32	38674	29965	59923	156440	6.01 MB	9.44	3.23	877.52	384	65945
64	151058	117261	234515	615960	23.6 MB	20.58	37.42	3044.72	397	69849
128	597010	463885	927763	2444312	93.6 MB	57.31	557.52	20867.06	423	77651

Table 1. Model sizes, memory usage and runtime for strictly alternating IMCs

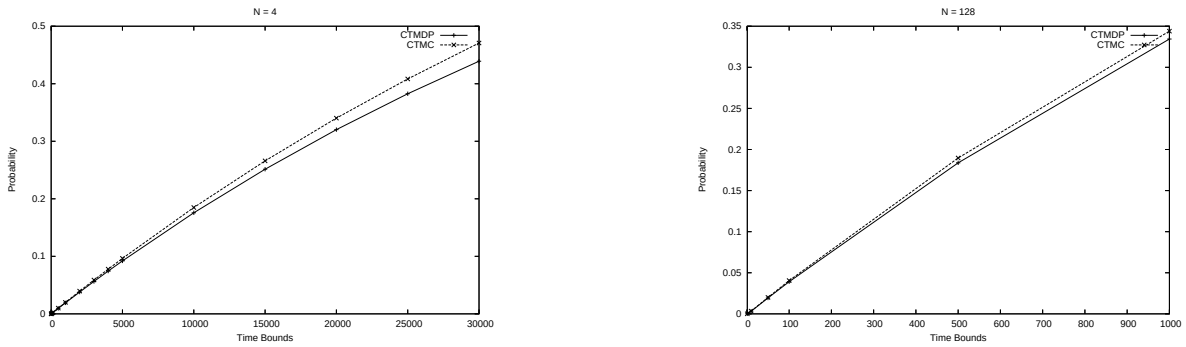


Figure 4. Different probabilities

[17] H. Hermanns, J.-P. Katoen, J. Meyer-Kayser, and M. Siegle. A tool for model-checking Markov chains. *Journal on Software Tools for Technology Transfer (STTT)*, 4(2):153–172, 2003.

[18] A. Hinton, M. Kwiatkowska, G. Norman, and D. Parker. PRISM: A Tool for Automatic Verification of Probabilistic Systems. In *TACAS*, pages 441–444. Springer, 2006.

[19] A. Jensen. Markoff Chains as an Aid in the Study of Markoff Processes. *Skandinavisk Aktuarietidsskrift*, pages 87–91, March 1953.

[20] J.-P. Katoen, M. Khattri, and I. S. Zapreev. A Markov Reward Model Checker. In *Quantitative Evaluation of Systems (QEST)*, pages 243–244. IEEE Computer Society, 2005.

[21] J. G. Kemeny and J. L. Snell. *Finite Markov Chains*. Van Nostrand, 1960.

[22] N.A. Lynch, I. Saias, and R. Segala. Proving Time Bounds for Randomized Distributed Algorithms. In *Symposium on the Principles of Distributed Computing*, pages 314–323, 1994.

[23] M. F. Neuts. *Matrix-Geometric Solutions in Stochastic Models: An Algorithmis Approach*. Dover, 1981.

[24] A. Pogosyants, R. Segala, and N. A. Lynch. Verification of the Randomized Consensus Algorithm of Aspnes and Herlihy: a Case Study. In *Workshop on Distributed Algorithms (WDAG'97)*, volume 1320, pages 111–125. Springer-Verlag, 1997.

[25] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley, 1994.

[26] M. A. Salsburg, D. Lifka, and R. S. Mitchell. A Management Framework For Petabyte-Scale Disk Storage. In *Int. CMG Conference*, pages 767–782, 2005.

[27] R. Segala. *Modeling and Verification of Randomized Distributed Real-Time Systems*. PhD thesis, Department of Electrical Engineering and Computer Science, MIT, 1995.

[28] R. Segala and N. Lynch. Probabilistic simulations for probabilistic processes. *Nordic Journal of Computing*, 2(2):250–273, 1995.

[29] SVL. Project Website, March 2006. `http://www.inrialpes.fr/vasy/cadp/man/svl.html`.

[30] R. J. van Glabbeek and W. P. Weijland. Branching time and abstraction in bisimulation semantics. *J. ACM*, 43(3):555–600, 1996.

[31] N. Wolovick and S. Johr. A Characterization of Meaningful Schedulers for Continuous-time Markov Decision Processes. In *Formal Modeling and Analysis of Times Systems (FORMATS)*, pages 352–367. Springer, 2006.