Three-Valued Abstraction for Stochastic Systems

Von der Fakultät für Mathematik, Informatik und Naturwissenschaften
der Rheinisch-Westfälischen Technischen Hochschule Aachen
zur Erlangung des akademischen Grades eines Doktors
der Naturwissenschaften genehmigte Dissertation

vorgelegt von

Diplom-Informatiker

Daniel Klink

aus

Daun

Berichter: Prof. Dr. Ir. Joost-Pieter Katoen
Michael Huth, PhD
Prof. Dr. Ing. Stefan Kowalewski

Tag der mündlichen Prüfung: 11. März 2010

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über
http://dnb.d-nb.de abrufbar.

ISBN 978-3-86853-408-5

Sternstr. 18, 80538 München
Tel.: 089/66060798
www.dr.hut-verlag.de

1. Auflage 2010

Abstract

Formal methods are a mathematical tool for modeling and verifying qualitative and quantitative aspects of systems of a wide variety of types. A most acknowledged technique in this context is the *model checking* approach that allows for an automated verification of system models with respect to sets of formal requirements. The underlying analysis techniques systematically and exhaustively explore all configurations of a system, resulting in a high memory consumption. To deal with this bottleneck, several solutions have been proposed, ranging from the use of efficient data structures to a number of reduction techniques. This thesis is concerned with *abstraction*, that is, with the reduction of available information in the system's model, and the question of what properties are preserved when applying abstraction.

We are focusing on the more intricate quantitative properties of probabilistic timed systems that can be expressed in PCTL and CSL, probabilistic variants of the computation tree logic (CTL). The models under consideration are discrete-time and continuous-time Markov chains (DTMCs, CTMCs) as well as interactive Markov chains (IMCs) that extend CTMCs with functional behavior and provide facilities for compositional modeling. Markov chains are the underlying low-level models for many modeling formalisms that are used, amongst others, in the area of reliability, dependability and performance analysis as well as in systems biology.

In this thesis, abstraction is implemented by merging states of a system's model; when states with different behavior are to be merged, nondeterminism is added to the corresponding abstract state to capture the uncertainty on what concrete state is occupied when looking at the abstract picture. Due to this newly introduced nondeterminism, we obtain abstractions that may contain more (but no less) possible behavior compared to the given concrete system; hence, when analyzing the abstraction, we obtain safe over- and

underapproximations of the results for the concrete model. This allows us to derive conclusions from the abstraction where the concrete model is too large to be analyzed or even of infinite size. However, if too much information is discarded – resulting in a coarse abstraction – the analysis results may be inconclusive. In order to handle such inconclusive (partial) results, we develop a model checking framework based on *three-valued* logics with an additional truth value *don't know*.

When generalizing abstraction for CTMCs to the compositional framework of IMCs, we propose to perform abstraction at the component level instead of building a monolithic model of the system first. This approach allows for significant reductions of the peak memory requirements. We investigate the formal relation between the resulting abstract compositional model and the given concrete one. Further, we show that the abstraction of several distinct components can be used to introduce symmetries that are not present in the concrete model but facilitate enormous savings in the compositional abstraction.

Besides the theoretical underpinnings of abstraction and model checking for stochastic systems, we also provide efficient algorithms for the analysis of abstract models and we show empirically that abstraction complements long-standing analysis techniques in queueing theory. Further, we present an extension of the abstraction framework that allows for the efficient analysis of a well-known, hard-to-solve case study from systems biology: *enzyme catalyzed substrate conversion*.

Zusammenfassung

Der Begriff der *Formalen Methoden* umfasst diverse Techniken zur mathematischen Modellierung und Verifikation von qualitativen und quantitativen Eigenschaften unterschiedlichster Systeme. Besonders hervorzuheben ist das *Model Checking*, ein automatisiertes Verfahren zur Verifikation von Systemen bezüglich gegebener formaler Spezifikationen. Die zugrunde liegenden Analyseverfahren basieren auf einer systematischen und vollständigen Untersuchung aller Konfigurationen des Systems, was zu einem hohen Speicherbedarf führt. Die hierzu entwickelten Gegenmaßnahmen reichen von der Verwendung effizienter Datenstrukturen bis zu einer breiten Palette von Reduktionstechniken. Diese Arbeit beschäftigt sich mit der *Abstraktion* – der Reduktion der über das System verfügbaren Informationsmenge – sowie mit der Frage nach den durch Abstraktion präservierten Eigenschaften.

Wir konzentrieren uns auf die komplexeren quantitativen Eigenschaften probabilistischer (Echt-)Zeit Systeme, die in PCTL und CSL, probabilistischer Varianten der *Computation Tree Logik* (CTL), ausgedrückt werden können. Wir betrachten dazu zeitdiskrete und zeitkontinuierliche Markov-Ketten (DTMCs, CTMCs) sowie interaktive Markov-Ketten (IMCs), die das Spektrum von CTMCs um funktionales Verhalten erweitern und für die kompositionelle Modellierung von komplexen Systemen geeignet sind. Die verschiedenen Varianten von Markov-Ketten bilden die mathematische Grundlage vieler Modellierungssprachen, unter anderem aus den Bereichen der verlässlichen Systeme, der Leistungsbewertung und der Systembiologie.

In dieser Arbeit wird die Abstraktion durch das Zusammenlegen beliebig vieler Systemkonfigurationen zu abstrakten Zuständen erreicht. Durch die Zusammenlegung von Konfigurationen mit unterschiedlichem Systemverhalten muss jedoch zusätzlicher Nichtdeterminismus in das abstrakte Modell integriert werden, um die Unbestimmtheit bezüglich des konkreten Verhal-

tens zum Ausdruck zu bringen. Dieser zusätzliche Nichtdeterminismus führt dazu, dass das abstrakte System ein Verhalten zeigen kann, welches im konkreten System nicht zugelassen ist; das Verhalten des konkreten Systems kann allerdings in der Abstraktion nachempfunden werden. Daher können im abstrakten Modell gewisse Eigenschaften des konkreten Modells unter- bzw. überapproximiert werden. Das bedeutet, dass auch dann Schlüsse über die Eigenschaften des konkreten Modells gezogen werden können, wenn das konkrete Modell aufgrund seiner Größe nicht analysiert werden kann. Andererseits kann nicht ausgeschlossen werden, dass interessante Eigenschaften im abstrakten System weder verifiziert noch falsifiziert werden können, falls zu viele Informationen durch Abstraktion verloren gegangen sind. Um mit solchen ergebnislosen Analysen von (Teil-)Eigenschaften umgehen zu können, basiert das hier entwickelte *Model Checking* für abstrakte Modelle auf einer dreiwertigen Logik mit den Wahrheitswerten *ja*, *nein* und *unbekannt*.

Bei der Generalisierung des Abstraktionsverfahrens für CTMCs hin zu den kompositionellen IMCs verfolgen wir den Ansatz, die Abstraktion bereits auf Komponentenebene statt auf Systemebene durchzuführen. Dies hat den Vorteil, dass der Speicherbedarf bei der Analyse eines abstrakten kompositionellen Modells bedeutend reduziert wird. Wir untersuchen die formale Beziehung zwischen konkreten Modellen und ihren Abstraktionen und zeigen weiterhin, dass durch die Abstraktion verschiedener Komponenten Symmetrien entstehen können, die eine weitere drastische Reduktion des Speicherbedarfs ermöglichen.

Neben den theoretischen Grundlagen für Abstraktion und *Model Checking* von stochastischen Systemen entwickeln wir effiziente Algorithmen zur Analyse der abstrakten Modelle und zeigen an einer Fallstudie inwieweit Abstraktion die etablierten Techniken aus der Warteschlangentheorie bereichern können. Außerdem entwickeln wir ein erweitertes Abstraktionsverfahren, mit dem wir ein wohlbekanntes aber schwierig zu analysierendes Modell zur kinetischen Beschreibung von Enzymreaktionen untersuchen können.

Acknowledgements

In my first years as undergraduate, I never would have imagined to take a doctoral degree some day. Certainly, I did not expect to work on stochastic systems after I had passed the basic courses. It was almost an accident that I came into contact with Joost-Pieter Katoen regarding possible topics for my *Diplom* thesis, yet, his suggestion for me to work on *abstraction for continuous-time Markov chains* should determine the direction of my research for years to come. Despite one or two lost fights with the *CTMDP beast*, I found this area to be truly fascinating and I want to encourage everyone to get involved with stochastic systems.

I am very grateful to Joost-Pieter for his invaluable guidance during the last years, for his confidence in my ideas, and for providing an excellent environment for my doctoral studies. I also want to thank Martin Leucker and Verena Wolf who helped me to understand the stochastic and (just as important) the academic world better. They have been – together with Anne Remke, Boudewijn R. Haverkort and Martin R. Neuhäußer – highly valued co-authors and worthy *opponents* in many fruitful discussions. Many thanks go to Michael Huth and Stefan Kowalewski for their comments and suggestions that helped improving this thesis.

Further, I want to thank all the people I had the pleasure to work with in Aachen in the *MOVES* group, the *programming languages and verification* group as well as the *hybrid systems* group: Alexandru Mereacre, Arnd Gehrmann, Carsten Fuhs, Carsten Kern, Carsten Otto, David Jansen, Elke Ohlenforst, Erika Àbràham, Etienne Lozes, Fabian Emmes, Haidi Yue, Henrik Bohnenkamp, Ivan Zapreev, Jürgen Giesl, Martin Plücker, Nils Jansen, Peter Schneider-Kamp, René Thiemann, Sabrina von Styp, Stefan Rieger, Steffi Swiderski, Thomas Noll, Tingting Han, Ulrich Loup, Xin Chen, and all the students I have got to know over the last years. Special greetings go

to my room mate Jonathan Heinen, may his conservatism always prevail ☺. I also want to thank everyone involved in the DFG Research Training Group *AlgoSyn* and the people participating in the *VOSS/ROCKS* project for great opportunities to talk about ongoing work and present new ideas in a friendly environment.

Finally, I want to thank my family and friends who always supported (and endured) me, and who sometimes had to remind me that not just everybody does his doctoral and that it is actually a great thing to do. Above all, thanks go to my wife Myriam for 14 wonderful years that I would not trade away for all the degrees in the world.

Contents

1 Introduction

Computer systems are gaining control over almost every aspect of our lives. Internet applications, mobile devices, medical systems and car safety systems are just a few examples where correctness and reliability are of utmost importance. Flaws in the design of such systems can jeopardize data privacy and service availability, lead to severe financial penalties and can even result in catastrophic events with human casualties and lasting damage to the environment. Consequently, more and more effort is put into the verification of critical systems. This thesis extends upon a prominent verification framework – model checking – and pushes existing boundaries of that approach. The concept of abstraction – the reduction of available information – is investigated with the goal to check properties for systems of huge and even infinite size.

1.1 Abstraction in the context of model checking

In the past decades, formal methods have been developed to aid the design of correct and reliable systems. A tremendously successful automated verification technique is the so-called *model checking* approach [BK08]. The basic concept is to systematically and exhaustively explore all possible configurations of a given system, more precisely, all states of a mathematical *model* of the system, and check whether certain properties are fulfilled. A system's *model* is either provided manually, say, during a model driven design process [KWB03], or it is (automatically) derived from actual program

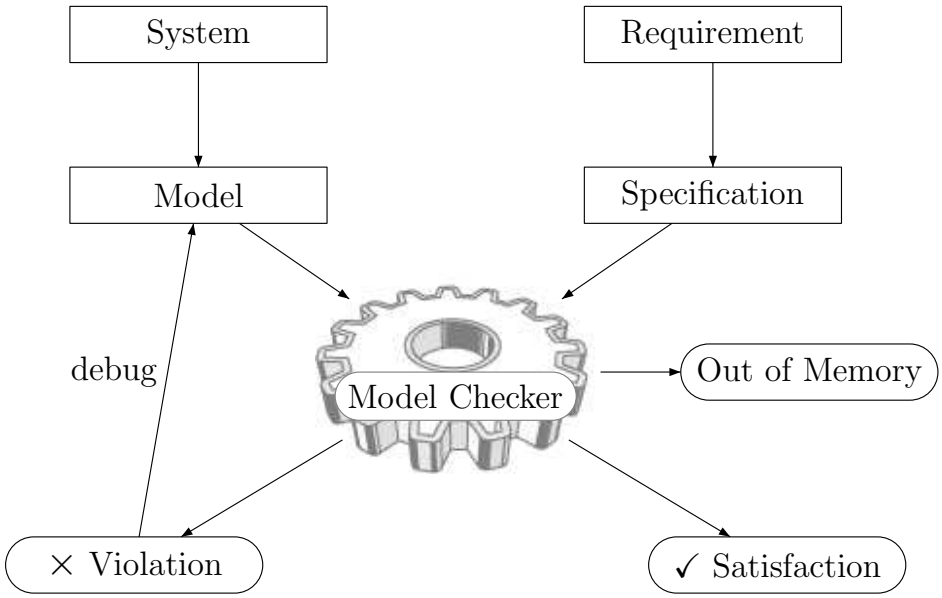


Figure 1.1: The model checking approach.

code [CKSY05]. The requirements on the system have to be formally specified as formulas (LTL, CTL, ...), automata or in high-level specification languages. Typical requirements are for example "the system may never enter a deadlock situation" or "at most one process may occupy the critical section at a time". If model checking is successful, the system's model is reported to *satisfy* the requirements. Otherwise, that is, if a requirement is violated, a counterexample is provided, serving as debug information that can be used to revise a flawed system design. For large systems, it also may happen that during the computations, the memory requirements exceed the available memory in which case nothing can be said about the satisfaction or violation of the requirements (cf. Figure 1.1).

Besides the modeling of the system and the specification of requirements itself, one of the biggest challenges in the context of model checking we face today is the so-called *state space explosion* problem: on the one hand, models are developed in high-level modeling formalisms such as guarded-command

languages [Dij75], statecharts [Har87] or Petri nets [Pet62]. On the other hand, for the verification of such models, usually the underlying low-level models (labeled transition systems) have to be built; these are often several orders of magnitude larger than the high-level description or may even have an infinite state space. By applying sophisticated reduction techniques and by using efficient data structures, nowadays, models with billions of states can be verified [CCG+02].

Probabilistic/stochastic model checking. In this thesis, we focus on systems that embrace probabilistic and stochastic aspects. Such systems are encountered in many areas, not only in computer science and engineering, but also in, say, systems biology and quantum computing. We investigate both, *probabilistic* systems where time is assumed to be discrete, i.e. clocked, as well as *stochastic* systems where time is modeled continuously. Examples where such advanced models are inevitable include programs with randomized variables, systems with unreliable components such as sensors or communication channels, and systems for which the mean time to failure of its components is of interest.

In recent years, model checking has been extended to models with transition probabilities and residence time distributions such as discrete-time and continuous-time Markov chains [BK08, BHHK03]. Requirements on such systems are typically quantitative, therefore logics such as PCTL and CSL have been proposed which, for example, allow to specify that "the probability for a breakdown between two inspections is at most 0.5%" or that "an IP address has been obtained within 10 seconds with a probability of at least 95%". While there has been quite some progress in the applicability of formal methods to probabilistic and stochastic systems – models with millions of states can be verified by state of the art probabilistic model checkers [KZH+09, KNP09] – a technical inevitability is that in general, a probability vector of the size of the state space has to be maintained rather than a bit-vector as for classical model checking. Thus, finding ways to fight the

state space explosion problem is even more crucial in this setting. The list of techniques that are used for the reduction of the state space of probabilistic models ranges from bisimulation minimization (which in contrast to the classical setting actually pays off [KKZJ07]) over partial order reduction [GB06] to specialized algorithms for certain classes of nicely structured models [RHC07]. In this thesis, we focus on a reduction method known as *abstraction* [CGL94, DJJL01].

Abstraction. Usually, abstraction is understood as a generalization that may come with a certain loss of information. For example, one may abstract from the value of a program variable that does not appear in the requirements. However, performing abstraction may yield an abstract model for which some properties cannot be verified anymore, e.g., because the value of the variable has a crucial impact on the control flow of the program; in this case, we say that the abstraction is too coarse. By iteratively *refining* the abstraction and starting over with a richer abstract model, conclusive results may finally be obtained. A prominent variant of this procedure is the CEGAR approach (counterexample-guided abstraction refinement, [CGJ$^+$00]) for which an extension to probabilistic model checking has been proposed recently [HWZ08].

The abstraction methods presented in this thesis are based on a partitioning of the state space. For example, we may group all states of a program for which the sum of two given variables are the same. Then, it is no longer possible to decide in any state of the abstract model if the minimum of the two variables is, say, zero. On the other hand, it is also not possible to disregard this property: the sum of two values x and $y = -x$ is obviously the same for all $x, y \in \mathbb{N}$, however, for $x = y = 0$ the minimum is zero whereas for $y = -x \neq 0$ the minimum is strictly smaller than zero. In such a situation, we want to say that we *don't know* if the property does hold or not. Hence, we consider *three-valued abstraction* and a model checking framework that

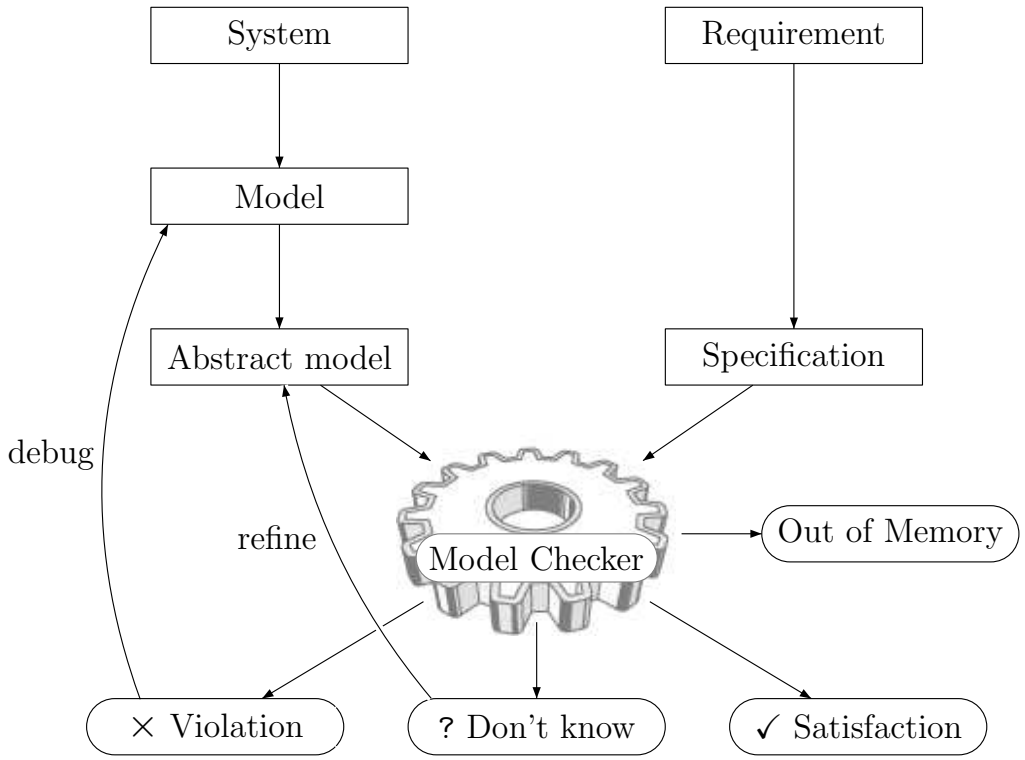


Figure 1.2: Abstraction in the context of model checking.

allows to reason with *don't know* as a third truth value. The interpretation of the results for three-valued model checking are (cf. Figure 1.2)

Violation: "the system design is flawed" and we have to debug the system's model,

Don't know: "the abstraction was too coarse" and we have to refine the partitioning of the state space,

Satisfaction: "the system is fulfilling the requirements" and we are done.

Compositionality. Manually building a *correct* model for a system is one of the most difficult and error prone tasks in the context of model checking. Fortunately, in practice, systems can often be described very naturally as a collection of interacting (concurrent) components; usually, designing each component by itself and then *plugging* the components together is much simpler than developing a *monolithic* model of the system in one shot. Therefore, many high-level modeling formalisms support compositional modeling [BHH+06, HHK02, GH94].

For the verification of non-probabilistic compositional models, several approaches have been taken that follow the Assume-Guarantee paradigm [Jon83]: assuming that a component's context satisfies certain specifications, guarantees are generated that can be used to verify other components and finally the complete system. Another more recent approach in this area extends upon three-valued abstraction [SG07]. However, in the probabilistic and especially in the stochastic world, compositional verification is still widely unexplored; notable exceptions are [SL95, KKNP08]. In this thesis, we develop the foundations necessary for performing abstraction to components of *interactive Markov chains* [HHK02], a compositional modeling formalism providing means to describe stochastic behavior.

1.2 Contributions

In the following, we give a brief overview of what to expect from this thesis. To classify this work with respect to the research field described before, the contributions are mainly in the area of abstraction of infinite-state and compositional models with probabilistic aspects. So far, especially abstraction for stochastic (continuous-time) systems has scarcely been studied. One contribution is the lifting of abstraction and model checking for abstract models to the continuous-time domain. First, we introduce the theoretical foundations (cf. [KKLW07a]):

- We propose to first *uniformize* the concrete continuous-time model before applying abstraction. *Uniformization* is a standard technique for transient analysis – today's numerical model checking tools strongly rely on this technique – and thus, uniformization adds no additional effort to the procedure.

- We show that lower and upper bounds for reachability probabilities in the concrete model can be obtained by analyzing abstractions and we also provide efficient algorithms for doing so.

- Reachability analysis is then exploited for developing a three-valued model checking framework that reports *satisfaction* or *violation* if the reachability probability bounds obtained from the abstract model are tight enough with respect to the given specifications; otherwise, if the abstraction is too coarse and therefore the resulting bounds are far off, the result will be *don't know*.

In addition to these more theoretical results, we show how abstraction and model checking techniques complement well-established techniques in queueing theory *in practice* (cf. [KRHK09]).

We then isolate a fundamental weakness of standard abstraction techniques that are based on so-called *stepwise simulation* and present a remedy for this weakness (cf. [KKLW08]):

- Instead of considering only single transitions to direct successors, sequences of transitions are merged to abstract transitions.

- We show that we still can obtain lower and upper bounds for reachability probabilities from the obtained abstractions; further, we adapt the efficient algorithms from [KKLW07a] to this setting.

- The feasibility of the approach is shown in terms of a case study from systems biology.

Another major step forward is our work on compositional abstraction for interactive Markov chains (cf. [KKN09]):

- We propose to perform abstraction on the component level; the (abstract) monolithic model we need for the analysis of compositional stochastic models is then built directly from the abstract components. This approach allows for radical reductions of the peak memory consumption during the construction of the monolithic model.
- As we show, it is most beneficial to have one abstraction for several similar concrete components such that symmetries in a compositional model can be exploited.

To our knowledge, for compositional systems exhibiting stochastic behavior, such as interactive Markov chains, no abstraction facilities have been available so far.

This thesis is based on the following publications:

[KKLW07a] Joost-Pieter Katoen, Daniel Klink, Martin Leucker, and Verena Wolf. Three-valued abstraction for continuous-time Markov chains. In *International Conference on Computer-Aided Verification (CAV)*, volume 4590 of *LNCS*, pages 316–329. Springer, 2007.

[KKLW08] Joost-Pieter Katoen, Daniel Klink, Martin Leucker, and Verena Wolf. Abstraction for stochastic systems by Erlang's method of stages. In *International Conference on Concurrency Theory (CONCUR)*, volume 5201 of *LNCS*, pages 279–294. Springer, 2008.

[KRHK09] Daniel Klink, Anne Remke, Boudewijn R. Haverkort, and Joost-Pieter Katoen. Time-bounded reachability in tree-structured QBDs by abstraction. In *International Conference on Quantitative Evaluation of Systems (QEST)*, pages 133–142. IEEE CS, 2009.

[KKN09] Joost-Pieter Katoen, Daniel Klink, and Martin R. Neuhäußer. Compositional abstraction for stochastic systems. In *International Conference on Formal Modelling and Analysis of Timed Systems (FORMATS)*, volume 5813 of *LNCS*, pages 195–211. Springer, 2009.

1.3 Structure of the thesis

In Chapter 2, we define some notations and give the necessary background on stochastic models and suitable logics. Further, we recall bisimulation minimization in terms of a reduction method. In Chapter 3, we introduce two abstraction methods for fully probabilistic models with discrete and continuous time; we also establish a formal relation between concrete and abstract models: stepwise probabilistic simulation. In Chapter 4, we are concerned with the analysis of abstract models and the preservation of properties when performing abstraction. We present three-valued model checking and show that definite results w.r.t. an abstraction carry over to the concrete model, more precisely, if a property is definitely satisfied or violated in the abstract model, the same applies to the concrete model. The chapter is concluded with a case study from queueing theory where sensible results can be obtained by using abstraction and where other techniques are not applicable.

In Chapter 5, we point out a shortcoming of the abstraction methods presented so far. Basically, stepwise simulation prevents obtaining tight abstractions for a significant class of models. Therefore we generalize the framework from stepwise to k-stepwise simulation. We show how to obtain respective abstractions and give preservation results that are needed for model checking. Finally, a well known case study from the area of systems biology, called *enzyme catalysed substrate conversion*, is analyzed successfully by choosing a proper partitioning and quite large values for k.

In Chapter 6, we extend abstraction as introduced in Chapter 3 to interactive Markov chains. Instead of simply abstracting the monolithic model

of a system, we propose to abstract on component level and show how to obtain a proper abstract monolithic model from abstract components. This approach allows for radical reductions of the peak size of the state space during construction of the monolithic model, especially when symmetries can be exploited that are introduced by abstraction. Finally, Chapter 7 concludes the thesis.

2
Preliminaries

In this chapter, we define notations used throughout the thesis and give some mathematical background. Further, we recall the concept of Markov chains and Markov decision processes, both, for the discrete-time and the continuous-time setting. We also introduce the temporal logics PCTL and CSL, which are suitable for expressing properties of these models, and recall results on the time complexity for model checking such properties. Finally, bisimulation minimization for discrete-time Markov chains is presented in the spirit of abstraction and the limits of this method are exemplified.

2.1 Notations

For function $f : X \times \ldots \times X \to \mathbb{R}_{\geq 0}$ and sets $X_0, \ldots, X_n \subseteq X$, we define $f(X_0, \ldots, X_n) = \sum_{x_0 \in X_0, \ldots, x_n \in X_n} f(x_0, \ldots, x_n)$, provided the sum exists. Whenever a set X_i is a singleton, we may omit brackets. Function $f(x_0, \ldots, x_{i-1}, \cdot, x_{i+1}, \ldots, x_n) : X \to \mathbb{R}_{\geq 0}$ is given by $x_i \mapsto f(x_0, \ldots, x_i, \ldots, x_n)$. For $f : X \to \mathbb{R}_{\geq 0}$ and any $x \in X$, $y \in \mathbb{R}_{\geq 0}$, we define $f' = f[x \mapsto y]$ by $f'(x) = y$ and $f'(z) = f(z)$ for all $z \neq x$.

We call $\mu : X \to [0, 1]$ with $\mu(X) = 1$ a distribution with support $\mathit{supp}(\mu) = \{x \in X \mid \mu(x) > 0\}$. The set of all distributions on X is denoted by $\mathit{distr}(X)$. Further, by $\{x_0 \mapsto p_0, \ldots, x_n \mapsto p_n\}$ with $\sum_{i=0}^{n} p_i = 1$ we denote a distribution μ with $\mu(x_i) = p_i$ for all $i \in \{0, \ldots, n\}$.

Three-valued truth domain. Let AP be a finite set of atomic propositions. For convenience, we fix AP for the remainder of this thesis. Typically, a proposition is interpreted over the *two-valued truth domain* $\mathbb{B}_2 = \{\bot, \top\}$, meaning that a proposition is either *true*, denoted by $\top$, or *false*, denoted by $\bot$; the isomorphism $\mathbf{1} : \mathbb{B}_2 \mapsto \{0, 1\}$, identifies truth values $\bot$ and $\top$ with 0 and 1 respectively. However, as by abstraction states may be grouped from which only some but not all satisfy a certain atomic proposition, we resort to a three-valued interpretation.

The *three-valued truth domain* $\mathbb{B}_3 = \{\bot, ?, \top\}$ adds the element ? to $\mathbb{B}_2$, which should be read as *don't know* or *indefinite*. $\mathbb{B}_3$ forms a lattice with ordering $\bot < ? < \top$ and *meet* ($\sqcap$) and *join* ($\sqcup$) of two elements are defined as usual to yield their minimum and maximum, respectively. Moreover, $\mathbb{B}_3$ turns into a *de Morgan* lattice when defining complementation $\cdot^c$ such that $\bot$ and $\top$ are complementary to each other and $?^c = ?$. When a proposition, and later on also a formula, evaluates to $\bot$ or $\top$, we call the result *definitely* true or false respectively, while we call it *indefinite* otherwise.

Measure theory. A *sample space* is a non-empty set Ω of possible outcomes of an experiment of chance. A set $\mathcal{B} \subseteq 2^\Omega$ is a *σ-algebra* over Ω if it contains Ω, $\Omega \backslash E$ for each $E \in \mathcal{B}$, and the union of any countable sequence of sets from $\mathcal{B}$. The subsets of Ω that are elements of $\mathcal{B}$ are called *measurable*. The σ-algebra *generated* by a countable set $\mathcal{E}$, denoted by $\langle \mathcal{E} \rangle$, is the smallest σ-algebra that contains all elements of $\mathcal{E}$.

A *probability space* is a triple $(\Omega, \mathcal{B}, Pr)$, where Ω is a sample space, $\mathcal{B}$ is a σ-algebra over Ω, and $Pr : \mathcal{B} \to [0, 1]$ is such that $Pr(\Omega) = 1$ and $Pr(\bigcup_{i=0}^{\infty} E_i) = \sum_{i=0}^{\infty} Pr(E_i)$ for any sequence $E_0, E_1, \ldots$ of pairwise disjoint sets of $\mathcal{B}$. We call Pr a *probability measure*. We may refer to Pr as a *distribution on* Ω if Ω is a countable set and $\mathcal{B} = 2^\Omega$, the power set of Ω. For $x \in \Omega$, we abbreviate $Pr(\{x\})$ by $Pr(x)$ and we may consider Pr as a function $Pr : \Omega \to [0, 1]$.

2.2 Stochastic processes

In this section, stochastic processes and some of their properties are introduced in as far as they are needed in the context of this thesis. A formally rigorous and comprehensive introduction can be found in [Fel68].

A *stochastic process* is a collection of random variables $\{X(t) \mid t \in T\}$, defined on a probability space. The parameters in T are assumed to represent points in time. If T is countable, we speak of *discrete* time; in this case, typically we have $T = \mathbb{N}$ where $t \in T$ is interpreted as t time units (seconds, hours, years, ...) after the process has been initially started. Otherwise, if time is *continuous*, we usually have $T = \mathbb{R}_{\geq 0}$. For the domain of $X(t)$ – also called the state space – we restrict ourselves to the discrete setting, i.e., we assume the state space to be countable.

A stochastic process has the *Markov property* iff for all $t' \in T$:

$$Pr(\{X(t+t') = s' \mid \forall z \leq t : X(z) = s_z\}) = Pr(\{X(t+t') = s' \mid X(t) = s_t\})$$

Intuitively, the Markov property holds, if the probability to end up in some state s' after any t' time units have elapsed does *not* depend on the history of the process (the values of $X(z)$ for $z \leq t$), but only on the current state s_t.

Another important property that often is presumed is *time-homogeneity*. Formally, a stochastic process is time-homogeneous, iff for all $t, t' \in T$:

$$Pr(\{X(t+t') = s' \mid X(t) = s\}) = Pr(\{X(t') = s' \mid X(0) = s\})$$

This means that the subsequent states of the process do not depend on the time that has passed since the process has been started.

Example 2.2.1. Let $\{X(t) \mid t \in \mathbb{N}\}$ be a Markovian, time-homogeneous, discrete-time stochastic process with state space S. Further let

$$Pr(\{X(t+1) = s' \mid X(t) = s\}) = \mathbf{P}(s, s') \text{ for all } s, s' \in S$$

and $\mu_0 \in \mathit{distr}(S)$, where $\mu_0(s)$ is the probability for the process to be in state s at time 0. By the rule of conditional probability, we compute:

$$\begin{aligned}
&Pr(\{X(n) = s_n, X(n-1) = s_{n-1}, \ldots, X(0) = s_0\}) \\
&= Pr(\{X(n) = s_n \mid X(n-1) = s_{n-1}\}) \cdot \ldots \cdot \\
&\quad Pr(\{X(1) = s_1 \mid X(0) = s_0\}) \cdot Pr(\{X(0) = s_0\}) \\
&= \mu_0(s_0) \cdot \mathbf{P}(s_0, s_1) \cdot \ldots \cdot \mathbf{P}(s_{n-1}, s_n)
\end{aligned}$$

2.3 Discrete-time models

First, we recall discrete-time Markov chains (DTMCs), a fully probabilistic model, and discrete-time Markov decision processes (MDPs), a nondeterministic extension of DTMCs. These models have been studied in length and efficient model checking algorithms have been developed and successfully applied to models from areas such as randomized distributed algorithms, planning, and communication protocols [Her90, LSS94, Rab82, KNPS06, KNS03].

Markov chains are also called $\frac{1}{2}$-player games where $\frac{1}{2}$ refers to the fact that a probabilistic choice, e.g. heads or tails, can also be made by a player who may choose freely, but not vice versa. Consequently, Markov decision processes are also called $1\frac{1}{2}$-player games as they extend Markov chains by one source of nondeterminism, a player who may choose freely. Due to the close relation to game theory, the formalisms will be exemplified by a simple game of luck in this section.

2.3.1 Discrete-time Markov chains

A discrete-time Markov chain (DTMC for short) is a stochastic process with a possibly infinite, yet countable, set of states S and an initial distribution μ_0; $\mu_0(s)$ for $s \in S$ denotes the probability that the DTMC will start in state s. The probability to move in one step from state s to a *successor* s' is given by a transition probability function $\mathbf{P}(s, s')$, for any pair $s, s' \in S$. We require each state to have at least one successor state that is reachable with a positive

probability. Each state may satisfy certain atomic propositions, which is described by a mapping L. A DTMC is *time discrete* as no explicit notion of real-time is part of the definition, yet each transition may be identified with the passage of a single time unit.

Definition 2.3.1 (DTMC)**.** A *Discrete-Time Markov Chain* (DTMC) is a tuple $(S, \mathbf{P}, L, \mu_0)$ with a non-empty, countable set of states S, transition probability function $\mathbf{P} : S \times S \to [0,1]$ satisfying $\mathbf{P}(s, S) = 1$ for all $s \in S$, labeling function $L : S \times AP \to \mathbb{B}_2$, and initial distribution $\mu_0 \in \mathit{distr}(S)$.

A *path* in a DTMC $\mathcal{M} = (S, \mathbf{P}, L, \mu_0)$ is an (infinite) sequence $\rho = s_0\, s_1\, s_2 \ldots$, sometimes written as $s_0 \to s_1 \to s_2 \ldots$; $\rho[i]$ denotes the $(i{+}1)$-st state of ρ, i.e., $\rho[i] = s_i$, and $\rho[i..j]$ denotes the sequence $s_i\, s_{i+1} \ldots s_j$. A finite prefix $\varrho = s_0\, s_1\, s_2 \ldots s_n$ of a path $\rho = s_0\, s_1\, s_2 \ldots$ is called a *path fragment*. The last element of ϱ is denoted by $\varrho\!\downarrow$. We write $\mathit{Path}_s^{\mathcal{M}}$ ($\mathit{Pathf}_s^{\mathcal{M}}$) for the set of all paths (path fragments, respectively) that start with state s. Sometimes we restrict the length of path fragments, e.g. we write $\mathit{Pathf}_{s,\leq n}^{\mathcal{M}}$ for the set of all path fragments starting in s with length of at most n.

The probability measure $\mathit{Pr}^{\mathcal{M}}$ on sets of paths is constructed as follows. Let Ω be the set $\mathit{Path}^{\mathcal{M}} = \bigcup_{s \in S} \mathit{Path}_s^{\mathcal{M}}$, and $\mathcal{C}(s_0 \ldots s_n)$ be the *(basic) cylinder set* of $s_0 \ldots s_n$, i.e., the set of all paths in $\mathit{Path}^{\mathcal{M}}$ with prefix $s_0 \ldots s_n \in \mathit{Pathf}^{\mathcal{M}}$. Let $\mathcal{B}$ be the σ-algebra generated by all cylinder sets, i.e., $\mathcal{B} = \langle (\mathcal{C}(s_0 \ldots s_n))_{s_0 \ldots s_n \in \mathit{Pathf}^{\mathcal{M}}} \rangle$. We define

$$\mathit{Pr}^{\mathcal{M}}(\mathcal{C}(s_0 \ldots s_n)) = \mu_0(s_0) \cdot \textstyle\prod_{i=0}^{n-1} \mathbf{P}(s_i, s_{i+1}).$$

Then, by Caratheodory's extension theorem, $\mathit{Pr}^{\mathcal{M}}$ extends to a probability measure on $\mathcal{B}$ in a unique manner [Fel68]. We sometimes write $\mathit{Pr}_s^{\mathcal{M}}$ instead of $\mathit{Pr}^{\mathcal{M}}$ in case s is the unique initial state of $\mathcal{M}$, i.e., $\mu_0(s) = 1$ and $\mu_0(s') = 0$ for $s' \neq s$, denoted $\mu_0 = \{s \mapsto 1\}$ in the following. We omit superscript $\mathcal{M}$ whenever $\mathcal{M}$ is clear from the context.

Reachability probabilities. The probability distribution after n-steps in a DTMC is called the n-th step *transient-state* probability distribution. Formally, for initial distribution μ_0 and the transition probability function $\mathbf{P}$ interpreted as matrix, the n-th step transient-state probability vector is given by $\mu_0 \cdot \mathbf{P}^n$. Transient probabilities can be exploited to obtain step-bounded reachability probabilities, i.e. probabilities for reaching some state in B after at most n transitions,

$$Pr(Reach^{\leq n}(B)) = Pr(\{\rho \in Path \mid \exists i \leq n : \rho[i] \in B\})$$

in a DTMC by transforming it such that all states $s \in B$ are absorbing, i.e. $\mathbf{P}(s, s) = 1$ for all $s \in B$.

Example 2.3.1. Consider an infinite state DTMC $(\{s_i \mid i \in \mathbb{Z}\}, \mathbf{P}, L, \{s_0 \mapsto 1\})$ where $\mathbf{P}(s_i, s_{i+j}) = \frac{1}{6}$ for all $i \in \mathbb{Z}$, $1 \leq j \leq 6$ and $L(s, goal) = \top$ iff $s = s_{42}$. Intuitively, this models a simple game of luck where the goal is to see *exactly* 42 pips by throwing a dice repeatedly.

For example, the sequence of pip values 3 1 4 1 5 corresponds to the path fragment $s_0 \to s_3 \to s_4 \to s_8 \to s_9 \to s_{14}$ in the DTMC and the probability for throwing this sequence of values is $\mathbf{P}(s_0, s_3) \cdot \ldots \cdot \mathbf{P}(s_9, s_{14}) = (\frac{1}{6})^5 = 0.0001286\ldots$. The set of sequences that win this game, i.e. that reach state s_{42} after some time, is given by:

$$\{v_0 v_1 \ldots \mid \forall i \in \mathbb{N} : v_i \in \{1, \ldots, 6\} \wedge \exists n \in \mathbb{N} : \textstyle\sum_{i=0}^{n} v_i = 42\}$$

This can also be described as a set of paths in the DTMC model:

$$W = \{s_0 \to u_1 \to u_2 \to \ldots \in Path \mid \exists n \in \mathbb{N} : u_n = s_{42}\}$$

Note that for $\varrho = s_0 \to s_{42}$, say, it holds $\varrho \in W$, but $\Pr(\mathcal{C}(\varrho)) = \mu_0(s_0) \cdot \mathbf{P}(s_0, s_{42}) = 0$. The probability for winning the game is consequently the sum over the probabilities of all cylinder sets that describe paths with a prefix ending in s_{42}:

$$Pr(W) = \textstyle\sum_{\varrho \in Pathf : \varrho\downarrow = s_{42} \wedge \exists \rho \in W \cap \mathcal{C}(\varrho)} Pr(\varrho) = 0.2857141\ldots$$

2.3.2 Discrete-time Markov decision processes

Discrete-time Markov decision processes (DTMDPs, in short MDPs) consist of an infinite set of states S, an initial distribution μ_0 and a labeling function L as known from Markov chains. In contrast to DTMCs, the behavior of an MDP is nondeterministic. More precisely, in each state, one out of *finitely* many actions a can be chosen. Further, each action induces a probability distribution over the set of states and thus, the probability to move in one step from a state s via action a to a successor s' is given by the three-dimensional transition probability function $\mathbf{P}(s, a, s')$. Note that in contrast to other popular models [Seg95, Her02], for all actions a there is *at most one* a transition emanating a state in an MDP. Opposed to standard literature, we consider MDPs to have a *three-valued* labeling function, such that we can express uncertainty on the satisfaction of an atomic proposition in a state.

As for DTMCs, we require deadlock-freeness: at least one action has to be enabled in each state, i.e. for each state s there exists an action a such that $\mathbf{P}(s, a, S) = 1$.

Definition 2.3.2 (MDP)**.** A *Discrete-Time Markov Decision Process* is a tuple $(S, A, \mathbf{P}, L, \mu_0)$ with a non-empty, finite set of states S, a finite set of actions A, a three-dimensional transition probability function $\mathbf{P} : S \times A \times S \to [0, 1]$ satisfying $\mathbf{P}(s, a, S) \in \{0, 1\}$ for all $s \in S$, $a \in A$, *three-valued* labeling function $L : S \times AP \to \mathbb{B}_3$, and initial distribution $\mu_0 \in \mathit{distr}(S)$.

A *path* in an MDP $\mathcal{M} = (S, A, \mathbf{P}, L, \mu_0)$ is an (infinite) sequence $\rho = s_0\, a_0\, s_1\, a_1\, s_2 \ldots$, sometimes written as $s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \ldots$; a finite prefix of a path ρ is called a path fragment $\varrho = s_0\, a_0\, s_1\, a_1\, s_2 \ldots s_{n-1}\, a_{n-1}\, s_n$. We use the same notations as for DTMCs: $\rho[i]$, $\varrho\downarrow$, $\mathit{Path}_s^{\mathcal{M}}$ and $\mathit{Pathf}_s^{\mathcal{M}}$. Further, the set of *enabled* actions in state $s \in S$ is denoted $A(s) = \{a \in A \mid \mathbf{P}(s, a, S) = 1\}$ and the set of probability distributions that are available in s is given by $\mathbf{T}(s) = \{\mathbf{P}(s, a, \cdot) \in \mathit{distr}(S) \mid a \in A(s)\}$.

Probability measures in MDPs depend on the nondeterministic choices that are made in each state of the system. The instance making these choices is called the *scheduler*, *strategy*, *adversary* or *policy*. In the following example, *strategy* may fit the best, however, throughout the thesis the term *scheduler* will be used.

Example 2.3.2. Reconsider the game of luck introduced in Example 2.3.1. By extending the rules of the game, we can give the player some influence on the outcome of the game: before the dice is thrown, the player may decide if all pips count (as before), or alternatively if only 1 and 2 pips should count and on 3 to 6, the counting direction is switched. This allows the player to reach states s_i from states s_j where $i < j$. This is especially important when more that 42 pips have been counted so far.

This set of rules cannot be modeled as a DTMC due to the lack of nondeterminism, instead, we model it as a discrete-time MDP $(S, A, \mathbf{P}, L, \mu_0)$ with state space $S = \{(s_i, \circ) \mid i \in \mathbb{Z}, d \in \{+, -\}\}$ where the first component is as before and the second component indicates the counting direction, and with $A = \{all, sw\}$. Action $all \in A$ corresponds to the choice where *all* pips count and action $sw \in A$ corresponds to the alternative choice where the counting direction is attempted to *switch*. For all $i \in \mathbb{Z}$, $\circ \in \{+, -\}$ and $a \in A$, the transition probabilities are given by:

$$\mathbf{P}((s_i, \circ), a, (s', \circ')) = \begin{cases} \frac{1}{6} & \text{if } s' = s_{i \circ j} \wedge \circ' = \circ \wedge 1 \leq j \leq 6 \wedge a = all \\ \frac{1}{6} & \text{if } s' = s_{i \circ j} \wedge \circ' = \circ \wedge 1 \leq j \leq 2 \wedge a = sw \\ \frac{2}{3} & \text{if } s' = s \wedge \circ' \neq \circ \wedge 3 \leq j \leq 6 \wedge a = sw \\ 0 & \text{otherwise} \end{cases}$$

The labeling function is given by $L((s, \circ)) = \top$ iff $s = s_{42}$ and initially, the counter is set to 0 and the direction is *up*, i.e. $\mu_0 = \{(s_0, +) \mapsto 1\}$.

As for DTMCs, a path (fragment) reflects the course of a single play in the game, e.g., throwing the same sequence 3 1 4 1 5 results in different pip counts, depending on the player's choices. Always choosing *all* yields

$$(s_0, +) \xrightarrow{all} (s_3, +) \xrightarrow{all} (s_4, +) \xrightarrow{all} (s_8, +) \xrightarrow{all} (s_9, +) \xrightarrow{all} (s_{14}, +)$$

whereas choosing *switch* for the second and third throw results in:

$$(s_0,+) \overset{all}{\to} (s_3,+) \overset{switch}{\longrightarrow} (s_4,+) \overset{switch}{\longrightarrow} (s_4,-) \overset{all}{\to} (s_3,-) \overset{all}{\to} (s_{-2},-)$$

For this set of rules, there are strategies such that a player can win every play in the game on the long run. For example, choosing *sw* in states $(s_i,+)$ with $i > 42$ and in states $(s_i,-)$ with $i < 42$ and by choosing *all* otherwise is such a strategy. On the other side, even the worst imaginable player cannot loose almost surely as the probability to, e.g., throw 42 ones in a row, thus not being able to switch the counting direction and ending up in the *goal* state, is positive.

Schedulers classes

As shown in Example 2.3.2, nondeterminism in MDPs is resolved by schedulers (strategies, ...). The choice of a scheduler may depend on the current state or on the history of the current run, it may be a deterministic choice or a randomized one. According to these distinctions we define the following types of schedulers:

- stationary Markovian deterministic (SMD or simple schedulers)

  $$D : S \to A \quad \text{such that} \quad D(s) \in A(s)$$

- stationary Markovian randomized (SMR)

  $$D : S \to \mathit{distr}(A) \quad \text{such that} \quad \mathit{supp}(D(s)) \subseteq A(s)$$

- history-dependent, deterministic (HD)

  $$D : \mathit{Pathf} \to A \quad \text{such that} \quad D(s) \in A(s)$$

- history-dependent, randomized (HR)

  $$D : \mathit{Pathf} \to \mathit{distr}(A) \quad \text{such that} \quad \mathit{supp}(D(s)) \subseteq A(s)$$

HR-schedulers are the most general ones in this setting. By restricting the support to a single successor, one obtains HD-schedulers; forcing the scheduler to decide the same for all path fragments ϱ, ϱ' with $\varrho\downarrow = \varrho'\downarrow$ yields stationary schedulers. In the following, the set of all history-dependent schedulers on $\mathcal{M}$ is denoted $\mathcal{D}^{\mathcal{M}}$.

For an MDP, each scheduler induces its own stochastic process, in general, an infinite state DTMC. Basically, the induced DTMC results from unwinding the MDP and fixing the choices according to the given scheduler. Formally, a scheduler $D : \mathit{Pathf}^{\mathcal{M}} \to \mathit{distr}(A)$ on MDP $\mathcal{M} = (S, A, \mathbf{P}, L, \mu_0)$ induces the DTMC $\mathcal{M}_D = (S_D, \mathbf{P}_D, L_D, \mu_0)$ where $S_D = \mathit{Pathf}^{\mathcal{M}}$ is the infinite, yet countable, state space, $L_D(\varrho) = L(\varrho\downarrow)$ for all $\varrho \in \mathit{Pathf}^{\mathcal{M}}$ and

$$\mathbf{P}_D(\varrho, \varrho') = \begin{cases} \sum_{a \in A} D(\varrho)(a) \cdot \mathbf{P}(\varrho\downarrow, a, s') & \text{if } \varrho' = \varrho \to s' \\ 0 & \text{otherwise.} \end{cases}$$

Now, for the resulting *induced DTMCs*, probability measures can be computed; of special interest are the supremum and the infimum over all schedulers under consideration, corresponding to the best case and worst case properties of a system.

In the discrete-time setting an important result states that, regarding extrema of reachability probabilities, HR-schedulers are no more powerful than simple schedulers, i.e. there is no HR-scheduler yielding better or worse reachability probabilities than the best and the worst simple scheduler [dA97]. This result does not carry over to the *continuous-time* setting as we will see later on.

2.4 Continuous-time models

In the following, continuous-time variants of Markov chains and Markov decision processes are introduced. While many concepts are adapted in a straightforward manner, quite some extra effort has to be put into the analysis of time-bounded reachability probabilities; especially for continuous-time

Markov decision processes, it is notoriously difficult to intuitively understand the semantics and to analyze time-bounded properties efficiently.

2.4.1 Continuous-time Markov chains

Continuous-time Markov chains consist of all components of a DTMC, countable state space S, transition probability function $\mathbf{P}$, labeling function L and an initial distribution μ_0. In addition, an exit rate vector λ parameterizes the continuous distributions describing the residence times in each state. These distributions have to be *memoryless* in order to preserve the Markov property of the model, i.e. for all $t, t' \in \mathbb{R}_{\geq 0}$ it has to hold

$$\Pr(X > t + t' \mid X > t) = \Pr(X > t').$$

As a fact, the *only* continuous probability distributions that are memoryless are exponential distributions.

Definition 2.4.1 (CTMC)**.** A Continuous-Time Markov Chain (CTMC) is a tuple $(S, \mathbf{P}, L, \mu_0, \lambda)$ with S, $\mathbf{P}$, L, and μ_0 as before, and *exit rate function* $\lambda : S \to \mathbb{R}_{>0}$.

The quantity $\lambda(s)$ determines the random, exponentially distributed residence time in state s, that is, $1 - e^{-\lambda(s)\cdot t}$ is the probability to take a transition emanating from s within t time units. If s is the current state, a transition occurs after an *average* residence time of $\frac{1}{\lambda(s)}$. The *time-dependent transition probability* to move from s to s' within t time units is given by $\mathbf{P}(s, s', t) = \mathbf{P}(s, s') \cdot (1 - e^{-\lambda(s)\cdot t})$.

A *path* in a CTMC $\mathcal{M} = (S, \mathbf{P}, L, \mu_0, \lambda)$ is an alternating sequence $\sigma = s_0\, t_0\, s_1\, t_1\, s_2 \ldots$ with $s_i \in S$ and $t_i \in \mathbb{R}_{>0}$ for all $i \in \mathbb{N}$. The time stamps t_i denote the amount of time spent in state s_i. As the probability to reside zero time units in a state is zero, all time stamps t_i are strictly positive. For a path $s_0\, t_0\, s_1\, t_1\, s_2 \ldots$ we may also write $s_0 \xrightarrow{t_0} s_1 \xrightarrow{t_1} s_2 \ldots$. Again, $\sigma[i]$ denotes the $(i{+}1)$-st state of a path, i.e., $\sigma[i] = s_i$. By $\sigma@t$ we denote the state of

σ occupied at time t, i.e. $\sigma@t = s_i$ where i is the smallest index such that $t < \sum_{j=0}^{i} t_j$. A *path fragment* ς in a CTMC is a finite prefix of a path ending with a state. As for DTMCs, we write $Path_s^{\mathcal{M}}$ ($Pathf_s^{\mathcal{M}}$) for the set of all paths (path fragments) of $\mathcal{M}$ that start with s, and, $\cdot\downarrow$ for the last state of a path fragment. By $Path^{\mathcal{M}}$, we denote the set of all paths in $\mathcal{M}$. For sets of time-abstract paths in a CTMC, i.e. paths where the time stamps have been masked, we add subscript *abs* to $Path^{\mathcal{M}}$.

The probability measure $Pr^{\mathcal{M}}$ on the sample space $\Omega = Path^{\mathcal{M}}$ is defined in a similar way as for DTMCs. A cylinder set $\mathcal{C}(s_0\, I_0\, s_1\, \ldots\, I_{n-1}\, s_n)$, however, now depends on intervals $I_i = (0, z_i]$, $z_i \in \mathbb{R}_{>0}$, $i \in \{0, 1, \ldots, n-1\}$ and contains all paths $u_0\, t_0\, u_1\, t_1 \ldots \in Pathf^{\mathcal{M}}$ with $u_i = s_i$, $t_i \in I_i$, for $0 \leq i < n$, and $u_n = s_n$. Let $F(s, I)$ denote the probability of leaving state s within interval $I = (0, z]$, given by

$$F(s, I) = \int_0^z \lambda(s) \cdot e^{-\lambda(s)\cdot t}\, dt = 1 - e^{-\lambda(s)\cdot z}.$$

Then, $Pr^{\mathcal{M}}$ is given by

$$Pr^{\mathcal{M}}(\mathcal{C}(s_0\, I_0\, \ldots\, I_{n-1}\, s_n)) = \mu_0(s_0) \cdot \prod_{i=0}^{n-1} \mathbf{P}(s_i, s_{i+1}) \cdot F(s_i, I_i)$$

and extends to a probability measure on $\mathcal{B}$ in a unique manner. As for DTMCs, we write $Pr_s^{\mathcal{M}}$ if s is the unique initial state. Sub- and superscripts are sometimes omitted to enhance readability.

The *time-abstract probabilistic behavior* of a CTMC $\mathcal{M} = (S, \mathbf{P}, L, \mu_0, \lambda)$ is described by its *embedded DTMC* $emb(\mathcal{M}) = (S, \mathbf{P}, L, \mu_0)$.

A CTMC is *uniform* if all of its states have the same exit rate λ, i.e., $\lambda(s) = \lambda \in \mathbb{R}_{>0}$ for all $s \in S$. As we will see in the following, uniform CTMCs allow a more accurate form of abstraction, and there exist efficient algorithms for computing time-bounded reachability probabilities. A non-uniform CTMC can be transformed into a uniform one which has identical (time-bounded) reachability probabilities. To this end, we fix a uniform exit rate $\lambda \geq \sup_{s\in S} \lambda(s)$ for all states. This replaces the average residence time $\frac{1}{\lambda(s)}$ of state s by a shorter (or equal) one of length $\frac{1}{\lambda}$. To increase the

time spent in a state s with $\lambda(s) < \lambda$, it is equipped with a self-loop with probability $1 - \frac{\lambda(s)}{\lambda}$.

Definition 2.4.2 (Uniformization)**.** Let $\mathcal{M} = (S, \mathbf{P}, L, \mu_0, \lambda')$ be a CTMC and let uniformization rate $\lambda \in \mathbb{R}_{>0}$ with $\lambda \geq \sup_{s \in S} \lambda'(s)$. Then, $\mathit{unif}_\lambda(\mathcal{M}) = (S, \mathbf{P}_\lambda, L, \mu_0, \lambda)$ is a *uniform CTMC* with $\lambda(s) = \lambda$ for all $s \in S$ and

$$\mathbf{P}_\lambda(s, s') = \begin{cases} \frac{\lambda'(s)}{\lambda} \cdot \mathbf{P}(s, s') & \text{if } s' \neq s, \\ \frac{\lambda'(s)}{\lambda} \cdot \mathbf{P}(s, s') + 1 - \frac{\lambda'(s)}{\lambda} & \text{if } s' = s. \end{cases}$$

For $\lambda = \sup_{s \in S} \lambda'(s)$, we may drop the subscript and write $\mathit{unif}(\mathcal{M})$. In the literature [GM84], uniformization is typically defined as a transformation of a CTMC $\mathcal{M}$ into the DTMC $\mathit{emb}(\mathit{unif}(\mathcal{M}))$. For technical convenience, we consider uniformization as a CTMC-to-CTMC transformation.

We illustrate DTMCs and CTMCs by state-transition graphs as usual. A state s for which $\mu_0(s) > 0$ is equipped with a small incoming arrow labeled with $\mu_0(s)$. For CTMCs the (time-abstract) transition probabilities are given by transition labels and exit rates are given by state labels. Note that this differs from standard literature where a transition from state s to s' is labeled by transition rate $\mathbf{R}(s, s') = \mathbf{P}(s, s') \cdot \lambda(s)$, if $\lambda > 0$. We sometimes omit states, labels or transitions in illustrations if they are not relevant for the matter to explain.

Example 2.4.1. Consider the CTMC in Figure 2.1 (left) where $\lambda(s) = \lambda(v) = 1$ and $\lambda(u_1)$, $\lambda(u_2)$ and $\lambda(u_3)$ are 2, 100, and 10 respectively. The initial probabilities are $\mu_0(u_1) = \mu_0(u_3) = \frac{1}{6}$, $\mu_0(u_2) = \frac{2}{3}$ and zero for all other states. Let $\lambda = 100$ be the uniformization rate. Then $\mathbf{P}_\lambda(u_1, v) = \frac{2}{100} \cdot \frac{2}{3} = \frac{1}{75}$ and $\mathbf{P}_\lambda(u_1, s) = \frac{2}{100} \cdot \frac{1}{3} = \frac{1}{150}$. Thus, u_1 is equipped with a self-loop of probability $\frac{2}{100} \cdot 0 + 1 - \frac{2}{100} = \frac{98}{100}$. The self-loop probabilities of the remaining states are determined in a similar way, cf. Figure 2.1 (right). State u_2 has no self-loop as $\mathbf{P}(u_2, u_2) = 0$ and $\lambda(u_2) = \lambda$.

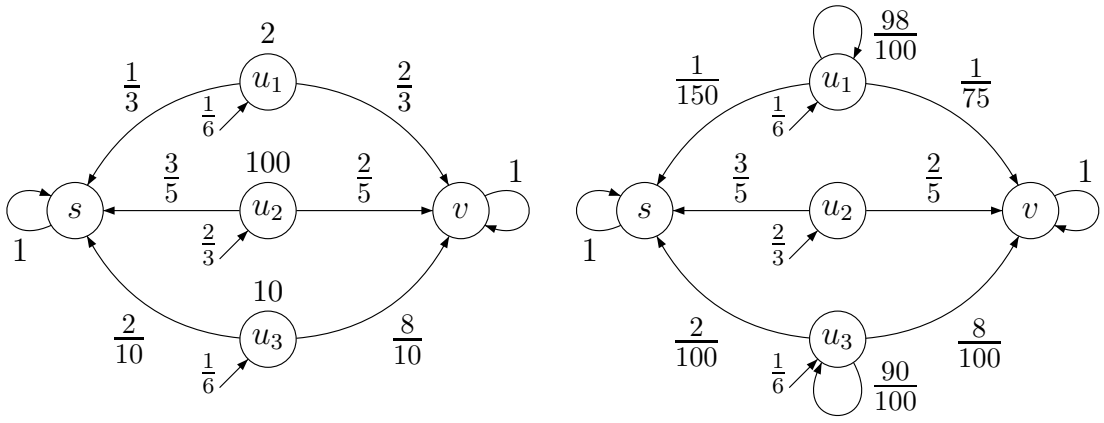


Figure 2.1: Uniformization with exit rate $\lambda = 100$

Poisson probabilities. Poisson processes model systems where events, such as the arrival of customers, occur independently from one another with the same *rate* and infinitely often. A Poisson process with parameter $\lambda \in \mathbb{R}_{>0}$ can be interpreted as a special infinite state uniform CTMC $(S, \mathbf{P}, L, \mu_0, \lambda)$ where $S = \{s_i \mid i \in \mathbb{N}\}$, $\mathbf{P}(s_i, s_{i+1}) = 1$ for all $i \in \mathbb{N}$, $L(s, ap) = \bot$ for all $s \in S$, $ap \in AP$, and $\mu_0 = \{s_0 \mapsto 1\}$.

In a Poisson process with parameter $\lambda \in \mathbb{R}_{>0}$, the probability for $n \in \mathbb{N}$ events occurring within $t \in \mathbb{R}_{\geq 0}$ time units, the *n-th Poisson probability* $\varphi_{\lambda,t}(n)$ corresponds to the transient probability for state s_n at time t:

$$\psi_{\lambda,t}(n) = e^{-\lambda \cdot t} \cdot \frac{(\lambda \cdot t)^n}{n!}$$

The naive computation of Poisson probabilities is numerically unstable due to the large factors. However, the algorithm by Fox and Glynn [FG88], implemented in state-of-the-art model checkers such as MRMC and PRISM, can be used even for large parameters.

Time-bounded reachability probabilities. A *uniform* CTMC $\mathcal{M} = (S, \mathbf{P}, L, \mu_0, \lambda)$ can now be seen as a composition of the embedded DTMC $(S, \mathbf{P}, L, \mu_0)$ and a Poisson process with parameter λ. The time-abstract

probabilistic behavior is completely captured by the embedded DTMC, thus, it carries enough information for computing unbounded reachability probabilities; all the timing information is contained in the Poisson process. This fact is also used for the efficient computation of time-bounded reachability probabilities $Pr(Reach_{\leq t}(B)) = Pr(\{\sigma \in Path \mid \exists t' \leq t : \sigma@t' \in B\})$ in uniformized CTMCs. As for DTMCs this can be computed in terms of transient probabilities in the transformed CTMC where $\mathbf{P}(s,s) = 1$ for all $s \in S$. Let $\mathbf{Q}$ with $\mathbf{Q}(s,s') = \mathbf{P}(s,s') - \mathbf{1}(s,s')$, then (cf. [BHHK03]):

$$\begin{aligned} Pr(Reach_{\leq t}(B)) &= \mu_0 \cdot e^{\lambda \cdot t \cdot \mathbf{Q}} \\ &= \mu_0 \cdot \textstyle\sum_{i=0}^{\infty} \frac{(\lambda \cdot t \cdot \mathbf{Q})^i}{i!} \\ &= \mu_0 \cdot \textstyle\sum_{i=0}^{\infty} e^{-\lambda \cdot t} \frac{(\lambda \cdot t)^i}{i!} \cdot \mathbf{P}^i \\ &= \textstyle\sum_{i=0}^{\infty} \psi_{\lambda,t}(i) \cdot (\mu_0 \cdot \mathbf{P}^i) \end{aligned}$$

While i-th step transient probabilities $\mu_0 \cdot \mathbf{P}^i$ can be computed via the embedded DTMC, the probability to have i steps in t time units, $\psi_{\lambda,t}(i)$, is calculated through the associated Poisson process. To obtain exact reachability probabilities, one has to compute the infinite sum over all i-th step transient probabilities times the probability for exactly i steps in t time units. However, a truncation point $k_\varepsilon \in \mathbb{N}$ can be computed a priori for given error bound $\varepsilon \in \mathbb{R}_{>0}$, such that k_ε is the smallest value satisfying:

$$\textstyle\sum_{i=0}^{\infty} \psi_{\lambda,t}(i) \cdot \underbrace{(\mu_0 \cdot \mathbf{P}^i)}_{\leq 1} - \sum_{i=0}^{k_\varepsilon} \psi_{\lambda,t}(i) \cdot \underbrace{(\mu_0 \cdot \mathbf{P}^i)}_{\leq 1} \leq \sum_{i=k_\varepsilon+1}^{\infty} \psi_{\lambda,t}(i) \leq \varepsilon$$

Hence, truncating at k_ε yields an ε-approximation of $Pr(Reach_{\leq t}(B))$. For large $\lambda \cdot t$, the truncation point k_ε tends to be of the order $\mathcal{O}(\lambda \cdot t)$.

Phase-type distributions. Phase-type distributions are a popular class of distributions as they can approximate general distributions arbitrarily close, yet, they can be characterized by a finite CTMC with a designated absorbing state. The time until absorption in such a CTMC is *phase-type distributed* [Neu81]. Phase-type distributions that can be characterized by a CTMC with k states plus one absorbing state are called *of the order* k (PH_k).

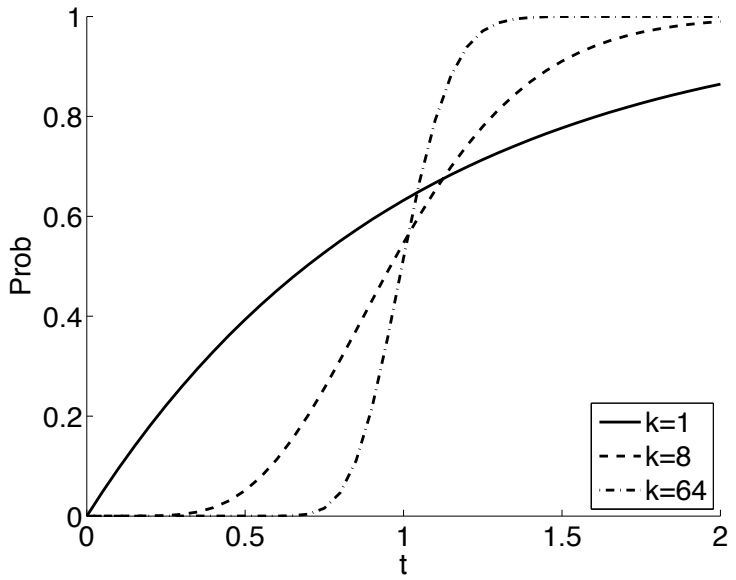


Figure 2.2: Erlang-k distributions with rate $\lambda = k$.

There exists quite some research on fitting PH-distributions onto given general distributions [HRTT04, TBT06] as well as on the reduction of PH-distributions [PH08b, PH08a].

Example 2.4.2. A special PH_k-distribution is the Erlang-k distribution: the distribution of the sum of k independent but identically (exponentially) distributed random variables. Intuitively, it describes the probability for k events, e.g. arrivals of customers, within a certain amount of time. The CTMC $(\{s_0, \ldots, s_k\}, \mathbf{P}, L, \{s_0 \mapsto 1\}, \lambda)$ with $\mathbf{P}(s_i, s_{i+1}) = 1$ for all $i < k$ and $\mathbf{P}(s_k, s_k) = 1$ characterizes an Erlang-k distribution with parameter $\lambda \in \mathbb{R}_{>0}$.

The mean of an Erlang-k distribution is given by $\frac{k}{\lambda}$ and the variance by $\frac{k}{\lambda^2}$. Thus, in order to model a step function that switches from, say, 0 to 1 at $t = 1$ very quickly, an Erlang-k distribution with parameter $\lambda = k$ can be used. As shown in Figure 2.2, a higher value for k implies a smaller variance, resulting in a steeper curve. With respect to the corresponding CTMCs, the curves from Figure 2.2 show the probabilities for reaching the absorbing state within t time units.

2.4.2 Continuous-time Markov decision processes

Continuous-time Markov decision processes (CTMDPs) extend MDPs in the same way as CTMCs extend DTMCs. However, the rise in complexity is much higher in the continuous-time setting. For example, the scheduler classes are much more diverse. Furthermore, the uniformization technique presented for CTMCs does not preserve the exact time-bounded reachability probabilities for (time-abstract) HD-schedulers, nor can it be adapted properly. Research on analysis techniques for time-bounded reachability in non-uniform CTMDPs is still at its very beginning [NZ09] while for HD-schedulers on uniform CTMDPs, an efficient algorithm has been implemented in MRMC [KZH+09], capable of analyzing models with millions of states. Thus, the construction of uniform CTMDPs is a worthy goal.

Definition 2.4.3 (CTMDP)**.** A *Continuous-Time Markov Decision Process* is a tuple $(S, A, \mathbf{P}, L, \mu_0, \lambda)$ with S, A, $\mathbf{P}$, L, μ_0 as before, and exit rate vector $\lambda : S \times A \to \mathbb{R}_{>0}$.

A CTMDP $(S, A, \mathbf{P}, L, \mu_0, \lambda)$ is called *locally uniform*, iff $\lambda(s, a) = \lambda(s, b)$ for all $s \in S$ and all $a, b \in A$; it is called *(globally) uniform*, iff $\lambda(s, a) = \lambda(u, b)$ for all $s, u \in S$ and all $a, b \in A$.

A *timed path* in a CTMDP $\mathcal{M} = (S, A, \mathbf{P}, L, \mu_0, \lambda)$ is an alternating sequence $\sigma = s_0\, a_0\, t_0\, s_1\, a_1\, t_1\, s_2 \ldots$ with $s_i \in S$, $a_i \in A$ and $t_i \in \mathbb{R}_{>0}$ for all $i \in \mathbb{N}$. A *path fragment* ς in a CTMDP is a finite prefix of a path ending with a state. For a path $s_0\, a_0\, t_0\, s_1\, a_1\, t_1\, s_2 \ldots$ we may also write $s_0 \xrightarrow{a_0,t_0} s_1 \xrightarrow{a_0,t_1} s_2 \ldots$. We use the same notations as for CTMCs: $\sigma[i]$, $\sigma\downarrow$, $\sigma@t$, $Path_s^{\mathcal{M}}$ and $Pathf_s^{\mathcal{M}}$.

As for MDPs, probability measures can only be computed with respect to a scheduler. In the continuous-time setting, the schedulers' decision may depend on the amount of time that has been spent in previously visited states; an undesirable consequence is that there exist schedulers which are *not measurable*. An example for such a scheduler in a CTMDP with $|A(s)| \geq 2$ for all states s chooses action a iff a total of $t \in V$ time units has elapsed, where V

is a *Vitali set*, a non-constructive uncountable set of irrational numbers. This problem is investigated in [WJ06], and a class of *measurable time-dependent schedulers* is defined. In this thesis, we will focus on time-abstract schedulers that base their decisions only on sequences of states and actions and are therefore measurable.

We illustrate MDPs and CTMDPs by a variant of state-transition graphs where solid arcs represent the nondeterministic choice between actions; dashed arrows represent (time-abstract) transition probabilities, induced by the action that is chosen with the solid arc they are attached to (cf. Figure 2.3). As for Markov chains, a state s for which $\mu_0(s) > 0$ is equipped with a small incoming arrow labeled with $\mu_0(s)$. We may omit states, labels or transitions in illustrations.

Example 2.4.3. Consider the CTMDP $\mathcal{M}$ depicted in Figure 2.3 (left) with exit rate $\lambda(s) = 1$ for all $s \neq s_2$ and $\lambda(s_2) = \frac{1}{2}$. Naively applying *uniformization* as introduced for CTMCs with uniformization rate 2 yields the CTMDP $\mathcal{M}_2$ depicted in Figure 2.3 (right). While in $\mathcal{M}$, there exist four HD-schedulers (which are actually simple schedulers), in $\mathcal{M}_2$ there is an infinite number of HD-schedulers that may decide differently, e.g. for histories s_0 and $s_0 \xrightarrow{a} s_0$; for such a scheduler, there is no corresponding scheduler whatsoever in $\mathcal{M}$. Intuitively, in contrast to $\mathcal{M}$, in $\mathcal{M}_2$ HD-schedulers have approximate timing information: e.g. for history s_0, it is clear that no time may have elapsed in the system yet, for $s_0 \xrightarrow{a} s_0$, on average, $\frac{1}{\lambda}$ time units have already passed.

More scheduler classes. The definitions of scheduler classes presented for MDPs are also applicable to CTMDPs by only considering the sets of time-abstract paths. By allowing schedulers to base their choice on timed paths of a CTMDP, one obtains new, more expressive scheduler classes:

- timed, history-dependent, deterministic (THD)
  $D : \mathit{Pathf} \to A \quad \text{s.t.} \quad D(s) \in A(s)$ and D is measurable

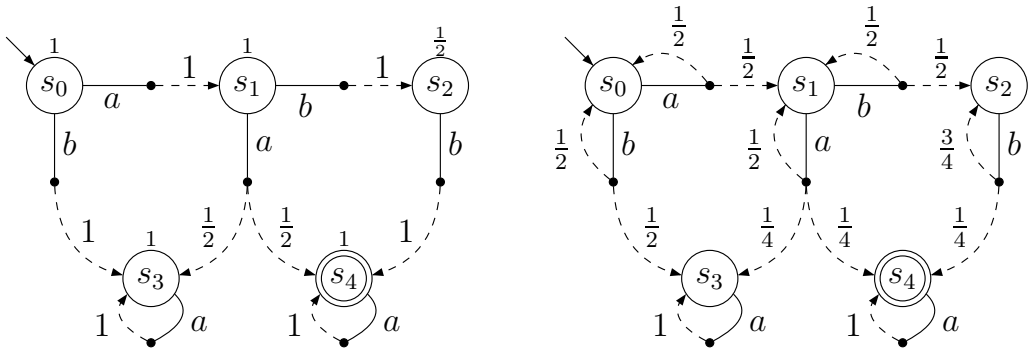


Figure 2.3: A *locally uniform* continuous-time Markov decision process (left) and a version, *globally uniformized with* $\lambda = 2$ (right).

- timed, history-dependent, randomized (THR)
  $D : \mathit{Pathf} \to \mathit{distr}(A)$ s.t. $\mathit{supp}(D(s)) \subseteq A(s)$ and D is measurable

Example 2.4.4. Consider the uniform CTMDP $\mathcal{M}_2$ from Example 2.4.3. The probabilities to start in s_0 and reach s_4 within a certain amount of time t are depicted in Figure 2.4 for different schedulers and scheduler classes. First, recall that in the worst case, a scheduler always chooses b in state s_0 and thus, state s_4 will never be reached; therefore, the infimum over all schedulers of any class is 0 for all $t \in \mathbb{R}_{\geq 0}$. Further, note that w.r.t. minimal and maximal time-bounded reachability probabilities, deterministic HD-schedulers yield the same values as randomized ones [BHKH05].

For (measurable) THD-schedulers, we obtain the greatest maximal reachability probabilities (cf. Figure 2.4). E.g. for $t = 3$, the difference to the next best scheduler class, HD-schedulers, is significant. The supremum for simple (SMD) schedulers is obtained by selecting scheduler D with $D(s_0) = D(s_1) = a$ for a time bound up to approximately $t = 3$ and scheduler D' with $D'(s_0) = a$, $D'(s_1) = b$ otherwise; the resulting (non-differentiable) curve is below the one for HD-schedulers.

Note that supremum and infimum of time-bounded reachability probabilities for simple schedulers are the same in $\mathcal{M}$ and $\mathcal{M}_2$. Together with

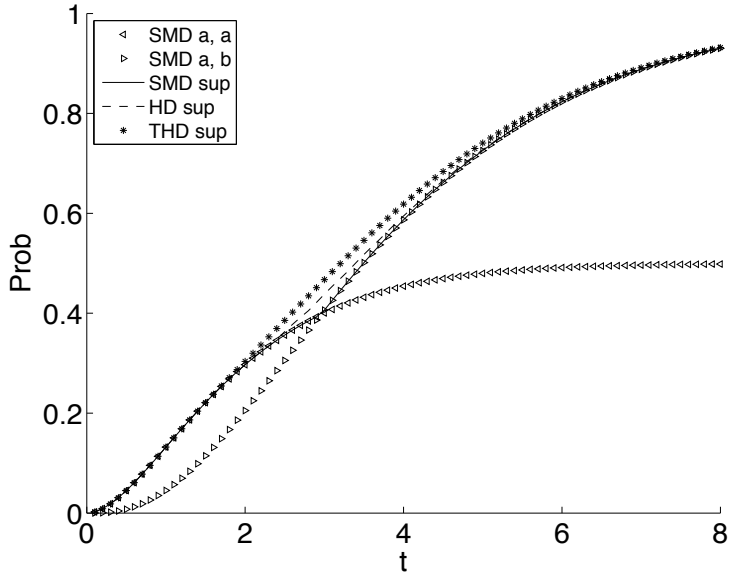


Figure 2.4: Time-bounded reachability probabilities in the CTMDP in Fig. 2.3 (right) for some schedulers and scheduler classes.

our previous observation from Example 2.4.3 (the sets of simple and HD-schedulers are identical for $\mathcal{M}$), this shows that *naive uniformization* for CTMDPs does not preserve reachability measures for HD-schedulers.

For an in-depth discussion of scheduler classes for CTMDPs and (local) uniformization of CTMDPs, see [NSK09].

2.5 Probabilistic branching-time logics

Probabilistic CTL. Probabilistic computation tree logic (PCTL, [HJ94]) extends computation tree logic (CTL, [CES83]) by replacing existential and universal path quantification by a probability operator, denoted $\mathcal{P}$. The syntax of PCTL is given by the following grammar:

$$\varphi ::= \mathit{true} \mid a \mid \varphi \wedge \varphi \mid \neg\varphi \mid \mathcal{P}_{\bowtie p}(\mathcal{X}\varphi) \mid \mathcal{P}_{\bowtie p}(\varphi \mathcal{U}^{\leq i} \varphi)$$

where $\bowtie \in \{<, \leq, \geq, >\}$, $p \in [0,1]$, $i \in \mathbb{N} \cup \{\infty\}$, and $a \in AP$. The until operator is decorated with a so-called *step bound* i indicating after how many

$$
\begin{aligned}
&[\![\mathit{true}]\!](s) = \top \\
&[\![a]\!](s) \quad = L(s,a) \\
&[\![\neg\varphi]\!](s) \;= ([\![\varphi]\!](s))^c \\
&[\![\varphi_1 \wedge \varphi_2]\!](s) = [\![\varphi_1]\!](s) \sqcap [\![\varphi_2]\!](s) \\
&[\![\mathcal{P}_{\bowtie p}(\Phi)]\!](s) = \top, \text{ iff } Pr_s(\{\varrho \in Path_s^{\mathcal{M}} \mid [\![\Phi]\!](\varrho) = \top\}) \bowtie p \\
&[\![\mathcal{X}\varphi]\!](\varrho) \quad = \top, \text{ iff } [\![\varphi]\!](\varrho[1]) = \top \\
&[\![\varphi_1 \mathcal{U}^{\leq i} \varphi_2]\!](\varrho) = \top, \text{ iff } \exists i' \leq i \colon ([\![\varphi_2]\!](\varrho[i']) = \top \\
&\qquad\qquad\qquad\qquad \wedge\, \forall i'' \leq i' \colon [\![\varphi_1]\!](\varrho[i'']) = \top)
\end{aligned}
$$

Table 2.1: Semantics of PCTL

steps an until formula has to be satisfied. Taking $i = \infty$ resembles the usual semantics of an until operator, and we may drop i in this case to simplify notation. For example, formula $\mathcal{P}_{>0.5}(\mathit{true}\,\mathcal{U}^{\leq 12}\mathit{goal})$ asserts that the probability to reach a *goal* state within 12 steps is greater than $\frac{1}{2}$.

The semantics of PCTL is defined with respect to DTMCs, and is given in Table 2.1.

Continuous Stochastic Logic. The Continuous Stochastic Logic (CSL, [ASSB00, BHHK03]) is similar to PCTL but is considered with respect to CTMCs. Syntactically, it differs from PCTL in the until operator which is now equipped with a time bound:

$$\varphi ::= \mathit{true} \mid a \mid \neg\varphi \mid \varphi \wedge \varphi \mid \mathcal{P}_{\bowtie p}(\mathcal{X}^{\leq t}\varphi) \mid \mathcal{P}_{\bowtie p}(\varphi \mathcal{U}^{\leq t} \varphi)$$

where $\bowtie, p$ and a are as before, but $t \in \mathbb{R}_{\geq 0} \cup \{\infty\}$. For example, the property to reach a *down* state in a CTMC within 5.2 time units, via *premium* states, with probability at most 10^{-4} can be formulated by the CSL formula $\mathcal{P}_{\leq 0.0001}(\mathit{premium}\,\mathcal{U}^{\leq 5.2}\mathit{down})$. As we will justify later on, CTMCs are to be

$$
\begin{aligned}
[\![\mathcal{P}_{\trianglelefteq p}(\Phi)]\!](s) &= \top, \text{ iff } Pr_s(\{\sigma \in Path_s^{\mathcal{M}} \mid [\![\Phi]\!](\sigma) = \top\}) \trianglelefteq p \\
[\![\mathcal{X}^{\leq t}\varphi]\!](\sigma) &= \top, \text{ iff } \sigma = s \xrightarrow{t'} s' \ldots \wedge\ t' \leq t \ \wedge\ [\![\varphi]\!](s') = \top \\
[\![\varphi_1 \mathcal{U}^{\leq t} \varphi_2]\!](\sigma) &= \top, \text{ iff } \exists t' \leq t : ([\![\varphi_2]\!](\sigma@t') = \top \\
&\qquad \wedge \forall t'' \leq t' : [\![\varphi_1]\!](\sigma@t'') = \top)
\end{aligned}
$$

Table 2.2: Semantics of CSL

uniformized prior to abstraction. As a CTMC is *weakly* probabilistic bisimilar to its uniformized counterpart [BKHW05], the validity of next-formulas has to be checked on the original CTMC instead; this is due to the introduction of self-loops. We do not consider the next operator in the remainder of the thesis; it has been dealt with in [Hut05]. Further, we omit the steady-state operator [BHHK03]. The parts of the semantics of CSL that differ from PCTL are listed in Table 2.2. Note that the semantics of the $\mathcal{P}$-operator looks the same as for PCTL, but now involves probabilities over paths in CTMCs rather than DTMCs.

Time complexity. PCTL (and CSL) model checking is performed inductively on the structure of the formula φ like for CTL model checking. This yields a time-complexity which is linear in the size of φ. Atomic formulas and Boolean connectives are dealt with in the usual way. Checking time-bounded and (unbounded) until-formulas reduces to computing reachability probabilities. Standard reachability probabilities involve solving a linear equation system which can be done in $\mathcal{O}(|S|^3)$ where $|S|$ is the number of states in the Markov chain [HJ94]. Time-bounded reachability probabilities involve solving a Volterra integral equation system. Alternatively, a reduction to transient analysis can be done yielding a time complexity in $\mathcal{O}(|S|^2 \cdot \lambda \cdot t)$ where λ is the uniformization rate and t is the time bound [BHHK03].

Logics for Markov decision processes. To extend PCTL and CSL to Markov decision processes (cf. [BdA95]), the semantics of the probabilistic quantifier $\mathcal{P}$ can be adjusted such that a formula is fulfilled or violated iff the probability measures for *all* schedulers under consideration fulfill or violate the given constraint. If neither is the case, i.e. if there exists a scheduler D for which the formula in question, say φ, is satisfied in state s and for another scheduler D' it is violated, the semantics should be *don't know*, formally $[\![\varphi]\!](s) = ?$.

2.6 Bisimulation minimization

Bisimulation minimization is an important reduction method for all kinds of models from labeled transition systems to CTMDPs [Buc94, NK07]. We will now define bisimulation in terms of a relation over states and then formulate bisimulation minimization as an abstraction method.

Definition 2.6.1 (Bisimulation relation)**.** Let $\mathcal{M} = (S, \mathbf{P}, L, \mu_0)$ be a DTMC and $R : S \times S$ be an equivalence relation such that sRu implies

- $L(s, ap) = L(u, ap)$ for all $ap \in AP$, and
- $\mathbf{P}(s, V) = \mathbf{P}(u, V)$ for all $V \in S/R$,

where S/R denotes the quotient space under R. Then, R is a *bisimulation relation* and states $s, u \in S$ with sRu are *bisimilar*. Note that there always exists a unique coarsest bisimulation relation.

The main idea is to aggregate states that cannot be distinguished one from another in terms of the labeling, the nondeterministic choices available and the successor states. From a game theoretic view, states are bisimilar, if they can mimic each others behavior in the current as well as in every possible step that may follow, that is, in the presence of nondeterminism, if there are schedulers that can mimic one another.

The aggregation of states can be described by an *abstraction* operator $\alpha : S \to S'$ mapping from the state space S of the given *concrete* model $\mathcal{M}$ to S' in the reduced *abstract* model $\mathcal{M}'$. In order to obtain a bisimulation minimization, we require that S' is the bisimulation quotient and α maps concrete states to their representative in S'. The *concretization* function $\gamma : S' \to 2^S$ returns the sets of concrete states that are represented by abstract ones. We can now understand bisimulation minimization as a form of abstraction.

Definition 2.6.2 (Bisimulation minimization)**.** Let $R : S \times S$ be the coarsest bisimulation equivalence on DTMC $\mathcal{M} = (S, \mathbf{P}, L, \mu_0)$. The minimized DTMC w.r.t. R is $\mathcal{M}' = (S/R, \mathbf{P}', L', \mu_0')$, induced by $\alpha : S \to S/R$ with $\alpha(s) = \alpha(u)$ iff sRu and

- $L'(s', ap) = L(s, ap)$ for all $ap \in AP$, $s \in \gamma(s')$,
- $\mathbf{P}'(s', v') = \mathbf{P}(s, \gamma(v'))$ for all $v' \in S'$, $s \in \gamma(s')$,
- $\mu_0'(v') = \mu_0(\gamma(v'))$ for all $v' \in S'$.

The above definitions can be lifted directly to CTMCs $\mathcal{M}$ and $\mathcal{M}'$ by requiring *bisimilar* states to have the same exit rate; for uniform CTMCs, this is trivially fulfilled. In the minimized CTMC, all abstract states have to have the same exit rates as the concrete states they represent.

Lifting bisimulation minimization to Markov decision processes requires that for every action a and every state s, if there exists an a-successor distribution in s, then there has to be a *bisimilar* a-successor distribution in $\alpha(s)$. For general CTMDPs, bisimulation has been thoroughly investigated in [NK07].

Example 2.6.1. Consider the game of luck from Example 2.3.1. First, note that any α with $\alpha(s_{41}) = \alpha(s_{42})$ does not induce a bisimilar DTMC as $L(s_{41}, \mathit{goal}) = \bot \neq L(s_{42}, \mathit{goal})$. By induction, any α with $\alpha(s_i) = \alpha(s_{i+1})$

for $i < 42$ does not induce a bisimilar DTMC either. Intuitively, all states s_i with $i < 42$ can be distinguished via the probability for reaching the *goal* state s_{42}. From all other states, however, only states are reachable that have no label. Thus, we may choose $\alpha : S \to S'$ such that $\alpha(s_i) = s'_i$ if $i \leq 42$ and $\alpha(s_i) = s'_{>42}$ otherwise. For all $s, u \in S$ with $s \neq u$ and $\alpha(s) = \alpha(u) = s'_{42}$ it holds $L(s, ap) = \bot = L(u, ap)$ for all $ap \in AP$ and $\sum_{v \in \gamma(s'_{>42})} \mathbf{P}(s, v) = 1 = \sum_{v \in \gamma(s'_{>42})} \mathbf{P}(u, v)$. For states $s'_i \in S'$ with $i \leq 42$, the concretizations $\gamma(s'_i)$ yield singleton sets $\{s_i\}$ such that all the conditions in Definition 2.6.1 are fulfilled trivially. Thus, the induced *abstract* DTMC is indeed bisimilar to the given one.

Note that bisimulation minimization does not yield a finite state DTMC for this game of luck. While, for example, states $(s_{41}, +)$ and $(s_{43}, -)$ are bisimilar, the reduced MDP is still of infinite size as any states $(s_i, \circ)$ and $(s_{i+1}, \circ)$ are not bisimilar for $i \in \mathbb{Z}$ and $\circ \in \{+, -\}$. However, if there is some $n \in \mathbb{N}$ with $\mu_0(s_i) = 0$ for all $i < n$, states $s_{n-1}, s_{n-2}, \ldots$ can be disregarded as they are not reachable, resulting in a finite state space. When considering the interactive game of luck from Example 2.3.2, all states are reachable.

3

Abstraction for infinite-state Markov chains

Abstraction is the art of omitting details while still preserving essential information. What is essential lies in the eye of the beholder, yet, in a mathematical context the policy on information preservation should be a conservative one: what you can tell about the abstraction should apply to the original as well.

In this chapter, we introduce two basic abstraction techniques for discrete-time models where sets of concrete states are partitioned via abstraction functions. Unlike for bisimulation minimization, it is not required that the states to be merged are labeled the same; this is one way how details, even essential ones, can be omitted. The first technique, using MDPs as abstract models, has been originally presented in [DJJL01]; the second technique describes abstract behavior by intervals of probabilities [Hut05, FLW06, KKLW07b]. In the latter, it is necessary to check intervals for consistency; we show that this can be done efficiently using a normalization function. Exemplarily, we will lift the interval abstraction to the continuous-time setting [KKLW07a] and formally compare both techniques [KRHK09] by the same formalism that relates concrete models and their abstractions: stepwise probabilistic simulation.

3.1 MDP abstraction

In the following, we show how to abstract infinite-state DTMCs and MDPs to – preferably finite state – MDPs, thus, this method will be referred to as *MDP abstraction*. The basic idea is to associate *sets of distributions* that are present in the concrete model to the corresponding states in the abstract model: e.g., for abstraction function $\alpha : S \to S'$, distribution $\mathbf{P}(s, \cdot)$ in each state $s \in S$ in a given DTMC, there should be a corresponding distribution in the abstract state $s' = \alpha(s)$. This ensures that the behavior of the concrete model is actually reflected in the abstract model.

Definition 3.1.1 (MDP abstraction)**.** Let $\mathcal{M} = (S, A, \mathbf{P}, L, \mu_0)$ be a MDP and $\alpha : S \to S'$ an abstraction function such that $\{\mu \in \mathbf{T}(s) \mid s \in \gamma(s')\}$ is finite for all $s' \in S'$. The MDP abstraction induced by α (denoted $\alpha_{MDP}(\mathcal{M})$) is given by $(S', A', \mathbf{P}', L', \mu_0')$ where

- $A' = \{a_{\mu'} \mid \exists s \in S, \mu \in \mathbf{T}(s) : \forall u' \in S' : \mu'(u') = \mu(\gamma(u'))\}$ is *finite*,
- for all $s', u' \in S'$,
$$\mathbf{P}'(s', a_{\mu'}, u') = \begin{cases} \mu'(u') = \mu(\gamma(u')) & \text{if } \mu \in \mathbf{T}(s) \text{ for some } s \in \gamma(s') \\ 0 & \text{otherwise} \end{cases}$$
- for all $s' \in S'$, $ap \in AP$
$$L'(s', ap) = \begin{cases} \top & \text{if } L(s, ap) = \top \text{ for all } s \in \gamma(s') \\ \bot & \text{if } L(s, ap) = \bot \text{ for all } s \in \gamma(s') \\ ? & \text{otherwise,} \end{cases}$$
- for all $u' \in S'$ it holds $\mu_0'(u') = \mu_0(\gamma(u'))$.

Example 3.1.1. Consider the DTMC $\mathcal{M} = (S, \mathbf{P}, L, \mu_0)$ depicted in Figure 3.1 where $L(s, goal) = \top$ iff $s \in \{u_1, u_2, u_3, v_2\}$ and let $supp(\mu_0) \subseteq \{s_1, \ldots, s_5\}$. Further, let abstraction function $\alpha : S \to \{s', u', v'\}$ be given by $\alpha(s_i) = s'$ for all $i \in \{1, \ldots 5\}$ and $\alpha(u_i) = u'$, $\alpha(v_i) = v'$ for all

$i \in \{1, 2, 3\}$. As all u-states as well as all v-states are bisimilar according to Definition 2.6.1, no nondeterminism arises from merging those sets of states. Consequently, as shown in Figure 3.2, only one choice is available in states u' and v' in the *MDP abstraction*. In contrast, for most of the s-states, the probabilities to take a transition to one of the states in $\gamma(s')$, $\gamma(u')$, and $\gamma(v')$ respectively differ. Hence, these states are not pairwise bisimilar and in order to capture all the possible behavior in the concrete model, we have to make each distribution that is present in the concrete states in $\gamma(s')$ available in s' by a nondeterministic choice. In Figure 3.2, each nondeterministic choice in s' is labeled with the corresponding states in the concrete model; the actions $a_{\mu'}$ are omitted in the figure. Note that s_1 and s_2 are both corresponding to the same choice as $\mathbf{P}(s_1, \gamma(x)) = \mathbf{P}(s_2, \gamma(x))$ for all $x \in \{s', u', v'\}$.

Formally, $\alpha_{MDP}(\mathcal{M}) = (\{s', u', v'\}, A', \mathbf{P}', L', \mu'_0)$ where

- $A' = \{a_{\mu'_1}, a_{\mu'_2}, a_{\mu'_3}, a_{\mu'_4}, a_{\mu'_5}, a_{\mu'_6}\}$ for
  - $\mu'_1 = \{u' \mapsto 1\}$
  - $\mu'_2 = \{v' \mapsto 1\}$
  - $\mu'_3 = \{u' \mapsto \frac{4}{5}, v' \mapsto \frac{1}{5}\}$
  - $\mu'_4 = \{s' \mapsto \frac{3}{5}, u' \mapsto \frac{1}{5}, v' \mapsto \frac{1}{5}\}$
  - $\mu'_5 = \{s' \mapsto \frac{2}{5}, u' \mapsto \frac{1}{2}, v' \mapsto \frac{1}{10}\}$
  - $\mu'_6 = \{u' \mapsto \frac{2}{5}, v' \mapsto \frac{3}{5}\}$
- $\mathbf{P}'(u', a_{\mu'_1}, \cdot) = \mu'_1$, $\mathbf{P}'(v', a_{\mu'_2}, \cdot) = \mu'_2$,
  $\mathbf{P}'(s', a_{\mu'_i}, \cdot) = \mu'_i$ for $i \in \{3, \ldots, 6\}$,
  and $\mathbf{P}'(x, a, y) = 0$ otherwise,
- $L'(s', \mathit{goal}) = \bot$, $L'(u', \mathit{goal}) = \top$ and $L'(v', \mathit{goal}) = ?$,
- $\mu'_0 = \{s' \mapsto 1\}$.

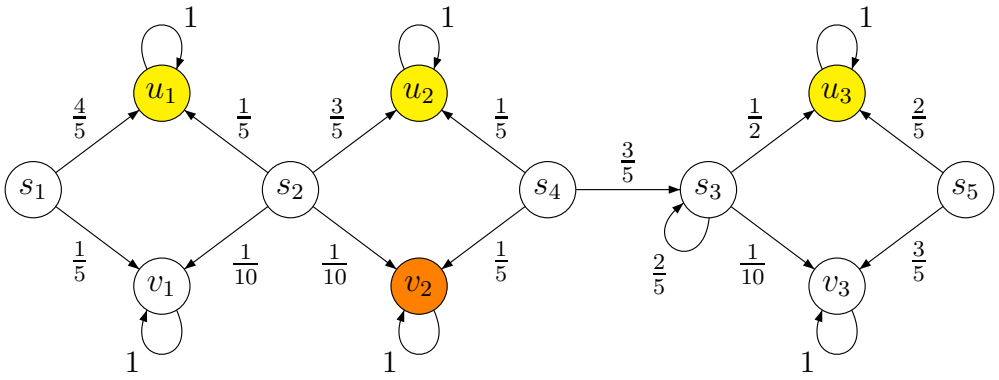


Figure 3.1: An example DTMC.

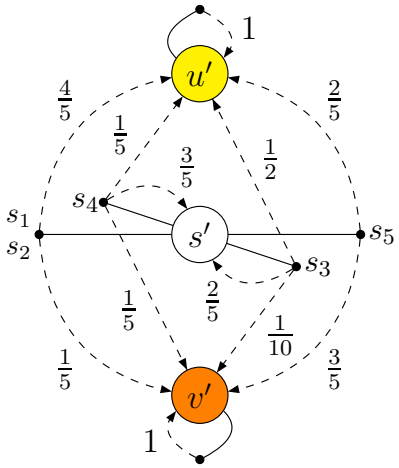


Figure 3.2: An MDP abstraction.

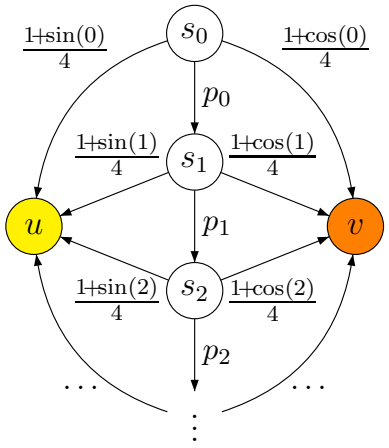


Figure 3.3: An infinite-state DTMC.

It is important to understand that not any infinite-state Markov chain yields a finite MDP abstraction. The following example reveals this limitation.

Example 3.1.2. Consider the Markov chain $\mathcal{M}$ in Figure 3.3 with state space $\{u, v\} \cup \{s_i \mid i \in \mathbb{N}\}$ and $p_i = \frac{2-\sin(i)-\cos(i)}{4}$ for all $i \in \mathbb{N}$. The set of probability distributions that are associated to the set of all s_i-states is countably infinite as sine and cosine are continuous and periodic with $2 \cdot \pi$. Hence, there is no abstraction function α for which $\alpha_{MDP}(\mathcal{M})$ has finite state space *and* finite action set.

3.2 Interval abstraction

In this section an alternative abstraction technique is introduced. We present the concept of *abstract Markov chains* (AMCs), in fact Markov chains whose edges are equipped with *intervals* of probabilities. We show how AMCs can be considered as abstractions of Markov chains; first, we consider AMCs for discrete time, then we will discuss how to lift abstraction to the continuous-time setting.

The discrete-time setting. The main idea behind *interval abstraction* is to keep track of the minimal and maximal probabilities for taking exactly one transition leading from a partition to a successor partition (possibly the same one). In the abstract model, these minimal and maximal probabilities form lower and upper bounds of transition probability intervals, i.e. for a given Markov chain with transition probabilities $\mathbf{P}$ and abstraction function α, lower and upper probability bounds for an abstract transition from, say, s' to u', can be computed by:

$$\mathbf{P}^l(s', u') = \inf_{s \in \gamma(s')} \mathbf{P}(s, \gamma(u'))$$
$$\mathbf{P}^u(s', u') = \sup_{s \in \gamma(s')} \mathbf{P}(s, \gamma(u'))$$

This idea does not only work for Markov chains, but also for abstract Markov chains and Markov decision processes. For simplicity, we do not elaborate on the latter.

Definition 3.2.1 (Abstract DTMC)**.** An *abstract DTMC* (ADTMC for short) is a tuple $(S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0)$ with S, L and μ_0 as for MDPs, and

- $\mathbf{P}^l, \mathbf{P}^u : S \times S \to [0, 1]$, transition probability bound matrices with $\mathbf{P}^l(s, S) \leq 1 \leq \mathbf{P}^u(s, S) < \infty$ for all $s \in S$ and $\mathbf{P}^l(s, u) \leq 1 \leq \mathbf{P}^u(s, u)$ for all $s, u \in S$.

An example ADTMC is depicted in Figure 3.4. The intervals $[\mathbf{P}^l(s, u), \mathbf{P}^u(s, u)]$ specify the ranges of possible transition probabilities between states s and u. In fact, for any state s there is a nondeterministic choice between the distributions $\mu \in \mathit{distr}(S)$ with $\mu(u) \in [\mathbf{P}^l(s, u), \mathbf{P}^u(s, u)]$ for all $u \in S$. Let $\mathbf{T}(s)$ be the (possibly infinite) set of all such distributions in state s. If set $\mathbf{T}(s)$ is a singleton and $L(s, a) \neq ?$ for all $a \in AP$, $s \in S$, then $\mathcal{M}$ can be considered as a DTMC.

ADTMCs are basically labeled *interval* DTMCs [KU02, Sku06], except that we allow for infinite state spaces, and do not have intervals on initial distributions but point intervals. They are also closely related to MDPs

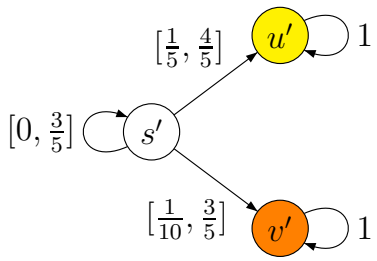


Figure 3.4: An abstract Markov chain (AMC).

as the basic ingredients – probabilism and nondeterminism – are integrated similarly. Differences are on the one hand that ADTMCs allow for an infinite number of distributions in each state and on the other hand that the set $\mathbf{T}(s)$ is always *convex* (as we will establish later, cf. Lemma 4.1.1) whereas in MDPs $\mathbf{T}(s)$ can be an arbitrary but finite set of distributions.

A path in an ADTMC is a sequence $\rho = s_0 \to s_1 \to \ldots$; we adopt the notions for paths (and path fragments) in DTMCs, e.g., $\rho[i]$, $Path_s^{\mathcal{M}}$, $Pathf_s^{\mathcal{M}}$, and so forth. As for MDPs, probability measures for ADTMCs can only be specified with respect to a given scheduler that resolves the nondeterminism introduced by the intervals. As the definition of paths and path fragments are the same for MDPs and AMCs, schedulers are also classified the same, that is, we distinguish stationary, history-dependent and (in the continuous-time setting) timed schedulers; however, schedulers on AMCs do not choose distributions indirectly via actions out of $A(s)$, but directly from the set of all distributions $\mathbf{T}(s)$ respecting the intervals.

As for MDPs, each scheduler induces its own stochastic process, e.g. an HD scheduler $D : Pathf^{\mathcal{M}} \to \bigcup_{s \in S} \mathbf{T}(s)$ on ADTMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0)$ induces the DTMC $\mathcal{M}_D = (S_D, \mathbf{P}_D, L_D, \mu_0)$ where $S_D = Pathf^{\mathcal{M}}$ is the state space, $L_D(\varrho) = L(\varrho\downarrow)$ for all $\varrho \in Pathf^{\mathcal{M}}$ and

$$\mathbf{P}_D(\varrho, \varrho') = \begin{cases} D(\varrho)(s') & \text{if } \varrho' = \varrho \to s' \\ 0 & \text{otherwise.} \end{cases}$$

The abstraction of an ADTMC is now formally defined as follows.

Definition 3.2.2 (Interval abstraction)**.** The *interval abstraction* of ADTMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0)$ with respect to abstraction function $\alpha : S \to S'$, denoted $\alpha(\mathcal{M})$, is given by $(S', \tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u, \tilde{L}, \tilde{\mu}_0)$ where

- for all $s', u' \in S'$,

  $\tilde{\mathbf{P}}^l(s', u') = \inf_{s \in \gamma(s')} \mathbf{P}^l(s, \gamma(u'))$,
  $\tilde{\mathbf{P}}^u(s', u') = \min\{1, \sup_{s \in \gamma(s')} \mathbf{P}^u(s, \gamma(u'))\}$,

- for all $s' \in S'$, $ap \in AP$

  $$\tilde{L}(s', ap) = \begin{cases} \top & \text{if } L(s, ap) = \top \text{ for all } s \in \gamma(s') \\ \bot & \text{if } L(s, ap) = \bot \text{ for all } s \in \gamma(s') \\ ? & \text{otherwise,} \end{cases}$$

- for all $u' \in S'$ it holds $\tilde{\mu}_0(u') = \mu_0(\gamma(u'))$.

The definition of $\tilde{\mathbf{P}}^l$ is self-explanatory. As the supremum of $\mathbf{P}^u(s, \gamma(u'))$ may exceed one, $\tilde{\mathbf{P}}^u(s', u')$ is defined as the minimum of one and the supremum of all transition probabilities from concrete states in s' to u'. Note that, the definitions of the labeling function and the initial distribution are the same as for MDP abstraction.

Example 3.2.1. Consider the DTMC in Figure 3.1 that has been subject to MDP abstraction in Example 3.1.1. The corresponding *interval abstraction* of this DTMC yields the ADTMC in Figure 3.4. The state space and the labeling is exactly as for MDP abstraction, however, the transition structure is quite different. To compute, say, the probability bounds for taking a transition from abstract state s' to abstract state u', the minimal and maximal probabilities to take corresponding transitions in the concrete model have to be determined. The minimal probability of such a transition in the concrete model is $\frac{1}{5}$ for leaving s_4 towards u_2. The maximum is $\frac{4}{5}$ as for s_2 there are two ways to reach a state in u, the overall probability to do so is just the sum

of the probabilities ($\frac{1}{5}+\frac{3}{5}$), and there is no other state in s for which there is a larger probability to reach states in u. This yields the transition probability interval $[\frac{1}{5}, \frac{4}{5}]$ for the abstract transition from s to u. The intervals for the other abstract transitions are computed similarly.

The following lemma states that the class of ADTMCs is closed under the above presented notion of abstraction, i.e., abstracting an ADTMC by $\alpha : S \to S'$ yields an ADTMC which is finite iff S' is finite.

Lemma 3.2.1. *For any ADTMC $\mathcal{M}$ with state space S and abstraction function $\alpha : S \to S'$, the interval abstraction $\alpha(\mathcal{M})$ is an ADTMC.*

Proof. Let ADTMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0)$, abstraction function $\alpha : S \to S'$ and $\alpha(\mathcal{M}) = (S', \tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u, \tilde{L}, \tilde{\mu}_0)$. As there is nothing to show for the labeling function, and it is easy to see that $\tilde{\mu}_0 \in \mathit{distr}(S')$, it remains to prove that $\tilde{\mathbf{P}}^l$ and $\tilde{\mathbf{P}}^u$ map to $[0,1]$ and $\tilde{\mathbf{P}}^l(s', S') \leq 1 \leq \tilde{\mathbf{P}}^u(s', S') < \infty$ for all $s' \in S'$. The former obligation follows by easy derivation. It also follows directly from Definition 3.2.2 that $\tilde{\mathbf{P}}^u(s', S') < \infty$. For all $s' \in S'$ we derive:

$$\begin{aligned}
\tilde{\mathbf{P}}^l(s', S') &= \textstyle\sum_{u' \in S'} \inf_{s \in \gamma(s')} \mathbf{P}^l(s, \gamma(u')) \\
&\leq \textstyle\sum_{u' \in S'} \mathbf{P}^l(\hat{s}, \gamma(u')) \qquad\qquad \text{for all } \hat{s} \in \gamma(s') \\
&= \mathbf{P}^l(\hat{s}, S) \leq 1
\end{aligned}$$

and

$$\begin{aligned}
\tilde{\mathbf{P}}^u(s', S') &= \textstyle\sum_{u' \in S'} \min\{1, \sup_{s \in \gamma(s')} \mathbf{P}^u(s, \gamma(u'))\} \\
&\geq \textstyle\sum_{u' \in S'} \min\{1, \mathbf{P}^u(\hat{s}, \gamma(u'))\} \qquad \text{for all } \hat{s} \in \gamma(s') \\
&\geq \min\{1, \textstyle\sum_{u' \in S'} \mathbf{P}^u(\hat{s}, \gamma(u'))\} \\
&= \min\{1, \underbrace{\mathbf{P}^u(\hat{s}, S)}_{\geq 1}\} = 1.
\end{aligned}$$

□

The continuous-time setting. Let us now consider CTMCs. In order to apply a similar abstraction technique to CTMCs, it is natural to group

states and consider again intervals of transition probabilities. The main technical complication, however, is that besides the transition probabilities, we have to determine the residence time in an abstract state that results from concrete states with possibly distinct residence time distributions. Let $\mathcal{M} = (S, \mathbf{P}, L, \mu_0, \lambda)$ be a CTMC and $\alpha : S \to S'$ an abstraction function. The probability to move from a state s to $U \subseteq S$ within t time units is given by $\mathbf{P}(s, U, t) = \mathbf{P}(s, U) \cdot (1 - e^{-\lambda(s) \cdot t})$. Taking minimal and maximal probabilities as under- and over-approximation — as in the discrete case — respectively, suggests to define

$$\begin{aligned} \mathbf{P}^l(s', u', t) &= \inf_{s \in \gamma(s')} \mathbf{P}(s, \gamma(u'), t), \\ \mathbf{P}^u(s', u', t) &= \sup_{s \in \gamma(s')} \mathbf{P}(s, \gamma(u'), t). \end{aligned} \tag{3.1}$$

Observe that the functions $\mathbf{P}^l(s', u', t)$ and $\mathbf{P}^u(s', u', t)$ (considered as functions ranging over t) are in general not of the form $p \cdot (1 - e^{-\lambda \cdot t})$ for fixed $p \in [0, 1]$ and $\lambda \in \mathbb{R}_{>0}$. This is illustrated by the following example.

Example 3.2.2. Consider the non-uniform CTMC $\mathcal{M}$ in Figure 2.1 and $\alpha : S \to S'$ with $\alpha(s) = s'$, $\alpha(v) = v'$ and $\alpha(u_i) = u'$ for all $i \in \{1, 2, 3\}$. The functions $f = \mathbf{P}(u_1, v, \cdot)$, $f' = \mathbf{P}(u_2, v, \cdot)$, and $f'' = \mathbf{P}(u_3, v, \cdot)$ are plotted in Figure 3.5. It is easy to see that $\mathbf{P}^l(u', v', t)$ and $\mathbf{P}^u(u', v', t)$, if defined as in Equation 3.1, are not differentiable and hence, not of the form $p \cdot (1 - e^{-\lambda \cdot t})$. In general, these functions get more complex as the number of transitions between states in u' and v' increases.

One could combine the infimum (supremum) of an abstract state's exit rates with the infimum (supremum) of the time-independent transition probabilities to define an appropriate under- and over-approximation. This yields, however, a rather coarse abstraction as indicated by the plot of the functions g and g' in Figure 3.5. We therefore propose to abstract a CTMC after uniformization, cf. Definition 2.4.2. The advantage of abstracting uniform CTMCs with exit rate λ, say, is that for any $U, V \subseteq S$ and $t \in \mathbb{R}_{\geq 0}$, the

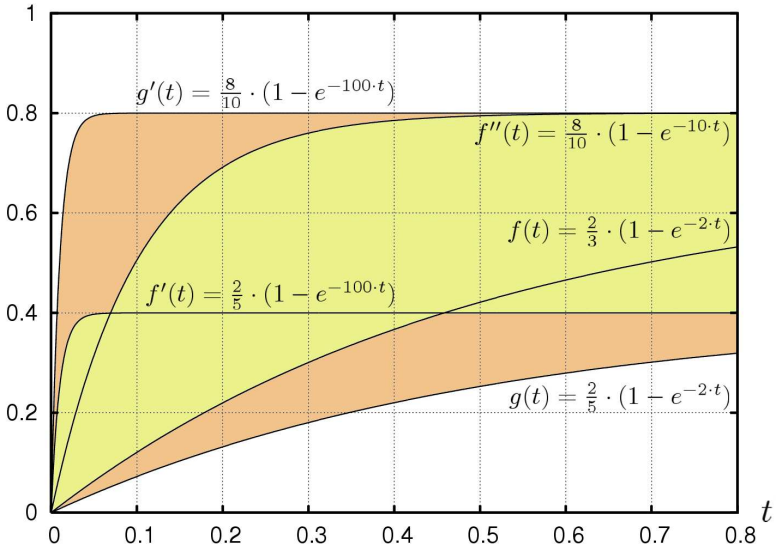


Figure 3.5: Abstraction for non-uniform CTMCs.

infimum/supremum of the time-bounded transition probabilities can be expressed via the infimum/supremum over time-abstract transition probabilities as follows:

$$\begin{aligned} \inf_{s \in U} \mathbf{P}(s, V, t) &= (1 - e^{-\lambda \cdot t}) \cdot \inf_{s \in U} \mathbf{P}(s, V) \\ &\leq (1 - e^{-\lambda \cdot t}) \cdot \sup_{s \in U} \mathbf{P}(s, V) \\ &= \sup_{s \in U} \mathbf{P}(s, V, t). \end{aligned}$$

By applying abstraction to uniform CTMCs we obtain uniform abstract CTMCs. As we will show later, the uniformity also enables us to adapt existing algorithms for time-bounded reachability probabilities for verifying abstract CTMCs. Recall that CTMC $\mathcal{M}$ and *unif*($\mathcal{M}$) are weakly bisimilar and thus the validity of any (next-free) CSL formula in $\mathcal{M}$ is preserved by *unif*($\mathcal{M}$), cf. [BKHW05].

Definition 3.2.3 (Abstract CTMC)**.** An *abstract CTMC (ACTMC)* is a tuple $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0, \lambda)$ with S, $\mathbf{P}^l$, $\mathbf{P}^u$, L, and μ_0 as before and $\lambda \in \mathbb{R}_{>0}$, the uniform exit rate for all states.

An ACTMC with $\mathbf{P}^l = \mathbf{P}^u$ and $L(s,a) \neq ?$ for all $s \in S$ and $ap \in AP$ can be considered as the uniform CTMC $(S, \mathbf{P}^l, L, \mu_0, \lambda)$. As for ADTMCs, let $\mathbf{T}(s)$ denote the set of distributions in s. A path in an ACTMC is a sequence $\sigma = s_0 \xrightarrow{t_0} s_1 \xrightarrow{t_1} \ldots$ if $t_i \in \mathbb{R}_{>0}$, for all $i \in \mathbb{N}$. In the sequel, we adopt the notations for paths (and path fragments) in CTMCs for ACTMCs.

Definition 3.2.4 (Continuous-time interval abstraction)**.** The *interval abstraction* of ACTMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0, \lambda)$ induced by abstraction function $\alpha : S \to S'$, denoted $\alpha(\mathcal{M})$, is the ACTMC $(S', \tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u, \tilde{L}, \tilde{\mu}_0, \lambda)$ with $\tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u$, $\tilde{L}$, and $\tilde{\mu}_0$ as in Definition 3.2.2.

It follows directly from Lemma 3.2.1 that $\alpha(\mathcal{M})$ is indeed an ACTMC. The same way as the time-abstract behavior of CTMCs is represented by DTMCs, the time-abstract behavior of an ACTMC can be represented by an ADTMC. The *embedded ADTMC* of ACTMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0, \lambda)$ is given by $emb(\mathcal{M}) = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0)$.

In the following, we use the term *abstract Markov chain (AMC)* to refer to both ADTMCs and ACTMCs. Thus, AMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0)$ represents the ADTMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0)$ and the ACTMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0, \lambda)$ where the uniform exit rate λ is *not* always given explicitly.

3.3 Normalization

When transitions are labeled with intervals of probabilities, it is possible that not every combination of probabilities of transitions emanating from a state yields a probability distribution. Consider, e.g., the AMC in Figure 3.6 (left). It suggests that it is possible to move from s_0 to s_1 with probability one. Then the probabilities to move to s_2 and s_3 must be zero, which, however, is not part of the respective intervals. AMCs that do not suffer from this problem are called delimited (in the context of interval Markov chains such a model is said to have the F-property [Wei01, Sku06]).

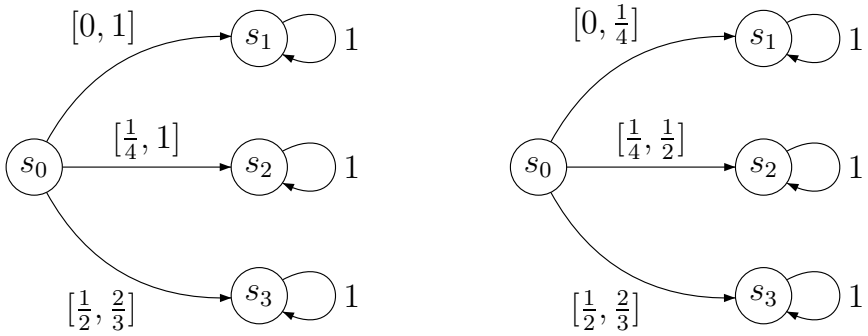


Figure 3.6: Transforming an AMC (left) into a delimited one (right)

Definition 3.3.1 (Delimited AMC)**.** An AMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0)$ is *delimited* iff for any $s, u \in S$ and $p \in [\mathbf{P}^l(s,u), \mathbf{P}^u(s,u)]$, there exists $\mu \in \mathbf{T}(s)$ with $\mu(u) = p$.

Abstracting a MC yields a delimited AMC, however, abstracting a delimited AMC is not guaranteed to yield a delimited AMC. The following procedure, called *normalization*, aims to transform a given AMC into a delimited one. The main principle is to remove values $p \in [\mathbf{P}^l(s,u), \mathbf{P}^u(s,u)]$ for states s, u for which there is no $\mu \in \mathbf{T}(s)$ with $\mu(u) = p$.

Definition 3.3.2 (Normalization)**.** For AMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0)$, the *normalization* of $\mathcal{M}$, denoted $\eta(\mathcal{M})$, is the AMC $(S, \tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u, L, \mu_0)$ where for all $s, u \in S$:

$$\tilde{\mathbf{P}}^l(s,u) = \max\{\mathbf{P}^l(s,u), 1 - \mathbf{P}^u(s, S \setminus \{u\})\}, \quad \text{and}$$
$$\tilde{\mathbf{P}}^u(s,u) = \min\{\mathbf{P}^u(s,u), 1 - \mathbf{P}^l(s, S \setminus \{u\})\}.$$

For the sake of brevity, we sometimes write $\eta(\mathbf{P}^l, \mathbf{P}^u) = (\tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u)$ instead of $\eta(\mathcal{M}) = (S, \tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u, L, \mu_0)$.

Example 3.3.1. Consider the AMC in Figure 3.6 (left). Normalization yields new upper bounds for the transitions from s_0 to s_1 and s_0 to s_2 as shown in Figure 3.6 (right):

$$\begin{aligned}\tilde{\mathbf{P}}^u(s_0,s_1) &= \min\{\mathbf{P}^u(s_0,s_1), 1-\mathbf{P}^l(s_0,s_2)-\mathbf{P}^l(s_0,s_3)\}\\ &= \min\{1, 1-\tfrac{1}{4}-\tfrac{1}{2}\} = \tfrac{1}{4} \quad \text{and}\\ \tilde{\mathbf{P}}^u(s_0,s_2) &= \min\{\mathbf{P}^u(s_0,s_2), 1-\mathbf{P}^l(s_0,s_1)-\mathbf{P}^l(s_0,s_3)\}\\ &= \min\{1, 1-0-\tfrac{1}{2}\} = \tfrac{1}{2}\end{aligned}$$

The upper bound from s_0 to s_3 remains unchanged as $\{s_2 \mapsto \frac{1}{3}, s_3 \mapsto \frac{2}{3}\} \in \mathbf{T}(s_0)$. Furthermore, no lower bound is adapted, as the adjustments to upper bounds for transitions from s_0 to s_1 and s_0 to s_2 ensure that for all $s, u \in S$, there exists $\mu \in \mathbf{T}(s)$ with $\mu(u) = \mathbf{P}^l(s,u)$:

$$\tilde{\mathbf{P}}^u(s_0,s_1) + \mathbf{P}^l(s_0,s_2) + \mathbf{P}^l(s_0,s_3) + \overbrace{\mathbf{P}^l(s_0,s_0)}^{=0} = 1 \text{ and}$$

$$\tilde{\mathbf{P}}^u(s_0,s_2) + \mathbf{P}^l(s_0,s_1) + \mathbf{P}^l(s_0,s_3) + \mathbf{P}^l(s_0,s_0) = 1.$$

The need to amend the probability bound $\mathbf{P}^l(s,u)$ (or $\mathbf{P}^u(s,u)$) only depends on the set of bounds $\mathbf{P}^u(s,v)$ ($\mathbf{P}^l(s,v)$, respectively) for $v \neq u$, cf. Definition 3.3.2. The following lemma states that amending $\mathbf{P}^l(s,u)$ implies that bounds $\mathbf{P}^u(s,v)$ are unchanged for all $v \neq u$. Similarly, a change of $\mathbf{P}^u(s,u)$ implies that bounds $\mathbf{P}^l(s,v)$ are unchanged for all $v \neq u$.

Lemma 3.3.1. *Let AMC* $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0)$ *and* $\eta(\mathbf{P}^l, \mathbf{P}^u) = (\tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u)$. *Then for any* $s, u \in S$*:*

1. $\tilde{\mathbf{P}}^l(s,u) \neq \mathbf{P}^l(s,u)$ *implies* $\forall v \neq u : \tilde{\mathbf{P}}^u(s,v) = \mathbf{P}^u(s,v)$
2. $\tilde{\mathbf{P}}^u(s,u) \neq \mathbf{P}^u(s,u)$ *implies* $\forall v \neq u : \tilde{\mathbf{P}}^l(s,v) = \mathbf{P}^l(s,v)$.

Proof. We provide the proof for the first claim; the proof of the second claim goes along similar lines. Assume $\tilde{\mathbf{P}}^l(s,u) \neq \mathbf{P}^l(s,u)$. By Definition 3.3.2, it follows $1 - \mathbf{P}^u(s, S \setminus \{u\}) > \mathbf{P}^l(s,u)$. Let $v \neq u$. By Definition 3.3.2, we

have $\tilde{\mathbf{P}}^u(s,v) = \min\{\mathbf{P}^u(s,v), 1 - \mathbf{P}^l(s, S \setminus \{v\})\}$. To show that $\tilde{\mathbf{P}}^u(s,v) = \mathbf{P}^u(s,v)$, it suffices to show that $1 - \mathbf{P}^l(s, S \setminus \{v\}) > \mathbf{P}^u(s,v)$. We derive:

$$\begin{aligned}
1 - \mathbf{P}^l(s, S \setminus \{v\}) &= 1 - \mathbf{P}^l(s, S \setminus \{u,v\}) - \mathbf{P}^l(s,u) \\
&> 1 - \mathbf{P}^l(s, S \setminus \{u,v\}) - (1 - \mathbf{P}^u(s, S \setminus \{u\})) \\
&= \mathbf{P}^u(s, S \setminus \{u\}) - \mathbf{P}^l(s, S \setminus \{u,v\}) \\
&\geq \mathbf{P}^u(s, S \setminus \{u\}) - \mathbf{P}^u(s, S \setminus \{u,v\}) \\
&= \mathbf{P}^u(s,v)
\end{aligned}$$

□

The next result states that normalization is idempotent.

Lemma 3.3.2. *For any AMC $\mathcal{M}$, $\eta(\mathcal{M}) = \eta(\eta(\mathcal{M}))$.*

Proof. Let $(\tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u) = \eta(\mathbf{P}^l, \mathbf{P}^u)$ and $(\hat{\mathbf{P}}^l, \hat{\mathbf{P}}^u) = \eta(\eta(\mathbf{P}^l, \mathbf{P}^u))$. We prove that $\tilde{\mathbf{P}}^l = \hat{\mathbf{P}}^l$. For $\tilde{\mathbf{P}}^u = \hat{\mathbf{P}}^u$ the proof goes along similar lines. We distinguish two cases:

1. $\tilde{\mathbf{P}}^l(s,u) \neq \mathbf{P}^l(s,u)$. We derive:

$$\begin{aligned}
\hat{\mathbf{P}}^l(s,u) &= \max\{\tilde{\mathbf{P}}^l(s,u), 1 - \tilde{\mathbf{P}}^u(s, S \setminus \{u\})\} && \text{(Def. 3.3.2)} \\
&= \max\{\tilde{\mathbf{P}}^l(s,u), 1 - \mathbf{P}^u(s, S \setminus \{u\})\} && \text{(Lemma 3.3.1)} \\
&= \tilde{\mathbf{P}}^l(s,u).
\end{aligned}$$

   In the last step we use that from $\tilde{\mathbf{P}}^l(s,u) \neq \mathbf{P}^l(s,u)$ and Definition 3.3.2, it follows $1 - \mathbf{P}^u(s, S \setminus \{u\}) = \tilde{\mathbf{P}}^l(s,u)$.

2. $\tilde{\mathbf{P}}^l(s,u) = \mathbf{P}^l(s,u)$. We distinguish two cases:

   (a) $\tilde{\mathbf{P}}^u(s,v) = \mathbf{P}^u(s,v)$ for any $v \neq u$. Then it follows directly from Definition 3.3.2 that:

$$\tilde{\mathbf{P}}^l(s,u) = \mathbf{P}^l(s,u) \geq 1 - \mathbf{P}^u(s, S \setminus \{u\}) = 1 - \tilde{\mathbf{P}}^u(s, S \setminus \{u\}).$$

(b) $\tilde{\mathbf{P}}^u(s,v) \neq \mathbf{P}^u(s,v)$ for some $v \neq u$. We first observe that this together with Definition 3.3.2 implies $\tilde{\mathbf{P}}^u(s,v) = 1 - \mathbf{P}^l(s, S \setminus \{v\})$ $(*)$. Moreover, from Lemma 3.3.1, it follows $\tilde{\mathbf{P}}^l(s,\hat{s}) = \mathbf{P}^l(s,\hat{s})$ for any $\hat{s} \neq v$ $(**)$.

By Definition 3.3.2, $\hat{\mathbf{P}}^l(s,u) = \max\{\tilde{\mathbf{P}}^l(s,u), 1 - \tilde{\mathbf{P}}^u(s, S \setminus \{u\})\}$. To prove $\hat{\mathbf{P}}^l(s,u) = \tilde{\mathbf{P}}^l(s,u)$ we show that $1 - \tilde{\mathbf{P}}^u(s, S \setminus \{u\}) \leq \tilde{\mathbf{P}}^l(s,u)$. This goes as follows:

$$
\begin{aligned}
& 1 - \tilde{\mathbf{P}}^u(s, S \setminus \{u\}) \\
&= 1 - \tilde{\mathbf{P}}^u(s, S \setminus \{u,v\}) - \tilde{\mathbf{P}}^u(s,v) \\
&= 1 - \tilde{\mathbf{P}}^u(s, S \setminus \{u,v\}) - (1 - \mathbf{P}^l(s, S \setminus \{v\}) && (*) \\
&= \mathbf{P}^l(s, S \setminus \{v\}) - \tilde{\mathbf{P}}^u(s, S \setminus \{u,v\}) \\
&= \tilde{\mathbf{P}}^l(s, S \setminus \{v\}) - \tilde{\mathbf{P}}^u(s, S \setminus \{u,v\}) && (**) \\
&= \tilde{\mathbf{P}}^l(s,u) + \tilde{\mathbf{P}}^l(s, S \setminus \{u,v\}) - \tilde{\mathbf{P}}^u(s, S \setminus \{u,v\}) \\
&\leq \tilde{\mathbf{P}}^l(s,s').
\end{aligned}
$$

□

From the previous results, we conclude that the worst case time complexity of normalizing an AMC is quadratic in the size of its state space. The following result is concerned with the correctness of normalization. It asserts that normalization indeed yields a delimited AMC.

Lemma 3.3.3. *For any AMC $\mathcal{M}$, $\eta(\mathcal{M})$ is delimited.*

Proof. Let $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0)$ and $\eta(\mathcal{M}) = (S, \tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u, L, \mu_0)$. We show that $\eta(\mathcal{M})$ is delimited, i.e., for any $p \in \left[\tilde{\mathbf{P}}^l(s,s'), \tilde{\mathbf{P}}^u(s,s')\right]$ there exists a distribution $\mu \in \mathbf{T}_{\eta(\mathcal{M})}(s)$ with $\mu(s') = p$.

The proof is by contraposition. Assume that no such μ exists for some $p \in \left[\tilde{\mathbf{P}}^l(s,s'), \tilde{\mathbf{P}}^u(s,s')\right]$. Then either $p + \tilde{\mathbf{P}}^u(s, S \setminus \{s'\}) < 1$ or $p + \tilde{\mathbf{P}}^l(s, S \setminus \{s'\}) > 1$. In the first case, we derive using Definition 3.3.2:

$$
\tilde{\mathbf{P}}^l(s,s') \;\leq\; p \;<\; 1 - \tilde{\mathbf{P}}^u(s, S \setminus \{s'\}) \;\leq\; 1 - \mathbf{P}^u(s, S \setminus \{s'\})
$$

which contradicts the assumption that $\eta(\mathcal{M}) = (\tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u)$. In the latter case, it follows using Definition 3.3.2:

$$\tilde{\mathbf{P}}^u(s,s') \;\geq\; p \;>\; 1-\tilde{\mathbf{P}}^l(s,S\setminus\{s'\}) \;\geq\; 1-\mathbf{P}^l(s,S\setminus\{s'\})$$

which is also a contradiction. Thus, no such p exists. □

It is easy to see that the normalization procedure of Definition 3.3.2 is consistent with the notion of schedulers since $\mathbf{T}_{\mathcal{M}}(s) = \mathbf{T}_{\eta(\mathcal{M})}(s)$ for any state $s \in S$.

Lemma 3.3.4. *For any AMC $\mathcal{M}$, we have $\mathcal{D}^{\mathcal{M}} = \mathcal{D}^{\eta(\mathcal{M})}$.*

Proof. It holds that $\mathcal{D}^{\mathcal{M}} = \mathcal{D}^{\eta(\mathcal{M})} \iff Path_{abs}(\mathcal{M}) = Path_{abs}(\eta(\mathcal{M}))$. We prove the second statement.

"$\subseteq$" Proof by contradiction: assume $Path_{abs}(\mathcal{M}) \supset Path_{abs}(\eta(\mathcal{M}))$, then there is a path $\rho \in Path_{abs}(\mathcal{M})$ with $\rho \notin Path_{abs}(\eta(\mathcal{M}))$. For at least one position i in ρ with $\rho[i] = s_i, \rho[i{+}1] = s_{i+1}$, there exists $\mu \in \mathbf{T}_{\mathcal{M}}(s_i)$ with $p = \mu(s_{i+1})$ such that $p \notin [\mathbf{P}^l_{\eta(\mathcal{M})}(s_i,s_{i+1}), \mathbf{P}^u_{\eta(\mathcal{M})}(s_i,s_{i+1})]$. This can only be the case, if η removed p from the interval, i.e.

- $p > 1-\mathbf{P}^l_{\mathcal{M}}(s_i, S\setminus\{s_{i+1}\}) \implies p^l = \mathbf{P}^l_{\mathcal{M}}(s_i, S\setminus\{s_{i+1}\}) + p > 1$
  or
- $p < 1-\mathbf{P}^u_{\mathcal{M}}(s_i, S\setminus\{s_{i+1}\}) \implies p^u = \mathbf{P}^u_{\mathcal{M}}(s_i, S\setminus\{s_{i+1}\}) + p < 1.$

Since p^l is the smallest and p^u is the largest possible sum for any μ and $p^u < 1 < p^l$, there is no valid distribution incorporating p. This contradicts $\mu \in \mathbf{T}_{\mathcal{M}}(s_i)$. Thus $Path_{abs}(\mathcal{M}) \subseteq Path_{abs}(\eta(\mathcal{M}))$.

"$\supseteq$" As $[\mathbf{P}^l_{\mathcal{M}}(s,s'), \mathbf{P}^u_{\mathcal{M}}(s,s')] \supseteq [\mathbf{P}^l_{\eta(\mathcal{M})}(s,s'), \mathbf{P}^u_{\eta(\mathcal{M})}(s,s')]$ for all $s, s' \in S$, at every position i in each $\rho \in Path_{abs}(\eta(\mathcal{M}))$, for all $\mu \in \mathbf{T}_{\eta(\mathcal{M})}(\rho[i])$ it holds $p = \mu(\rho[i+1]) \in [\mathbf{P}^l_{\mathcal{M}}(\rho[i],\rho[i+1]), \mathbf{P}^u_{\mathcal{M}}(\rho[i],\rho[i+1])]$. Thus, $\rho \in Path_{abs}(\mathcal{M})$.

□

Hence, if η is applied to an AMC the resulting AMC is delimited and has the same schedulers. Summarizing, the *normalization* function introduced in Definition 3.3.2 is efficient due to Lemma 3.3.1 (at most $|S|$ bounds have to be adjusted) and Lemma 3.3.2 (it is idempotent). Applying it to an ADTMC results in a delimited ADTMC (Lemma 3.3.3) for which the set of schedulers, and thus the induced behavior, is the same (Lemma 3.3.4).

We conclude this section by addressing the relationship between abstraction of DTMCs and of CTMCs. As normalization and abstraction do not affect the exit rates, the following diagram commutes.

$$\begin{array}{ccccccc} \mathcal{M} & \xrightarrow{\alpha} & \tilde{\mathcal{M}} & \xrightarrow{\eta} & \hat{\mathcal{M}} & & \} \text{ (A)CTMCs} \\ \Big\downarrow{\scriptstyle emb} & & & & \Big\downarrow{\scriptstyle emb} & & \\ \mathcal{N} & \xrightarrow{\alpha} & \tilde{\mathcal{N}} & \xrightarrow{\eta} & \hat{\mathcal{N}} & & \} \text{ (A)DTMCs} \end{array}$$

Thus, abstraction of CTMCs can be regarded as a conservative extension of abstraction of DTMCs.

3.4 Probabilistic simulation

This section studies the relationship between an AMC $\mathcal{M}$ and its abstraction $\alpha(\mathcal{M})$. The central notion for this relationship is a probabilistic variant of (forward) *simulation* on Kripke structures. Simulation relations are preorders on the state space requiring that whenever state s' simulates s, denoted $s \preceq s'$, then s' can mimic all stepwise behavior of s but in addition may also perform steps that cannot be matched by s. For AMCs, we consider a variant of Jonsson and Larsen's seminal notion of probabilistic simulation [JL91].

Definition 3.4.1. Let $\mu \in \mathit{distr}(S)$, $\mu' \in \mathit{distr}(S')$, and $R \subseteq S \times S'$, then μ is *simulated* by μ' w.r.t. R, denoted $\mu \preceq_R \mu'$, if there exists a weight function $\Delta : S \times S' \to [0,1]$ such that for all $u \in S$ and $v \in S'$:

$$\text{(a) } \Delta(u,v) > 0 \Rightarrow uRv, \qquad \text{(b) } \Delta(u,S) = \mu(u), \qquad \text{(c) } \Delta(S,v) = \mu'(v).$$

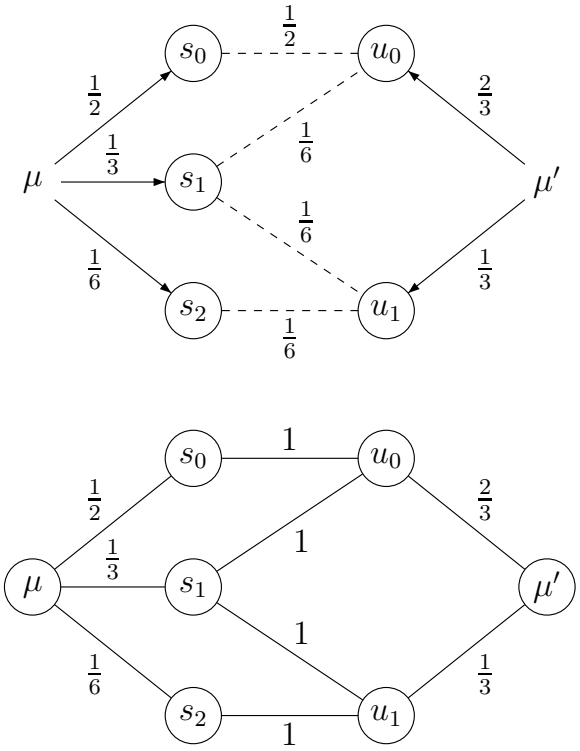


Figure 3.7: Simulation for distributions (top) modeled as maximal flow problem (bottom).

We say that $\preceq_R$ is the lifting of relation R on states to distributions over states.

Example 3.4.1. Simulation for distributions can also be understood as a maximal flow [BEMC00]. Consider $S = \{s_0, s_1, s_2\}$, $S' = \{u_0, u_1\}$ and $\mu \in \mathit{distr}(S)$, $\mu' \in \mathit{distr}(S')$ as depicted in Figure 3.7 (top). Then, the weight function Δ relating s_0 and s_1 with u_0 as well as s_1 and s_2 with u_1 as indicated by the dashed lines in Figure 3.7 (top) is the solution of the maximal flow problem in Figure 3.7 (bottom) where μ and μ' are source and sink and edges are labeled with capacities.

Definition 3.4.2 (Probabilistic simulation)**.** Let $\mathcal{M}$ and $\mathcal{M}'$ be AMCs with state spaces S, S', initial distributions μ_0, μ_0', and labeling functions L and L', respectively. Then, $R \in S \times S'$ is a *probabilistic simulation* relation w.r.t. $\mathcal{M}$ and $\mathcal{M}'$, if:

1. $\mu_0 \preceq_R \mu_0'$, and
2. for all $(s, s') \in R$
   (a) for all $ap \in AP$ it holds $L'(s', ap) \neq ? \Rightarrow L'(s', ap) = L(s, ap)$, and
   (b) for all $\mu \in \mathbf{T}(s)$ there exists $\mu' \in \mathbf{T}(s')$ with $\mu \preceq_R \mu'$.

AMC $\mathcal{M}$ is simulated by $\mathcal{M}'$, denoted $\mathcal{M} \preceq \mathcal{M}'$, if there exists a probabilistic simulation w.r.t. $\mathcal{M}$ and $\mathcal{M}'$, and, if $\mathcal{M}$ and $\mathcal{M}'$ are ACTMCs, they have the same exit rate.

The following result asserts that abstraction preserves probabilistic simulation, i.e. all scheduler choices in the concrete model can be mimicked in the abstraction.

Theorem 3.4.1. *For any AMC $\mathcal{M}$ with state space S and abstraction function $\alpha : S \to S'$ it holds $\mathcal{M} \preceq \alpha(\mathcal{M})$.*

Proof. Let $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0)$ and $\alpha(\mathcal{M}) = \tilde{\mathcal{M}} = (S', \tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u, \tilde{L}, \tilde{\mu}_0)$. We prove $\mathcal{M} \preceq \alpha(\mathcal{M})$ by showing that $R = \{(s, \alpha(s)) \mid s \in S\}$ is a probabilistic simulation. By Definition 3.2.2, $\tilde{\mu}_0(\tilde{u}) = \mu_0(\gamma(\tilde{u}))$, and it directly follows $\mu_0 \preceq_R \tilde{\mu}_0$. It remains to prove condition (2) of Definition 3.4.2.

(2a) Let sRs'. There is nothing to show if $\tilde{L}(s', ap) = ?$. Now let $\tilde{L}(s', ap) = \theta \in \{\bot, \top\}$. Then by Definition 3.2.2, $\tilde{L}(s', ap) = \theta$ implies $L(s, ap) = \theta$ for all $s \in \gamma(s')$.

(2b) Let sRs' and $\mu \in \mathbf{T}_{\mathcal{M}}(s)$. We construct a distribution $\mu' \in \mathbf{T}_{\tilde{\mathcal{M}}}(s')$ and weight function $\Delta : S \times S'$ and prove that these fulfill the conditions of Definition 3.4.1. For $u' \in S'$ let

$$\mu'(u') = \mu(\gamma(u')) \tag{3.2}$$

and

$$\Delta(u, u') = \begin{cases} \mu(u) & \textit{if } uRu' \\ 0 & \textit{otherwise.} \end{cases} \tag{3.3}$$

Function μ' is a probability distribution if μ is so:

$$\textstyle\sum_{u' \in S'} \mu'(u') = \sum_{u' \in S', u \in \gamma(u')} \mu(u) = \sum_{u \in S} \mu(u) = 1$$

We now show that $\mu' \in \mathbf{T}_{\tilde{\mathcal{M}}}(s')$, i.e. that $\tilde{\mathbf{P}}^l(s', u') \leq \mu'(u') \leq \tilde{\mathbf{P}}^u(s', u')$:

$$\begin{array}{lll}
\tilde{\mathbf{P}}^l(s', u') & = \inf_{\bar{s} \in \gamma(s')} \mathbf{P}^l(\bar{s}, \gamma(u')) & \text{(Def. 3.2.2)} \\
& \leq \mathbf{P}^l(s, \gamma(u')) & (s \in \gamma(s')) \\
& \leq \mu(\gamma(u')) & (\mu \in \mathbf{T}_{\mathcal{M}}(s)) \\
& = \mu'(u') & \text{(Eq. 3.2)} \\
& \leq \mathbf{P}^u(s, \gamma(u')) & (\mu \in \mathbf{T}_{\mathcal{M}}(s)) \\
& \leq \sup_{\bar{s} \in \gamma(s')} \mathbf{P}^u(\bar{s}, \gamma(u')) & (s \in \gamma(s')) \\
& = \tilde{\mathbf{P}}^u(s', u') & \text{(Def. 3.2.2)}
\end{array}$$

We finally check the conditions of Definition 3.4.1. Condition (a) is fulfilled trivially, since $\Delta(u, v) = 0$ if $(u, v) \notin R$.

For condition (b), we calculate for $u \in S$:

$$\begin{array}{lll}
\Delta(u, S') & = \sum_{u' \in S'} \Delta(u, u') & \text{(Def. of } \Delta, R) \\
& = \Delta(u, \alpha(u)) & (\Delta(u, u') > 0 \Rightarrow uRu') \\
& = \mu(u) & \text{(see Eq. (3.3))}
\end{array}$$

For $u' \in S'$, we compute:

$$\begin{array}{lll}
\Delta(S, u') & = \sum_{u \in S} \Delta(u, u') & \text{(Def. of } \Delta) \\
& = \sum_{u \in \gamma(u')} \Delta(u, u') = \mu(\gamma(u')) & \text{(see Eq. (3.3))} \\
& = \mu'(u') & \text{(see Eq. (3.2))}
\end{array}$$

□

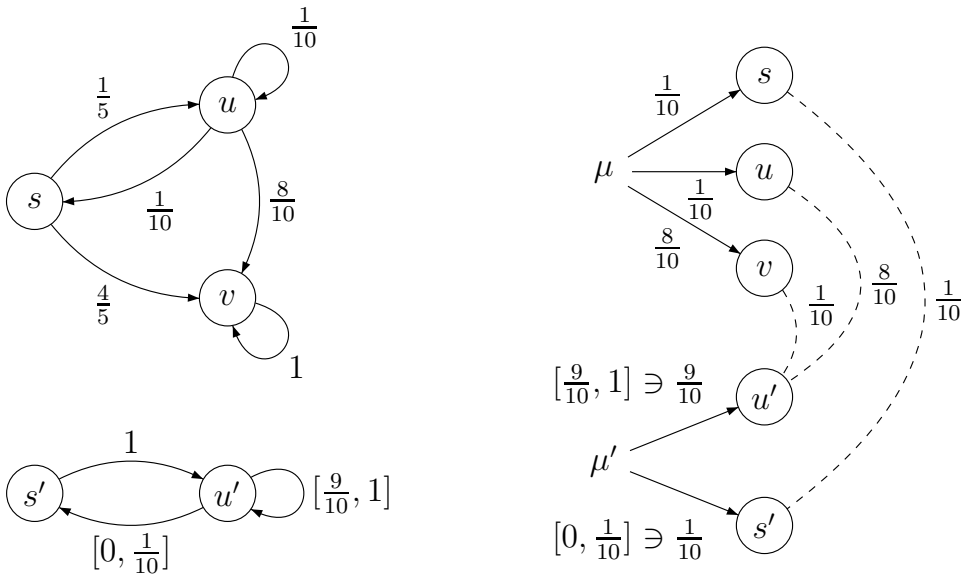


Figure 3.8: An example Markov chain (left, top), an abstraction (left, bottom) and a simulation relation between u and u' (right)

Example 3.4.2. Consider the MC in Figure 3.8 (left, top) and the abstraction induced by α where $\gamma(s') = \{s\}$ and $\gamma(u') = \{u, v\}$ (left, bottom). For example, assuming u and u' are labeled the same, u is simulated by u' as (see Figure 3.8, right):

$$\mu = \mathbf{P}(u, \cdot) = (\tfrac{1}{10}, \tfrac{1}{10}, \tfrac{8}{10}) \preceq \mu' = (\tfrac{1}{10}, \tfrac{9}{10}) \in \mathbf{T}(u').$$

We will now reconsider Example 3.1.2 that showed that MDP abstraction cannot always be used to obtain finite abstractions. In the following, we will see how to come up with a finite *interval abstraction*.

Example 3.4.3. Consider the Markov chain $\mathcal{M} = (S, \mathbf{P}, L, \mu_0)$ in Figure 3.3 (p. 41) with $p_i = \frac{2-\sin(i)-\cos(i)}{4}$ for all $i \in \mathbb{N}$. The term $\frac{2-\sin(i)-\cos(i)}{4}$ gets extreme for $i \in \mathbb{N}$ that minimizes (maximizes) $\sin(i) + \cos(i)$. As both, sinus and cosine are continuous and periodic with period 2π and π is irregular, the extreme values of $\sin(i) + \cos(i)$ are the same for $i \in \mathbb{N}$ and $i \in \mathbb{R}$. As $\frac{d(\sin(i)+\cos(i))}{di} = \cos(i) - \sin(i) = 0$ iff $\sin(i) = \cos(i)$, it follows that for

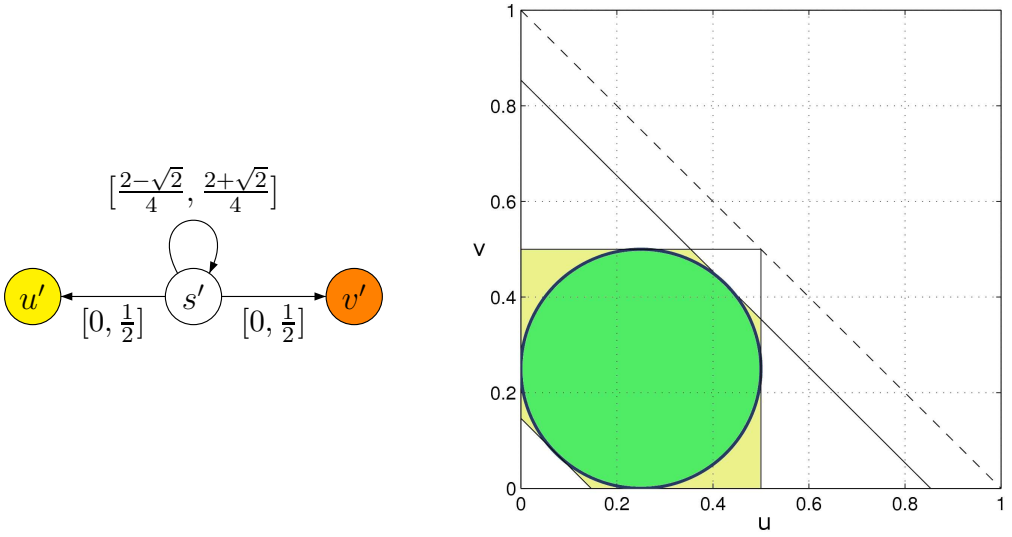


Figure 3.9: An interval abstraction of the infinite-state DTMC from Figure 3.3 (left) and a visualization of concrete and abstract behavior (right).

$i = \arctan(1) = \{\frac{1}{4}\pi + j\pi \mid j \in \mathbb{Z}\}$ the values are extreme, in fact, the infimum and supremum for $\sin(i) + \cos(i)$ are $-\sqrt{2}$ and $\sqrt{2}$. From these preliminary considerations it follows that the abstraction of $\mathcal{M}$ induced by α as given in Example 3.1.2 is the AMC depicted in Figure 3.9 (left).

Let us compare the (nondeterministic) behavior in the abstraction with the concrete behavior. For state s', the probability to take a transition to u' and v' respectively is in $[0, \frac{1}{2}]$. In Figure 3.9 (right), these restrictions are represented by the square with edge length $\frac{1}{2}$. Furthermore, the self-loop probability from s' to s' is in $[\frac{2-\sqrt{2}}{4}, \frac{2+\sqrt{2}}{4}]$, hence, the probability mass that is left for taking a transition to u' or v' is in $1 - [\frac{2-\sqrt{2}}{4}, \frac{2+\sqrt{2}}{4}] = [\frac{2-\sqrt{2}}{4}, \frac{2+\sqrt{2}}{4}] \cong [0.15, 0.85]$. In Figure 3.9 (right), this is indicated by the solid diagonal lines. What is left as possible abstract behavior is represented by the hexagonal shape.

Now consider the concrete behaviors. Plotting the coordinates $(\mu(u), \mu(v))$ for all distributions $\mu \in \mathbf{P}(s_i, \cdot)$ with $i \in \mathbb{N}$ yields a circle with radius $\frac{1}{4}$ and

center $(\frac{1}{4}, \frac{1}{4})$, which is completely contained in the hexagonal shape representing the choices in the interval abstraction. This implies, that all countably many choices we would have to associate to s' in the MDP abstraction can also be chosen in the interval abstraction, but not the other way around.

The remainder of this section is dedicated to a formal comparison of the two abstraction techniques. We show that, given a Markov chain and an abstraction function, interval abstractions are at least as abstract as MDP abstractions in terms of probabilistic simulation.

Theorem 3.4.2. *For any Markov chain $\mathcal{M}$ with state space S and abstraction function $\alpha : S \to S'$ (for which $\alpha_{MDP}(\mathcal{M})$ is defined), it holds:*

$$\mathcal{M} \preceq \alpha_{MDP}(\mathcal{M}) \preceq \alpha(\mathcal{M})$$

Proof. Let $\mathcal{M} = (S, \mathbf{P}, L, \mu_0)$ be a Markov chain and $\alpha : S \to S'$ an abstraction function for which $\alpha_{MDP}(\mathcal{M}) = (S', A', \mathbf{P}', L', \mu_0')$ is the induced MDP abstraction with transitions $\mathbf{T}_{MDP}$ and $\alpha(\mathcal{M}) = (S', \tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u, \tilde{L}, \tilde{\mu}_0)$ is the induced interval abstraction with transitions $\mathbf{T}$. A slight variant of the first part $\mathcal{M} \preceq \alpha_{MDP}(\mathcal{M})$ has been shown in [DJJL01], thus we concentrate on proving that $\alpha_{MDP}(\mathcal{M}) \preceq \alpha(\mathcal{M})$. Let $R : S' \times S'$ with $s' R \tilde{s}$ iff $\gamma(s') = \gamma(\tilde{s})$, that is, we compare abstract states representing the same set of concrete states. As we consider the same α for both abstractions, this implies $s' R \tilde{s}$ iff $s' = \tilde{s}$.

First, note that $L' = \tilde{L}$ and $\mu_0' = \tilde{\mu}_0$ by definition; thus conditions (1) and (2a) in Definition 3.4.2 are fulfilled trivially. It remains to show the satisfaction of condition (2b), i.e. for all $s' R \tilde{s}$ it holds that for any $\mu' \in \mathbf{T}_{MDP}(s')$, there exists $\tilde{\mu} \in \mathbf{T}(\tilde{s})$ with $\mu' \preceq_R \tilde{\mu}$. As $s' R \tilde{s}$ iff $s' = \tilde{s}$, it suffices to show that $\mathbf{T}_{MDP}(s') \subseteq \mathbf{T}(s')$ for all $s' \in S'$.

By the definition of $\mathbf{T}_{MDP}(s')$, we have that for any $\mu' \in \mathit{distr}(S')$ it holds $\mu' \in \mathbf{T}_{MDP}(s')$ iff there exists $s \in \gamma(s')$ with $\mu'(u') = \mu(\gamma(u'))$ for $\mu = \mathbf{P}(s, \cdot)$.

For such a $\mu \in \mathit{distr}(S)$, by Definition 3.2.2, it holds for all $u' \in S'$:

$$\begin{aligned} \tilde{\mathbf{P}}^l(s', u') &= \inf_{\hat{s} \in \gamma(s')} \mathbf{P}(\hat{s}, \gamma(u')) \\ &\leq \mu(\gamma(u')) = \mu'(u') \\ &\leq \min(1, \sup_{\hat{s} \in \gamma(s')} \mathbf{P}(\hat{s}, \gamma(u'))) = \tilde{\mathbf{P}}^u(s', u') \end{aligned} \tag{3.4}$$

Further, the set $\mathbf{T}(s')$ contains all distributions respecting the intervals given by $\tilde{\mathbf{P}}^l$ and $\tilde{\mathbf{P}}^u$ in the interval abstraction, i.e.

$$\mathbf{T}(s') = \{\mu' \in \mathit{distr}(S') \mid \mu'(u') \in [\tilde{\mathbf{P}}^l(s', u'), \tilde{\mathbf{P}}^u(s', u')] \text{ for all } u' \in S'\}.$$

Thus, from inequation (3.4) it follows that $\mu' \in \mathbf{T}(s')$. □

3.5 Summary and discussion

In this chapter, we introduced two rather similar abstraction techniques: MDP abstraction and interval abstraction. While the former technique is technically easier to deal with – no normalization is necessary – the latter yields more compact representations and finite abstractions for a larger class of infinite-state models. We showed that both, MDP abstraction and interval abstraction, simulate the given concrete model; moreover, for a fixed abstraction function, the resulting AMC obtained by interval abstraction simulates the MDP obtained by MDP abstraction. In other words, interval abstraction yields more abstract results than MDP abstraction.

We lifted abstraction for discrete-time models to the continuous-time setting; to our knowledge, this has not been done before [KKLW07a]. However, an orthogonal technique for the abstraction of CTMCs based on a partial ordering of the state space has been presented in [CVV08]; investigating the possible gains by combining both techniques is a worthy objective for future research. Another direction that needs to be explored is, how to automatically find an adequate partitioning of the state space, and, for the orthogonal technique, a partial ordering on the state space.

4

Model checking abstract Markov chains

In the previous chapter, we discussed how to omit details by grouping states in order to obtain manageable (finite) abstractions for infinite models; we focused on AMCs as the presented ideas are easily transferable to MDPs. In the following, we will investigate which information is preserved by abstraction and how to derive that information effectively.

The abstract models we obtain by the presented techniques are subject to nondeterminism, thus, we cannot compute fixed probability measures, say reachability probabilities, as for deterministic models; instead we may compute minimal and maximal probabilities w.r.t. a set of schedulers resolving the nondeterminism. We strive to compute minimal and maximal probabilities for abstractions that are safe lower and upper bounds of the actual probabilities in the concrete model; therefore, we first discuss which class of schedulers we have to consider in order to obtain such safe bounds. An essential result is that so-called *extreme schedulers* suffice for computing minimal (maximal) measures.

Further, we present algorithms for computing reachability probabilities on AMCs, i.e. step- and time-bounded reachability as well as unbounded reachability. These algorithms are, as we shall see, the key ingredients of the model checking procedure for PCTL and CSL. Finally, in a case study from the area of queueing theory, we show how abstraction can yield quite precise results where standard techniques are not feasible at all.

4.1 Extreme probability measures

When minimizing (or maximizing) probability measures, say for reaching a set of goal states within a given amount of steps or time, we have to refer to a *class* of schedulers. In this section, we discuss which class of schedulers is appropriate for abstraction, that is, what information on the history of a systems development schedulers need to know in order to yield bounds for the actual probability measures. Then, we reduce the set of schedulers to a manageable subset that preserves extreme measures and therefore is referred to as *extreme schedulers*.

In general, it is not clear whether stationary, history-dependent or timed schedulers (cf. Section 2.4.2, p. 28) yield safe lower and upper bounds. We have already shown that for time-bounded reachability in CTMDPs these three scheduler classes yield pairwise different results. The same arguments apply to ACTMCs as well and therefore it is important to discuss which scheduler class is appropriate in the context of abstraction. While for *unbounded* reachability stationary schedulers are known to suffice [dA97], in the following example we will see that this is not the case for bounded reachability.

Example 4.1.1. Let CTMC $\mathcal{M} = (S, \mathbf{P}, L, \mu_0, \lambda)$ be as depicted in Figure 4.1 (top) with $k \in \mathbb{N}_{>0}$. Basically, there are two paths from s_0 to s_g where for both, the probability to finally reach s_g is $\frac{1}{2}$ and the time it takes is distributed the same on both paths, namely Erlang-$(3k{+}2)$.

Let $\alpha : S \to S'$ be given by $\gamma(s') = \{s\}$ for all $s' \neq \bar{s}'_0$ and $\gamma(\bar{s}'_0) = \{\bar{s}_0, \bar{u}_0\}$. In $\alpha(\mathcal{M})$ as depicted in Figure 4.1 (middle), both paths are now split into a prefix (ending in $\bar{s}'_0$) and a suffix; note that the prefix $s'_0 \to s'_1 \to \ldots \to \bar{s}'_0$ is short, i.e. *fast*, but half of the probability is lost for reaching s'_g; in contrast, the prefix $s'_0 \to u'_1 \to \ldots \to \bar{s}'_0$ is long, i.e. *slow*, but all of the probability may still flow to s'_g finally. By considering the history that led to $\bar{s}'_0$ in $\alpha(\mathcal{M})$, it is easy to decide which of the suffixes has to be chosen in order to mimic the behavior of $\mathcal{M}$. Note that it is not necessary to take the timing

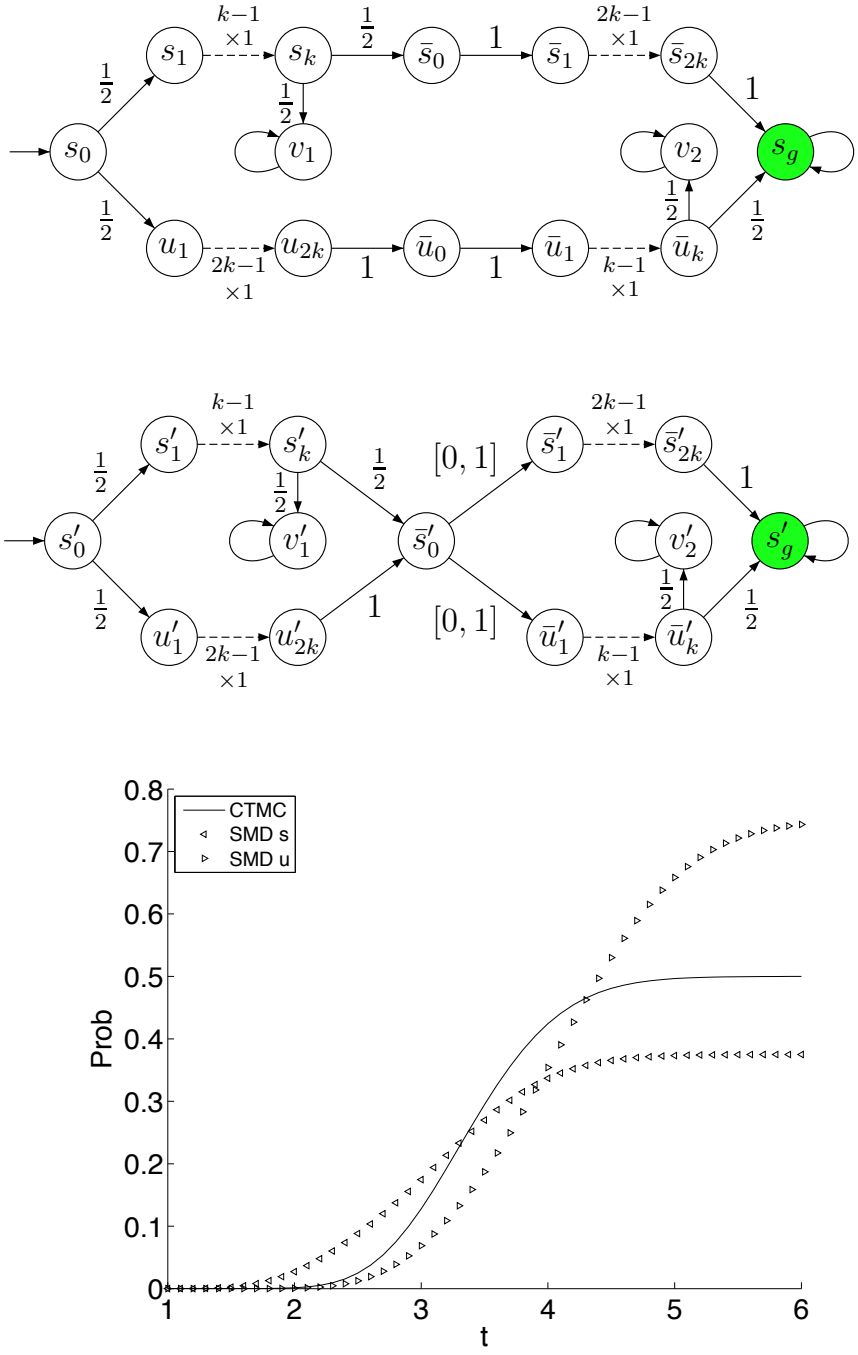


Figure 4.1: Stationary (SMD) schedulers do not suffice: the supremum over stationary schedulers in the ACTMC (middle) is lower than the probability in the concrete model (top) for, e.g. $t = 4$ and $\lambda = k = 10$ (bottom).

information into account: e.g. for any two timed paths $\varsigma, \varsigma' \in Path_{\mathcal{M}}$ with $\varsigma[j] = \varsigma'[j] \in \gamma(\bar{s}'_0)$ for some $j \in \mathbb{N}_{>0}$ and $\varsigma[i] = \varsigma'[i]$ for all $i \in \{0, \ldots, j\}$, the successor probability distribution in $\alpha(\mathcal{M})$ simulating the original CTMC in the $(j+1)$-st state of both paths must be the same.

If one restricts to simple (SMD) schedulers, $\mathcal{M}$ cannot be mimicked anymore; moreover, choosing successor $\bar{s}'_1$ with probability 1 (or $\bar{u}'_1$ respectively) for all histories does not result in a lower or an upper bound for time-bounded reachability probabilities in $\mathcal{M}$ (cf. Figure 4.1, bottom); this remains to be true for any other distribution over $\{\bar{s}'_1, \bar{u}'_1\}$.

Similar observations can be made for step-bounded reachability.

As shown in the example above, we need to consider history-dependent schedulers, however, it is not necessary to consider timed ones as will be proven later on. The following lemma justifies why for AMCs, in contrast to MDPs, randomization does not yield a larger class of schedulers as their deterministic counterparts. It states that the classes of schedulers are closed under convex combinations.

Lemma 4.1.1. *Let $\mathcal{M}$ be an AMC, $D_i \in \mathcal{D}^{\mathcal{M}}$ and $p_i \in [0,1]$ for all $i \in \mathbb{N}$ such that $\sum_{k=0}^{\infty} p_i = 1$. Then, $\sum_{i=0}^{\infty} p_i \cdot D_i \in \mathcal{D}^{\mathcal{M}}$.*

Proof. Let $D_i \in \mathcal{D}^{\mathcal{M}}$, and $p_i \in [0,1]$ for all $i \in \mathbb{N}$ with $\sum_{i=0}^{\infty} p_i = 1$. First, observe that for all $\varrho \in Path^{\mathcal{M}}_{abs}$ with $\varrho\downarrow = s$, it holds $D_i(\varrho) \in \mathbf{T}_{\mathcal{M}}(s)$ for all $i \in \mathbb{N}$ and $D(\varrho) = \sum_{i=0}^{\infty} p_i \cdot D_i(\varrho) \in distr(S)$. Let $\mu = D(\varrho)$ and $\mu_i = D_i(\varrho)$. Then for all $i \in \mathbb{N}$, $u \in S$ we have $\mathbf{P}^l(s,u) \leq \mu_i(u) \leq \mathbf{P}^u(s,u)$ which implies

$$\mathbf{P}^l(s,u) \leq \textstyle\sum_{i=0}^{\infty} p_i \mu_i(s) = \mu(u) \leq \mathbf{P}^u(s,u).$$

Hence, $D(\varrho) \in \mathbf{T}_{\mathcal{M}}(s)$ and therefore $D \in \mathcal{D}^{\mathcal{M}}$. □

Recall that our main objective is to obtain algorithms for minimal (maximal) reachability probabilities. In order to restrict to a manageable class of schedulers, i.e. schedulers that do not face an infinite number of choices, we focus on schedulers that may only select so-called *extreme* distributions:

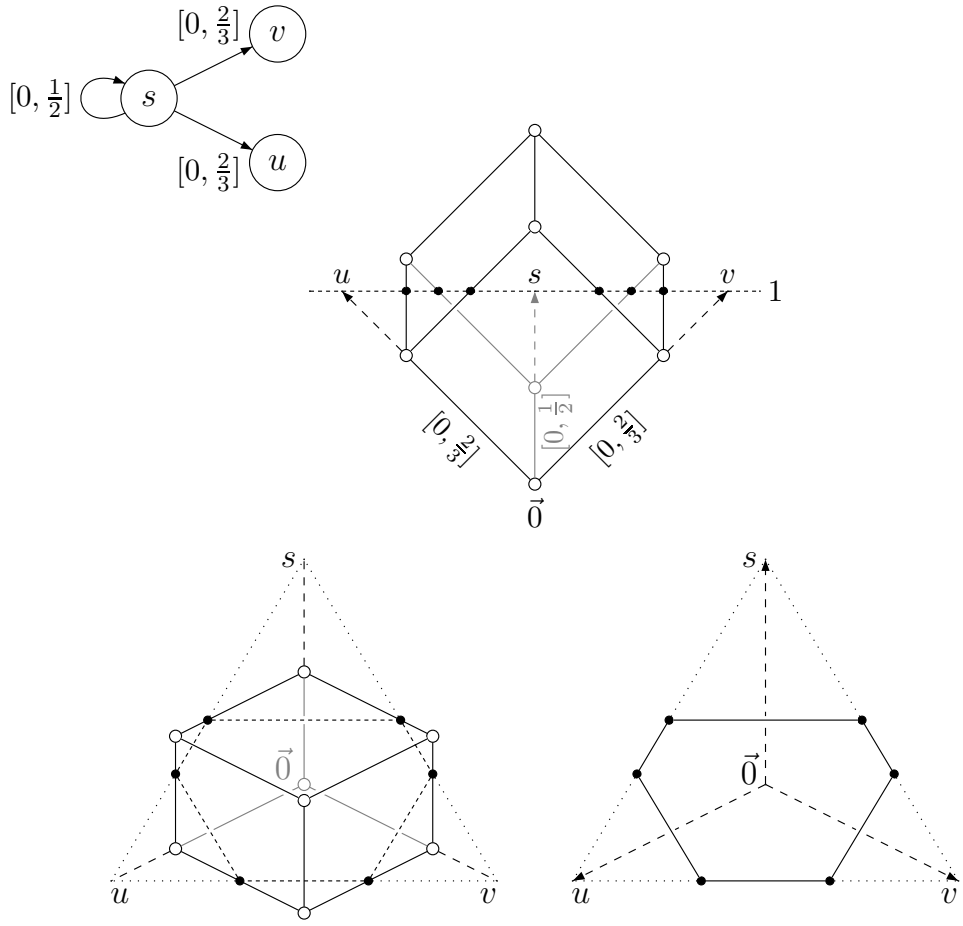


Figure 4.2: A geometric interpretation of intervals (cube, white nodes), set *distr*(S) (triangle, dotted lines) and extreme distributions (black nodes).

Consider the cube in Figure 4.2. It represents all combinations of values that can be chosen from the three probability intervals $[0, \frac{1}{2}]$, $[0, \frac{2}{3}]$ and $[0, \frac{2}{3}]$ of the AMC in Figure 4.2 (top, left). The set $\mathit{distr}(\{s, u, v\})$ is represented by the dotted triangle in Figure 4.2 (bottom, left). Hence, all points in the intersection of the cube and the triangle are distributions respecting the interval bounds (bottom, right). The six bold vertices spanning the intersection may serve as a finite representation of the infinite set of choices: every distribution $\mu \in \mathbf{T}(s)$ can be constructed as a convex combination of the six *extreme* distributions which span the intersection; that is, a scheduler choosing randomly from extreme distributions can make the same decisions as a scheduler choosing deterministically (or randomly) from the set of all distributions respecting the intervals.

Definition 4.1.1 (Extreme distributions)**.** Let AMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0)$ be delimited, $s \in S$ and $U \subseteq S$. We define $\mathit{extr}(\mathbf{P}^l, \mathbf{P}^u, U, s) \subseteq \mathbf{T}(s)$ such that for any $\mu \in \mathit{extr}(\mathbf{P}^l, \mathbf{P}^u, U, s)$, it holds

1. $U = \varnothing$ implies $\mu(u) = \mathbf{P}^l(s, u) = \mathbf{P}^u(s, u)$ for any $u \in S$
2. $U \neq \varnothing$ implies that for some $u \in U$ either
   (a) $\mu(u) = \mathbf{P}^l(s, u)$ and
   $\mu \in \mathit{extr}(\eta(\mathbf{P}^l, \mathbf{P}^u[(s, u) \mapsto \mu(u)]), U \setminus \{u\}, s)$, or
   (b) $\mu(u) = \mathbf{P}^u(s, u)$ and
   $\mu \in \mathit{extr}(\eta(\mathbf{P}^l[(s, u) \mapsto \mu(u)], \mathbf{P}^u), U \setminus \{u\}, s)$.

Distribution $\mu \in \mathbf{T}(s)$ is called *extreme* if $\mu \in \mathit{extr}(\mathbf{P}^l, \mathbf{P}^u, S, s)$.

Here, $\mathit{extr}(\mathbf{P}^l, \mathbf{P}^u, U, s)$ denotes the subset of distributions in state s for which all probabilities of states not in U are extreme. The recursive nature of this definition stems from the fact that fixing an (extreme) probability for one of the states in U possibly restricts the choices for the remaining states. Note that $\mathit{extr}(\mathbf{P}^l, \mathbf{P}^u, U, s)$ is not uniquely defined as the order in which probabilities are fixed for states in U may restrict the choices for the remaining states.

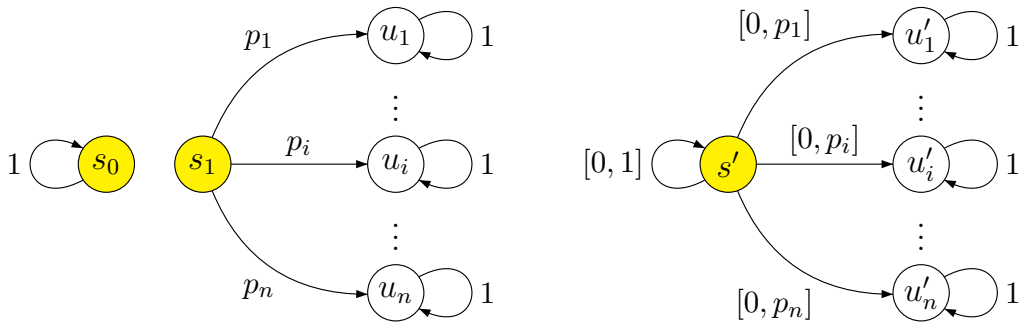


Figure 4.3: A Markov chain (left) and an abstraction with 2^n extreme distributions in s'.

Further, in case $\{s \in S \mid \mathbf{P}^l(s,S) > 0\}$ is finite, the set $extr(\mathbf{P}^l, \mathbf{P}^u, S, s)$ is finite, whereas the set $\mathbf{T}(s)$ is typically uncountably infinite as the transition intervals are dense. While reducing the set of significant distributions from infinite to finite size, the number of distributions may still be exponential in the size of the state space; this is shown in the following example. Yet, minimal and maximal step- and time-bounded reachability probabilities can be computed in polynomial time in the size of the state space as we will see in Sections 4.2 and 4.3.

Example 4.1.2. Consider the AMC in Figure 4.3 (right) resulting from abstraction of the Markov chain (left) by merging s_0 and s_1 to s'. Let $p_i > 0$ for all $1 \leq i \leq n$ and note that $\sum_{i=1}^{n} p_i = 1$. In s' there are 2^n different extreme distributions as for each $U \subseteq \{u'_1, \ldots, u'_n\}$ there is a distinct extreme distribution $\mu \in \mathbf{T}(s')$ with $\mu(s') = 1 - \mu(U)$ and $\mu(u'_i) = p_i$ iff $u'_j \in U$.

A history-dependent deterministic scheduler $E : Pathf \to distr(S)$ that chooses extreme distributions only, i.e. for all $\varrho \in Pathf$ it holds $E(\varrho) \in extr(\mathbf{P}^l, \mathbf{P}^u, S, \varrho\downarrow)$, is called *extreme*. Sets of HD schedulers that are restricted to extreme ones are denoted $\mathcal{E}^{\mathcal{M}}$. For AMC $\mathcal{M}$ and scheduler $D \in \mathcal{D}^{\mathcal{M}}$, let $\mathcal{M}_D$ be the induced Markov chain and let $Pr^{\mathcal{M}_D}$, or simply Pr^D, denote the

probability measure associated with $\mathcal{M}_D$, and $\mathcal{B}_{\mathcal{M}}$ the σ-algebra generated by cylinder sets of paths in $\mathcal{M}$. We may use Pr^D as a measure on $\mathcal{B}_{\mathcal{M}}$ rather than for sets of paths in $\mathcal{M}_D$, as every state in $\mathcal{M}_D$ is a path fragment of $\mathcal{M}$, e.g., for ADTMC $\mathcal{M}$ and scheduler D, the probability of the cylinder set $\mathcal{C}(s_1\, s_2\, s_3) \subset Path^{\mathcal{M}}$ equals the measure of $\mathcal{C}(s_1\ s_1s_2\ s_1s_2s_3) \subset Path^{\mathcal{M}_D}$, i.e., $Pr^D(\mathcal{C}(s_1\ s_2\ s_3)) = Pr^{\mathcal{M}_D}(\mathcal{C}(s_1\ s_1s_2\ s_1s_2s_3))$.

Later in this section, we will show that extreme schedulers constitute an important class of schedulers as minimum (and maximum) reachability probabilities under extreme schedulers are optimal, i.e., allowing non-extreme schedulers does not yield better minima (maxima). In fact, this applies not only to reachability probabilities, but to more general events.

Optimal choices. Before we take a closer look at general measures under extreme schedulers, we show how to optimally choose an extreme distribution over successor states for which a certain *rating* is given. Intuitively, the higher the rating of a successor, the more beneficial it is to minimize or maximize the probability for taking a transition to that successor. In the following, we focus on minimization; for maximization the argument can be made analogously.

Definition 4.1.2. Let $\mathcal{M} = (S, \mathbf{P}_0^l, \mathbf{P}_0^u, L, \mu_0)$ be an AMC, $s \in S$ and $\nu : S \to [0,1]$ a *rating* of the states; further, let $\{u_0, u_1, \ldots\} = S$ be the state space ordered such that $\nu(u_i) \leq \nu(u_j)$ iff $i \leq j$. Then, we define $\min^{\mathcal{M}}(s,\nu) = \mu' \in \mathit{extr}(\mathbf{P}_0^l, \mathbf{P}_0^u, S, s)$ by $\mu'(u_i) = \mathbf{P}_i^u(s, u_i)$ for

$$(\mathbf{P}_{i+1}^l, \mathbf{P}_{i+1}^u) = \eta(\mathbf{P}_i^l[(s,u_i) \mapsto \mathbf{P}_i^u(s,u_i)], \mathbf{P}_i^u) \quad \text{for } i \in \mathbb{N}.$$

Intuitively, $\min^{\mathcal{M}}(s,\nu)$ maximizes the probability for choosing successor states $u_0, u_1, \ldots$ of s iteratively (where $\nu(u_i) \leq \nu(u_j)$ iff $i \leq j$) while respecting the transition probability intervals in $\mathcal{M}$. In order to avoid choosing values that do not yield a distribution, normalization is applied after each step.

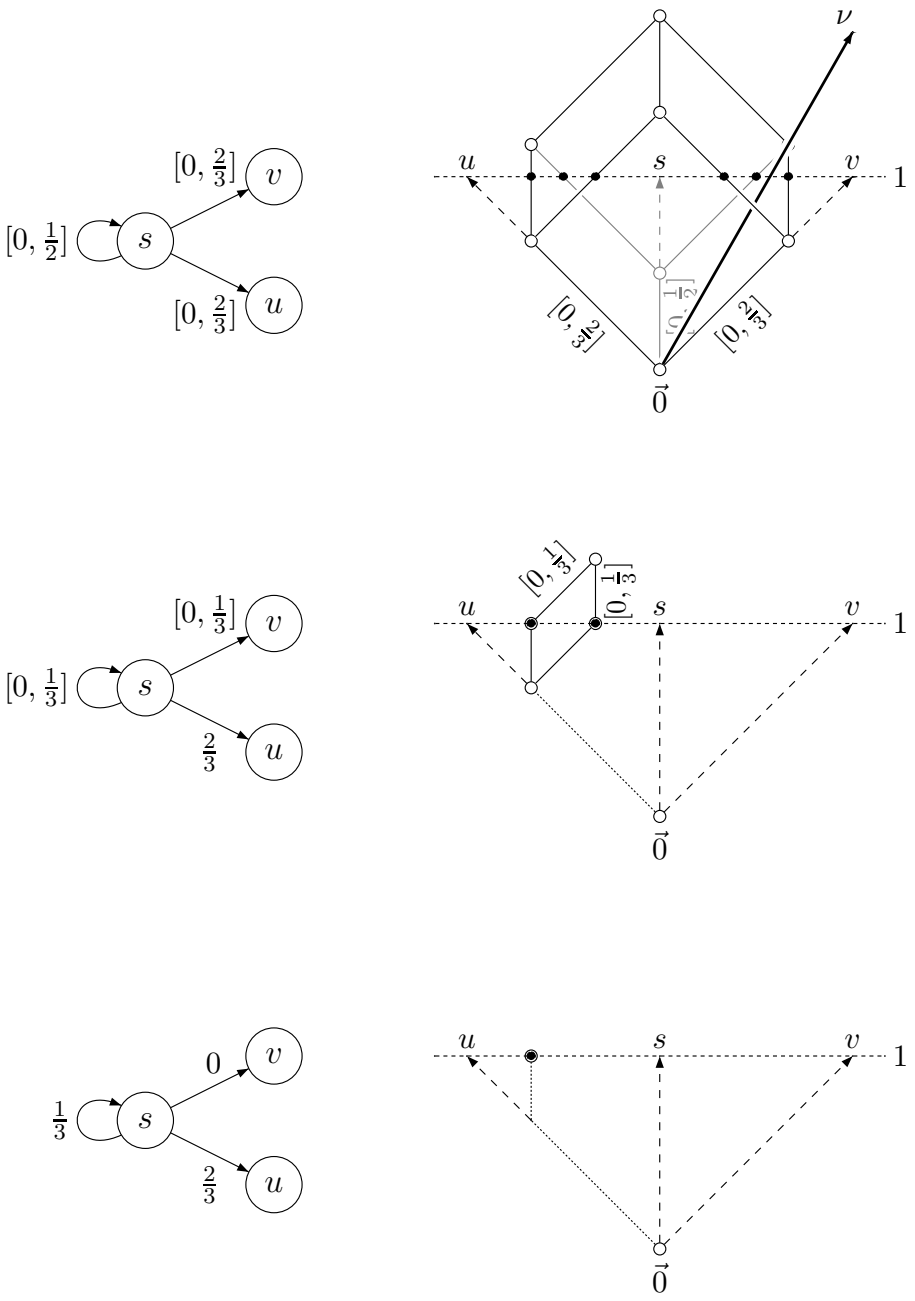


Figure 4.4: Iteratively computing the optimal choice in s for minimizing w.r.t. rating $\nu = \{s \mapsto \frac{3}{4}, u \mapsto 0, v \mapsto 1\}$.

Example 4.1.3. Let us consider the cube example from Figure 4.2 again and let us compute $\min^{\mathcal{M}}(s, \nu)$ with $\nu = \{s \mapsto \frac{3}{4}, u \mapsto 0, v \mapsto 1\}$. The AMC $\mathcal{M}$, its graphical representation and the rating ν are depicted in Figure 4.4 (top). Basically, the minimization function will determine an extreme distribution such that no other extreme distribution (represented by black dots) is smaller in terms of the projection to the ν vector. This is achieved by performing the following computations:

1. as u has the smallest rating $\nu(u) = 0$, maximize the choice for u, i.e. set $\mathbf{P}_1^l(s, u) = \mathbf{P}_0^u(s, u) = \frac{2}{3}$ and compute the normalization of the resulting choices (see Figure 4.4, middle),

2. as s has the second smallest rating $\nu(s) = \frac{3}{4}$, maximize the choice for s by setting $\mathbf{P}_2^l(s, s) = \mathbf{P}_1^u(s, s) = \frac{1}{3}$ and compute the normalization of the resulting choices (see Figure 4.4, bottom),

3. as all other choices have been made, $\mathbf{P}_2^l(s, v) = \mathbf{P}_2^u(s, v) = 0$ must already be determined as $\mathcal{M}$ is delimited.

Hence, we obtain extreme distribution $\min^{\mathcal{M}}(s, \nu) = \{u \mapsto \frac{2}{3}, s \mapsto \frac{1}{3}\}$. Let us validate that the minimization function indeed made its choices to minimize w.r.t. the given rating:

extreme distribution μ	$\sum_{u \in S} \mu(u) \cdot \nu(u)$
$\{u \mapsto \frac{2}{3}, s \mapsto \frac{1}{3}\}$	$\frac{1}{4} = 0.25$
$\{u \mapsto \frac{2}{3}, v \mapsto \frac{1}{3}\}$	$\frac{1}{3} = 0.333\ldots$
$\{s \mapsto \frac{1}{2}, u \mapsto \frac{1}{2}\}$	$\frac{3}{8} = 0.375$
$\{s \mapsto \frac{1}{2}, v \mapsto \frac{1}{2}\}$	$\frac{7}{8} = 0.875$
$\{v \mapsto \frac{2}{3}, u \mapsto \frac{1}{3}\}$	$\frac{2}{3} = 0.666\ldots$
$\{v \mapsto \frac{2}{3}, s \mapsto \frac{1}{3}\}$	$\frac{11}{12} = 0.916\ldots$

Obviously, for $\min^{\mathcal{M}}(s, \nu) = \{u \mapsto \frac{2}{3}, s \mapsto \frac{1}{3}\}$, the weighted sum over the ratings is minimal.

The following lemma formalizes the observation made in the above example regarding the minimality w.r.t. a rating for distributions obtained by the minimization function from Definition 4.1.2.

Lemma 4.1.2. *Let $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0)$ be an AMC, $s \in S$ and $\nu : S \to [0,1]$ a rating of the state space. For all $\mu \in \mathbf{T}^{\mathcal{M}}(s)$:*

$$\sum_{u \in S} \min^{\mathcal{M}}(s,\nu)(u) \cdot \nu(u) \leq \sum_{u \in S} \mu(u) \cdot \nu(u)$$

Proof. Let $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0)$ be an AMC, $s \in S$, $\nu : S \to [0,1]$ and let $\mu \in \mathbf{T}^{\mathcal{M}}(s)$ be arbitrary. Recall that by definition, $\mu_{\min} = \min^{\mathcal{M}}(s,\nu)$ maximizes the probability for choosing successor states $u_0, u_1, \ldots$ of s iteratively (where $\nu(u_i) \leq \nu(u_j)$ iff $i \leq j$). Moreover, it is the only distribution $\mu \in \mathbf{T}^{\mathcal{M}}(s)$ for which $\sum_{j=0}^{i} \mu(u_i)$ is maximal for all $i \in \mathbb{N}$ as we show by contraposition:

Assume, $\mu \in \mathbf{T}^{\mathcal{M}}(s) \setminus \{\mu_{\min}\}$ and for all $i \in \mathbb{N}$, we have that $\sum_{j=0}^{i} \mu(u_i)$ is maximal w.r.t. $\mathbf{T}^{\mathcal{M}}(s)$. Then, there exists $i \in \mathbb{N}$, with $\mu(u_i) \neq \mu_{\min}(u_i)$ and for all $j < i$ it holds $\mu(u_j) = \mu_{\min}(u_j)$. Consider two cases:

1. Let $\mu(u_i) < \mu_{\min}(u_i)$, then $\sum_{j=0}^{i} \mu(u_i) < \sum_{j=0}^{i} \mu_{\min}(u_i)$ contradicting our assumption regarding μ.

2. Let $\mu(u_i) > \mu_{\min}(u_i)$, then for $\mathbf{P}_i^u$ as given in Definition 4.1.2 it holds $\mathbf{P}_i^u(s,u_i) \geq \mu(u_i) > \mu_{\min}(u_i)$ which contradicts $\mu_{\min} = \min^{\mathcal{M}}(s,\nu)$.

Thus, it holds $\sum_{j=0}^{i} \mu_{\min}(u_j) \geq \sum_{j=0}^{i} \mu(u_j)$ for all $i \in \mathbb{N}$. This allows us to construct a weight function $\Delta : S \times S \to [0,1]$ relating distributions $\mu_{\min}$ and $\mu \in \mathbf{T}(s)$ with $\Delta(u_i, S) = \mu_{\min}(u_i)$, $\Delta(S, u_j) = \mu(u_j)$ and an additional condition $\Delta(u_i, u_j) > 0 \implies i \leq j$. For all $i, j \in \mathbb{N}$, we define:

$$\Delta(u_i,u_j) = \left| \left[\sum_{k=0}^{i-1} \mu_{\min}(u_k), \sum_{k=0}^{i} \mu_{\min}(u_k) \right] \cap \left[\sum_{k=0}^{j-1} \mu(u_k), \sum_{k=0}^{j} \mu(u_k) \right] \right|$$

Then, for all $i, j \in \mathbb{N}$, the above conditions are satisfied as

$$\begin{aligned}
\Delta(u_i, S) &= \left| \left[\textstyle\sum_{k=0}^{i-1} \mu_{\min}(u_k), \sum_{k=0}^{i} \mu_{\min}(u_k) \right] \cap [0,1] \right| = \mu_{\min}(u_i) \\
\Delta(S, u_j) &= \left| [0,1] \cap \left[\textstyle\sum_{k=0}^{j-1} \mu(u_k), \sum_{k=0}^{j} \mu(u_k) \right] \right| = \mu(u_j) \\
\Delta(u_i, u_j) &= \left| \left[\textstyle\sum_{k=0}^{i-1} \mu_{\min}(u_k), \sum_{k=0}^{i} \mu_{\min}(u_k) \right] \cap \left[\textstyle\sum_{k=0}^{j-1} \mu(u_k), \sum_{k=0}^{j} \mu(u_k) \right] \right| \\
&= 0, \quad \text{if } i > j
\end{aligned}$$

where the last equation follows from

$$\textstyle\sum_{k=0}^{i-1} \mu_{\min}(u_k) \geq \sum_{k=0}^{i-1} \mu(u_k) \geq \sum_{k=0}^{j} \mu(u_k) \quad \text{for all } j \in \mathbb{N} \text{ with } i > j.$$

Intuitively, when redistributing $\mu_{\min}$ to μ by Δ, it is not necessary to assign some probability from $\mu_{\min}(u_i)$ to $\mu(u_j)$ for any $i > j$ as exemplified in the following picture:

<table>
<tr><td colspan="2">$\mu_{\min}(u_0)$</td><td colspan="3">$\mu_{\min}(u_1)$</td><td>$\mu_{\min}(u_2)$</td></tr>
<tr><td>$\Delta(u_0, u_0)$</td><td>$\Delta(u_0, u_1)$</td><td>$\Delta(u_1, u_1)$</td><td>$\Delta(u_1, u_2)$</td><td>$\Delta(u_1, u_3)$</td><td>$\Delta(u_2, u_3)$</td></tr>
<tr><td>$\mu(u_0)$</td><td colspan="2">$\mu(u_1)$</td><td>$\mu(u_2)$</td><td colspan="2">$\mu(u_3)$</td></tr>
</table>

We now compute for any $\mu \in \mathbf{T}^{\mathcal{M}}(s)$ and for a corresponding Δ as defined above

$$\begin{aligned}
& \textstyle\sum_{i \in \mathbb{N}} \mu_{\min}(u_i) \cdot \nu(u_i) \\
=\ & \textstyle\sum_{i \in \mathbb{N}} \Delta(u_i, S) \cdot \nu(u_i) && (\text{Def. } \Delta, \mu_{\min}) \\
=\ & \textstyle\sum_{i,j \in \mathbb{N}} \Delta(u_i, u_j) \cdot \nu(u_i) \\
=\ & \textstyle\sum_{j \in \mathbb{N}} \sum_{i=0}^{j} \Delta(u_i, u_j) \cdot \nu(u_i) && (\text{Def. } \Delta) \\
\leq\ & \textstyle\sum_{j \in \mathbb{N}} \sum_{i=0}^{j} \Delta(u_i, u_j) \cdot \nu(u_j) && (\forall i \leq j : \nu(u_i) \leq \nu(u_j) \\
=\ & \textstyle\sum_{i,j \in \mathbb{N}} \Delta(u_i, u_j) \cdot \nu(u_j) && (\text{Def. } \Delta) \\
=\ & \textstyle\sum_{j \in \mathbb{N}} \Delta(S, u_j) \cdot \nu(u_j) \\
=\ & \textstyle\sum_{j \in \mathbb{N}} \mu(u_j) \cdot \nu(u_j) && (\text{Def. } \Delta)
\end{aligned}$$

concluding the proof. □

With the above lemma, we can now show that extreme distributions do suffice for obtaining extreme measures in an abstract Markov chain.

Theorem 4.1.3. *For any AMC $\mathcal{M}$ and measurable set $Q \in \mathcal{B}_{\mathcal{M}}$:*

$$\inf_{E\in\mathcal{E}^{\mathcal{M}}} Pr^{E}(Q) = \inf_{D\in\mathcal{D}^{\mathcal{M}}} Pr^{D}(Q)$$
$$\sup_{E\in\mathcal{E}^{\mathcal{M}}} Pr^{E}(Q) = \sup_{D\in\mathcal{D}^{\mathcal{M}}} Pr^{D}(Q)$$

Proof. Let μ_0 be the initial distribution of $\mathcal{M}$ and assume that $\mathcal{M}$ is delimited. As the arguments are similar for the supremum and the infimum, we give details only for the infimum. Let us first treat the discrete-time case, i.e., we assume that $\mathcal{M}$ is an ADTMC.

Discrete-time case:
Before proving the claim, let us fix some notations. For all $n \in \mathbb{N} \cup \{-1\}$, let $\mathcal{E}_n \subset \mathcal{E}^{\mathcal{M}}$ be the set of schedulers that choose *extreme distributions for histories of length* $\leq n$, i.e. $\mathcal{E}_{-1} = \mathcal{D}$ and for any $E_n \in \mathcal{E}_n$ it holds $E_n(\varrho) \in extr(\mathbf{P}^l, \mathbf{P}^u, S, \varrho\downarrow)$ for all $\varrho \in Path\!f^{\mathcal{M}}_{\leq n}$. Then, $\mathcal{E}^{\mathcal{M}} = \bigcap_{n\in\mathbb{N}} \mathcal{E}_n$ where $\mathcal{E}_{n-1} \supseteq \mathcal{E}_n$ for all $n \in \mathbb{N}$.

Further, for $Q \in \mathcal{B}_{\mathcal{M}}$ and $\varrho \in Path\!f^{\mathcal{M}}$, let $Q_\varrho = Q \cap \mathcal{C}(\varrho)$ denote the subset of Q where $\rho \in Q_\varrho$ iff $\rho = \varrho \to \ldots \in Q$; note that $Q = \bigcup_{\varrho\in Path\!f^{\mathcal{M}}_{=n}} Q_\varrho$ for all $n \in \mathbb{N}$ and $\Pr^D(Q_\varrho)/\Pr^D(\mathcal{C}(\varrho))$ is the conditional probability under scheduler D for paths with a given prefix $\varrho \in Path\!f^{\mathcal{M}}$ being in Q. Hence, for any $n \in \mathbb{N}$ we have:

$$\Pr^D(Q) = \textstyle\sum_{\varrho\in Path\!f^{\mathcal{M}}_{=n}} \mu_0(\varrho[0]) \cdot \prod_{i=0}^{n-1} D(\varrho[0..i])(\varrho[i{+}1]) \cdot \frac{\Pr^D(Q_\varrho)}{\Pr^D(\mathcal{C}(\varrho))}$$

By contraposition, we now show that for all $D \in \mathcal{D}^{\mathcal{M}}$, there exists $E \in \mathcal{E}^{\mathcal{M}}$ such that $Pr^E(Q) \leq Pr^D(Q)$. Assume that for some $D \in \mathcal{D}^{\mathcal{M}}$, it holds $\Pr^E(Q) > \Pr^D(Q)$ for all $E \in \mathcal{E}^{\mathcal{M}}$. Hence, $D \in \mathcal{D}^{\mathcal{M}} \setminus \mathcal{E}^{\mathcal{M}} = \mathcal{D}^{\mathcal{M}} \setminus (\bigcap_{n\in\mathbb{N}} \mathcal{E}_n)$, i.e. there exists a smallest $n \in \mathbb{N}$, such that $D \notin \mathcal{E}_n$ and $D^{\mathcal{M}} \in \mathcal{E}_{n-1}$.

We now define $E_n \in \mathcal{E}_n$ based on $D \in \mathcal{E}_{n-1}$. Let $E_n(\varrho) = D(\varrho)$ for all $\varrho \in Path\!f^{\mathcal{M}}_{\neq n}$ and $E_n(\varrho) = \min^{\mathcal{M}}(\varrho\downarrow, \nu)$ for all $\varrho \in Path\!f^{\mathcal{M}}_{=n}$ and for

$\nu : S \to [0,1]$ with $\nu(u) = \frac{\Pr^D(Q_{\varrho u})}{\Pr^D(\mathcal{C}(\varrho u))}$ for all $u \in S$. From Lemma 4.1.2 it follows that for all $\varrho \in Path^{\mathcal{M}}_{=n}$:

$$\sum_{i\in\mathbb{N}} E_n(\varrho)(u_i) \cdot \frac{\Pr^D(Q_{\varrho u_i})}{\Pr^D(\mathcal{C}(\varrho u_i))} \leq \sum_{i\in\mathbb{N}} D(\varrho)(u_i) \cdot \frac{\Pr^D(Q_{\varrho u_i})}{\Pr^D(\mathcal{C}(\varrho u_i))}$$

With $Q = \bigcup_{\varrho\in Path^{\mathcal{M}}_{=n}} Q_\varrho$ for all $n \in \mathbb{N}$ and with $\varrho, \varrho' \in Path^{\mathcal{M}}_{=n}$ and $\varrho \neq \varrho'$ implies $Q_\varrho \cap Q_{\varrho'} = \varnothing$, we obtain:

$$\begin{aligned}
&\Pr^{E_n}(Q)\\
&= \textstyle\sum_{\varrho\in Path^{\mathcal{M}}_{=n}} \mu_0(\varrho[0]) \cdot \prod_{i=0}^{n-1} E_n(\varrho[0..i])(\varrho[i{+}1]) \cdot \frac{\Pr^{E_n}(Q_\varrho)}{\Pr^{E_n}(\mathcal{C}(\varrho))}\\
&= \textstyle\sum_{\varrho\in Path^{\mathcal{M}}_{=n}} \mu_0(\varrho[0]) \cdot \prod_{i=0}^{n-1} E_n(\varrho[0..i])(\varrho[i{+}1]) \cdot \sum_{i\in\mathbb{N}} E_n(\varrho)(u_i) \frac{\Pr^{E_n}(Q_{\varrho u_i})}{\Pr^{E_n}(\mathcal{C}(\varrho u_i))}\\
&= \textstyle\sum_{\varrho\in Path^{\mathcal{M}}_{=n}} \mu_0(\varrho[0]) \cdot \prod_{i=0}^{n-1} D(\varrho[0..i])(\varrho[i{+}1]) \cdot \sum_{i\in\mathbb{N}} E_n(\varrho)(u_i) \frac{\Pr^{D}(Q_{\varrho u_i})}{\Pr^{D}(\mathcal{C}(\varrho u_i))}\\
&\leq \textstyle\sum_{\varrho\in Path^{\mathcal{M}}_{=n}} \mu_0(\varrho[0]) \cdot \prod_{i=0}^{n-1} D(\varrho[0..i])(\varrho[i{+}1]) \cdot \sum_{i\in\mathbb{N}} D(\varrho)(u_i) \frac{\Pr^{D}(Q_{\varrho u_i})}{\Pr^{D}(\mathcal{C}(\varrho u_i))}\\
&= \textstyle\sum_{\varrho\in Path^{\mathcal{M}}_{=n}} \mu_0(\varrho[0]) \cdot \prod_{i=0}^{n-1} D(\varrho[0..i])(\varrho[i{+}1]) \cdot \frac{\Pr^{D}(Q_{\varrho})}{\Pr^{D}(\mathcal{C}(\varrho))}\\
&= \Pr^D(Q)
\end{aligned}$$

Hence, there does not exist a smallest $n \in \mathbb{N}$ such that for all $E_n \in \mathcal{E}_n$ it holds $\Pr^{E_n}(Q) > \Pr^D(Q)$. Thus we come to the conclusion that for any $D \in \mathcal{D}^{\mathcal{M}}$ there exists $E \in \mathcal{E}^{\mathcal{M}}$ such that $\Pr^E(Q) \leq \Pr^D(Q)$ and, in the discrete-time setting, the theorem holds.

Continuous-time case:
Before proving the theorem in the continuous-time setting we have to discuss how probabilities of general events are measured in CTMDP-like models. The basic idea is to define measures inductively over so-called *combined transitions*, measurable sets of delays and successor states. In the following, we provide the basic definitions that are necessary for this proof; for a more complete introduction, we refer to [WJ06].

Formally, in a CTMDP or an ACTMC, a *combined transition* $\xrightarrow{[a,b]} U$ with respect to a history-dependent deterministic scheduler D is a measurable

rectangle consisting of a time interval $[a, b] \subseteq \mathbb{R}_{\geq 0}$ and successor states $U \subseteq S$ where its measure is given by:

$$p_D(\varsigma, \xrightarrow{[a,b]} U) = \int_{\mathbb{R}_{\geq 0}} \lambda e^{-\lambda t} dt \cdot \int_S \mathbf{1}(t \in [a,b]) \cdot \mathbf{1}(s \in U) \cdot D(\varsigma_{abs})(ds)$$

Intuitively, $p_D(\varsigma, \xrightarrow{[a,b]} U)$ is the probability w.r.t. scheduler D and for the time-abstract history $\varsigma_{abs} \in Path^{\mathcal{M}}_{abs}$, to take a transition within $t \in [a, b]$ time units to a successor $u \in U$. To derive this probability, a Lebesgue-integral is taken over all possible delays $t \in \mathbb{R}_{\geq 0}$ in the current state and all possible successor states $s \in S$; combinations of delay and successor state that are not in $\xrightarrow{[a,b]} U$ do not contribute to the probability measure due to the indicator functions. Note that by disallowing randomized schedulers, we save ourselves the necessity to integrate over the schedulers randomized choices as done in [WJ06].

We only consider untimed schedulers, therefore the integral over delays t and the one over successor states are independent of each other. Further, the integral over successors is only over a countable set, thus it is merely a countable sum. With these arguments, we derive:

$$\begin{aligned} p_D(\varsigma, \xrightarrow{[a,b]} U) &= \int_{\mathbb{R}_{\geq 0}} \lambda e^{-\lambda t} dt \cdot \left(\int_S \mathbf{1}(t \in [a,b]) \cdot \mathbf{1}(s \in U) \cdot D(\varsigma_{abs})(ds) \right) \\ &= \int_{\mathbb{R}_{\geq 0}} \mathbf{1}(t \in [a,b]) \cdot \lambda e^{-\lambda t} dt \cdot \left(\int_S \mathbf{1}(s \in U) \cdot D(\varsigma_{abs})(ds) \right) \\ &= \int_a^b \lambda e^{-\lambda t} dt \cdot \left(\sum_{u \in U} D(\varsigma_{abs})(u) \right) \\ &= \sum_{u \in U} D(\varsigma_{abs})(u) \cdot \int_a^b \lambda e^{-\lambda t} dt \end{aligned}$$

We lift p_D to arbitrary measurable rectangles (sets of combined transitions) $R \in \mathbb{R}_{\geq 0} \times S$. Let $R^s = \{t \in \mathbb{R}_{\geq 0} \mid (\xrightarrow{t} s) \in R\}$ for all $s \in S$, then

$$p_D(\varsigma, R) = \sum_{s \in S} D(\varsigma_{abs})(s) \cdot \int_{R^s} \lambda e^{-\lambda t} dt$$

Note that this definition is consistent with the definition for combined transitions $\xrightarrow{[a,b]} U$. For $R = \xrightarrow{[a,b]} U$ and $s \notin U$ it holds $R^s = \varnothing$ and thus $\int_{R^s} \lambda e^{-\lambda t} dt = 0$. Hence, for $R = \xrightarrow{[a,b]} U$ we obtain:

$$p_D(\varsigma, R) = \sum_{s \in S} D(\varsigma_{abs})(s) \cdot \int_{R^s} \lambda e^{-\lambda t} dt = \sum_{s \in U} D(\varsigma_{abs})(s) \cdot \int_a^b \lambda e^{-\lambda t} dt$$

Now, let us proceed with discussing the inductive definition of a probability measure $\Pr_D : \mathcal{B}_{\mathcal{M}} \to [0,1]$ for fixed initial distribution μ_0 of CTMDP (or ACTMC) $\mathcal{M}$ and deterministic history-dependent scheduler $D \in \mathcal{D}^{\mathcal{M}}$. The inductive definition is over the length of paths, where in each step, sets Q_m of timed paths of length m are extended by combined transitions (measurable rectangles) to sets Q_{m+1} of timed paths of length $m+1$:

$$\begin{aligned}\Pr_D^0(Q_0) &= \textstyle\sum_{s \in Q_0} \mu_0(s)\\ \Pr_D^{m+1}(Q_{m+1}) &= \textstyle\int_{Pathf^{\mathcal{M}}_{=m}} \Pr_D^m(d\varsigma) \cdot \int_{\mathbb{R}_{\geq 0} \times S} \mathbf{1}(\varsigma \xrightarrow{t} s \in Q_{m+1}) \cdot p_D(\varsigma, d(\xrightarrow{t} s))\end{aligned}$$

By the Ionescu-Tulcea theorem, the measure is then uniquely lifted from finite paths of unbounded length to infinite ones [ADD00, Theorem 2.7.2]. We denote the resulting measure as $\Pr_D$.

We now derive an alternative definition for $\Pr_D^{m+1}(Q_{m+1})$. Let $m \in \mathbb{N}$, $Q_0 \subseteq S$ and for all $i \in \{0, \ldots, m\}$, let $Q_{i+1} = Q_i \times R_{i+1}$ with measurable rectangles $R_{i+1} \subseteq \mathbb{R}_{\geq 0} \times S$ (that is, $\varsigma \xrightarrow{t} s \in Q_{i+1} \implies \varsigma \in Q_i$), then we compute

$$\begin{aligned}&\Pr_D^{m+1}(Q_{m+1})\\ &= \textstyle\int_{Pathf^{\mathcal{M}}_{=m+1}} \Pr_D^{m+1}(d\varsigma) \cdot \mathbf{1}(\varsigma \in Q_{m+1})\\ &= \textstyle\int_{Pathf^{\mathcal{M}}_{=m}} \Pr_D^{m}(d\varsigma) \cdot \Big(\int_{\mathbb{R}_{\geq 0} \times S} \mathbf{1}(\varsigma \xrightarrow{t} s \in Q_{m+1}) \cdot p_D(\varsigma, d(\xrightarrow{t} s))\Big)\\ &= \textstyle\int_{Pathf^{\mathcal{M}}_{=m}} \Pr_D^{m}(d\varsigma) \cdot \mathbf{1}(\varsigma \in Q_m) \cdot \Big(\int_{R_{m+1}} p_D(\varsigma, d(\xrightarrow{t} s))\Big)\\ &= \textstyle\int_{Pathf^{\mathcal{M}}_{=m}} \Pr_D^{m}(d\varsigma) \cdot \mathbf{1}(\varsigma \in Q_m) \cdot \Big(\sum_{s_{m+1} \in S} D(\varsigma_{abs})(s_{m+1}) \cdot \int_{R_{m+1}^{s_{m+1}}} \lambda e^{-\lambda t} dt\Big)\\ &= \textstyle\int_{Pathf^{\mathcal{M}}_{=0}} \Pr_D^{0}(d\varsigma) \cdot \mathbf{1}(\varsigma \in Q_0) \cdot \Big(\sum_{s_1 \in S} D(\varsigma_{abs})(s_1) \cdot \int_{R_1^{s_1}} \lambda e^{-\lambda t} dt\\ &\quad \cdot \ \ldots \ \cdot \Big(\sum_{s_{m+1} \in S} D(\varsigma_{abs}\, s_m(s_{m+1}) \cdot \int_{R_{m+1}^{s_{m+1}}} \lambda e^{-\lambda t} dt\Big) \ldots\Big)\\ &= \textstyle\sum_{s_0 \in Q_0} \mu_0(s_0) \cdot \Big(\sum_{s_1 \in S} D(s_0)(s_1) \cdot \int_{R_1^{s_1}} \lambda e^{-\lambda t} dt\\ &\quad \cdot \ \ldots \ \cdot \Big(\sum_{s_{m+1} \in S} D(s_0\, s_1\, \ldots\, s_m)(s_{m+1}) \cdot \int_{R_{m+1}^{s_{m+1}}} \lambda e^{-\lambda t} dt\Big) \ldots\Big)\\ &= \textstyle\sum_{s_0 \in Q_0} \mu_0(s_0) \cdot \Big(\sum_{s_1 \in S} D(s_0)(s_1) \cdot \ \ldots \ \cdot \Big(\sum_{s_{m+1} \in S} D(s_0\, \ldots\, s_m)(s_{m+1})\\ &\qquad\qquad \cdot \int_{R_1^{s_1}} \lambda e^{-\lambda t} dt \cdot \ \ldots \ \cdot \int_{R_{m+1}^{s_{m+1}}} \lambda e^{-\lambda t} dt\Big) \ldots\Big)\\ &= \textstyle\sum_{\varrho \in Pathf^{abs}_{=m+1}} \mu_0(\varrho[0]) \cdot \prod_{i=0}^{m} D(\varrho[0..i])(\varrho[i+1]) \cdot \prod_{i=0}^{m} \int_{R_{i+1}^{\varrho[i+1]}} \lambda e^{-\lambda t} dt\end{aligned}$$

where $Pathf^{abs}$ is shorthand for $Pathf^{\mathcal{M}}_{abs}$, the set of time-abstract paths in $\mathcal{M}$. With this definition, we now show that for all $m, n \in \mathbb{N}$ with $l = \min(m, n)$:

$$\Pr^m_D(Q_m) = \sum_{\varrho' \in Pathf^{abs}_{=l}} \mu_0(\varrho'[0]) \cdot \prod_{i=0}^{l-1} D(\varrho'[0..i])(\varrho'[i{+}1]) \cdot \frac{\Pr^m_D(Q_{\varrho'})}{\Pr^m_D(\mathcal{C}(\varrho'))}$$

We distinguish two cases: for $l = m < n \in \mathbb{N}$, by definition it follows directly

$$\Pr^m_D(Q_m) = \sum_{\varrho' \in Pathf^{abs}_{=m}} \mu_0(\varrho'[0]) \cdot \prod_{i=0}^{m-1} D(\varrho'[0..i])(\varrho'[i{+}1]) \cdot \frac{\Pr^m_D(Q_{\varrho'})}{\Pr^m_D(\mathcal{C}(\varrho'))}$$

and for $m \geq n = l \in \mathbb{N}$ we compute:

$$\begin{aligned}
&\Pr^m_D(Q_m) \\
&= \textstyle\sum_{\varrho \in Pathf^{abs}_{=m}} \mu_0(\varrho[0]) \prod_{i=0}^{m-1} D(\varrho[0..i])(\varrho[i{+}1]) \prod_{i=0}^{m-1} \int_{R^{\varrho[i+1]}_{i+1}} \lambda e^{-\lambda t} dt \\
&= \textstyle\sum_{\varrho \in Pathf^{abs}_{=m}} \mu_0(\varrho[0]) \prod_{i=0}^{n-1} D(\varrho[0..i])(\varrho[i{+}1]) \\
&\quad \textstyle\prod_{i=n}^{m-1} D(\varrho[0..i])(\varrho[i{+}1]) \prod_{i=0}^{m-1} \int_{R^{\varrho[i+1]}_{i+1}} \lambda e^{-\lambda t} dt \\
&= \textstyle\sum_{\varrho' \in Pathf^{abs}_{=n}} \mu_0(\varrho'[0]) \prod_{i=0}^{n-1} D(\varrho'[0..i])(\varrho'[i{+}1]) \\
&\quad \textstyle\sum_{\varrho \in Pathf^{abs}_{=m} : \varrho = \varrho' \cdots} \prod_{i=n}^{m-1} D(\varrho[0..i])(\varrho[i{+}1]) \prod_{i=0}^{m-1} \int_{R^{\varrho[i+1]}_{i+1}} \lambda e^{-\lambda t} dt \\
&= \textstyle\sum_{\varrho' \in Pathf^{abs}_{=n}} \mu_0(\varrho'[0]) \prod_{i=0}^{n-1} D(\varrho'[0..i])(\varrho'[i{+}1]) \\
&\quad \textstyle\left(\mu_0(\varrho'[0]) \prod_{i=0}^{n-1} D(\varrho'[0..i])(\varrho'[i{+}1]) \middle/ \mu_0(\varrho'[0]) \prod_{i=0}^{n-1} D(\varrho'[0..i])(\varrho'[i{+}1])\right) \\
&\quad \textstyle\sum_{\varrho \in Pathf^{abs}_{=m} : \varrho = \varrho' \cdots} \prod_{i=n}^{m-1} D(\varrho[0..i])(\varrho[i{+}1]) \prod_{i=0}^{m-1} \int_{R^{\varrho[i+1]}_{i+1}} \lambda e^{-\lambda t} dt \\
&= \textstyle\sum_{\varrho' \in Pathf^{abs}_{=n}} \mu_0(\varrho'[0]) \prod_{i=0}^{n-1} D(\varrho'[0..i])(\varrho'[i{+}1]) \\
&\quad \textstyle\left(\sum_{\varrho \in Pathf^{abs}_{=m} : \varrho = \varrho' \cdots} \mu_0(\varrho[0]) \prod_{i=0}^{m-1} D(\varrho[0..i])(\varrho[i{+}1]) \prod_{i=0}^{m-1} \int_{R^{\varrho[i+1]}_{i+1}} \lambda e^{-\lambda t} dt\right) \\
&\quad \textstyle\Big/ \left(\mu_0(\varrho'[0]) \prod_{i=0}^{n-1} D(\varrho'[0..i])(\varrho'[i{+}1])\right)\Big) \\
&= \textstyle\sum_{\varrho' \in Pathf^{abs}_{=n}} \mu_0(\varrho'[0]) \prod_{i=0}^{n-1} D(\varrho'[0..i])(\varrho'[i{+}1]) \frac{\Pr^m_D(Q_{\varrho'})}{\Pr^m_D(\mathcal{C}(\varrho'))}
\end{aligned}$$

For any $n \in \mathbb{N}$, with the Ionescu-Tulcea theorem and with $l = \min(\infty, n) = n$, we obtain that

$$\Pr_D(Q) = \sum_{\varrho' \in Pathf^{abs}_{=n}} \mu_0(\varrho'[0]) \cdot \prod_{i=0}^{n-1} D(\varrho'[0..i])(\varrho'[i{+}1]) \cdot \frac{\Pr_D(Q_{\varrho'})}{\Pr_D(\mathcal{C}(\varrho'))}$$

holds, just as in the discrete-time setting.

Now that we have recalled the inductive definition of probability measures of general events in the continuous-time setting and tailored it for deterministic history-dependent schedulers, the proof of the theorem can be shown, following the same lines as in the discrete-time setting, i.e., it can be shown by contraposition that for all $D \in \mathcal{D}^{\mathcal{M}}$, there exists $E \in \mathcal{E}^{\mathcal{M}}$ such that $\Pr_E(Q) \leq \Pr_D(Q)$ (cf. p. 75 f.). □

By Theorem 4.1.3, it suffices to consider extreme schedulers for the infimum/supremum of the probability of measurable sets in AMCs.

4.2 Step-bounded reachability

This section focuses on step-bounded reachability probabilities, i.e. the probability to reach a state in the set of goal states B within a fixed number of steps i. For $n, i \in \mathbb{N}$, we define:

$$\begin{aligned} \mathit{Reach}^{=n}(B) &= \{\varsigma \in \mathit{Path}^{\mathcal{M}} \mid \varsigma[n] \in B \text{ and for all } k < n, \varsigma[k] \notin B\} \\ \mathit{Reach}^{\leq i}(B) &= \textstyle\bigcup_{n=0}^{i} \mathit{Reach}^{=n}(B) \end{aligned}$$

Let $\mathcal{M}'$ be an AMC with state space S' and assume $\mathcal{M} \preceq \mathcal{M}'$, i.e., there exists a probabilistic simulation $R \subseteq S \times S'$ satisfying the conditions in Definition 3.4.2. The sets $B \subseteq S$ and $B' \subseteq S'$ are called *compatible* (w.r.t. R) iff for all $s \in S$, $s' \in S'$ we have sRs' implies $s \in B$ iff $s' \in B'$.

We are interested in the relationship between the reachability probabilities $\mathit{Reach}^{\leq i}(B)$ in $\mathcal{M}$ and $\mathit{Reach}^{\leq i}(B')$ in $\mathcal{M}'$. A time-abstract path fragment $s_0 \to \ldots \to s_n$ in $\mathcal{M}$ with $s_n \in B$ can be mimicked in a stepwise manner by a time-abstract path fragment $s'_0 \to \ldots \to s'_n$ in $\mathcal{M}'$ if $s_i R s'_i$ for all $0 \leq i \leq n$ and $s'_n \in B'$. Since $s_i R s'_i$ implies that any distribution in $\mathbf{T}(s_i)$ has a matching distribution in $\mathbf{T}(s'_i)$, it follows that for each scheduler $D \in \mathcal{D}^{\mathcal{M}}$ we can construct a scheduler $D' \in \mathcal{D}^{\mathcal{M}'}$ such that the probabilities of the corresponding cylinder sets agree:

$$Pr^{D}_{s_0}(\mathcal{C}(s_0 \ldots s_n)) = Pr^{D'}_{s'_0}(\mathcal{C}(s'_0 \ldots s'_n)).$$

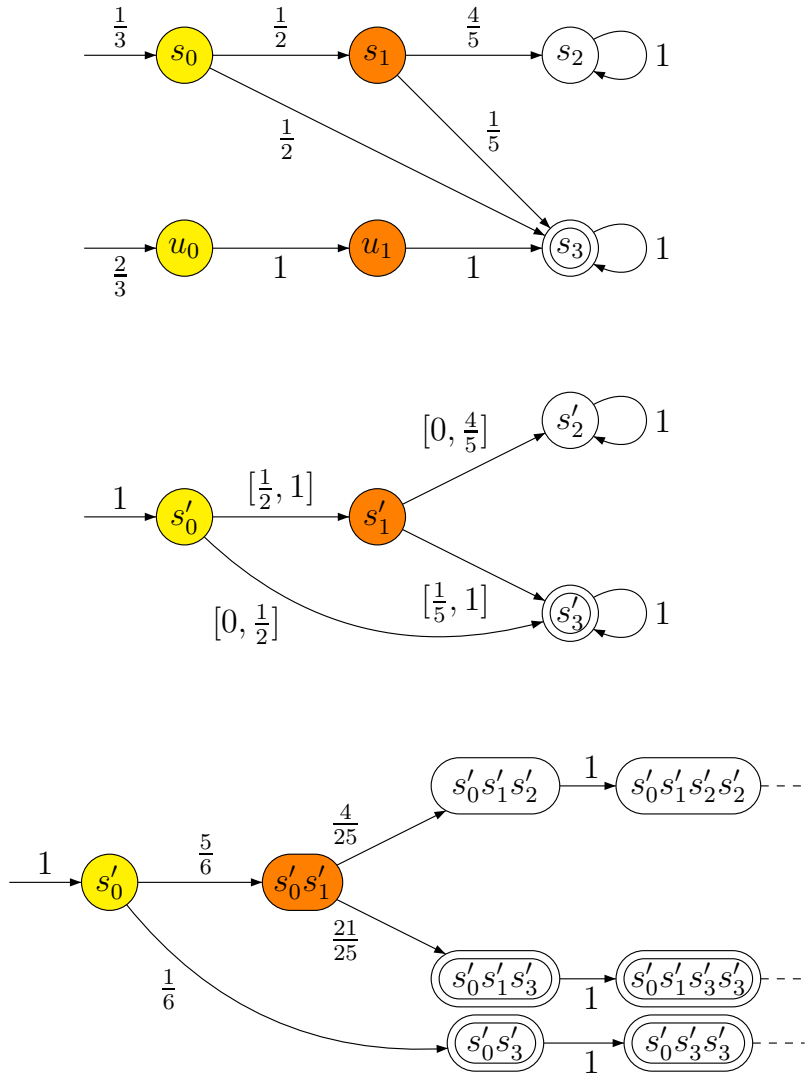


Figure 4.5: An example Markov chain $\mathcal{M}$ (top), one of its abstractions $\mathcal{M}' = \alpha(\mathcal{M})$ (middle) and the Markov chain $\mathcal{M}'_D$ (bottom) induced by a scheduler simulating $\mathcal{M}$.

A similar idea is used for the events $Reach^{\leq i}(B)$ and $Reach^{\leq i}(B')$ and initial distributions μ_0 and μ'_0 (rather than s_0 and s'_0). The main principle of this construction is illustrated by the following example.

Example 4.2.1. Consider the Markov chain $\mathcal{M}$ in Figure 4.5 (top) and its abstraction $\mathcal{M}' = \alpha(\mathcal{M})$ (middle) where $\gamma(s'_0) = \{s_0, u_0\}$, $\gamma(s'_1) = \{s_1, u_1\}$ and $\gamma(s'_i) = \{s_i\}$ for $i \in \{2, 3\}$. The aim is to obtain a scheduler D on $\mathcal{M}'$ that mimics the behavior of $\mathcal{M}$. The idea here is to construct D such that sets of paths in the induced Markov chain $\mathcal{M}'_D$ have the same probability as their counterparts in $\mathcal{M}$. This requires the computation of the conditional probabilities:

$$\mathbf{P}'_D(s'_0, s'_0s'_1) = \tfrac{\frac{1}{3}\cdot\frac{1}{2}+\frac{2}{3}\cdot 1}{\frac{1}{3}+\frac{2}{3}} = \tfrac{5}{6} \qquad \mathbf{P}'_D(s'_0, s'_0s'_3) = \tfrac{\frac{1}{3}\cdot\frac{1}{2}}{\frac{1}{3}+\frac{2}{3}} = \tfrac{1}{6}$$

$$\mathbf{P}'_D(s'_0s'_1, s'_0s'_1s'_2) = \tfrac{\frac{1}{3}\cdot\frac{1}{2}\cdot\frac{4}{5}}{\frac{1}{3}\cdot\frac{1}{2}+\frac{2}{3}\cdot 1} = \tfrac{4}{25} \qquad \mathbf{P}'_D(s'_0s'_1, s'_0s'_1s'_3) = \tfrac{\frac{1}{3}\cdot\frac{1}{2}\cdot\frac{1}{5}+\frac{2}{3}\cdot 1\cdot 1}{\frac{1}{3}\cdot\frac{1}{2}+\frac{2}{3}\cdot 1} = \tfrac{21}{25}$$

For example, the probability to move from s'_0 to s'_1 in $\mathcal{M}'$ such that the probability is matched to move from s_0 and u_0 to s_1 or u_1 in $\mathcal{M}$ is calculated by summing up the probabilities to reach s_1 or u_1 in $\mathcal{M}$ and dividing by the probability to reach one of the corresponding predecessors in $\mathcal{M}$, s_0 or u_0.

The scheduler D we aim at is the one that induces the Markov chain shown in Figure 4.5 (bottom).

Theorem 4.2.1. *Let $\mathcal{M}$ and $\mathcal{M}'$ be AMCs with $\mathcal{M} \preceq \mathcal{M}'$. For any compatible sets B, B', $i \in \mathbb{N}$ and $D \in \mathcal{D}^{\mathcal{M}}$ there exists $D' \in \mathcal{D}^{\mathcal{M}'}$ with*

$$Pr^D(Reach^{\leq i}(B)) = Pr^{D'}(Reach^{\leq i}(B')).$$

Proof. Before proving our results, we first fix some notation. Let S and S' be the state spaces, and μ_0 and μ'_0 the initial distributions of $\mathcal{M}$ and $\mathcal{M}'$, respectively. Let R be the probabilistic simulation that relates $\mathcal{M}$ and $\mathcal{M}'$.

The main part of the proof is an induction over i that proves that for every $D \in \mathcal{D}^{\mathcal{M}}$ and for all $s \in S$, $s' \in S'$ with sRs' there exists $D' \in \mathcal{D}^{\mathcal{M}'}$ such that

$$\Pr_s^D(Reach^{\leq i}(B)) = \Pr_{s'}^{D'}(Reach^{\leq i}(B')). \tag{4.1}$$

Note that D' depends on s, s' and D. Thus, our induction hypothesis is that for every triple (s, s', D) there is a D' such that (4.1) holds. In the last part of the proof we show that from this the theorem follows.

We abbreviate $\Pr_{\hat{s}}^{\hat{D}}(\mathit{Reach}^{\leq i}(\hat{B}))$ by $\Pr_{\hat{s}}^{\hat{D}}(i, \hat{B})$ for any $\hat{D}$, $\hat{s}$, $\hat{B}$ and prove Equation (4.1) as follows: For the induction basis, first assume that $s \in B$. This implies $s' \in B'$ and therefore $\Pr_s^D(0, B) = \Pr_{s'}^{D'}(0, B') = 1$ for all D, D'. If $s \notin B$ we have $s' \notin B'$ and therefore $\Pr_s^D(0, B) = \Pr_{s'}^{D'}(0, B') = 0$ for all $D, D' \in \mathcal{D}^{\mathcal{M}}$.

For the induction step, we assume that s, s' with sRs' and $D \in \mathcal{D}^{\mathcal{M}}$ are given and we have to construct D' such that $\Pr_s^D(i+1, B) = \Pr_{s'}^{D'}(i+1, B')$. We define D' inductively over the length of path fragments. Assume that $D(s) = \mu$. As sRs' there exists a function Δ as in Def. 3.4.2. We define $D'(s') = \mu'$ where $\mu'(u') = \Delta(S, u')$ for all $u' \in S'$.

Let D_u be the scheduler such that $D_u(\varrho) = D(s \to \varrho)$ for all $\varrho \in \mathit{Path}f_{abs}^{\mathcal{M}}$ with $\varrho[0] = u$. From the induction hypothesis, for D_u, $u, u' \in S$ with uRu' there is a $D_{u,u'} \in \mathcal{D}^{\mathcal{M}'}$ such that $\Pr_u^{D_u}(i, B) = \Pr_{u'}^{D_{u,u'}}(i, B')$. If not uRu' let $D_{u,u'}$ be an arbitrary scheduler. For $\varrho' \in \mathit{Path}f_{abs}^{\mathcal{M}'}$ we define

$$D'(s' \to \varrho') = \textstyle\sum_{u \in S} \frac{\Delta(u,u')}{\Delta(S,u')} D_{u,u'}(\varrho')$$

where $u' = \varrho'[0]$. Let all choices of D' for $\varrho' \in \mathit{Path}f_{abs}^{\mathcal{M}'}$ with $\varrho'[0] \neq s'$ be arbitrary. Note that $\Delta(u, u') > 0$ implies uRu'. Moreover, by Lemma 4.1.1, a convex combination of schedulers is a scheduler and therefore D' is an element of $\mathcal{D}^{\mathcal{M}'}$. We calculate

$$\begin{aligned}
&\Pr_s^D(i+1, B) \\
&= \textstyle\sum_{u \in S} D(s)(u) \cdot \Pr_u^{D_u}(i, B) \\
&= \textstyle\sum_{u \in S} \mu(u) \cdot \Pr_u^{D_u}(i, B) \\
&= \textstyle\sum_{u \in S} \Delta(u, S') \cdot \Pr_u^{D_u}(i, B) && \text{(Def. 3.4.2)} \\
&= \textstyle\sum_{u' \in S'} \sum_{u \in S} \Delta(u, u') \cdot \Pr_u^{D_u}(i, B) \\
&= \textstyle\sum_{u' \in S'} \sum_{u \in S} \Delta(u, u') \cdot \Pr_{u'}^{D_{u,u'}}(i, B') && \text{(ind. hyp.)}
\end{aligned}$$

$$
\begin{aligned}
&= \textstyle\sum_{u' \in S'} \Delta(S,u') \sum_{u \in S} \frac{\Delta(u,u')}{\Delta(S,u')} \cdot \Pr_{u'}^{D_{u,u'}}(i, B') \\
&= \textstyle\sum_{u' \in S'} D'(s')(u') \sum_{u \in S} \frac{\Delta(u,u')}{\Delta(S,u')} \cdot \Pr_{u'}^{D_{u,u'}}(i, B') \\
&= \textstyle\sum_{u' \in S} \mu'(u') \sum_{u \in S} \frac{\Delta(u,u')}{\Delta(S,u')} \cdot \Pr_{u'}^{D_{u,u'}}(i, B') \\
&= \Pr_{s'}^{D'}(i+1, B')
\end{aligned}
$$

In the above calculation, we used the fact that if $\mathcal{M}$ and $\mathcal{M}'$ are ACTMCs the residence times in the states can be arbitrary. Therefore, for the first and last equality we do not have an extra factor for the probability to remain for a certain time in s or s'. Moreover, in the last equality we exploit the definition of D' and the law of total probability, i.e., we consider the conditional probability of reaching B' from s' via state u' and sum over all possible states u'.

This completes the proof of Equation (4.1). To show our claim, we generalize this result to initial distributions as follows. For a fixed scheduler $D \in \mathcal{D}^{\mathcal{M}}$ we can construct $\tilde{D} \in \mathcal{D}^{\mathcal{M}'}$ as the weighted sum of the schedulers $D'_{s,s'} \in \mathcal{D}^{\mathcal{M}'}$, where $D'_{s,s'}$ corresponds to the triple (s, s', D). Let Δ be the weight function that relates μ_0 and μ'_0 (cf. Def. 3.4.1). Then for $\varrho' \in Path_{abs}^{\mathcal{M}'}$ with $\varrho'[0] = s'$ we define

$$
\tilde{D}(\varrho') = \textstyle\sum_{s \in S} \frac{\Delta(s,s')}{\Delta(S,s')} \cdot D'_{s,s'}(\varrho')
$$

which directly implies that for all $i \geq 0$:

$$
\begin{aligned}
&\Pr^{D}(Reach^{\leq i}(B)) \\
&= \textstyle\sum_{s \in S} \mu_0(s) \cdot \Pr_s^D(Reach^{\leq i}(B)) \\
&= \textstyle\sum_{s \in S} \Delta(s,S') \cdot \Pr_s^D(Reach^{\leq i}(B)) \\
&= \textstyle\sum_{s' \in S'} \sum_{s \in S} \Delta(s,s') \cdot \Pr_s^D(Reach^{\leq i}(B)) \\
&= \textstyle\sum_{s' \in S'} \sum_{s \in S} \Delta(s,s') \cdot \Pr_{s'}^{D'_{s,s'}}(Reach^{\leq i}(B')) \\
&= \textstyle\sum_{s' \in S'} \Delta(S,s') \cdot \sum_{s \in S} \frac{\Delta(s,s')}{\Delta(S,s')} \cdot \Pr_{s'}^{D'_{s,s'}}(Reach^{\leq i}(B')) \\
&= \textstyle\sum_{s' \in S'} \mu'_0(s') \cdot \Pr_{s'}^{\tilde{D}}(Reach^{\leq i}(B')) \\
&= \Pr^{\tilde{D}}(Reach^{\leq i}(B')).
\end{aligned}
$$

□

For an AMC $\mathcal{M}$, we define the abbreviations

$$Pr^{\mathcal{M}}_{\inf}(i, B) = \inf_{D \in \mathcal{D}^{\mathcal{M}}} Pr^{D}(Reach^{\leq i}(B))$$
$$Pr^{\mathcal{M}}_{\sup}(i, B) = \sup_{D \in \mathcal{D}^{\mathcal{M}}} Pr^{D}(Reach^{\leq i}(B)).$$

The following corollary provides the key argument when showing the preservation of PCTL formulas for abstraction-based model checking in Section 4.5.

Corollary 4.2.2. *Let $\mathcal{M}$ and $\mathcal{M}'$ be AMCs with $\mathcal{M} \preceq \mathcal{M}'$ and let $i \in \mathbb{N}$ and B, B' be compatible. Then*

$$Pr^{\mathcal{M}'}_{\inf}(i, B') \leq Pr^{\mathcal{M}}_{\inf}(i, B) \leq Pr^{\mathcal{M}}_{\sup}(i, B) \leq Pr^{\mathcal{M}'}_{\sup}(i, B').$$

Proof. The claim follows directly from Theorem 4.2.1. □

This implies that our abstraction is conservative w.r.t. the minimal (maximal) probability to reach a set of states within i steps in the concrete model. More precisely, if in the abstract model the probability to reach B' within i steps is at least (at most) p, the probability to reach B within i steps in the concrete model is at least (at most) p as well.

Computation of step-bounded reachabilities. For the *computation* of $Pr^{\mathcal{M}}_{\inf}(i, B)$, it is sufficient to focus on ADTMCs. To see this, let $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0, \lambda)$ be an ACTMC and $\hat{\mathcal{M}} = emb(\mathcal{M}) = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0)$, then $\mathcal{D}^{\mathcal{M}} = \mathcal{D}^{\hat{\mathcal{M}}}$. For all $D \in \mathcal{D}^{\mathcal{M}}$ and all sets $B \subseteq S$,

$$Pr^{\mathcal{M}_D}(Reach^{\leq i}(B)) = Pr^{\hat{\mathcal{M}}_D}(Reach^{\leq i}(B)). \tag{4.2}$$

Further, for the computation of reachability probabilities, we assume that the ADTMCs have finite state spaces.

By Theorem 4.1.3, it is sufficient to consider extreme schedulers for the calculation of $Pr^{\mathcal{M}}_{\inf}(i, B)$ and $Pr^{\mathcal{M}}_{\sup}(i, B)$. As already stated above, with each ADTMC $\mathcal{M}$ we can associate an MDP that represents the behavior of $\mathcal{M}$ w.r.t. extreme schedulers. However, the cardinality of the action set of the

resulting MDP may be exponential in the size of the state space. Thus, in the following, we outline an algorithm for the calculation of minimal reachability probabilities in a finite-state, delimited ADTMC $\mathcal{M}$ exploiting the special structure of the model. The calculation of maximal reachability probabilities can be carried out in a similar way. A fixpoint characterization based on the same idea has been presented in [FLW06].

Value iteration. For MDPs, *value iteration* [Bel57] is a popular method for the computation of minimal (and maximal) reachability probabilities. The idea is to compute the result in a backwards manner, that is, in order to compute the probability for reaching some B-state within i steps, one utilizes the result for "within $i-1$ steps". More precisely, for MDP $\mathcal{M} = (S, A, \mathbf{P}, L, \mu_0)$ where all $s \in B$ are absorbing, it holds

$$\Pr_{inf}(i, B) = \textstyle\sum_{s_0 \in S} \mu_0(s_0) \cdot p_i^l(s_0)$$

where for all $s \in S$, $i \in \mathbb{N}_{>0}$:

$$\begin{aligned} p_i^l(s) &= \min\{\textstyle\sum_{u \in S} \mu(u) \cdot p_{i-1}^l(u) \mid \mu = \mathbf{P}(s, a, \cdot), a \in A\} \\ p_0^l(s) &= \mathbf{1}(s \in B) \end{aligned}$$

The naive adaptation of value iteration to AMC $\mathcal{M} = (S, A, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0)$ where all $s \in B$ are absorbing yields ($\Pr_{inf}(i, B) = \sum_{s_0 \in S} \mu_0(s_0) \cdot p_i^l(s_0)$ where for all $s \in S$, $i \in \mathbb{N}_{>0}$):

$$\begin{aligned} p_i^l(s) &= \min\{\textstyle\sum_{u \in S} \mu(u) \cdot p_{i-1}^l(u) \mid \mu \in \mathit{extr}(\mathbf{P}^l, \mathbf{P}^u, S, s)\} \\ p_0^l(s) &= \mathbf{1}(s \in B) \end{aligned}$$

Recall that the cardinality of $\mathit{extr}(\mathbf{P}^l, \mathbf{P}^u, S, s)$ is in $\mathcal{O}(2^{|S|})$, cf. Example 4.1.2. We now replace the brute force check on which $\mu \in \mathit{extr}(\mathbf{P}^l, \mathbf{P}^u, S, s)$ yields minimal probabilities by the iteratively computable $\min^{\mathcal{M}}(s, p_{i-1}^l) \in \mathit{extr}(\mathbf{P}^l, \mathbf{P}^u, S, s)$ from Definition 4.1.2. By Lemma 4.1.2, for $\mu = \min^{\mathcal{M}}(s, p_{i-1}^l)$ the sum $\sum_{u \in S} \mu(u) \cdot p_{i-1}^l(u)$ is minimal w.r.t. all extreme distributions in s.

The idea for the scheduler is to assign as much probability as possible to the successor state for which the probability is minimal to reach a goal state (within one step less than required from the current state on); the remaining probability mass is distributed over the remaining states in the same fashion.

For step-bound i and goal states B, we obtain $\Pr_{inf}(i, B) = \sum_{s_0 \in S} \mu_0(s_0) \cdot p_i^l(s_0)$ where for all $s \in S$, $i \in \mathbb{N}_{>0}$:

$$p_i^l(s) = \sum_{u \in S} \min^{\mathcal{M}}(s, p_{i-1}^l)(u) \cdot p_{i-1}^l(u)$$
$$p_0^l(s) = \mathbf{1}(s \in B)$$

Theorem 4.2.3. *For any AMC $\mathcal{M}$ with state space S, step-bound $i \in \mathbb{N}$ and set of goal states $B \subseteq S$, we can compute reachability probabilities $\Pr_{inf}(i, B)$ and $\Pr_{sup}(i, B)$ in time polynomial in the size of $\mathcal{M}$ and linear in the step-bound i.*

Proof. We focus on the proof for $\Pr_{inf}(i, B)$ and consider AMC $\mathcal{M} = (S, A, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0)$ where all $s \in B$ have been made absorbing. The minimization function $\min^{\mathcal{M}}$ has a complexity of $\mathcal{O}(|S|^3)$ as the recursion is of depth $|S|$ and in each iteration a normalization has to be performed, which is in $\mathcal{O}(|S|^2)$. Further, ordering the state space according to p_{i-1}^l can be done in $\mathcal{O}(|S| \cdot \log(|S|))$. Thus, with the polynomial complexity for the standard value iteration algorithm, we obtain an algorithm, polynomial in the size of the state space for the computation of step-bounded reachability probabilities. □

4.3 Time-bounded reachability

Let us now consider the continuous-time setting and focus on time-bounded reachability probabilities. Let $\mathcal{M}$ be an ACTMC with state space S. We are interested in the probability to reach a state in $B \subseteq S$ within $t \in \mathbb{R}_{\geq 0}$ time units. Let

$$Reach_{\leq t}(B) = \{\sigma \in Path^{\mathcal{M}} \mid \sigma@t' \in B \text{ for some } t' \in [0, t]\}.$$

Theorem 4.3.1. *Let $\mathcal{M}$ and $\mathcal{M}'$ be ACTMCs with $\mathcal{M} \preceq \mathcal{M}'$. For compatible sets B, B' of states, $t \in \mathbb{R}_{\geq 0}$ and $D \in \mathcal{D}^{\mathcal{M}}$ there exists $D' \in \mathcal{D}^{\mathcal{M}'}$ with*

$$Pr^{D}(Reach_{\leq t}(B)) = Pr^{D'}(Reach_{\leq t}(B')).$$

Proof. The probability to reach B in exactly $i \in \mathbb{N}$ steps and within $t \in \mathbb{R}_{\geq 0}$ time units is given by

$$Reach^{=i}_{\leq t}(B) = \{s_0 t_0 s_1 t_1 \ldots \in Path^{\mathcal{M}} \mid s_0, s_1, \ldots, s_{i-1} \notin B, \; s_i \in B \text{ and } \textstyle\sum_{j=0}^{i-1} t_j \leq t\}.$$

It is clear that the probability of $Reach^{=i}_{\leq t}(B)$ under the condition that exactly i steps are carried out in $[0, t)$ is given by $\Pr^{D}(Reach_{=i}(B))$. In a uniform (A)CTMC, the number of steps during time interval $[0, t)$ is Poisson distributed and therefore the probability of i steps in $[0, t)$ is given by

$$\psi_{\lambda,t}(i) = \frac{(\lambda \cdot t)^i \cdot e^{-\lambda \cdot t}}{i!}$$

where λ is the common exit rate of $\mathcal{M}$ and $\mathcal{M}'$ (cf. Definition 3.4.2). By the law of total probability,

$$\begin{aligned} \Pr^{D}(Reach_{\leq t}(B)) &= \textstyle\sum_{i=0}^{\infty} \Pr^{D}(Reach^{=i}_{\leq t}(B)) \\ &= \textstyle\sum_{i=0}^{\infty} \psi_{\lambda,t}(i) \cdot \Pr^{D}(Reach^{=i}(B)). \end{aligned}$$

We claim that, for all $i \in \mathbb{N}$ and all schedulers $D \in \mathcal{D}^{\mathcal{M}}$, there exists $D' \in \mathcal{D}^{\mathcal{M}'}$ such that

$$\Pr^{D}(Reach^{=i}(B)) = \Pr^{D'}(Reach^{=i}(B')).$$

This can be shown by a similar induction as in the proof of Theorem 4.2.1 and it yields

$$\begin{aligned} \Pr^{D}(Reach_{\leq t}(B)) &= \textstyle\sum_{i=0}^{\infty} \psi_{\lambda,t}(i) \cdot \Pr^{D}(Reach^{=i}(B)) \\ &= \textstyle\sum_{i=0}^{\infty} \psi_{\lambda,t}(i) \cdot \Pr^{D'}(Reach^{=i}(B')) \\ &= \textstyle\sum_{i=0}^{\infty} \Pr^{D'}(Reach^{=i}_{\leq t}(B')) \\ &= \Pr^{D'}(Reach_{\leq t}(B')) \end{aligned}$$

and the proof is complete. □

For ACTMC $\mathcal{M}$ we define the abbreviations

$$Pr^{\mathcal{M}}_{\inf}(t, B) = \inf_{D \in \mathcal{D}^{\mathcal{M}}} Pr^{D}(Reach_{\leq t}(B))$$
$$Pr^{\mathcal{M}}_{\sup}(t, B) = \sup_{D \in \mathcal{D}^{\mathcal{M}}} Pr^{D}(Reach_{\leq t}(B)).$$

The following lemma will provide the key argument when showing the preservation of CSL formulas for abstraction-based model checking in Section 4.5.

Corollary 4.3.2. *Let $\mathcal{M}$ and $\mathcal{M}'$ be ACTMCs with $\mathcal{M} \preceq \mathcal{M}'$, let B, B' be compatible sets and $t \in \mathbb{R}_{\geq 0}$. Then*

$$Pr^{\mathcal{M}'}_{\inf}(t, B') \leq Pr^{\mathcal{M}}_{\inf}(t, B) \leq Pr^{\mathcal{M}}_{\sup}(t, B) \leq Pr^{\mathcal{M}'}_{\sup}(t, B').$$

Proof. The claim follows directly from Theorem 4.3.1. □

Therefore, our abstraction is conservative w.r.t. the minimal probability to reach a set of states within t time units in the concrete model.

Computation of time-bounded reachabilities. By Theorem 4.1.3, it suffices to consider extreme schedulers if one is interested in $Pr^{\mathcal{M}}_{\inf}(t, B)$ or $Pr^{\mathcal{M}}_{\sup}(t, B)$. In the same way as an ADTMC can be represented by a MDP, we can interpret an ACTMC as a CTMDP, where the transition probabilities in each state are determined by the extreme distributions. Since we consider uniform ACTMCs, the associated CTMDP is a uniform CTMDP. For uniform CTMDPs, an efficient algorithm for the approximation of minimal and maximal probabilities (w.r.t. history-dependent schedulers) for timed reachability properties is known [BHKH05]. In the sequel, we present an adaption of this algorithm, which avoids the exponential blow-up of the CTMDP-representation. More precisely, the algorithm operates directly on the ACTMC, similar to the iteration for the step-bounded case. We concentrate on minimal probabilities. The computation of maximal probabilities goes along similar lines, so we omit these details.

A greedy algorithm. The algorithm presented in [BHKH05] for time-bounded reachability probabilities in uniform CTMDPs computes ε-approximations for a given error bound $\varepsilon > 0$ by summing up the probabilities to reach a goal state within a given amount of time and in exactly i steps for $i \in \{0, \ldots, n_\varepsilon\}$ where n_ε is a proper truncation point. It can be understood as a variant of the value iteration algorithm for MDPs, however, the algorithm's structure notably differs from the one for step-bounded reachability as the probability for *exactly* $i \in \mathbb{N}$ steps within t time units has to be factored into the probability to reach a goal state in *exactly* i steps.

In the following, for CTMDP $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0, \lambda)$ where all $s \in B$ are absorbing, the minimal probabilities $q_i^l(s)$ for paths $\sigma \in \bigcup_{j=i}^{n_\varepsilon} Reach_{\leq t}^{=j}(B)$ with $\sigma[i] = s$ are computed:

$$
\begin{aligned}
q_0^l(s) &= \psi_{\lambda,t}(0) \cdot \mathbf{1}(s \in B) + q_1^l(s) \\
q_i^l(s) &= \min\{\textstyle\sum_{u \in S} \mu(u) \cdot \nu(u) \mid \mu = \mathbf{P}(s, a, \cdot), a \in A\} \qquad \forall\ 0 < i \leq n_\varepsilon \\
q_{n_\varepsilon+1}^l(s) &= 0 \\
\text{where} \qquad \nu(u) &= \psi_{\lambda,t}(i) \cdot \mathbf{1}(u \in B) + q_{i+1}^l(u)
\end{aligned}
$$

Intuitively, $q_i^l(s)$ equals the successor probability distribution for which the probability to be in a goal state after the i-th transition plus the probability to reach a goal state after the i+1-st to n_ε-th transition is minimal.

To obtain an ε-approximation for $\Pr_{inf}(t, B)$ with the above given algorithm (denoted $\Pr_{inf}^{\varepsilon}(t, B)$ in the following), we still have to find a proper truncation point n_ε. Therefore, we choose n_ε such that the path fragments we consider for the computation of reachability probabilities are covering all but an ε-fraction of all relevant paths; formally, we rely on the following inequation:

$$\textstyle\sum_{i=n_\varepsilon+1}^{\infty} \psi_{\lambda,t}(i) \leq \varepsilon \quad \Longleftrightarrow \quad \sum_{i=0}^{n_\varepsilon} \psi_{\lambda,t}(i) \geq 1 - \varepsilon$$

From this, the truncation point n_ε can be easily computed a priori as shown in [Hav98].

As for step-bounded reachability, we adapt the algorithm by replacing the brute force check on the minimizing choice by the minimization function. Thus, we obtain an algorithm for the computation of ε-approximations of time-bounded reachability for ACTMC $\mathcal{M} = (S, A, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0, \lambda)$ where all $s \in B$ are absorbing, which has a time complexity that is polynomial in the size of the state space. For time-bound t, proper truncation point n_ε and goal states B, we obtain $\Pr^\varepsilon_{inf}(t, B) = \sum_{s_0 \in S} \mu_0(s_0) \cdot q^l_0(s_0)$ where for all $s \in S$, $i \in \{1, \ldots, n_\varepsilon\}$:

$$\begin{aligned} q^l_0(s) &= \psi_{\lambda,t}(0) \cdot \mathbf{1}(s \in B) + q^l_1(s) \\ q^l_i(s) &= \textstyle\sum_{u \in S} \min^{\mathcal{M}}(s, \nu)(u) \cdot \nu(u) \\ q^l_{n_\varepsilon+1}(s) &= 0 \\ \text{where} \qquad \nu(u) &= \psi_{\lambda,t}(i) \cdot \mathbf{1}(u \in B) + q^l_{i+1}(u) \end{aligned}$$

Analogously, we obtain an algorithm for the computation of upper bounds for time-bounded reachability probabilities. However, we have to add the error bound ε to the result as otherwise, the algorithm computes a value up to ε below the actual value. Hence, for time-bound t, proper truncation point n_ε and goal states B, we obtain $\Pr^\varepsilon_{sup}(t, B) = \sum_{s_0 \in S} \mu_0(s_0) \cdot q^u_0(s_0)$ where for all $s \in S$, $i \in \{1, \ldots, n_\varepsilon\}$:

$$\begin{aligned} q^u_0(s) &= \psi_{\lambda,t}(0) \cdot \mathbf{1}(s \in B) + q^u_1(s) + \varepsilon \\ q^u_i(s) &= \textstyle\sum_{u \in S} \max^{\mathcal{M}}(s, \nu)(u) \cdot \nu(u) \\ q^u_{n_\varepsilon+1}(s) &= 0 \\ \text{where} \qquad \nu(u) &= \psi_{\lambda,t}(i) \cdot \mathbf{1}(u \in B) + q^u_{i+1}(u) \end{aligned}$$

Theorem 4.3.3. *For any ACTMC $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0, \lambda)$, time-bound $t \in \mathbb{R}_{\geq 0}$ and set of goal states $B \subseteq S$, we can compute reachability probabilities $\Pr^\varepsilon_{inf}(t, B)$ and $\Pr^\varepsilon_{sup}(t, B)$ in time polynomial in the size of $\mathcal{M}$ and linear in the time-bound t and the exit rate λ.*

Proof. Follows directly from the complexity results provided in [BHKH05] and the arguments from the proof of Theorem 4.2.3. □

The section is concluded with a lemma, which states that the above algorithm yields an ε-accurate approximation of time-bounded reachability probabilities.

Lemma 4.3.4. *Let $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0, \lambda)$ be an ACTMC, $s \in S$, $B \subseteq S$, $t \in \mathbb{R}_{>0}$ and error margin ε, then:*

$$\Pr_{inf}(t, B) - \varepsilon \leq \Pr^{\varepsilon}_{inf}(t, B) \leq \Pr_{inf}(t, B)$$
$$\Pr_{sup}(t, B) + \varepsilon \geq \Pr^{\varepsilon}_{sup}(t, B) \geq \Pr_{sup}(t, B)$$

Proof. The claim follows directly from [BHKH05, Theorem 5]. □

4.4 Unbounded reachability

In this section we shortly address the probability to eventually reach a set of states in an AMC. For an AMC $\mathcal{M}$ with state space S, $B \subseteq S$ let the set

$$Reach(B) = \{\varsigma \in Path^{\mathcal{M}} \mid \varsigma[i] \in B \text{ for some } i \geq 0\}.$$

Let $\mathcal{M}'$ be an AMC with $\mathcal{M} \preceq \mathcal{M}'$, and let B, B' be compatible sets of states. By Theorem 4.2.1, for any $D \in \mathcal{D}^{\mathcal{M}}$ and $i \in \mathbb{N}$ there exists $D' \in \mathcal{D}^{\mathcal{M}'}$ with $Pr^D(Reach^{\leq i}(B)) = Pr^{D'}(Reach^{\leq i}(B'))$. As $Reach_{\hat{\mathcal{M}}}(\hat{B}) = \lim_{i\to\infty} Reach^{\leq i}_{\hat{\mathcal{M}}}(\hat{B})$ for all sets $\hat{B}$ of states of an AMC $\hat{\mathcal{M}}$ it follows that for any $D \in \mathcal{D}^{\mathcal{M}}$ there exists $D' \in \mathcal{D}^{\mathcal{M}'}$ with

$$Pr^D(Reach(B)) = Pr^{D'}(Reach(B')). \tag{4.3}$$

Let us abbreviate $\inf_{D\in\mathcal{D}^{\mathcal{M}}} Pr^D(Reach(B))$ by $Pr^{\mathcal{M}}_{\inf}(B)$ and let $Pr^{\mathcal{M}}_{\sup}(B)$ denote the supremum.

Lemma 4.4.1. *Let $\mathcal{M}$ and $\mathcal{M}'$ be AMCs with $\mathcal{M} \preceq \mathcal{M}'$ and let B, B' be compatible sets of states. Then*

$$Pr^{\mathcal{M}'}_{\inf}(B') \leq Pr^{\mathcal{M}}_{\inf}(B) \leq Pr^{\mathcal{M}}_{\sup}(B) \leq Pr^{\mathcal{M}'}_{\sup}(B').$$

Proof. This lemma is a direct implication of Equation (4.3). □

Therefore, our abstraction is conservative w.r.t. the minimal probability to eventually reach a set of states in the concrete model.

Computation of unbounded reachabilities. It is sufficient to consider ADTMCs for unbounded reachability probabilities. In addition, we can restrict the scheduler set to the subset of simple and extreme schedulers, denoted $\mathcal{ES}$ (cf. [dA97]).

Let $\mathcal{M}$ be an ADTMC with state space S. Then, for all $B \subseteq S$,

$$Pr^{\mathcal{M}}_{\inf}(B) = \inf_{E \in \mathcal{ES}^{\mathcal{M}}} Pr^{D}(Reach(B)),$$
$$Pr^{\mathcal{M}}_{\sup}(B) = \sup_{E \in \mathcal{ES}^{\mathcal{M}}} Pr^{D}(Reach(B)).$$

In order to compute unbounded reachability probabilities for MDPs with state space S and action set A, a linear programming problem with (at most) $|S| \cdot |A|$ equations is to be solved, e.g. using the simplex method which has exponential complexity in the number of equations. There also exist polynomial time algorithms such as the ellipsoid method [Kar84], however, in practice they are outperformed by simplex. By transforming an AMC into an MDP, this method can be adapted for the analysis of AMCs. However, in the worst case, the action set of the MDP is exponential in $|S|$, i.e. a system of exponentially many equations has to be solved.

Due to the difficulty of solving linear programming problems with many equations, in practice, value iteration is used to approximate solutions of $\Pr_{inf}(Reach(B))$. With this approach, the polynomial algorithm presented in Section 4.2 is applicable. However, one has to be very careful when using this *trick* as the results may be quite different from the actual results. This becomes clear when considering the Erlang-k distributions in Example 2.4.2 (p. 26). When using value iteration, one might decide to stop the computation after considering $k-1$ transitions in the underlying Markov chain as the probability to be in state s_k – the goal state – remains unchanged, namely *zero*. This changes when computing the probability to reach the goal state within k steps in which case we obtain probability *one*.

4.5 Three-valued model checking

The characterizations in the previous section in terms of extrema of bounded and unbounded reachability probabilities are the key building blocks for model checking of PCTL and CSL formulas. It remains to provide a three-valued semantics for these logics and, more importantly, to show that verification results on abstract Markov chains carry over to their concrete counterparts.

Three-valued semantics. For ADTMC $\mathcal{M}$, we define the satisfaction function $[\![\cdot]\!] : \text{PCTL} \to (S \cup Path^{\mathcal{M}} \to \mathbb{B}_3)$ inductively as shown in Table 4.1, where:

$$\Pr_{inf}(s, \Psi) = \Pr_{inf}(\{\varsigma \in Path_s^{\mathcal{M}} \mid [\![\Psi]\!](\varsigma) = \top\})$$
$$\Pr_{sup}(s, \Psi) = \Pr_{sup}(\{\varsigma \in Path_s^{\mathcal{M}} \mid [\![\Psi]\!](\varsigma) \neq \bot\})$$

$$\trianglelefteq, \triangleleft \in \{\leq, <\} \text{ with } \triangleleft = < \text{ iff } \trianglelefteq = \leq,$$
$$\trianglerighteq, \triangleright \in \{\geq, >\} \text{ with } \triangleright = > \text{ iff } \trianglerighteq = \geq,$$

$p \in [0,1]$, $i \in \mathbb{N} \cup \{\infty\}$ and $a \in AP$. Recall that the complement of a truth value is denoted $\cdot^c$, i.e. $\bot$ and $\top$ are complementary and $?^c = ?$. For ACTMC $\mathcal{M}$, the satisfaction function $[\![\cdot]\!] : \text{CSL} \to (S \cup Path^{\mathcal{M}} \to \mathbb{B}_3)$ is defined similarly, however, $\Pr_{inf}(s, \Psi)$ and $\Pr_{sup}(s, \Psi)$ concern timed paths. The time-bounded until operator is as shown in Table 4.2 where $t \in \mathbb{R}_{\geq 0} \cup \{\infty\}$.

Let us have a closer look at the semantics. For the propositional fragment the semantics is clear. A path ρ satisfies until formula $\varphi_1 \mathcal{U}^{\leq i} \varphi_2$ if φ_1 definitely holds ($\top$) until φ_2 definitely holds at the latest after i steps; similarly, σ satisfies $\varphi_1 \mathcal{U}^{\leq t} \varphi_2$ if φ_1 definitely holds until φ_2 definitely holds at the latest at time t. The until-formulas are violated ($\bot$), if either before φ_2 holds, φ_1 is violated, or if φ_2 is definitely violated within i steps, up to time t respectively. Otherwise, the result is indefinite (?). For untimed until, i.e. for $i = \infty$, the semantics is similar, but without any step bound.

$$
\begin{aligned}
&[\![\mathit{true}]\!](s) = \top \\
&[\![a]\!](s) \quad = L(s,a) \\
&[\![\neg\varphi]\!](s) \;= ([\![\varphi]\!](s))^c \\
&[\![\varphi_1 \wedge \varphi_2]\!](s) = [\![\varphi_1]\!](s) \sqcap [\![\varphi_2]\!](s)
\end{aligned}
$$

$$
[\![\mathcal{P}_{\trianglelefteq p}(\Psi)]\!](s) = \begin{cases} \top & \text{if } \Pr_{\mathit{sup}}(s,\Psi) \trianglelefteq p \\ \bot & \text{if } \Pr_{\mathit{inf}}(s,\Psi) \rhd p \\ ? & \text{otherwise} \end{cases}
$$

$$
[\![\mathcal{P}_{\trianglerighteq p}(\Psi)]\!](s) = \begin{cases} \top & \text{if } \Pr_{\mathit{inf}}(s,\Psi) \trianglerighteq p \\ \bot & \text{if } \Pr_{\mathit{sup}}(s,\Psi) \lhd p \\ ? & \text{otherwise} \end{cases}
$$

$$
[\![\varphi_1 \mathcal{U}^{\leq i} \varphi_2]\!](\varsigma) = \begin{cases} \top & \text{iff } \exists i' \leq i \colon ([\![\varphi_2]\!](\varsigma[i']) = \top \\ & \qquad\qquad \wedge \forall i'' \leq i' \colon [\![\varphi_1]\!](\varsigma[i'']) = \top) \\ \bot & \text{iff } \forall i' \leq i \colon ([\![\varphi_2]\!](\varsigma[i']) = \bot \\ & \qquad\qquad \vee \exists i'' \leq i' \colon [\![\varphi_1]\!](\varsigma[i'']) = \bot) \\ ? & \text{otherwise} \end{cases}
$$

Table 4.1: Three-valued semantics of PCTL.

$$
[\![\varphi_1 \mathcal{U}^{\leq t} \varphi_2]\!](\varsigma) = \begin{cases} \top & \text{iff } \exists t' \leq t \colon ([\![\varphi_2]\!](\varsigma@t') = \top \\ & \qquad\qquad \wedge \forall t'' \leq t' \colon [\![\varphi_1]\!](\varsigma@t'') = \top) \\ \bot & \text{iff } \forall t' \leq t \colon ([\![\varphi_2]\!](\varsigma@t') = \bot \\ & \qquad\qquad \vee \exists t'' \leq t' \colon [\![\varphi_1]\!](\varsigma@t'') = \bot) \\ ? & \text{otherwise} \end{cases}
$$

Table 4.2: Three-valued semantics of CSL.

To determine the satisfaction of $\mathcal{P}_{\geq p}(\Psi)$ we consider the minimal probability of the paths for which Ψ is definitely satisfied. If this probability is at least p, then paths where Ψ holds have measure at least p. For the violation of $\mathcal{P}_{\geq p}(\Psi)$ we consider the maximal probability of paths satisfying Ψ; if it is strictly less than p, then the formula is violated for sure. The definition of the semantics of $\mathcal{P}_{\bowtie p}(\Psi)$ for $\bowtie \in \{<, >, \leq\}$ follows by a similar argumentation.

Example 4.5.1. Consider the CTMC in Figure 4.6 (left) where $L(s, ap) = \top$ iff $s \in \{s_0, s_1\}$ and exit rate $\lambda = 12$. Starting in s_0 (s_1), the probability to reach a non-ap-state in 0.3 time units is approximately 0.9037 and 0.9328, respectively. Thus, formula $\varphi = ap \rightarrow \mathcal{P}_{\geq 0.9}(true\, \mathcal{U}^{\leq 0.3} \neg ap)$ is true in all states.

Consider the interval abstraction induced by $\gamma(s'_0) = \{s_0, u_0\}$, $\gamma(s'_1) = \{s_1\}$ and $\gamma(s'_2) = \{s_2\}$, depicted in Figure 4.6 (right): The lower and upper probability bounds to reach a non-ap-state in 0.3 time units from s_0 are about 0.8807 and 0.9037, respectively; as $L(s_0, ap) \neq L(u_0, ap)$, we obtain $L(s'_0, ap) = ?$. Hence,

$$\begin{aligned}[\![ap \rightarrow \mathcal{P}_{\geq 0.9}(true \mathcal{U}^{\leq 0.3} \neg ap)]\!](s'_0) &= ? \sqcup [\![\mathcal{P}_{\geq 0.9}(true \mathcal{U}^{\leq 0.3} \neg ap)]\!](s'_0) \\ &= ? \sqcup ? = ?.\end{aligned}$$

For $\mathcal{P}_{\geq 0.88}$ instead of $\mathcal{P}_{\geq 0.9}$, the formula would have been satisfied in the abstraction as well, while for $\mathcal{P}_{\geq 0.91}$ the result would still be ? since $? \sqcup \bot = ?$.

Model checking. As for CTL, model checking PCTL and CSL works by a bottom-up traversal of the parse tree of the formula φ. Boolean combinations of formulas as well as the $\mathcal{P}$-formulas are evaluated as expected. For the latter, however, we need the lower and upper probability bounds for the satisfaction/violation of an until-formula, which remains the only operator to discuss.

To compute the measure of paths definitely satisfying $\Psi = \varphi_1 \mathcal{U} \varphi_2$ ($\Psi = \varphi_1 \mathcal{U}^{\leq i} \varphi_2$ or $\Psi = \varphi_1 \mathcal{U}^{\leq t} \varphi_2$ respectively), it suffices to compute the measure

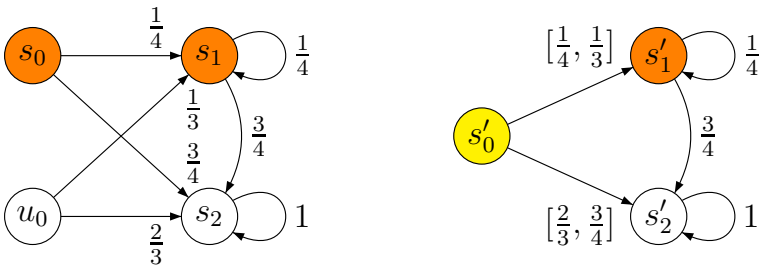


Figure 4.6: A Markov chain (left) and an abstraction with $\gamma(s'_0) = \{s_0, u_0\}$, $\gamma(s'_1) = \{s_1\}$ and $\gamma(s'_2) = \{s_2\}$.

of reaching states satisfying φ_2 (in steps bounded by i, time bounded by t respectively) along paths of states satisfying φ_1. By induction, we know which states do not satisfy φ_1. Transforming the Markov chains such that those states are absorbing yields a Markov chain in which a path satisfies $\varphi_1 \mathcal{U} \varphi_2$ iff a φ_2-state is reached. In other words, it remains to solve a reachability problem in the transformed Markov chain.

Recall that the given algorithm for computing time-bounded reachability approximates only with error margin ε. However, it can easily be guaranteed that the error due to approximation is accounted to ?.

Also note that step-bounded until formulas can be dealt with in the continuous-time setting as well, by dropping the exit rate and considering the resulting embedded DTMC.

The following theorems state that our framework developed so far can indeed be used for abstraction-based model checking. Intuitively, the theorems assert that the result of checking a PCTL / CSL formula in the abstract DTMC / CTMC agrees with the one for the more concrete model, unless the result is indefinite.

Theorem 4.5.1 (Preservation of PCTL)**.** *Let s and s' be two states of an ADTMC $\mathcal{M}$ with $s \preceq s'$. Then for all PCTL formulas φ:*

$$[\![\varphi]\!](s') \neq ? \text{ implies } [\![\varphi]\!](s) = [\![\varphi]\!](s').$$

Proof. By induction on the structure of PCTL formulas. Atomic formulas are *true* and $a \in AP$:

- $[\![true]\!](s') = \top = [\![true]\!](s)$
- $[\![a]\!](s') \neq ? \implies [\![a]\!](s') = L(s', a) = L(s, a) = [\![a]\!](s)$ since $s \preceq s'$.

Induction hypothesis: for all subformulas φ' of φ, and all states s, s' where $s \preceq s'$:

$$[\![\varphi']\!](s') \neq ? \implies [\![\varphi']\!](s) = [\![\varphi']\!](s') \qquad (*)$$

- $\varphi = \neg\varphi'$:
  For $[\![\varphi']\!](s') = ?$ we have $[\![\neg\varphi']\!](s') = ?$, hence there is nothing to be shown.
  Otherwise, $[\![\neg\varphi']\!](s') = ([\![\varphi']\!](s'))^c \overset{(*)}{=} ([\![\varphi']\!](s))^c = [\![\neg\varphi']\!](s)$.

- $\varphi = \varphi_1 \wedge \varphi_2$:
  For $[\![\varphi_1]\!](s') = \bot$ (and for $[\![\varphi_2]\!](s') = \bot$ analogously):

  $$\begin{aligned}[\![\varphi_1 \wedge \varphi_2]\!](s') &= [\![\varphi_1]\!](s') \sqcap [\![\varphi_2]\!](s') \\ &\overset{(*)}{=} [\![\varphi_1]\!](s) \sqcap [\![\varphi_2]\!](s') = \bot \sqcap [\![\varphi_2]\!](s') = \bot\end{aligned}$$

  For $[\![\varphi_1]\!](s') = [\![\varphi_2]\!](s') = \top$:

  $$[\![\varphi_1 \wedge \varphi_2]\!](s') = [\![\varphi_1]\!](s') \sqcap [\![\varphi_2]\!](s') \overset{(*)}{=} [\![\varphi_1]\!](s) \sqcap [\![\varphi_2]\!](s) = \top \sqcap \top = \top$$

  For $[\![\varphi_1]\!](s') = ?$, $[\![\varphi_2]\!](s') \neq \bot$ (and for $[\![\varphi_2]\!](s') = ?$, $[\![\varphi_1]\!](s') \neq \bot$ analogously):

  $$[\![\varphi_1 \wedge \varphi_2]\!](s') = [\![\varphi_1]\!](s') \sqcap [\![\varphi_2]\!](s') = ? \sqcap [\![\varphi_2]\!](s') = ?$$

  and thus $(*)$ holds trivially in this case.

- $\varphi = \mathcal{P}_{\trianglelefteq p}(\varphi_1 \mathcal{U}^{\leq i} \varphi_2)$:

  As argued before, model checking of a (step-bounded) until-formula can be reduced to a reachability analysis on an appropriately modified ADTMC $\tilde{\mathcal{M}} = (\tilde{S}, \tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u, \tilde{L}, \tilde{\mu}_0)$. Let $B = B' = \{s \in \tilde{S} \mid [\![\varphi_2]\!](s) = \top\}$ (or $B = B' = \{s \in \tilde{S} \mid [\![\varphi_2]\!](s) = \bot\}$ respectively) and $s \preceq s'$. Together with Corollary 4.2.2 and Lemma 4.4.1 we obtain:

  $$\Pr^{\mathcal{M}}_{inf}(s, \varphi_1 \mathcal{U}^{\leq i} \varphi_2) = \Pr^{\tilde{\mathcal{M}}}_{s,inf}(\mathit{Reach}(i, B)) \geq \Pr^{\tilde{\mathcal{M}}}_{s',inf}(\mathit{Reach}(i, B'))$$
  $$\Pr^{\mathcal{M}}_{sup}(s, \varphi_1 \mathcal{U}^{\leq i} \varphi_2) = \Pr^{\tilde{\mathcal{M}}}_{s,sup}(\mathit{Reach}(i, B)) \leq \Pr^{\tilde{\mathcal{M}}}_{s',sup}(\mathit{Reach}(i, B'))$$

  Intuitively this means that the probability for paths starting in s' and fulfilling (or violating) $\varphi_1 \mathcal{U}^{\leq i} \varphi_2$ is at most the probability of such paths starting in s, thus:

  $$\begin{aligned} & [\![\varphi]\!](s') = \top \\ \Longrightarrow\ & \Pr_{sup}(s, \varphi_1 \mathcal{U}^{\leq i} \varphi_2) \leq \Pr_{sup}(s', \varphi_1 \mathcal{U}^{\leq i} \varphi_2) \trianglelefteq p \\ \Longrightarrow\ & [\![\varphi]\!](s) = \top \end{aligned}$$

  and

  $$\begin{aligned} & [\![\varphi]\!](s') = \bot \\ \Longrightarrow\ & \Pr_{inf}(s, \varphi_1 \mathcal{U}^{\leq i} \varphi_2) \geq \Pr_{inf}(s', \varphi_1 \mathcal{U}^{\leq i} \varphi_2) \trianglerighteq p \\ \Longrightarrow\ & [\![\varphi]\!](s) = \bot \end{aligned}$$

  For $\mathcal{P}_{\trianglerighteq p}(\varphi_1 \mathcal{U}^{\leq i} \varphi_2)$ this can be shown analogously. □

Theorem 4.5.2 (Preservation of CSL)**.** *Let s and s' be two states of an ACTMC $\mathcal{M}$ with $s \preceq s'$. Then for all CSL formulas φ:*

$$[\![\varphi]\!](s') \neq ?\ \textit{implies}\ [\![\varphi]\!](s) = [\![\varphi]\!](s').$$

Proof. For atomic formulas and the boolean operators, the proof is as for preservation of CSL and also for $\varphi = \mathcal{P}_{\trianglelefteq p}(\varphi_1 \mathcal{U}^{\leq t} \varphi_2)$ the basic idea is as for step-bounded until. As slight modification, instead of Corollary 4.2.2 one has to refer to Corollary 4.3.2 when comparing time-bounded reachability probabilities for s and s'. □

Observe that the three-valued PCTL semantics on a DTMC (viewed as abstract DTMC) coincides with the two-valued PCTL semantics for DTMCs as well as the three-valued CSL semantics on a uniform CTMC (viewed as ACTMC) coincides with the two-valued CSL semantics for CTMCs (see Section 2). This shows that our abstraction is *conservative* for positive and negative verification results.

We conclude this section with complexity results.

Theorem 4.5.3. *For ADTMC $\mathcal{M}$ and a PCTL formula φ (without unbounded until subformulas), we can determine $[\![\varphi]\!]$ in time exponential (polynomial) in the size of $\mathcal{M}$ and linear in the size of φ.*

Proof. For the propositional subset of PCTL, $[\![\varphi]\!](s)$ can obviously be checked in time linear to the size of φ. The complexity for checking reachability properties, i.e. the unbounded until operator, is polynomial in the size of the induced MDP of $\mathcal{M}$. As for each state state, the induced MDP may have an exponential number of actions to choose from, in the worst case its size is exponential in the size of $\mathcal{M}$. For formulas without unbounded until subformulas, the polynomial time algorithm from Section 4.2 can be utilized, yielding a time complexity which is polynomial in the size of the state space. □

We want to point out that unbounded reachability probabilities for ADTMCs can be approximated in an iterative way using *value iteration*, though, complexity results cannot be provided. In the setting of ADTMCs this is especially favorable as extreme distributions are *calculated on-the-fly* (in polynomial time) in contrast to the MDP approach.

Theorem 4.5.4. *Given an ACTMC $\mathcal{M}$, a CSL formula φ without unbounded until subformulas, and an error margin ε, we can approximate $[\![\varphi]\!]$ in time polynomial in the size of $\mathcal{M}$ and linear in size of φ, λ and the highest time bound t occurring in φ (dependency on ε is omitted as ε is linear in $\lambda \cdot t$). In case the approximation yields $\top$ or $\bot$, the result is correct.*

Proof. For the propositional subset of CSL, $[\![\varphi]\!](s)$ can obviously be checked in time linear to the size of φ. The complexity for checking timed reachability properties can be derived from the complexity for uniform CTMDPs (see [BHKH05]), which is polynomial in the size of $\mathcal{M}$, and linear in the exit rate λ, the time bound t and the number of actions. Instead of checking all extreme distributions in an ACTMC, which would yield an exponential number of actions in a corresponding CTMDP, we can use the polynomial algorithm presented earlier to determine a distribution yielding the minimal reachability property. Thus the complexity of checking ACTMCs is as claimed. Correct answers for each, the $\top$ and $\bot$ case, are ensured by accounting the error to ?. □

Note that, when considering a formula with *nested* until subformulas, even for CTMCs it is a difficult task to ensure a maximal error for the approximation result. In the case of three-valued model checking this means that a larger fraction of the ? results may result from the approximation. However, when considering formulas without nested until subformulas, one can assure that the maximal error of the approximation of reachability probabilities will be at most ε. This implies that ? results are computed with an error of at most $2 \cdot \varepsilon$.

4.6 Case study: Tree-structured Quasi-Birth-Death processes

In the previous sections, we introduced abstraction techniques for discrete-time and continuous-time Markov chains with countable state space. We further showed how to analyze them in terms of (bounded) reachability probabilities and how to use these analyses for model checking abstractions. In the following, we apply these techniques to a case study from queueing theory where a queueing station with so-called preemptive last-in-first-out service policy is considered. We will define several ways to partition the state space and empirically show their impact on both, the size of the abstract state

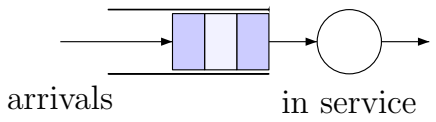


Figure 4.7: Queueing station.

space and the quality of the results. For that purpose, we analyze what is the probability to serve all but k jobs in the queue within t time units.

Preemptive LIFO queues. *Queueing stations* comprise of a number of servicing units and a queue for arriving jobs, cf. Figure 4.7. For identifying different classes of queueing stations with respect to parameters like the arrival distribution, the service distribution and the number of service units, the standard description scheme is the so-called *Kendall* notation. For example M|PH|1 denotes the class of queueing stations where jobs arrive with an exponentially distributed (*Markovian*) delay, and where jobs are serviced according to a *phase-type* distribution and by *one* server only. A further important parameter is the servicing discipline; examples are first-in-first-out (FIFO), last-in-first-out (LIFO) and their preemptive variants, where the servicing of the current job can be paused in order to service newly arriving jobs first.

PH|PH|1 queuing stations with FIFO service discipline for example correspond to plain *quasi-birth-death* (QBD) processes [LR99], therefore their state spaces just grow *linearly* with the number of queued jobs. This stems from the fact that jobs are not preempted, i.e., jobs are served until completion and only the service phase of the job currently in service needs to be encoded in the state. For such systems, *uniformization with representatives* is a feasible technique to compute transient probabilities [RHC07]. For queuing stations with *preemptive LIFO* service discipline, however, the

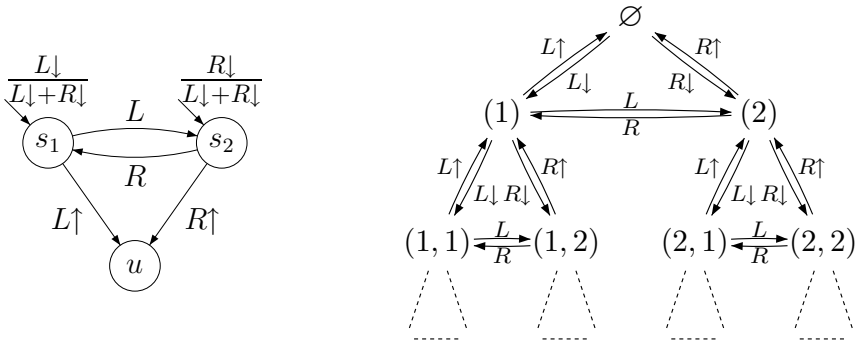


Figure 4.8: A CTMC representing a PH_2 distribution (left) and an example (binary) tree-structured QBD (right).

underlying models are *tree-structured* QBDs, whose size grows *exponentially* with the queue length. In principle, uniformization with representatives can be adapted to the analysis of tree-structured QBDs, but for PH_d distributed service times, n uniformization steps and a single starting state, one would have to consider $\mathcal{O}(d^n)$ states, which is practically infeasible. The same holds for techniques based on uniformization like in [Cia95, ZHHW08].

In the following we will show how to apply abstraction to obtain models with manageable size in this setting; for simplicity we restrict ourselves to M|PH|1 queues, however, the approach can also be applied to PH|PH|1 queues in the same manner.

Tree-structured quasi-birth death processes. Tree-structured quasi-birth death (QBD) processes [TSY95, YA99] are a class of infinite-state continuous-time Markov chains that can not only be used to model single-server queues with a LIFO service discipline [HA00], but also to analyze random access algorithms [vHB05], as well as priority queueing systems [vHB06]. Discrete-time tree-structured QBDs are equivalent to probabilistic pushdown automata [BKS05] and recursive Markov chains [EWY08]. However, in the

following, we introduce tree-structured QBDs using terminology from queueing theory.

Definition 4.6.1. A *d-ary tree-structured quasi-birth-death process* $\mathcal{M}$ is a non-uniform CTMC $(S, \mathbf{P}, L, \mu_0, \lambda)$ with state space:

$$S = \{(x_1, \ldots, x_n) \mid n \in \mathbb{N} \wedge \forall i \leq n : x_i \in \{1, \ldots, d\}\}$$

A state $\vec{x} = (x_1, \ldots, x_n)$ represents a queue of length n where jobs $1, \ldots, n-1$ have been preempted in phase $x_i \in \{1, \ldots, d\}$ and job n is currently in service in phase $x_n \in \{1, \ldots, d\}$. Transitions, represented by positive entries in $\mathbf{P}$, can only occur between a state and its parent, its children and its siblings. For $\vec{x}, \vec{y} \in S$ and $r_i\downarrow, r_i\uparrow, r_{i,j} \in \mathbb{R}_{\geq 0}$ for all $i, j \in \{1, \ldots, d\}$:

- $\mathbf{P}(\vec{x}, \vec{y}) = \begin{cases} \frac{r_{x_{m+1}\downarrow}}{\lambda(\vec{x})} & \text{if } \vec{x} = (x_1, \ldots, x_m), \text{ and } \vec{y} = (x_1, \ldots, x_{m+1}) \\ \frac{r_{x_{m+1}\uparrow}}{\lambda(\vec{x})} & \text{if } \vec{x} = (x_1, \ldots, x_{m+1}), \text{ and } \vec{y} = (x_1, \ldots, x_m) \\ \frac{r_{x_m, y_m}}{\lambda(\vec{x})} & \text{if } \vec{x} = (x_1, \ldots, x_m), \text{ and } \vec{y} = (x_1, \ldots, x_{m-1}, y_m) \\ 0 & \text{otherwise} \end{cases}$

- $\lambda(\vec{x}) = \begin{cases} \sum_{i=1}^{d} (r_i\downarrow + r_{x_m,i}) + r_m\uparrow & \text{if } \vec{x} = (x_1, \ldots, x_m) \\ \sum_{i=1}^{d} r_i\downarrow & \text{if } \vec{x} = (\,) \end{cases}$

The underlying state space of a preemptive LIFO M|PH$_2$|1 queue with overall arrival rate $L\downarrow + R\downarrow$ and service time distribution as depicted in Figure 4.8 (left) is the (binary) tree-structured QBD shown in Figure 4.8 (right), where $r_1\downarrow = L\downarrow$, $r_2\downarrow = R\downarrow$, $r_1\uparrow = L\uparrow$, $r_2\uparrow = R\uparrow$, $r_{1,2} = L$, $r_{2,1} = R$. Note that, in contrast to ordinary trees, in tree-structured QBDs, transitions between siblings are allowed. For convenience, in the following we may label the transitions with *rates* instead of probabilities; the probabilities can be obtained by simply dividing the rates by the exit rate of the state to be left.

State $\varnothing = (\,)$ represents the empty queue. Arriving jobs can either enter service phase 1 or 2, represented by the states (1) and (2). Due to the

preemptive LIFO service discipline, a new job arrival causes the preemption of the job currently in service and the service phase of the preempted job needs to be stored. This results in a tree-structured state space.

Note that it has been shown in [SvHB06], that every tree-structured QBD can be embedded in a binary tree-structured Markov chain with a special structure.

Partitioning a Tree-Structured QBD. A major issue in state-based abstraction is to come up with an adequate, i.e., small though precise, partitioning of the state space. Thus, we identify various possibilities to partition the state space of a tree-structured QBD based on different amounts of information to be encoded in abstract states; later, we will empirically investigate their influence on the accuracy, i.e. the difference between lower and upper bounds, of time-bounded reachability probabilities.

Every state of the tree-structured QBD represents

(P1) the number of jobs in the queue,

(P2) the service phases of the preempted jobs,

(P3) the phase of the job that is currently in service and

(P4) the precise order of jobs in the queue.

The states with m jobs, that are situated in the same layer of the tree, have the form $\vec{x} = (x_1, x_2, \ldots, x_m)$ where x_i gives the service phase of the i-th job in the queue. We abbreviate the prefix of $\vec{x}$ of length n by $\vec{x}{\downarrow_n}$ and the number of jobs in $\vec{x}$ in phase i by $\#_i\vec{x}$.

In the following, we present abstractions that preserve several of the above mentioned properties from (P1) to (P4). In order to obtain a *finite* abstract state space, we also apply counter abstraction, i.e., we cut the state space at layer n (denoted cut level in the following), which implies that property (P1) is only preserved for less than n customers.

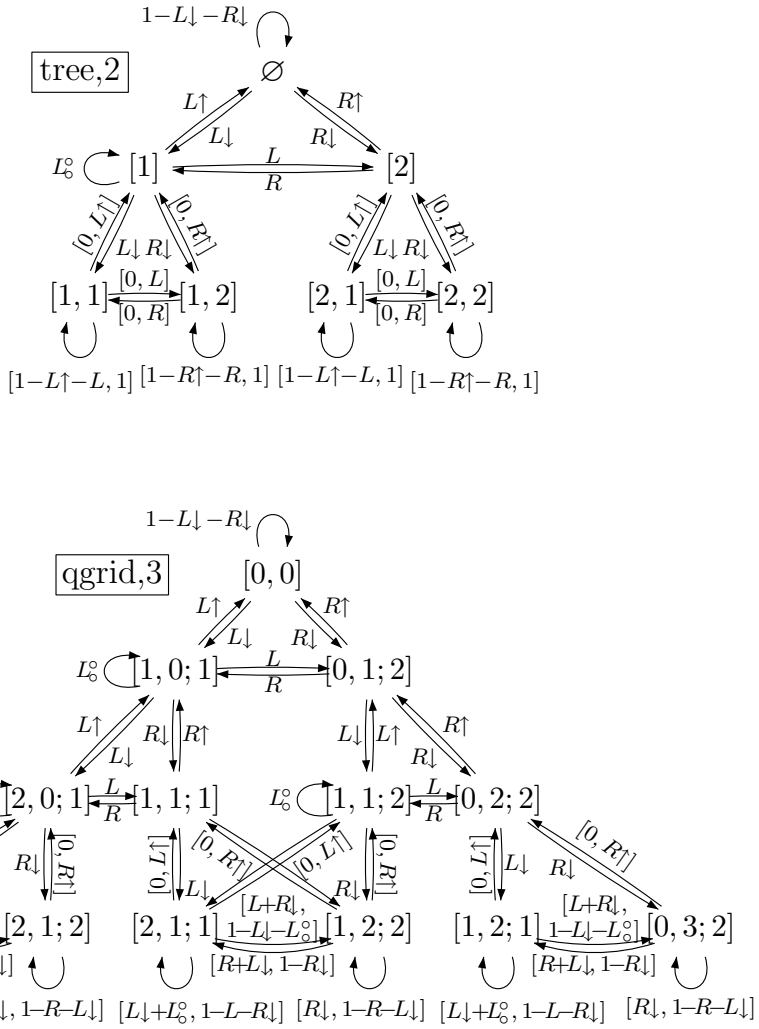


Figure 4.9: Interval abstractions $\mathcal{M}_{tree,2}$ and $\mathcal{M}_{qgrid,3}$ for $L\uparrow + L^\circ = R\uparrow$, $L + L_\circ = R$, and $L^\circ_\circ = L^\circ + L_\circ$.

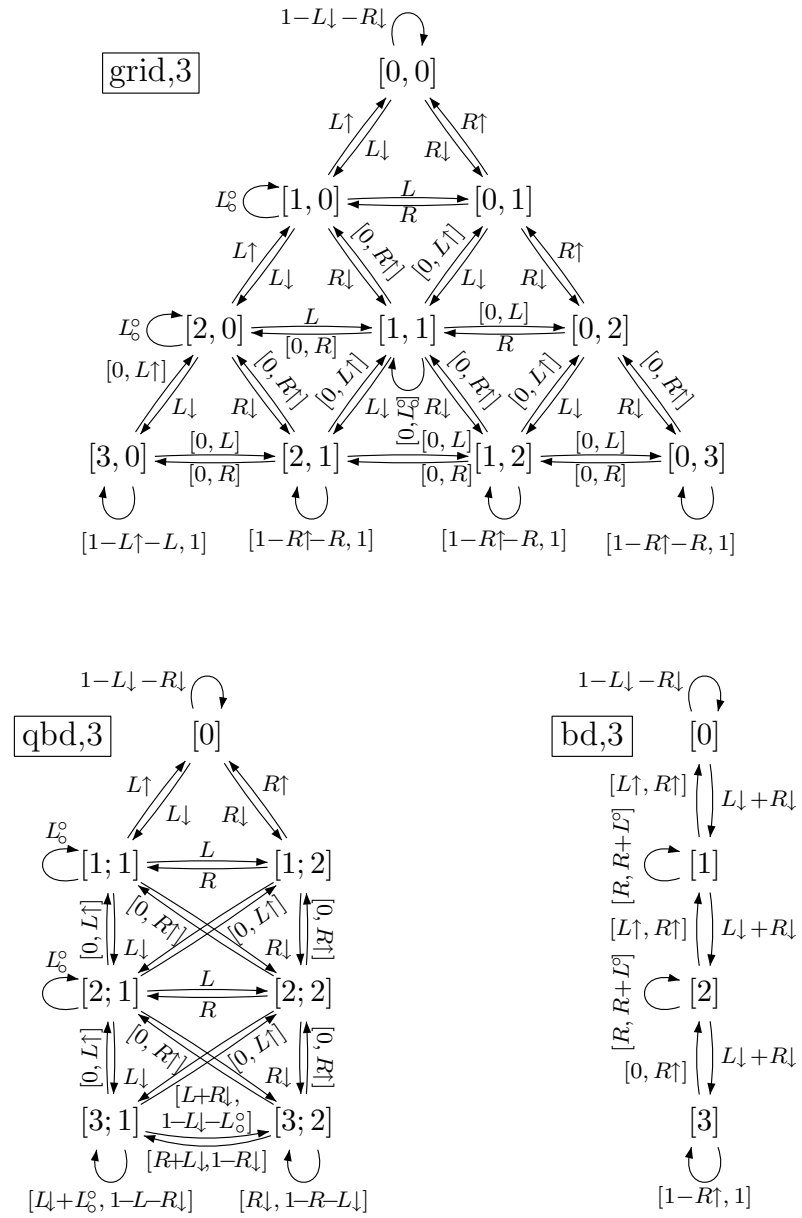


Figure 4.10: Interval abstractions $\mathcal{M}_{grid,3}$, $\mathcal{M}_{qbd,3}$ and $\mathcal{M}_{bd,3}$ for $L\uparrow + L^\circ = R\uparrow$, $L + L_\circ = R$, and $L^\circ_\circ = L^\circ + L_\circ$.

Let $\mathcal{M}$ be a tree-structured QBD with state space S. For partitioning scheme $ps \in \{tree, qgrid, grid, qbd, bd\}$ and cut level $n \in \mathbb{N}_{>0}$, we define abstraction function $\alpha_{ps,n} : S \to S_{ps,n}$ as follows. For $\vec{x} = (x_1, x_2, \ldots, x_m) \in S$, let

$$\alpha_{tree,n}(\vec{x}) = \begin{cases} [\vec{x}] & \text{if } m < n, \\ [\vec{x}{\downarrow_n}] & \text{otherwise;} \end{cases}$$

$$\alpha_{qgrid,n}(\vec{x}) = \begin{cases} [\#_1\vec{x}, \ldots, \#_d\vec{x};\ x_m] & \text{if } m < n, \\ [\#_1\vec{x}{\downarrow_{n-1}} + \#_1(x_m), \ldots, \\ \ \#_d\vec{x}{\downarrow_{n-1}} + \#_d(x_m);\ x_m] & \text{otherwise;} \end{cases}$$

$$\alpha_{grid,n}(\vec{x}) = \begin{cases} [\#_1\vec{x}, \ldots, \#_d\vec{x}] & \text{if } m < n, \\ [\#_1\vec{x}{\downarrow_n}, \ldots, \#_d\vec{x}{\downarrow_n}] & \text{otherwise;} \end{cases}$$

$$\alpha_{qbd,n}(\vec{x}) = \begin{cases} [m;\ x_m] & \text{if } m < n, \\ [n;\ x_m] & \text{otherwise;} \end{cases}$$

$$\alpha_{bd,n}(\vec{x}) = \begin{cases} [m] & \text{if } m < n, \\ [n] & \text{otherwise.} \end{cases}$$

For example, when using the *grid* scheme, all states with the same number of jobs (up to the n-th queued job) in phases 1, 2, ..., d, respectively, are grouped together.

Scheme *bd* preserves property $(P1)$ only, *grid* additionally preserves $(P2)$, whereas *qbd* additionally preserves $(P3)$. Scheme *tree* preserves all properties and *qgrid* preserves all but $(P4)$. Interval abstractions of the binary tree-structured QBD from Figure 4.8 (right) are shown in Figures 4.9 and 4.10. From those, it becomes clear that the partitioning schemes are named after the structure of the abstract models. Schemes *bd* and *qbd* yield chain-like structures similar to (quasi)-birth death processes, where the *qbd* scheme enhances the *bd* scheme by storing the phase of the job currently in service. Similarly the *qgrid* scheme enhances the *grid* scheme. The order of the abstract models size is given in the first row in Table 4.3.

ps	tree	qgrid	grid	qbd	bd
$\lvert S_{ps,n}\rvert$	$\mathcal{O}(d^n)$	$\mathcal{O}(d\cdot\binom{d+n}{d})$	$\mathcal{O}(\binom{d+n}{d})$	$\mathcal{O}(d\cdot n)$	$\mathcal{O}(n)$
#distrs	$< 1.5\times$	$< d\times$	$< d\times$	$< d\times$	$\leq d\times$

Table 4.3: Sizes of abstract models and average numbers of distributions per state (for $d > 1$).

Analysis. In the relevant literature, the analysis of (tree-structured) QBDs mostly focuses on steady-state probabilities as these can be obtained using matrix-geometric techniques [BLM03]. Another important performance measure is the *utilization* of a queue, i.e. the fraction of time the queue is in a busy state. A sensibly designed queue should have a utilization strictly below 100% as otherwise, on the long run, the number of waiting jobs will grow ad infinitum. On the other hand, a very small utilization might indicate an oversized design.

The measure we are concerned with is: *"If the queue is filled up to a certain level, what is the probability for the system to process all but k jobs within t time units?"* New jobs that arrive while serving older jobs should be completed as well. This measure is a typical example for a time-bounded reachability property and cannot be computed using steady-state analysis. Transient analysis however has received scant attention; the only existing approach is approximate [vVvHB06]. Recently, direct techniques based on uniformization or variants thereof [Gra91], have been proposed for reachability properties for general infinite-state CTMCs [ZHHW08] and for highly structured infinite-state CTMCs, such as standard QBDs [RHC07] and Jackson queueing networks [RH08]. However, they all lead to an exponential blow-up when applied to tree-structured QBDs. Whereas the techniques presented in the previous sections guarantee to yield upper and lower bounds of reachability probabilities, [vVvHB06] yields arbitrary approximations. In

addition, applying transient analysis to compute timed reachability probabilities requires an amendment of the CTMC which destroys the tree structure; therefore [vVvHB06] cannot be directly applied to this setting.

In the following, we perform extensive experiments for phase-type service distributions of different orders and analyze the influence of parameter setting and partitioning scheme on the quality of the results and on the size of the resulting abstract state space. To get an impression of how the system evolves over time, we compute probability bounds for gradually increasing time bounds up to a point where the system approaches an equilibrium. The average number of distributions per state never exceeds d (cf. second row in Table 4.3). Since we investigate phase-type service distributions of the order $d \leq 5$, with MDP abstraction, we obtain models that are not much larger than the ones obtained by interval abstraction. Due to Theorem 3.4.2, we therefore stick to MDP abstraction in the following. Abstract models are generated using a Java tool, and the analysis is done using MRMC version 1.4 [KZH+09]. All experiments were run on a standard desktop computer (Athlon X2 3800+, 2GB RAM).

E1. Firstly, we compare the *precision* of the abstractions induced by the partitionings presented in the previous section, check if the results are consistent and provide results for two sets of rates. We choose a low cut level for all partitioning schemes and goal level $k = 0$. Note that $k = 0$ is a special case as the ordering of the jobs is of no relevance in the concrete model as all jobs need to be served anyway before the goal level is reached. Hence, for all partitioning schemes preserving the number of customers per phase, as *tree*, *qgrid* and *grid*, the imprecisions have to result from the choice of the cut level.

E2. In the second experiment, we investigate the influence of the cut level on the imprecisions occurring for the special case $k = 0$. We focus on the *grid* scheme in this experiment.

E3. In a third experiment we compare the *precision* of the *grid*-abstractions for varying goal levels k and fixed cut level.

E4. We then present a refinement of the grid abstraction, where the tree is built in full breadth up to level c and the grid abstraction is applied for higher levels. We present results for varying c, fixed $k = 4$ and given cut level.

E5. In an additional experiment, we choose the cut levels such that the resulting abstract state spaces are of approximately the same size. Apart from the quality of the results, we compare the time it takes to generate and analyze the abstract models.

E6. To emphasize the generality of our approach we present results for the refined grid abstraction, as presented in Experiment 4, for a phase-type 5 distribution.

As standard parameters we choose $L{\downarrow} = 2$, $R{\downarrow} = 3$, $L = 4$, $R = 5$, $L{\uparrow} = 7.5$, $R{\uparrow} = 10$, with a resulting utilization, i.e. the percentage of time the queue is busy in the long run, of 57%.

E1. Partitioning schemes:
We compare three abstractions, $\mathcal{M}_{tree,12}$, $\mathcal{M}_{grid,12}$ and $\mathcal{M}_{bd,12}$, where we choose initial states that simulate $(1, 2, 1, 2, 1, 2, 1, 2)$ in the concrete model, namely $[1, 2, 1, 2, 1, 2, 1, 2]$, $[4, 4]$, and $[8]$ respectively, and take the empty queue as goal state. The resulting lower and upper probability bounds, for these abstractions are shown in Figure 4.11 (top) where the x-axis shows the time bounds. As expected, the most accurate results are obtained using *tree* abstraction. However, the difference between the obtained upper and lower bound is rather large for medium and large time bounds since the cut level 12 is too close to the initial state. The second best abstraction is *grid* abstraction, that is almost as good as *tree* abstraction, especially for low time

bounds. The *bd* abstraction, while having the least memory requirements, is clearly outperformed on the whole range of time bounds. Note that, for large time bounds, results are not as bad as for medium ones. This comes from the fact that upper bounds are derived by assuming that jobs are processed rather quickly while for lower bounds it is assumed that jobs are processed slowly. However, in both cases all jobs are processed eventually, due to a utilization which is less than one.

The results for the other partitionings are presented in Figure 4.11 (bottom). The initial states $[4, 4; 2]$, $[4, 4]$, and $[8; 2]$ simulate $(1, 2, 1, 2, 1, 2, 1, 2)$. *Qbd* abstraction performs significantly better than *bd* abstraction (left), but still is outperformed by *grid* abstraction for low and medium time bounds. For large time bounds (starting at about 4.75), *qbd* abstraction yields better lower bounds, even though the state space is significantly smaller than for *grid* abstraction. This comes from the fact that *qbd* abstraction *is aware* of the job that is currently processed. If that job is processed rather quickly, a transition to a higher level (away from the cut level) is more probable. This shows that *grid* and *qbd* abstractions are incomparable. Finally, combining the advantages of *grid* and *qbd* yields the inner pair of curves (*qgrid*). This scheme outperforms the other two abstractions and comes close to the *tree* scheme's results (left), however, with a drastically lower memory consumption (with a factor of 217).

Now, we consider an alternative set of parameters where rates are much more diverse: $L\downarrow = 1$, $R\downarrow = 5$, $L = 3$, $R = 15$, $L\uparrow = 8$, $R\uparrow = 20$, with a resulting utilization of 46%. As Figure 4.12 indicates, both *bd* and *qbd* abstractions perform badly with these parameters. The reason is that only the total number of jobs is stored and therefore in the best (worst) case, it is assumed that *only* short (long) jobs are present in the queue. On the contrary, the performance of *grid* and *qgrid* abstraction is excellent (lower and upper bounds are overlapping in the graph).

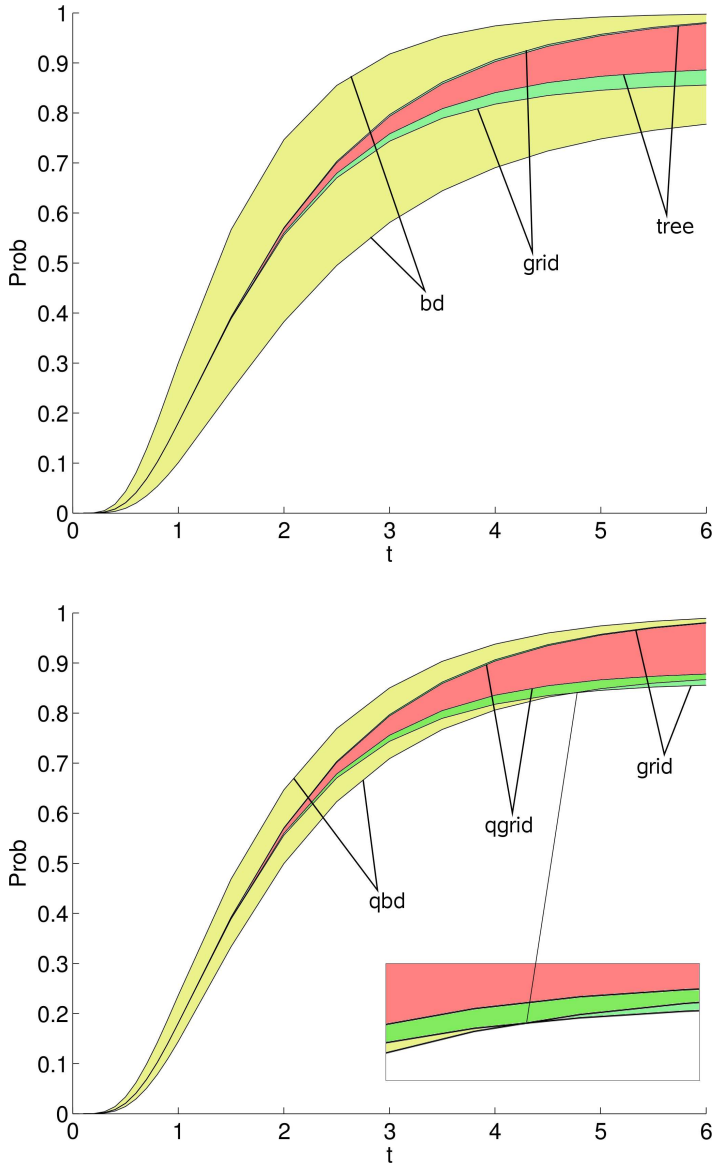


Figure 4.11: E1: Upper and lower bounds for *bd*, *grid*, *tree* abstractions (top) and *qbd*, *qgrid*, *grid* abstractions (bottom) with the same cut level and $k = 0$.

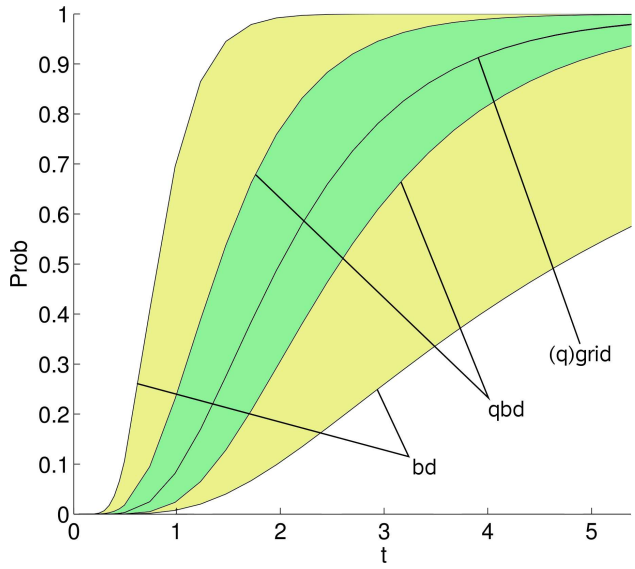


Figure 4.12: E1 with an alternative parameter set: Upper and lower bounds for *bd*, *qbd* and *(q)grid* abstractions with the same cut level and $k = 0$.

E2. Influence of the cut level:
In our second experiment, we investigate the influence of the cut level on the quality of the results. We show upper and lower bounds on the probability to reach the empty state from initial state $[4, 4]$ with *grid* abstraction and increasing cut level. The parameters are chosen as in experiment **E1**, except $L\downarrow = \frac{3}{10}$ and $R\downarrow = 5$ are larger, resulting in a very high utilization of 96% and, hence, a much larger range on the time-axis.

Figure 4.13 shows that *grid* abstraction is capable of providing lower and upper bounds that differ marginally, for large enough cut levels. As all the jobs need to be served to reach the empty state, the ordering of the jobs does not matter. Hence, for cut level $n \to \infty$, *grid* abstraction contains all the necessary information, if the empty state is chosen as goal state.

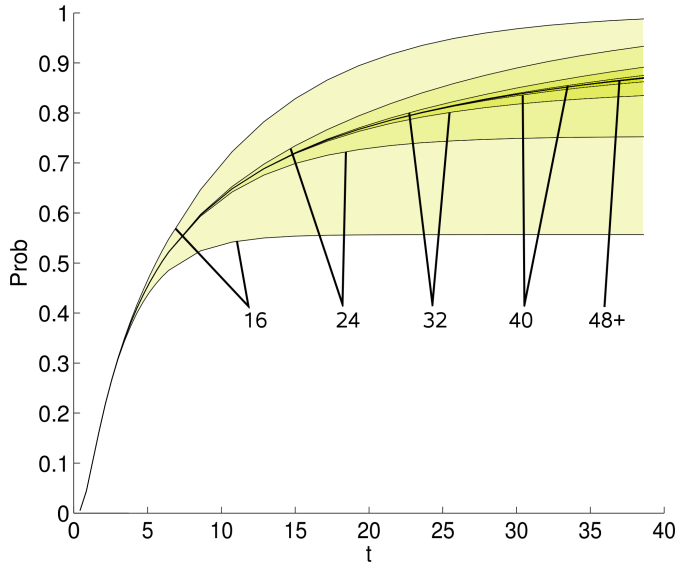


Figure 4.13: E2: Upper and lower bounds for *grid* abstractions with increasing cut levels and $k = 0$.

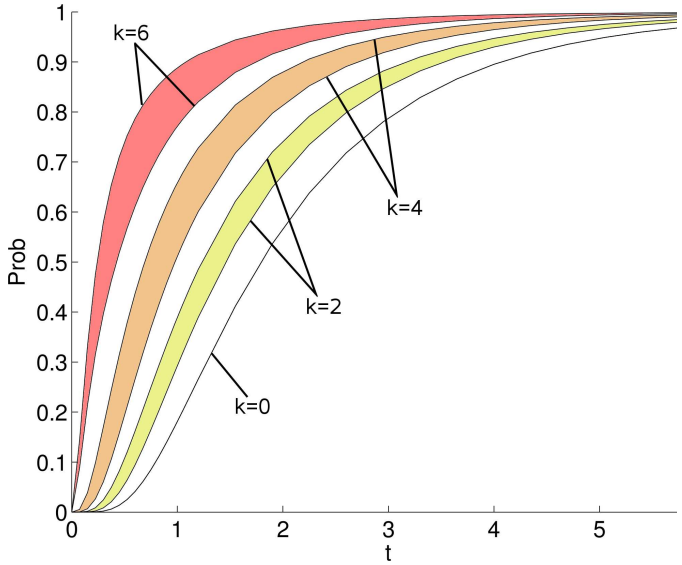


Figure 4.14: E3: Upper and lower bounds for *grid* abstraction with increasing goal levels k and fixed cut level.

E3. Influence of the goal level:
We compare results obtained with *grid* abstraction for a large enough cut level for increasing goal levels k. Figure 4.14 shows that with growing goal levels the difference between upper and lower probability bounds increases. This is due to the fact, that in this setting *grid* abstraction does not contain all necessary information, as the ordering of the jobs in the queue influences the time that is needed to serve all but k jobs.

E4. Refinement:
To obtain more precise results, the *grid* partitioning scheme is *refined*. Lack of information on the first jobs in the queue leads to less tight bounds. Hence, the abstract states are refined according to the ordering of the first jobs. We define $\alpha_{grid,c,n}$ such that states are merged for which the order of the first c job phases is the same and where the numbers of job phases for jobs $c+1$ up to n are equal as well:

$$\alpha_{grid,c,n}(\vec{x}) = \begin{cases} [\vec{x}{\downarrow_c}; \#_1(x_{c+1}, \ldots, x_m), & \\ \quad \ldots, \#_d(x_{c+1}, \ldots, x_m)] & \text{if } m < n, \\ [\vec{x}{\downarrow_c}; \#_1(x_{c+1}, \ldots, x_n), & \\ \quad \ldots, \#_d(x_{c+1}, \ldots, x_n)] & \text{otherwise.} \end{cases}$$

This still is a very natural partitioning scheme. For example, for a binary tree-structured QBD, $\alpha_{grid,2,8}((1)) = [(1); 0, 0]$ and $\alpha_{grid,2,8}((1,2,1,1,2,1)) = [(1,2); 3, 1]$.

In Figure 4.15, results for goal level $k = 4$ and refinement levels $c \in \{0, 2, 4\}$ are shown. If the refinement level c equals the goal level k, and if the cut level is large enough, the obtained lower and upper bounds are extremely tight.

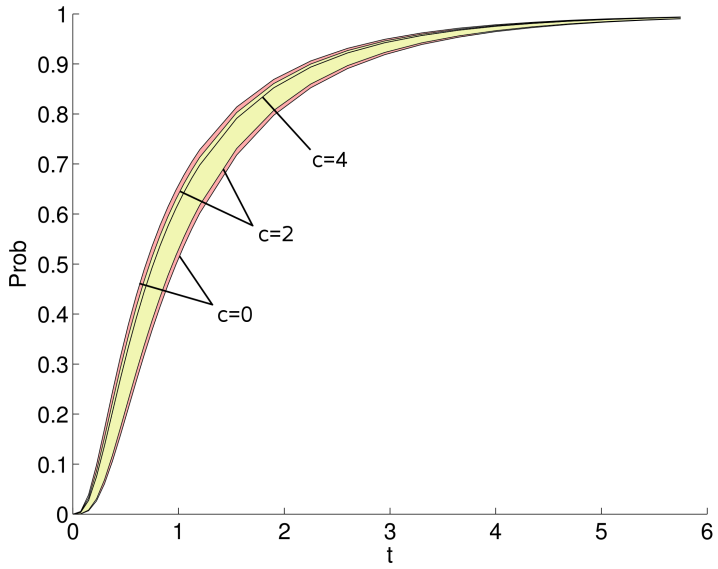
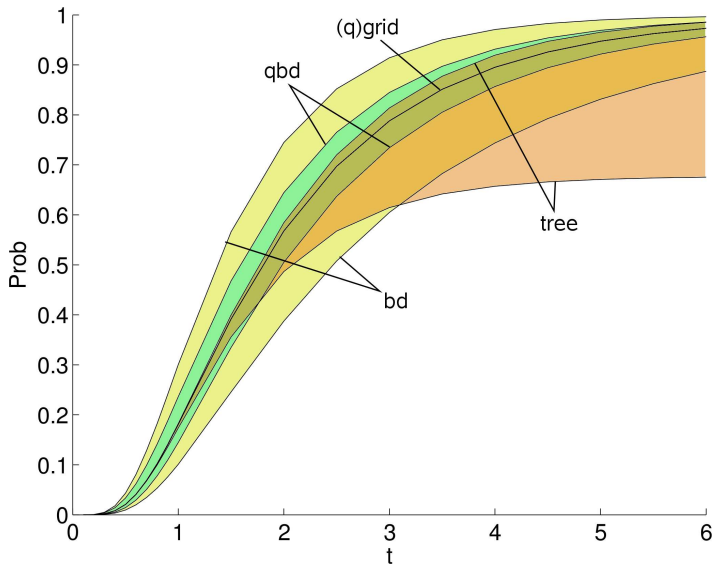


Figure 4.15: E4: Upper and lower bounds for *refined grid* abstraction with increasing c for $k = 4$ and fixed cut level.

E5. Similar sized state spaces:
While in the first experiment the cut level remained constant, we now compare the different abstractions, while keeping the size of the abstract state space approximately the same. We select the same initial states and the same goal state as in the first experiment, as well as the first set of rates. The goal of this experiment is to determine which properties are more important for the quality of the results, given a maximal number of states.

The cut level, the number of states in the abstract model and the time for generating the abstract state space and for computing probability bounds are given in the table in Figure 4.16. While *tree* and *bd* abstraction use the least computation time, their results, shown in Figure 4.16, are rather imprecise compared to the other abstractions. Long-term behavior is not captured very well by *tree* abstraction as the cut level is close to the initial state. For small time bounds, *bd* abstraction is the worst. Yet, it catches up with increasing time bounds, as it has the largest depth and therefore very little

probability mass is *lost* in the cut level. However, neither one can compete with *grid* and *qgrid* abstractions for which the lower and upper bounds are almost identical. Here, the *grid* scheme is favorable as the computation time is 12% smaller than for *qgrid*.



	depth	states	time/ms
tree	10	2047	989
qgrid	45	2071	1546
grid	63	2080	1378
qbd	1023	2047	1434
bd	2046	2047	1092

Figure 4.16: E5: Upper and lower bounds for *bd*, *tree*, *qbd* and *(q)grid* abstractions with similar sized state spaces for $k = 0$.

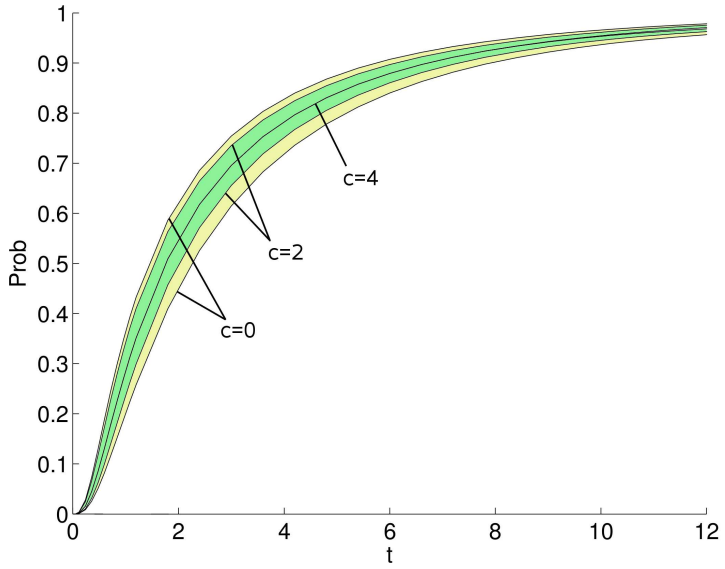


Figure 4.17: E6: Upper and lower bounds for *refined grid* abstractions with increasing c for $k = 4$ and fixed cut level.

E6. Phase-type 5 service distribution:
To emphasize the generality of our approach, we present results on a M|PH_5|1 queue with preemptive LIFO and a utilization of 77% and a phase-type distribution with the following parameters:

$$r{\downarrow} = \begin{pmatrix} 0.5 & 1 & 0.75 & 1.25 & 1.5 \end{pmatrix}$$
$$r{\uparrow} = \begin{pmatrix} 4 & 5 & 6 & 7 & 8 \end{pmatrix}$$
$$r = \begin{pmatrix} 2 & 0.5 & 2 & 1 & 1.5 \\ 0 & 2 & 1.5 & 2.5 & 1 \\ 0 & 0 & 2 & 2.5 & 2.5 \\ 0 & 0 & 0 & 4 & 3 \\ 0 & 0 & 0 & 0 & 7 \end{pmatrix}$$

For initial state $(2, 2, 2, 1, 1)$, goal level $k = 4$ and cut level 24, Figure 4.17 shows upper and lower bounds for the *refined grid* abstraction with $c \in$

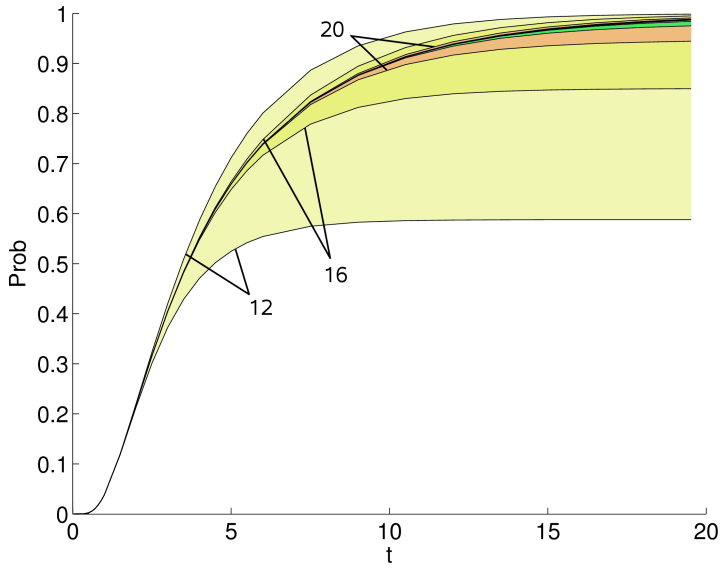


Figure 4.18: E6: Upper and lower bounds for *grid* abstractions with increasing cut level and $k = 0$.

$\{0, 2, 4\}$. Again, for matching goal and refinement level, the results differ only marginally.

In the following, we investigate how the time for abstraction and for the computation of probability bounds scales with the cut level. The goal level is fixed to $k = 0$.

The results in Figure 4.18 and in Table 4.4 show that for large enough cut level, the *grid* abstraction is capable of providing upper and lower bounds that differ only marginally and scale well with the size of the abstract model.

Note that the differences between upper and lower bounds of reachability probabilities listed in Table 4.4 cannot get smaller than $\varepsilon = 10^{-6}$ because MRMC computes ε-approximations of the probability bounds. In contrast, when forgoing abstraction, a massive number of states has to be considered (see Table 4.4, right part). For example, when computing time-bounded reachability for time-bound $t = 7.5$, by using abstraction, one obtains an ε-approximation (with $\varepsilon = 10^{-6}$) from an abstract model with less than

	grid abstraction								uniformization	
diff	grid 12	grid 16	grid 20	grid 24	grid 28	grid 32	grid 36	grid 40	trunc	$\approx$ states
2.5	0.0224	0.001	10^{-6}	10^{-6}	10^{-6}	10^{-6}	10^{-6}	10^{-6}	185	10^{129}
t 7.5	0.3117	0.0580	0.0062	0.0004	10^{-5}	10^{-6}	10^{-6}	10^{-6}	270	10^{188}
15	0.4054	0.1345	0.0376	0.0086	0.0015	0.0002	$2 \cdot 10^{-5}$	$3 \cdot 10^{-6}$	398	10^{278}
states	6188	20349	53130	118755	237336	435894	749398	1221759		
distributions	28666	96901	256796	579151	1164206	2146761	3701296	6047091		
time (h:mm:ss)	0:00:26	0:01:33	0:04:15	0:09:50	0:20:14	0:38:13	1:07:57	2:06:04		

Table 4.4: E6: Differences between lower and upper bounds in *grid* abstractions vs. size of the state space for uniformization with error bound $\varepsilon = 10^{-6}$.

half a million states. With standard uniformization, 10^{188} states have to be explored in the concrete model in order to compute an approximation with the same guaranteed error bound. However, even with a terabyte of available memory, CTMCs with more than 10^{14} states cannot be dealt with. This shows evidently that abstraction is an attractive approach to computing time-bounded reachability probabilities in tree-structured QBDs.

4.7 Summary and discussion

In this chapter, we showed how to analyze (finite) abstract Markov chains. The key point is the restriction of the set of history-dependent schedulers to *extreme* schedulers that only may choose from a finite set of distributions. We have proven that this restriction is not affecting the infimum and supremum of probability measures. Furthermore, ideas used in that proof have been exploited for developing efficient, polynomial-time algorithms for computing step-bounded and time-bounded reachability probabilities. With the results we obtained on the preservation of reachability probabilities in abstract models simulating a concrete one, we devised a three-valued model checking framework for PCTL and CSL based on such abstractions and showed its soundness. Finally, the feasibility of abstraction based model checking has been demonstrated by a case study from queueing theory for which other known techniques are only applicable to very small instances.

Related work. We now discuss some of the related work on abstraction for probabilistic systems and we also comment on existing links to following chapters when appropriate. Note that most of the advanced techniques have been developed for discrete-time models and cannot be lifted to the continuous-time setting easily.

As for interval abstraction, the main idea behind *magnifying lens abstraction* [dAP07] is to partition the state space and to compute the minimal and maximal probabilities for moving from one partition to another. The main

difference is that probabilities are considered for leaving a partition towards another partition (without visiting a third one) *within an arbitrary amount of steps*. This implies that the number of steps in the concrete model that correspond to one step in the abstract model is not bounded. As a result, step-bounded reachability probabilities are not preserved by *magnifying lens abstraction*. A further restriction is that the authors consider MDPs with finite state space only. However, combining interval abstraction (obtaining a finite abstract model) with magnifying lens abstraction (to compute unbounded reachability probabilities) might lift this restriction. In Chapter 5 of this thesis, we develop a generalization of interval abstraction that is partially inspired by this technique; instead of an *arbitrary* amount of steps, a fixed number of subsequent steps will be considered for abstraction.

Game-based abstraction [KNP06, KH09] is a technique that basically extends MDP abstraction [DJJL01]. When considering DTMCs as concrete models, the results obtained via game-based abstraction and MDP abstraction are identical. The important difference is that instead of one source of nondeterminism as in MDPs ($1\frac{1}{2}$-player games), in the abstract model used for game-based abstraction (so-called *simple stochastic games* or $2\frac{1}{2}$-player games) there are two independent sources of nondeterminism. This allows for the distinction between nondeterminism that is present in the original model, a concrete MDP, and the nondeterminism introduced during the process of abstraction. Then, for an optimal (maximal) solution in the concrete MDP, lower (upper) bounds can be computed by maximizing the strategy of the first player and minimizing (maximizing) the strategy of the second player. Analogously, lower and upper bounds can be computed for minimal solutions in the concrete MDP. The algorithms that are used for the computation of these bounds can – to our knowledge – not be lifted to a continuous-time variant of simple stochastic games easily. However, lifting *game-based abstraction* to the continuous-time setting requires substantial progress on the analysis of continuous-time $2\frac{1}{2}$-player games. With the availability of positive

results in that area, abstraction for *interactive Markov chains* as presented in Chapter 6 could be generalized in a similar way as MDP abstraction is by game-based abstraction.

Very recently, an abstraction technique for infinite-state continuous-time Markov chains – *sliding window abstraction* [HMW09] – has been introduced that takes advantage of the fact that in certain models, for a given fraction of time, only small fragments of the infinite state space are occupied with significant probabilities. Basically, the technique hides all states that are occupied with probabilities below a given threshold: a *window* is left open through which parts of the original state space are still visible. As time moves on, the probability mass may decrease in some states and increase in others such that the fraction of states to be considered changes constantly: the window is said to *slide* over the state space. The sustained loss of precision – when disregarding states with a positive probability that is below the threshold – is traceable, i.e. a maximal error for the resulting probabilities can be computed along the analysis. The technique has been shown to work well for a number of models from the area of systems biology. Combining *sliding window abstraction* with ideas presented in this thesis would be an interesting subject for future work.

In [Hut05], a more theoretical point of view is taken. Basically, it is shown that for infinite-state discrete-time Markov chains and PCTL formulas *without until*, finite-state abstractions can be constructed for which the same model checking results are obtained as for the original model. The abstraction proposed for PCTL formulas *including until* can be seen as a predecessor of interval abstraction as it has been introduced in the previous chapter. Algorithmic issues, such as normalization, are not elaborated on. Further, the continuous-time setting is not considered. More theoretical results related to this thesis are to be found in [HPW09], where PCTL completeness results are provided for an abstraction framework for Markov chains that uses three-

valued labels (but does not incorporate nondeterministic choices of transition probabilities) in abstract states.

5

Abstraction by Erlang's method of stages

In this chapter, one downside common to MDP and interval abstraction – in fact, to all abstraction techniques that rely on stepwise simulation – will be isolated and a solution is proposed; we will stick to interval abstraction, however, the idea is easily transferable. Furthermore, we focus on abstraction and the computation of probability bounds for step- and time-bounded reachability for continuous-time Markov chains; step-bounded reachability for the discrete-time case will not be made explicit. While we provide algorithms for the computation of bounds for reachability probabilities, we refrain from lifting the analysis results to model checking as it may be done just as for AMCs as shown in the previous chapter (cf. [KKLW08]).

Put in a nutshell, the approach in this chapter is to generalize interval abstraction towards considering *transition sequences* of a given length k as *abstract transitions*. We aim at abstractions that k-stepwise simulate the original model. As abstract model we introduce a generalization of ACTMCs where state residence times are no longer exponentially distributed but Erlang-j distributed (see Section 2.4.1) with $j \in \{1, \ldots, k\}$; we identify a class of schedulers that is well-suited for obtaining lower and upper bounds of reachability probabilities and we provide efficient algorithms that are inspired by these classes of schedulers. The chapter is concluded with a case study from systems biology, demonstrating the feasibility of the presented approach.

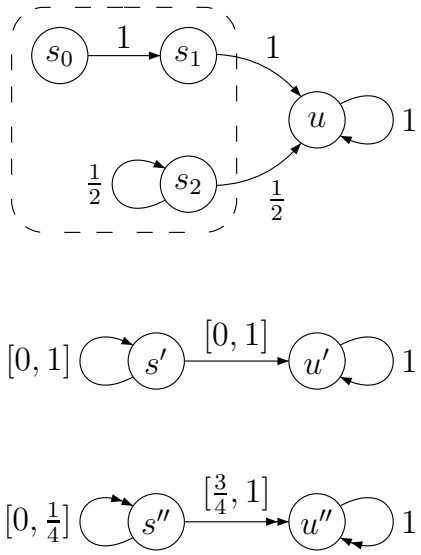


Figure 5.1: Concrete CTMC (top), an AMC abstraction (middle) and an abstraction by an Erlang-2 interval process (bottom).

5.1 Erlang-k interval processes

In the following, let us first investigate an example that reveals a shortcoming of stepwise simulation based abstraction methods. While abstraction as presented before is correct, we will find that in certain cases it may yield very coarse abstractions. However, we will see that it is possible to obtain better abstractions by generalizing the ideas from the previous chapters.

Example 5.1.1. Consider the Markov chain $\mathcal{M}$ shown in Figure 5.1 (top) and abstraction function $\alpha : S \to S'$ with $S' = \{s', u'\}$ where $\gamma(s') = \{s_0, s_1, s_2\}$ and $\gamma(u') = \{u\}$. Applying interval abstraction results in the abstract model $\alpha(\mathcal{M})$ depicted in Figure 5.1 (middle). As from s_0 the (time-abstract) probability to move to an s-state in exactly one step is 1, we have $\mathbf{P}^u(s', s') = 1$ and consequently $\mathbf{P}^l(s', u') = 0$. Further, the probability to move from s_1 to u is 1 and therefore $\mathbf{P}^u(s', u') = 1$ and $\mathbf{P}^l(s', s') = 0$. While $\alpha(\mathcal{M})$ is simulating $\mathcal{M}$ (and therefore allows for simulation of the exact

behavior of the concrete model $\mathcal{M}$), it is a pretty coarse abstraction: when always choosing the self-loop in state s' with probability 1, state u' will never be reached.

A more narrow interval is obtained when considering two consecutive transitions. For both s_0 and s_1, the (time-abstract) 2-step probability to move to u is 1 and for s_2, the 2-step probability to move to u is $\frac{3}{4}$. Thus, as depicted in Figure 5.1 (bottom), the probability to take the abstract transition from s'' (representing all s-states) to u'' (representing u) is within $\frac{3}{4}$ and 1; and almost surely, on the long run, state u'' will be reached from s'', no matter how values from the intervals are chosen.

Note, that Figure 5.1 (bottom) does not show an AMC and clearly, the relation between $\mathcal{M}$ and this abstraction is not a stepwise probabilistic simulation as it is the case for $\alpha(\mathcal{M})$.

In the remainder of this chapter we develop a methodology for abstractions as described in the above example. First, we introduce *Erlang-k interval processes* serving as abstract model (cf. Figure 5.1, bottom), then we establish a formal relation and discuss preservation of reachability probabilities.

Erlang-k interval processes are determined by two ingredients: an abstract discrete-time Markov chain as defined in Section 3.2 and a Poisson process, cf. Section 2.4.1. The former determines the probabilistic branching whereas residence times are governed by the latter. More precisely, the state residence time is the time until j further arrivals occur according to the Poisson process where $j \in \{1, \ldots, k\}$ is nondeterministically chosen. Thus, the residence times are Erlang-j distributed. Formally, the time until j arrivals have occurred in the Poisson process within t time units is given by the cumulative distribution function $F_{\lambda,j}(t) = 1 - \sum_{i=0}^{j-1} e^{-\lambda \cdot t} \cdot \frac{(\lambda \cdot t)^i}{i!}$. Thus, the probability for $\{k, k+1, \ldots, k+j-1\}$ arrivals with $j \geq 1$ is given by

$$\psi_{\lambda,t}(k, j) = \sum_{i=k}^{k+j-1} e^{-\lambda \cdot t} \cdot \frac{(\lambda \cdot t)^i}{i!}.$$

Definition 5.1.1 (Erlang-k interval process)**.** An *Erlang-k interval process* (for short, E$_k$IP) is a tuple $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0, \lambda, k)$, with S, $\mathbf{P}^l$, $\mathbf{P}^u$, L, μ_0 as for AMCs, $\lambda \in \mathbb{R}_{>0}$, a parameter of the associated Poisson process and $k \in \mathbb{N}_{>0}$.

Notions of paths, paths fragments, sets of paths and so forth are as for CTMCs. We depict E$_k$IPs by drawing the state-transition graph of the discrete part, i.e., the associated ADTMC with transitions labeled by $[\mathbf{P}^l(s,s'), \mathbf{P}^u(s,s')]$. The number of arrow heads distinguishes E$_k$IPs from ACTMCs (cf. Figure 5.1). The Poisson process that determines the residence times, as well as the marking of the initial state are often omitted.

An Erlang-1 interval process is an ACTMC. If additionally all intervals are singletons, the process is equivalent to a CTMC with $\mathbf{P}^l = \mathbf{P}^u = \mathbf{P}$. The set of transition probability functions for $\mathcal{M}$ is given by

$$\mathbf{T}_{\mathcal{M}}(s) = \{\mu \in \mathit{distr}(S) \mid \forall u \in S : \mu(u) \in [\mathbf{P}^l(s,u), \mathbf{P}^u(s,u)]\}$$

and the set of parameters for the residence time distribution is $\{1, \ldots, k\}$ for all $s \in S$. As for AMCs, for E$_k$IPs, not all combinations of values from the intervals yield probability distributions. The same normalization techniques as presented for AMCs in Section 3.3, can be applied here to disregard such combinations.

Each of the two ingredients of E$_k$IPs, the ADTMC and the Poisson process, are sources of nondeterminism. Hence, a scheduler has to decide for

1. a distribution according to the transition probability intervals, and
2. the number $j \in \{1, \ldots, k\}$ of arrivals in the Poisson process.

For the same reasons as for AMCs (cf. Section 4.1) we are considering history dependent schedulers. For E$_k$IP $\mathcal{M}$, a *history-dependent deterministic scheduler* is a function $D : \mathit{Path}f^{\mathcal{M}}_{abs} \to \mathit{distr}(S) \times \{1, \ldots, k\}$ such that $D(\varrho) \in \mathbf{T}_{\mathcal{M}}(\varrho\downarrow) \times \{1, \ldots, k\}$ for all $\varrho \in \mathit{Path}f^{\mathcal{M}}_{abs}$. The set of all history-dependent deterministic schedulers of $\mathcal{M}$ is denoted $\mathcal{D}^{\mathcal{M}}$. A scheduler $D \in$

$\mathcal{D}^{\mathcal{M}}$ induces a stochastic process with probability measure $\Pr^D$, for which $\Pr^D(\mathcal{C}(s_0)) = \mu_0(s_0)$ for all $s_0 \in S$, and for $s_{i+1} \in S$ and $I_i = [0, x_i] \subseteq \mathbb{R}_{\geq 0}$ with $i \in \{0, \ldots, n\}$ for all $n \in \mathbb{N}$,

$$\begin{aligned}
&\Pr^D(\mathcal{C}(s_0\, I_0 \ldots I_n\, s_{n+1})) \\
&= \Pr^D(\mathcal{C}(s_0\, I_0 \ldots I_{n-1}\, s_n)) \cdot F_{\lambda,j_n}(\sup I_n) \cdot \mu_{n+1}(s_{n+1}) \\
&= \mu_0(s_0) \cdot \textstyle\prod_{i=0}^{n} \left(F_{\lambda,j_i}(\sup I_i) \cdot \mu_{i+1}(s_{i+1})\right)
\end{aligned}$$

where $(\mu_{i+1}, j_i) = D(s_0\, s_1\, \ldots\, s_i)$. Similarly, for the time-abstract probability measure induced by D we have $\Pr^D_{emb}(\mathcal{C}(s_0)) = \mu_0(s_0)$ and

$$\Pr^D_{emb}(\mathcal{C}(s_0\, \ldots\, s_{n+1})) = \mu_0(s_0) \cdot \textstyle\prod_{i=0}^{n} \mu_{i+1}(s_{i+1}).$$

Note that in the time-abstract case, the probability measure is identical to the one for ADTMCs.

We are interested in the supremum/infimum (ranging over all schedulers) of the probability of measurable sets of paths. Clearly, the choice of j_i, the number of steps in the associated Poisson process in state s_i, may influence such quantities. For instance, on increasing j_i, time-bounded reachability probabilities will decrease as the expected state residence time (in s_i) becomes longer. However, we delay the discussion on the nondeterministic choice in the Poisson process until Sections 5.3 and 5.4, and focus on the choice of distribution μ_i according to the probability intervals in the following section.

5.2 Abstraction and simulation

In this section, we define a generalized notion of interval abstraction. For $k = 1$, as $\mathbf{P}^1 = \mathbf{P}$, this generalized abstraction operator is identical to abstraction of CTMCs in terms of ACTMCs as defined in Section 3.2.

Definition 5.2.1 (Abstraction)**.** For CTMC $\mathcal{M} = (S, \mathbf{P}, L, \mu_0, \lambda)$ and abstraction function $\alpha_k : S \to S'$ with $k \in \mathbb{N}_{>0}$, the induced abstraction is given by $\alpha_k(\mathcal{M}) = (S', \tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u, \tilde{L}, \tilde{\mu}_0, \lambda, k)$ where for $\mathbf{P}^k$, the k-th power of $\mathbf{P}$,

- for all $s', u' \in S'$:

  $\tilde{\mathbf{P}}^l(s', u') = \inf_{s \in \gamma(s')} \mathbf{P}^k(s, \gamma(u'))$
  $\tilde{\mathbf{P}}^u(s', u') = \sup_{s \in \gamma(s')} \mathbf{P}^k(s, \gamma(u'))$

- for all $s' \in S'$, $ap \in AP$:

$$\tilde{L}(s', ap) = \begin{cases} \top & \text{if } L(s, ap) = \top \text{ for all } s \in \gamma(s') \\ \bot & \text{if } L(s, ap) = \bot \text{ for all } s \in \gamma(s') \\ ? & \text{otherwise} \end{cases}$$

- for all $u' \in S'$: $\tilde{\mu}_0(u') = \mu_0(\gamma(u'))$

Lemma 5.2.1. *For any CTMC $\mathcal{M}$, any abstraction function α and $k \in \mathbb{N}_{>0}$, the induced abstraction $\alpha_k(\mathcal{M})$ is an Erlang-k interval process.*

Proof. The proof follows along the lines of the proof for Lemma 3.2.1. □

Example 5.2.1. Reconsider the CTMC $\mathcal{M}$ from Figure 5.1 (top) with exit rate $\lambda = 1$ and abstraction function $\alpha_k : S \to S'$ with $\gamma(s') = \{s_0, s_1, s_2\}$, $\gamma(u') = \{u\}$. For $k = 1$, the abstract Erlang-1 interval process $\alpha_1(\mathcal{M})$ is actually the depicted ACTMC. When choosing $k = 2$ we obtain the E_2IP $\alpha_2(\mathcal{M})$ depicted in Figure 5.1 (bottom; states are renamed to s'' and u'' respectively).

MDP abstraction could be defined similarly, however, the memory consumption tends to grow with k as shown in the following example.

Example 5.2.2. Consider the CTMC $\mathcal{M}$ in Figure 5.2 (left) and let α be such that $\gamma(s') = \gamma(s'') = \{s_0, s_1, s_2, s_3\}$ and $\gamma(u') = \gamma(u'') = \{u\}$. The interval abstraction $\alpha_2(\mathcal{M})$ is as depicted in Figure 5.2 (bottom, right). With a variant of MDP abstraction defined analogously to the one for interval abstraction in Definition 5.2.1, we would obtain the abstract model depicted in Figure 5.2 (top, right; probabilities for staying in s' are omitted). Note that whereas $k = 1$, i.e. standard MDP abstraction, yields two probability distributions $\{s' \mapsto 1\}$ and $\{s' \mapsto \frac{1}{3}, u' \mapsto \frac{2}{3}\}$ in the abstract state s', for $k = 2$

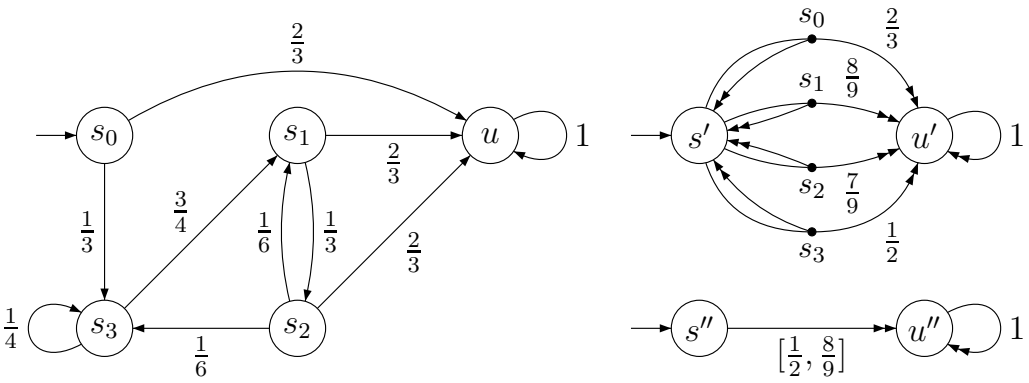


Figure 5.2: A CTMC $\mathcal{M}$ (left), an interval abstraction $\alpha_2(\mathcal{M})$ (bottom, right) and the corresponding MDP abstraction (top, right).

we already obtain four distributions. The maximal number of distributions is bounded by the number of states in each partition, therefore, $k = 3$ does not yield more than four distributions.

To be able to capture the difference between the semantics of CTMCs and E$_k$IPs, we define a variant of probabilistic simulation, expressing that basically one transition in the latter corresponds to k transitions in the former model. Note that the following definition relies on Definition 3.4.1, that lifts a relation on states to a relation on distributions over states.

Definition 5.2.2 (k-step simulation)**.** Let $\mathcal{M} = (S, \mathbf{P}, L, \mu_0, \lambda)$ be a CTMC and $\mathcal{M}' = (S', \tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u, \tilde{L}, \tilde{\mu}_0, \lambda)$ an E$_k$IP. Relation $R_k \subseteq S \times S'$ is a *k-step simulation* on $\mathcal{M}$ and $\mathcal{M}'$ iff for all $s \in S$, $s' \in S'$, sR_ks' implies:

1. for $\mu = \mathbf{P}^k(s, \cdot)$, there exists $\mu' \in \mathbf{T}_{\mathcal{M}'}(s')$ such that $\mu \preceq_k \mu'$
2. for all $ap \in AP$, $\tilde{L}(s', ap) \neq ?$ implies $\tilde{L}(s', ap) = L(s, ap)$

We write $s \preceq_k s'$ if sR_ks' for some k-step simulation R_k, and $\mathcal{M} \preceq_k \mathcal{M}'$ if $\mu_0 \preceq_k \mu_0'$.

Figure 5.3 illustrates how the k-step distribution $\mathbf{P}^k(s, \cdot)$ is related to a distribution on S': k steps in $\mathcal{M}$ are matched with a single step in $\mathcal{M}'$. For

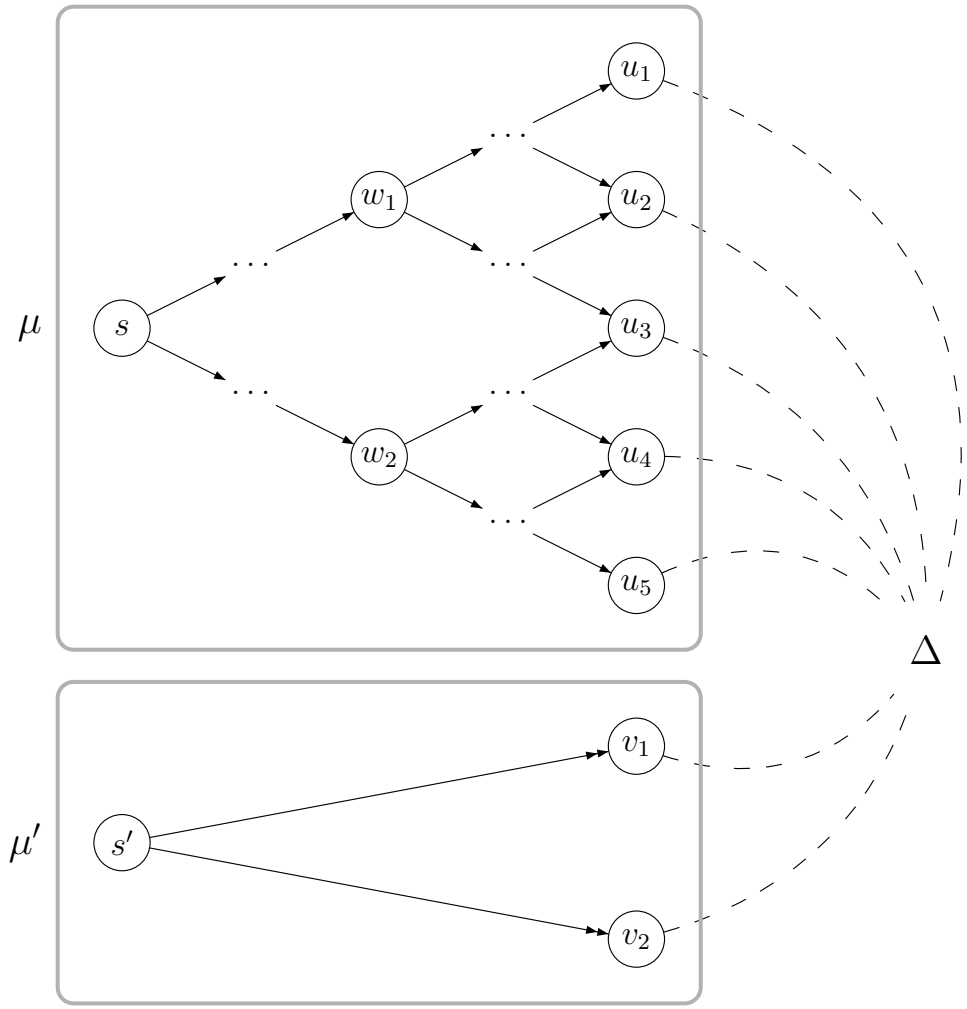


Figure 5.3: k-step simulation

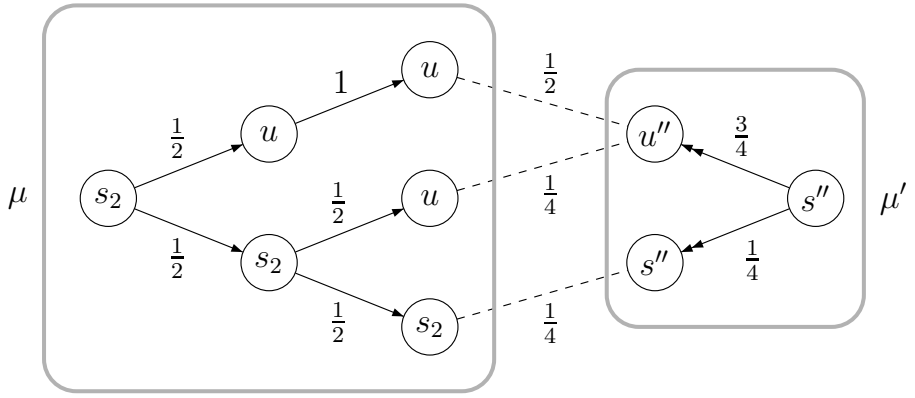


Figure 5.4: 2-step simulation

$k = 1$ this definition is consistent with the standard notion of simulation, cf. Section 3.4.

The following theorem states that for any $k \in \mathbb{N}_{>0}$, abstraction as given in Definition 5.2.1 yields an $\mathrm{E}_k\mathrm{IP}$ which k-stepwise simulates the given CTMC.

Theorem 5.2.2. *Let $\mathcal{M}$ be a CTMC with state space S and $\alpha_k : S \to S'$ an abstraction function with $k \in \mathbb{N}_{>0}$. Then, $\mathcal{M} \preceq_k \alpha_k(\mathcal{M})$.*

Proof. The proof goes along similar lines as the proof for Theorem 3.4.1. □

Example 5.2.3. Let us again consider the CTMC $\mathcal{M}$ from Figure 5.1 (top) and its abstraction $\alpha_2(\mathcal{M})$ shown in Figure 5.1 (bottom). The 2-step probability distribution in state s_2 in $\mathcal{M}$ is $\mu = \{s_2 \mapsto \frac{1}{4}, u \mapsto \frac{1}{2} + \frac{1}{4}\}$ as depicted in Figure 5.4. We choose $\mu'' = \{s'' \mapsto \frac{1}{4}, u'' \mapsto \frac{3}{4}\} \in \mathbf{T}_{\alpha_2(\mathcal{M})}(s'')$ and the obvious weight function, then with $u \preceq_2 u''$ (which can be shown easily) we obtain $s_2 \preceq_2 s''$. In a similar way it can be checked that $s_0 \preceq_2 s''$ and $s_1 \preceq_2 s''$ as suggested by Theorem 5.2.2.

In this section, we stressed that k-step simulation relates the transition probabilities of one transition in the abstract system to k transitions in

the concrete system. However, it does not say anything about the number $j \in \{1, \ldots, k\}$ of arrivals in the Poisson process, which has to be chosen appropriately to guarantee that the probability for reaching certain states within a given step or time bound is related in the concrete and the abstract system. This issue will be approached in the following sections. First, we only consider the *number* of transitions in the Poisson process to obtain step-bounded reachability probabilities, then we will discuss time-bounded reachability using the associated Poisson probabilities.

5.3 Step-bounded reachability

We now show that the previously proposed abstraction method can be used to efficiently derive bounds for the probability to reach a set $B \subseteq S$ in a CTMC $\mathcal{M} = (S, \mathbf{P}, L, \mu_0, \lambda)$. For that, we consider an $\mathrm{E}_k\mathrm{IP}$ $\mathcal{M}'$ with state space S' and $\mathcal{M} \preceq_k \mathcal{M}'$. In the following, we assume that $\mathbf{P}(s, s) = 1$ for all $s \in B$ as the behavior of $\mathcal{M}$ after visiting B can be ignored. As for AMCs, B and B' are *compatible* iff $s \preceq_k s'$ implies that $s \in B$ iff $s' \in B'$, for all $s \in S$, $s' \in S'$.

The k-step forward simulation (cf. Definition 5.2.2) is useful for relating transition probabilities in the concrete and the abstract system. However, to relate *bounded reachability probabilities* of concrete and abstract systems, we have to assess the abstract transitions with the right number j of new arrivals in the Poisson process associated with $\mathcal{M}'$. We will check for which choices of the number of arrivals, we obtain lower and upper bounds of step-bounded (and later also time-bounded) reachability probabilities.

Let us elaborate on this problem in more detail: first consider the case that n is the number of transitions taken in the concrete system to reach a certain goal state. Let ℓ and j be such that $n = \ell \cdot k + j$ and $j \in \{0, \ldots, k-1\}$. Clearly, the number of transitions in the abstract system only corresponds exactly to a multiple of the number of transitions in the concrete system if the remainder j equals 0. As this is generally not the case, we will obtain

lower and upper bounds for the probability of reaching a set of goal states that are not coinciding, even if in the abstraction we have $\mathbf{P}^l = \mathbf{P}^u$.

For more precise explanations let us consider the tree of state sequences shown in Figure 5.5 (top, left). Let the black nodes denote goal states. Taking the right branch, five transitions are needed to reach a goal state. For $k = 3$, this implies that two abstract transitions lead to a goal state. However, as $2 \cdot 3 = 6$, computing with two transitions and three arrivals in the Poisson process each, will not give the exact probability for reaching a goal state, but, as we show, a *lower bound*. Intuitively, the probability for reaching a goal state in Figure 5.5 (top, right) is computed. For an upper bound, one might first consider all states as goal states from the fourth state on in the right branch. This would give a rather coarse upper bound. We follow instead the idea depicted in Figure 5.5 (bottom, left): We consider two transitions for reaching the goal state, however, use three steps in the Poisson process for assessing the first transition, but only one step for assessing the last transition of a sequence of transitions. That is, we compute the reachability probability for the goal states as depicted in Figure 5.5 (bottom, left). Technically, it is beneficial to understand the situation as depicted in Figure 5.5 (bottom, right), i.e., to first consider one transition with only one step in the Poisson process and then to consider a sequence of transitions for which k steps are taken in the Poisson process.

As we will establish in the following Lemma 5.3.1, a tight bound for minimal reachability probability is obtained when the scheduler always chooses $j = k$, and a tight bound for maximal reachability probability is obtained when the scheduler chooses $j = 1$ in the initial state and $j = k$ otherwise. Consequently, we restrict our attention to the following scheduler classes:

$$\begin{aligned}
\mathcal{D}_l^{\mathcal{M}'} &= \{D \in \mathcal{D}^{\mathcal{M}'} \mid \forall \varrho \exists \mu_\varrho : D(\varrho) = (\mu_\varrho, k)\} \\
\mathcal{D}_u^{\mathcal{M}'} &= \{D \in \mathcal{D}^{\mathcal{M}'} \mid \forall \varrho \exists \mu_\varrho : D(\varrho) = (\mu_\varrho, 1) \text{ if } \varrho \in S', \\
&\qquad\qquad D(\varrho) = (\mu_\varrho, k) \text{ otherwise}\}.
\end{aligned}$$

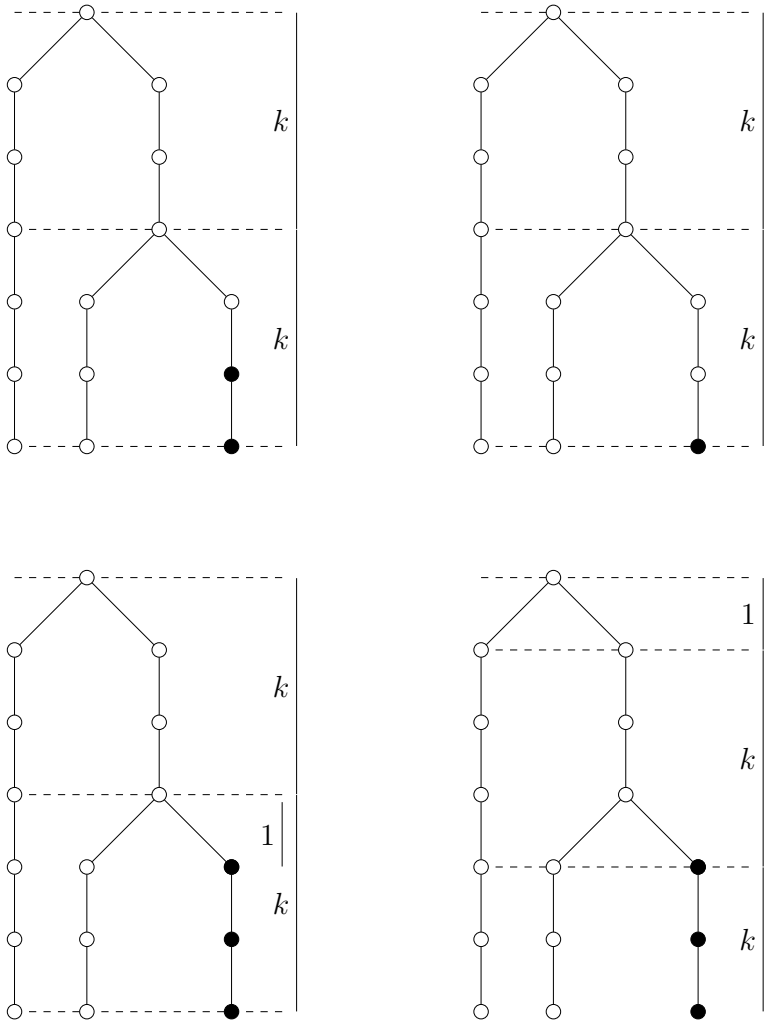


Figure 5.5: Reaching goals in stages of length k.

For these scheduler classes, the number of steps in the Poisson process of the abstract model give lower and upper bounds for the number of steps in a concrete model as follows. For i steps in the concrete model, in the abstraction, $\lfloor \frac{i}{k} \rfloor$ is the corresponding number of steps for a scheduler in $\mathcal{D}_l^{\mathcal{M}'}$ and yields a lower bound for i as $\lfloor \frac{i}{k} \rfloor \cdot k \leq i$; similarly $\lceil \frac{i}{k} \rceil$ is the corresponding number of steps for a scheduler in $\mathcal{D}_u^{\mathcal{M}'}$ and yields an upper bound for i. Let us consider an example to see how this argument can be used for computing step-bounded reachability probabilities.

Example 5.3.1. Let CTMC $\mathcal{M}$ with state space $S = \{s_0, s_1, \ldots\}$ and an E_3IP $\mathcal{M}'$ with $\mathcal{M} \preceq_3 \mathcal{M}'$ be as depicted in Figure 5.6 (top). Note that the transition probability intervals in $\mathcal{M}'$ are singletons, that is, the deviations of reachability probabilities in the concrete model and the probability bounds in the abstract model can only result from abstracting from the actual number of transitions that is necessary in the concrete model to reach a goal state. Assume that $\mathcal{M}$ either starts in state s_0 or s_1 and the initial state of $\mathcal{M}'$ is s'. In Figure 5.6 (bottom) the Poisson arrivals of $\mathcal{M}$ are illustrated as a chain of states in the leftmost column. The evolution of the discrete steps of $\mathcal{M}$ are given by the trees with root s_0 and s_1, respectively (where transition probability 1 is omitted). Similarly, the tree with root s' represents the transitions in $\mathcal{M}'$. The black nodes belong to set B and B', respectively. It is easy to see that a 3-step forward simulation on $\mathcal{M}$ and $\mathcal{M}'$ can be defined such that all the black nodes simulate each other. Furthermore, s' simulates s_0, s_1 and s_2 as well as u' simulates u_0, u_1, u_2 and u_3. The unnamed states are not reachable from either s_0 or s_1 in an exact multiple of 3 steps and therefore we do not depict their simulating states in the E_3IP. For any $D \in \mathcal{D}'$ we calculate step-bounded reachability probabilities in $\mathcal{M}$ and $\mathcal{M}'$ as follows:

$$
\begin{array}{lll}
\Pr_{s_0}(3, B) = \frac{1}{2} & \Pr_{s_1}(3, B) = \frac{1}{2} & \Pr_{s'}^{D}(1, B') = \frac{1}{2} \\
\Pr_{s_0}(4, B) = \frac{3}{4} & \Pr_{s_1}(4, B) = \frac{1}{2} & \\
\Pr_{s_0}(5, B) = \frac{7}{8} & \Pr_{s_1}(5, B) = \frac{1}{2} & \\
\Pr_{s_0}(6, B) = \frac{15}{16} & \Pr_{s_1}(6, B) = \frac{15}{16} & \Pr_{s'}^{D}(2, B') = \frac{15}{16}
\end{array}
$$

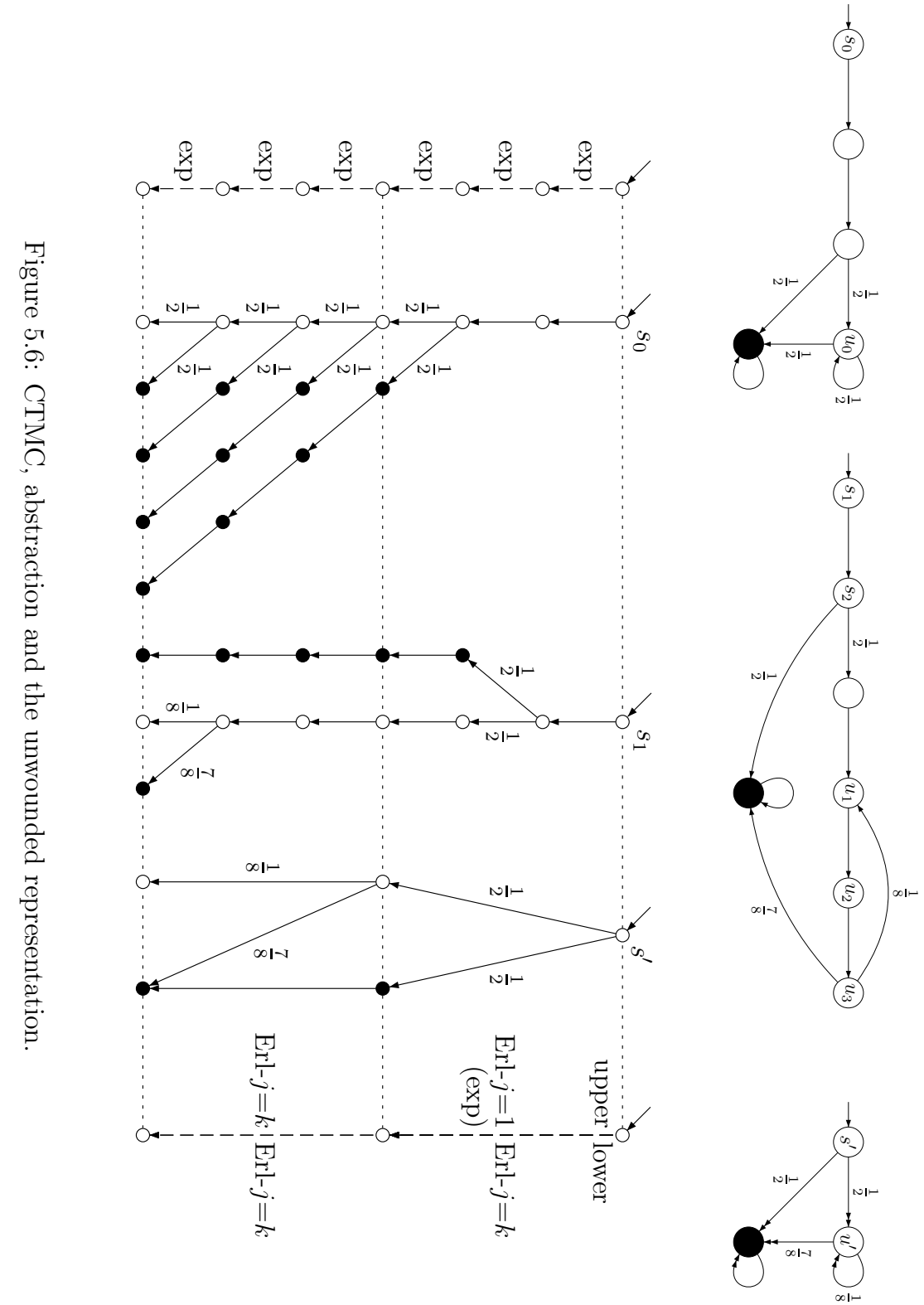


Figure 5.6: CTMC, abstraction and the unwounded representation.

Obviously, for $3 \leq i \leq 6$, $k = 3$, and initial states s_0 and s_1 in $\mathcal{M}$:

$$\Pr^D_{\mathcal{M}'}(Reach^{\leq \lfloor \frac{i}{k} \rfloor}(B')) \leq \Pr_{\mathcal{M}}(Reach^{\leq i}(B)) \leq \Pr^D_{\mathcal{M}'}(Reach^{\leq \lceil \frac{i}{k} \rceil}(B'))$$

We now show that the observation from the example above regarding lower and upper bounds for reachability probabilities is valid in general.

Lemma 5.3.1. *Let $\mathcal{M} \preceq_k \mathcal{M}'$ and let $B \subseteq S$, $B' \subseteq S'$ be compatible sets. Then there exists a scheduler $D \in \mathcal{D}^{\mathcal{M}'}$ such that for all $i \in \mathbb{N}$:*

$$\Pr^D_{\mathcal{M}'}(Reach^{\leq \lfloor \frac{i}{k} \rfloor}(B')) \leq \Pr_{\mathcal{M}}(Reach^{\leq i}(B)) \leq \Pr^D_{\mathcal{M}'}(Reach^{\leq \lceil \frac{i}{k} \rceil}(B'))$$

Proof. Let R_k be a k-step forward simulation on $\mathcal{M}$ and $\mathcal{M}'$. We prove by induction on i that for all $s \in S$, $s' \in S'$ with sR_ks' there exists a scheduler $D_{s,s'} \in \mathcal{D}^{\mathcal{M}'}$ with

$$\Pr^{D_{s,s'}}_{s'}(Reach^{=\lfloor \frac{i}{k} \rfloor}(B')) \leq \Pr_s(Reach^{=i}(B)) \leq \Pr^{D_{s,s'}}_{s'}(Reach^{=\lceil \frac{i}{k} \rceil}(B')). \quad (5.1)$$

As the sets of goal states B, B' cannot be left, the lemma follows directly from Equation (5.1) and the fact that for all $i, h \geq 0$, $D \in \mathcal{D}^{\mathcal{M}'}$ it holds

$$\Pr_s(Reach^{=i}(B)) = \Pr_s(Reach^{\leq i}(B))$$
$$\Pr^D_{s'}(Reach^{=h}(B')) = \Pr^D_{s'}(Reach^{\leq h}(B')).$$

We prove Equation (5.1) by first assuming that $0 \leq i < k$. We define the initial decision of $D_{s,s'} \in \mathcal{D}^{\mathcal{M}'}$ as follows: Let $\Delta : S \times S' \to [0,1]$ be as in Definition 5.2.2 for the pair sR_ks'. We set $D_{s,s'}(s') = \mu$ where

$$\mu(v') = \textstyle\sum_{v \in S : vR_kv'} \Delta(v, v') \quad \text{for all } v' \in S.$$

Since $0 \leq i < k$ we have $\lfloor \frac{i}{k} \rfloor = 0$ and $\lceil \frac{i}{k} \rceil \in \{0, 1\}$. If $s' \in B'$ then sR_ks' and the compatibility of B, B' imply that $s \in B$ and thus

$$\Pr^{D_{s,s'}}_{s'}(0, B') = \Pr^{D_{s,s'}}_{s'}(1, B') = 1 = \Pr_s(i, B).$$

Otherwise, that is, if $s' \notin B'$ then sR_ks' and the compatibility of B, B' imply that $s \notin B$. Let us distinguish the cases $\lceil \frac{i}{k} \rceil = 0$ and $\lceil \frac{i}{k} \rceil = 1$. In the former case, we have $i = 0$ and therefore $\Pr^{D_{s,s'}}_{s'}(0, B') = 0 = \Pr_s(i, B)$.

In the latter case, $0 < i < k$ and, as $\lfloor \frac{i}{k} \rfloor = 0$, this yields for the lower bound $\Pr_{s'}^{D_{s,s'}}(\lfloor \frac{i}{k} \rfloor, B') = 0 \leq \Pr_s(i, B)$.

For the upper bound we get for $\lceil \frac{i}{k} \rceil$

$$\begin{aligned}
\Pr_s(i,B) &= \textstyle\sum_{v \in B} \mathbf{P}^i(s,v) \\
&\leq \textstyle\sum_{v \in B} \mathbf{P}^k(s,v) && (*) \\
&= \textstyle\sum_{v \in B} \sum_{v' \in S'} \Delta(v,v') && (sR_ks') \\
&= \textstyle\sum_{v \in S} \sum_{v' \in B'} \Delta(v,v') && (**) \\
&= \textstyle\sum_{v' \in B'} \mu(v') && (sR_ks') \\
&= \Pr_{s'}^{D_{s,s'}}(1, B') \\
&= \Pr_{s'}^{D_{s,s'}}(\lceil \tfrac{i}{k} \rceil, B')
\end{aligned}$$

where $\mu = D(s')$. Here, $(*)$ holds because all B-states have self-loops with probability one and $(**)$ is true since

$$\begin{aligned}
\Delta(v,v') > 0 \implies (v,v') \in R_k &\implies v \in B \text{ iff } v' \in B' \\
&\implies \Delta(B, S' \setminus B') = 0 = \Delta(S \setminus B, B').
\end{aligned}$$

Let us now proceed with the induction step, that is, we assume that Equation (5.1) holds and prove that it is true for $i \to i+k$. Note that by induction hypothesis, for each pair vR_kv' there exists a scheduler $D_{v,v'} \in \mathcal{D}^{\mathcal{M}'}$ such that

$$\Pr_{v'}^{D_{v,v'}}(\lfloor \tfrac{i}{k} \rfloor, B') \leq \Pr_v(i,B) \leq \Pr_{v'}^{D_{v,v'}}(\lceil \tfrac{i}{k} \rceil, B').$$

Let $\varrho \in Path_{abs}^{\mathcal{M}'}$ and let v' be the first element of ϱ. We define $D_{s,s'}(s'\varrho)$ as a linear combination of the distributions $D_{v,v'}(\varrho)$. More precisely,

$$D_{s,s'}(s'\varrho) = \textstyle\sum_{v \in S} \left(\frac{\Delta(v,v')}{\Delta(S,v')} \cdot D_{v,v'}(\varrho) \right).$$

This implies that for $s' \notin B'$, $h > 0$

$$\Pr_{s'}^{D_{s,s'}}(h, B') = \sum_{v' \in S'} \left(\mu(v') \cdot \sum_{v \in S} \left(\tfrac{\Delta(v,v')}{\Delta(S,v')} \cdot \Pr_{v'}^{D_{v,v'}}(h-1, B') \right) \right). \qquad (5.2)$$

We calculate

$$
\begin{aligned}
&\Pr_s(i+k, B)\\
&= \textstyle\sum_{v\in S} \mathbf{P}^k(s,v) \cdot \Pr_v(i,B)\\
&= \textstyle\sum_{v\in S} \sum_{v'\in S'} \Delta(v,v') \cdot \Pr_v(i,B) && (sR_ks')\\
&= \textstyle\sum_{v'\in S'} \left(\Delta(S,v') \cdot \sum_{v\in S} \left(\frac{\Delta(v,v')}{\Delta(S,v')} \cdot \Pr_v(i,B)\right)\right)\\
&\geq \textstyle\sum_{v'\in S'} \left(\Delta(S,v') \cdot \sum_{v\in S} \left(\frac{\Delta(v,v')}{\Delta(S,v')} \cdot \Pr_{v'}^{D_{v,v'}}(\lfloor \tfrac{i}{k} \rfloor, B')\right)\right) && \text{(Ind. hyp.)}\\
&= \textstyle\sum_{v'\in S'} \left(\mu(v') \cdot \sum_{v\in S} \left(\frac{\Delta(v,v')}{\Delta(S,v')} \cdot \Pr_{v'}^{D_{v,v'}}(\lfloor \tfrac{i}{k} \rfloor, B')\right)\right) && (sR_ks')\\
&= \Pr_{s'}^{D_{s,s'}}(\lfloor \tfrac{i+k}{k} \rfloor, B') && \text{(Eq. (5.2))}\\
&= \Pr_{s'}^{D_{s,s'}}(\lfloor \tfrac{i}{k} \rfloor + 1, B')
\end{aligned}
$$

for the lower bound. For the upper bound, we obtain:

$$
\begin{aligned}
&\Pr_s(i+k, B)\\
&= \textstyle\sum_{v\in S} \mathbf{P}^k(s,v) \cdot \Pr_v(i,B)\\
&= \textstyle\sum_{v\in S} \sum_{v'\in S'} \Delta(v,v') \cdot \Pr_v(i,B) && (sR_ks')\\
&= \textstyle\sum_{v'\in S'} \left(\Delta(S,v') \cdot \sum_{v\in S} \left(\frac{\Delta(v,v')}{\Delta(S,v')} \cdot \Pr_v(i,B)\right)\right)\\
&\leq \textstyle\sum_{v'\in S'} \left(\Delta(S,v') \cdot \sum_{v\in S} \left(\frac{\Delta(v,v')}{\Delta(S,v')} \cdot \Pr_{v'}^{D_{v,v'}}(\lceil \tfrac{i}{k} \rceil, B')\right)\right) && \text{(Ind. hyp.)}\\
&= \textstyle\sum_{v'\in S'} \left(\mu(v') \cdot \sum_{v\in S} \left(\frac{\Delta(v,v')}{\Delta(S,v')} \cdot \Pr_{v'}^{D_{v,v'}}(\lceil \tfrac{i}{k} \rceil, B')\right)\right) && (sR_ks')\\
&= \Pr_{s'}^{D_{s,s'}}(\lceil \tfrac{i+k}{k} \rceil, B') && \text{(Eq. (5.2))}\\
&= \Pr_{s'}^{D_{s,s'}}(\lceil \tfrac{i}{k} \rceil + 1, B')
\end{aligned}
$$

Thus $\Pr_{s'}^{D_{s,s'}}(\lfloor \frac{i}{k} \rfloor + 1, B') \leq \Pr_s(i+k, B) \leq \Pr_{s'}^{D_{s,s'}}(\lceil \frac{i}{k} \rceil + 1, B')$, and we conclude that for all $i \in \mathbb{N}$ there exists a scheduler $D \in \mathcal{D}^{\mathcal{M}'}$ such that

$$\Pr_{\mathcal{M}'}^{D}(\mathit{Reach}^{=\lfloor \frac{i}{k} \rfloor}(B')) \leq \Pr_{\mathcal{M}}(\mathit{Reach}^{=i}(B)) \leq \Pr_{\mathcal{M}'}^{D}(\mathit{Reach}^{=\lceil \frac{i}{k} \rceil}(B'))$$

which completes the proof. □

From this lemma, we directly obtain that i-step bounded reachability probabilities in a CTMC are bounded by the minimal and maximal probabilities in a simulating $\mathrm{E}_k\mathrm{IP}$ to take $\lfloor \frac{i}{k} \rfloor$ to $\lceil \frac{i}{k} \rceil$ steps. Let

$$\Pr^{\mathcal{M}'}_{inf}(i, B') = \inf_{D \in \mathcal{D}^{\mathcal{M}'}_l} \Pr^D(Reach^{\leq i}(B'))$$
$$\Pr^{\mathcal{M}'}_{sup}(i, B') = \sup_{D \in \mathcal{D}^{\mathcal{M}'}_u} \Pr^D(Reach^{\leq i}(B')).$$

Corollary 5.3.2. *Let $\mathcal{M} \preceq_k \mathcal{M}'$ and let $B \subseteq S$, $B' \subseteq S'$ be compatible sets, then for all $i \in \mathbb{N}$:*

$$\Pr^{\mathcal{M}'}_{inf}(\lfloor \tfrac{i}{k} \rfloor, B') \leq \Pr^{\mathcal{M}}(i, B) \leq \Pr^{\mathcal{M}'}_{sup}(\lceil \tfrac{i}{k} \rceil, B')$$

Proof. Follows directly from Lemma 5.3.1. □

Note that the computation of step-bounded reachability probabilities in $\mathrm{E}_k\mathrm{IP}$s is done on the embedded ADTMC using the same algorithm as in Section 4.2.

5.4 Time-bounded reachability

In the previous section we focused on time-abstract probabilities, now we lift the results from that section to time-bounded reachability and discuss how to compute lower and upper bounds of time-bounded reachability probabilities.

Theorem 5.4.1. *Let $\mathcal{M}$ be a CTMC and $\mathcal{M}'$ an E_kIP with $\mathcal{M} \preceq_k \mathcal{M}'$. For $t \in \mathbb{R}_{\geq 0}$ and compatible sets B and B', there exist schedulers $D \in \mathcal{D}^{\mathcal{M}'}_l$, $D' \in \mathcal{D}^{\mathcal{M}'}_u$ with*

$$\Pr^D(Reach_{\leq t}(B')) \leq \Pr^{\mathcal{M}}(Reach_{\leq t}(B)) \leq \Pr^{D'}(Reach_{\leq t}(B')).$$

Proof. Let R_k be a k-step forward simulation on $\mathcal{M}$ and $\mathcal{M}'$ and assume that s and s' are the initial states of $\mathcal{M}$ and $\mathcal{M}'$ – the lifting to initial distributions is as usual. From Lemma 5.3.1 we know that for pair sR_ks' there is a scheduler $\hat{D} \in \mathcal{D}^{\mathcal{M}'}$ such that for all $i \in \mathbb{N}$:

$$\Pr^{\hat{D}}_{s'}(Reach^{\leq \lfloor \frac{i}{k} \rfloor}(B')) \leq \Pr_s(Reach^{\leq i}(B)) \leq \Pr^{\hat{D}}_{s'}(Reach^{\leq \lceil \frac{i}{k} \rceil}(B')) \ .$$

We lift this statement to time-bounded reachability by proving that there exist schedulers $D \in \mathcal{D}_l^{\mathcal{M}'}$, $D' \in \mathcal{D}_u^{\mathcal{M}'}$ such that for all $i \in \mathbb{N}$:

$$\Pr_{s'}^{D}(Reach_{\leq t}^{\leq \lfloor \frac{i}{k} \rfloor}(B')) \leq \Pr_s(Reach_{\leq t}^{\leq i}(B)) \leq \Pr_{s'}^{D'}(Reach_{\leq t}^{\leq \lceil \frac{i}{k} \rceil}(B')). \qquad (5.3)$$

where for any $i \in \mathbb{N}$, $t \in \mathbb{R}_{\geq 0}$ and set of goal states B, by $Reach_{\leq t}^{\leq i}(B)$ we denote the set of paths that reach a state in B within both, i steps and t time units. From this the theorem follows directly as for any D:

$$\Pr_s(Reach_{\leq t}(B)) = \lim_{i \to \infty} \Pr_s(Reach_{\leq t}^{\leq i}(B)), \text{ and}$$
$$\Pr_{s'}^{D}(Reach_{\leq t}(B')) = \lim_{h \to \infty} \Pr_{s'}^{D}(Reach_{\leq t}^{\leq h}(B))$$

The idea is to define two copies D and D' of $\hat{D}$ that choose the same distributions μ as $\hat{D}$ but arrivals j such that $D \in \mathcal{D}_l^{\mathcal{M}'}$ and $D' \in \mathcal{D}_u^{\mathcal{M}'}$. More precisely, if $\hat{D}(\varrho) = \mu$ (recall that we omit the choice of the number of arrivals for $\hat{D}$ as it plays no role for Lemma 5.3.1) we define $D(\varrho) = (\mu, k)$ for all $\varrho \in Path_{abs}^{\mathcal{M}'}$ and $D'(\varrho) = (\mu, 1)$ if $\varrho = s'$ and $D'(\varrho) = (\mu, k)$ otherwise. This yields:

$$\begin{aligned}
&\Pr_s(Reach_{\leq t}^{\leq i}(B)) \\
&= \textstyle\sum_{h=0}^{i} \left(\psi_{\lambda,t}(h) \cdot \Pr_s(Reach^{=h}(B))\right) \\
&= \textstyle\sum_{h=0}^{i} \left(\psi_{\lambda,t}(h) \cdot \Pr_s(Reach^{\leq h}(B))\right) && (*) \\
&\geq \textstyle\sum_{h=0}^{i} \left(\psi_{\lambda,t}(h) \cdot \Pr_{s'}^{D}(Reach^{\leq \lfloor \frac{h}{k} \rfloor}(B'))\right) && \text{(Lemma 5.3.1)} \\
&\geq \textstyle\sum_{h=0}^{\lfloor \frac{i}{k} \rfloor \cdot k} \left(\psi_{\lambda,t}(h) \cdot \Pr_{s'}^{D}(Reach^{\leq \lfloor \frac{h}{k} \rfloor}(B'))\right) && \text{(sum truncated)} \\
&= \textstyle\sum_{\ell=0}^{\lfloor \frac{i}{k} \rfloor} \left(\left(\sum_{h=\ell k}^{\ell k+k-1} \psi_{\lambda,t}(h)\right) \cdot \Pr_{s'}^{D}(Reach^{\leq \ell}(B'))\right) \\
&= \textstyle\sum_{h=0}^{\lfloor \frac{i}{k} \rfloor} \left(\psi_{\lambda,t}(hk,k) \cdot \Pr_{s'}^{D}(Reach^{\leq h}(B'))\right) \\
&= \textstyle\sum_{h=0}^{\lfloor \frac{i}{k} \rfloor} \left(\psi_{\lambda,t}(hk,k) \cdot \Pr_{s'}^{D}(Reach^{=h}(B'))\right) && (*) \\
&= \Pr_{s'}^{D}(Reach_{\leq t}^{\leq \lfloor \frac{i}{k} \rfloor}(B')) && (**)
\end{aligned}$$

Note that $(*)$ is true because B and B' cannot be left and for $(**)$ we exploit that D chooses always $j = k$. For the upper bounds we get:

$$
\begin{aligned}
&\Pr_s(Reach^{\leq i}_{\leq t}(B)) \\
&= \textstyle\sum_{h=0}^{i} \psi_{\lambda,t}(h) \cdot \Pr_s(Reach^{\leq h}(B)) \\
&\leq \textstyle\sum_{h=0}^{i} \psi_{\lambda,t}(h) \cdot \Pr^{D'}_{s'}(Reach^{\leq \lceil \frac{h}{k} \rceil}(B')) && \text{(Lemma 5.3.1)} \\
&\leq \qquad \psi_{\lambda,t}(0) \cdot \Pr^{D'}_{s'}(Reach^{\leq 0}(B')) \\
&\qquad + \textstyle\sum_{h=1}^{\lceil \frac{i}{k} \rceil \cdot k} \psi_{\lambda,t}(h) \cdot \Pr^{D'}_{s'}(Reach^{\leq \lceil \frac{h}{k} \rceil}(B')) && \text{(sum extended)} \\
&= \qquad \psi_{\lambda,t}(0) \cdot \Pr^{D'}_{s'}(Reach^{\leq 0}(B')) \\
&\quad + \textstyle\sum_{\ell=1}^{\lceil \frac{i}{k} \rceil} \left(\sum_{h=(\ell-1)k+1}^{\ell k} \psi_{\lambda,t}(h) \right) \cdot \Pr^{D'}_{s'}(Reach^{\leq \ell}(B')) \\
&= \qquad \psi_{\lambda,t}(0) \cdot \Pr^{D'}_{s'}(Reach^{\leq 0}(B')) \\
&\qquad + \textstyle\sum_{\ell=1}^{\lceil \frac{i}{k} \rceil} \psi_{\lambda,t}((\ell-1)k+1, k) \cdot \Pr^{D'}_{s'}(Reach^{\leq \ell}(B')) \\
&= \Pr^{D'}_{s'}(Reach^{\leq \lceil \frac{i}{k} \rceil}_{\leq t}(B'))
\end{aligned}
$$

Here, for the last step we use the fact that D' chooses $j = 1$ for the first step and $j = k$ for the remaining steps.
This implies that Equation (5.3) is true which completes the proof of the theorem. □

The following corollary is a direct result of the theorem above. It states that when comparing reachability probabilities of a CTMC with those of a simulating $\mathrm{E}_k\mathrm{IP}$ $\mathcal{M}'$, in the *worst (best) case* $\mathcal{M}'$ will have a smaller (larger) time-bounded reachability probability, when restricting to the scheduler class $\mathcal{D}_l^{\mathcal{M}'}$ ($\mathcal{D}_u^{\mathcal{M}'}$). Let

$$
\begin{aligned}
\Pr^{\mathcal{M}'}_{inf}(t, B') &= \inf_{D \in \mathcal{D}_l^{\mathcal{M}'}} \Pr^D(Reach_{\leq t}(B')) \\
\Pr^{\mathcal{M}'}_{sup}(t, B') &= \sup_{D \in \mathcal{D}_u^{\mathcal{M}'}} \Pr^D(Reach_{\leq t}(B')).
\end{aligned}
$$

Corollary 5.4.2. *Let $\mathcal{M}$ be a CTMC and $\mathcal{M}'$ an E_kIP with $\mathcal{M} \preceq_k \mathcal{M}'$. Let $t \in \mathbb{R}_{\geq 0}$ and B be compatible with B', then:*

$$
\Pr^{\mathcal{M}'}_{inf}(t, B') \leq \Pr^{\mathcal{M}}(t, B) \leq \Pr^{\mathcal{M}'}_{sup}(t, B')
$$

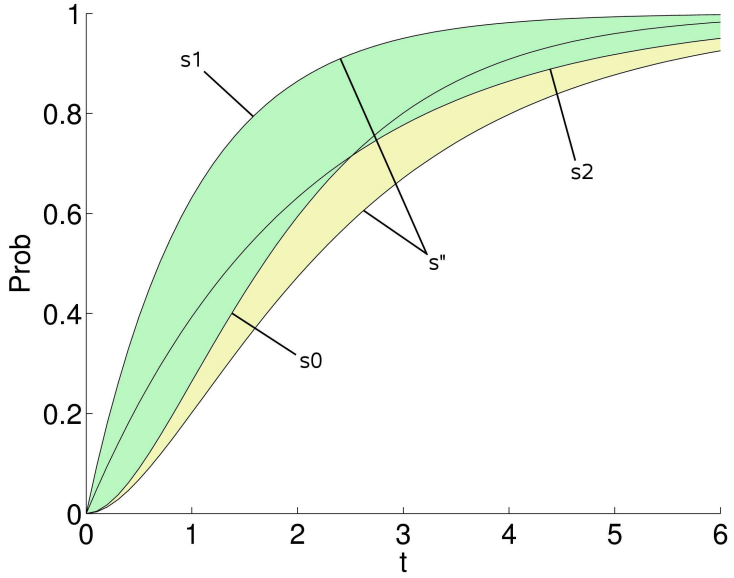


Figure 5.7: Concrete vs. abstract behavior over time.

Example 5.4.1. Let us consider the CTMC $\mathcal{M}$ and its abstractions from Figure 5.1 once more. The plot in Figure 5.7 shows the probability (bounds) to reach u'' within t time units if the Erlang-2 interval process starts in s'' with probability 1 and in s_0, s_1 and s_2 in the CTMC, respectively. For the Erlang-2 interval process, the infimum over all choices for values from the intervals is taken and the resulting curve is obviously below the infimum in the CTMC. The supremum coincides with the probabilities for s_1; it is obtained when choosing the residence time to be Erlang-1 (exponentially) distributed and always selecting $\{u'' \mapsto 1\} \in \mathbf{T}(s'')$.

Computation of time-bounded reachability. As for uniform CTMCs, we can efficiently calculate time-bounded reachability probabilities in $\mathcal{M}'$, using time-abstract reachability probabilities and the probability for the number of Poisson arrivals in a certain range. More specifically, after i transitions in $\mathcal{M}'$, the number of arrivals in the associated Poisson process is among $i{\cdot}k, i{\cdot}$

$k{+}1, \ldots, i{\cdot}k{+}(k{-}1)$, if $D \in \mathcal{D}_l^{\mathcal{M}'}$, and among $(i{-}1){\cdot}k{+}1, (i{-}1){\cdot}k{+}2, \ldots, i{\cdot}k$, if $D \in \mathcal{D}_u^{\mathcal{M}'}$. For $B \subseteq S_{\mathcal{M}'}$, $i \in \mathbb{N}$ let $Reach^{=i}(B) = \{\sigma \in Path_{\mathcal{M}'} \mid \sigma[i] \in B\}$. Using $\psi_{\lambda,t}$ for the respective Poisson probabilities, we thus obtain for $\mathrm{E}_k\mathrm{IP}$ $\mathcal{M}'$, $t \in \mathbb{R}_{\geq 0}$ and $B \subseteq S_{\mathcal{M}'}$

$$\Pr^D(Reach_{\leq t}(B)) = \textstyle\sum_{i=0}^{\infty} \psi_{\lambda,t}(\sum_{h=0}^{i-1} j_h, j_i) \cdot \Pr^D_{emb}(Reach^{=i}(B))$$

where $j_i = k$ for all $i \in \mathbb{N}$ if $D \in \mathcal{D}_l^{\mathcal{M}'}$ and $j_0 = 1$, $j_i = k$ for $i \in \mathbb{N}_{>0}$ if $D \in \mathcal{D}_u^{\mathcal{M}'}$.

The difference to the computation of time-bounded reachability for abstract CTMCs in the previous chapter is in the factor from the Poisson process only. Therefore adapting the algorithm from Section 4.2 is straightforward. For $\mathrm{E}_k\mathrm{IP}$ $\mathcal{M} = (S, \mathbf{P}^l, \mathbf{P}^u, L, \mu_0, \lambda, k)$, time bound $t \in \mathbb{R}_{\geq 0}$, goal states B and error bound ε, we obtain $\Pr^{\varepsilon}_{inf}(t, B) = \sum_{s_0 \in S} \mu_0(s_0) \cdot q_0^l(s_0)$ where for all $s \in S$, $i \in \{1, \ldots, n_\varepsilon\}$:

$$\begin{aligned} q_0^l(s) &= \psi_{\lambda,t}(0,k) \cdot \mathbf{1}(s \in B) + q_1^l(s) \\ q_i^l(s) &= \textstyle\sum_{u \in S} \min^{\mathcal{M}}(s, \bar{\mu})(u) \cdot \bar{\mu}(u) \\ q_{n_\varepsilon+1}^l(s) &= 0 \end{aligned}$$

$$\text{where} \qquad \bar{\mu}(u) = \psi_{\lambda,t}(i \cdot k, k) \cdot \mathbf{1}(u \in B) + q_{i+1}^l(u)$$

Similarly, we obtain $\Pr^{\varepsilon}_{sup}(t, B) = \sum_{s_0 \in S} \mu_0(s_0) \cdot q_0^u(s_0)$ where for all $s \in S$, $i \in \{1, \ldots, n_\varepsilon\}$:

$$\begin{aligned} q_0^u(s) &= \psi_{\lambda,t}(0,1) \cdot \mathbf{1}(s \in B) + q_1^u(s) + \varepsilon \\ q_i^u(s) &= \textstyle\sum_{u \in S} \min^{\mathcal{M}}(s, \bar{\mu})(u) \cdot \bar{\mu}(u) \\ q_{n_\varepsilon+1}^u(s) &= 0 \end{aligned}$$

$$\text{where} \qquad \bar{\mu}(u) = \psi_{\lambda,t}(1 + (i-1) \cdot k, k) \cdot \mathbf{1}(u \in B) + q_{i+1}^u(u)$$

Note that, as in Section 4.3, we add the error bound ε to the supremum as otherwise the computed result is up to ε below the actual value.

Lemma 5.4.3. *For an E_kIP $\mathcal{M}'$, set of goal states B, $t \in \mathbb{R}_{\geq 0}$ and error margin $\varepsilon \in \mathbb{R}_{>0}$:*

$$\Pr_{inf}^{\mathcal{M}'}(t, B) \geq \Pr_{inf}^{\varepsilon}(t, B) \geq \Pr_{inf}^{\mathcal{M}'}(t, B) - \varepsilon$$

$$\Pr_{sup}^{\mathcal{M}'}(t, B) \leq \Pr_{sup}^{\varepsilon}(t, B) \leq \Pr_{sup}^{\mathcal{M}'}(t, B) + \varepsilon$$

Proof. The proof follows directly from [BHKH05, Theorem 5]. □

The results presented so far can be lifted to model checking CSL formulas on E_kIPs in the same manner as it has been done for ACTMCs before and thus will not be discussed in detail. We also skip the discussion of unbounded reachability and proceed with a case study from systems biology.

5.5 Case study: Enzyme catalyzed substrate conversion

Markovian models are well established for the analysis of biochemical reaction networks [BB01, vK07]. Typically, such networks are described by a set of reaction types and the involved molecular species, e.g., the different types of molecules. The occurrence of a reaction changes the species' populations as molecules are produced and/or consumed.

We focus on an enzymatic reaction network with four molecular species: enzyme (E), substrate (S), complex (C) and product (P) molecules. The three reaction types R_1, R_2, R_3 are given by the following rules:

$$R_1 : E + S \xrightarrow{c_1} C, \qquad R_2 : C \xrightarrow{c_2} E + S, \qquad R_3 : C \xrightarrow{c_3} E + P.$$

The species on the left hand of the arrow (also called *reactants*) describe how many molecules of a certain type are consumed by the reaction and those on the right hand describe how many are produced. For instance, one molecule of type E and S is consumed by reaction R_1 and one C molecule is produced. The constants $c_1, c_2, c_3 \in \mathbb{R}_{>0}$ determine the probability of the reactions as explained below.

Concrete model. The temporal evolution of the system is represented by a CTMC as follows (cf. [BSW06]): A state corresponds to a population vector $\vec{x} = (x_E, x_S, x_C, x_P) \in \mathbb{N}^4$ and transitions are triggered by chemical reactions. The change of the current population vector $\vec{x}$ caused by a reaction of type R_m, $m \in \{1, 2, 3\}$ is expressed as a vector $\vec{v}_m$ where

$$\vec{v}_1 = (-1, -1, 1, 0), \qquad \vec{v}_2 = (1, 1, -1, 0), \qquad \vec{v}_3 = (1, 0, -1, 1).$$

Obviously, reaction R_m is only possible if vector $\vec{x} + \vec{v}_m$ contains no negative entries. Given an initial state $\vec{s} = (s_E, s_S, 0, 0)$, it is easy to verify that the set of reachable states is given by

$$S = \{(x_E, x_S, x_C, x_P) \mid x_E + x_C = s_E,\ x_S + x_C + x_P = s_S\}.$$

The probability that a reaction of type R_m occurs within a certain time interval is determined by the function $r_m : S \to \mathbb{R}_{>0}$. The value $r_m(\vec{x})$ is proportional to the number of distinct combinations of R_m's reactants:

$$\begin{aligned} r_1((x_E, x_S, x_C, x_P)) &= c_1 \cdot x_E \cdot x_S \\ r_2((x_E, x_S, x_C, x_P)) &= c_2 \cdot x_C \\ r_3((x_E, x_S, x_C, x_P)) &= c_3 \cdot x_C \end{aligned}$$

We define the transition matrix $\mathbf{P}$ of the CTMC by $\mathbf{P}(\vec{x}, \vec{x} + \vec{v}_m) = \frac{r_m(x)}{\lambda}$ with exit rate

$$\lambda \geq \max_{\vec{x} \in S}(r_1(\vec{x}) + r_2(\vec{x}) + r_3(\vec{x})).$$

Thus, state $\vec{x}$ has outgoing transitions $\vec{x} \xrightarrow{r_m(x)/\lambda} \vec{x} + \vec{v}_m$ for all m with $\vec{x} + \vec{v}_m > \vec{0}$ and the self-loop probability in $\vec{x}$ is:

$$\mathbf{P}(\vec{x}, \vec{x}) = 1 - \frac{r_1(\vec{x}) + r_2(\vec{x}) + r_3(\vec{x})}{\lambda}$$

Finally, let $L((x_E, x_S, x_C, x_P), i) = (x_P, i)$, that is, we label every state by the number of product molecules.

For initial state $(2, 4, 0, 0)$ and parameter set $c_1 = c_2 = 1$ and $c_3 = 0.001$, the transition structure of the underlying CTMC is depicted in Figure 5.8 (top). Transitions are labeled with values $r_m(\vec{x})$ rather than probabilities. Intuitively, the larger $r_m(\vec{x})$, the *faster* the transition; for these parameters, reactions of type R_1 and R_2 are considerably *faster* than R_3 reactions. In the following, states $\vec{x}$ with $L(\vec{x}, 4) = \top$ are considered to be goal states.

Abstract model. Let the CTMC $\mathcal{M} = (S, \mathbf{P}, L, \mu_0, \lambda)$ be given as described above. Let abstraction function $\alpha_k : S \to S'$ be given by

$$\alpha_k((x_E, x_S, x_C, x_P)) = [x_P] \quad \text{for all} \quad (x_E, x_S, x_C, x_P) \in S$$

i.e., we group all states in which the number of molecules of type P is the same. The induced abstractions $\alpha_1(\mathcal{M})$ and $\alpha_2(\mathcal{M})$ are depicted in Figure 5.8 (middle, bottom). While $\alpha_1(\mathcal{M})$ is an ACTMC where, by always deciding for self-loops with probability 1, it is possible that the goal state [4] will never be reached, $\alpha_2(\mathcal{M})$ is an E_2IP with a positive probability for eventually reaching [4], even in the worst case. The crucial difference when abstracting k consecutive transitions instead of single ones has already been observed in the minimal Example 5.2.1. Here, the states $\vec{x}$ with *no* complex molecules take the role of s_0 in the minimal example, i.e., there is no transition leading out of $\gamma(\alpha(\vec{x}))$, resulting in $\mathbf{P}^u(\vec{x}, \vec{x}) = 1$ and $\mathbf{P}^l(\vec{x}, \vec{y}) = 0$ for all $\vec{y} \neq \vec{x}$.

While it is clear that in this case study, for lower bounds of reachability probabilities, we get better results by applying Erlang-k interval abstraction, it is by no means obvious, how much better those results can get. In the following, we will investigate this question, especially with respect to the choice of k.

We are interested in the probability that within time t the number of type P molecules reaches threshold $n = s_S$, the maximum number of P molecules. We apply labels $AP = \{0, 1, \ldots, n\}$ and for $0 \leq a \leq n$ let $L(x, a) = \top$ if $x = (x_E, x_S, x_C, x_P)$ with $x_P = a$ and $L(x, a) = \bot$ otherwise. For the initial populations, we fix $s_E = 20$ and vary s_S between 50 and 2000.

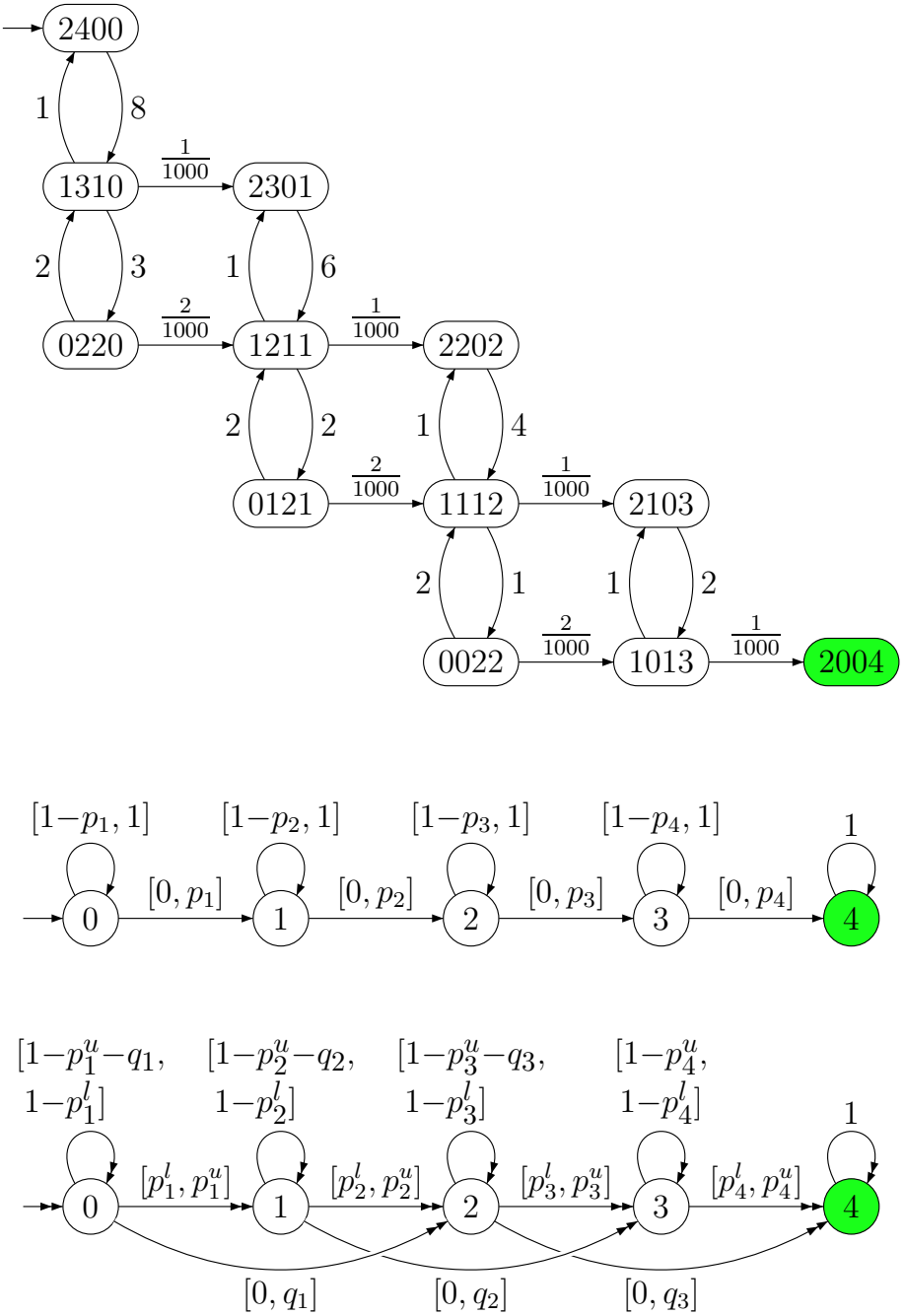


Figure 5.8: Concrete model where transitions are labeled with r_m; initially the system has 2 enzyme and 4 substrate molecules (top); an interval abstraction with $k = 1$ (middle) and with $k = 2$ (bottom).

Stiffness. In many biological systems, components act on time scales that differ by several orders of magnitude which leads to *stiff* models. Traditional numerical analysis methods perform poorly in the presence of stiffness because a large number of very small time steps has to be considered. For the enzymatic reaction, stiffness arises if $c_2 \gg c_3$ and results in a high self-loop probability in most states because λ is large compared to $r_1(x)+r_2(x)+r_3(x)$. Thus, even in case of a small number $|S|$ of reachable states, computing reachability probabilities is extremely time consuming. We show how our abstraction method can be used to efficiently verify properties of interest even for stiff parameter sets. We choose a realistic parameter set of $c_1 = c_2 = 1$ and $c_3 = 0.001$. Note that the order of magnitude of the expected time until threshold $n = s_S = 300$ is reached is 10^4 for these parameters.

Some important remarks are necessary at this point. Abstraction techniques rely on the construction of small abstract models by disregarding details of the concrete model as the latter is too large to be solved efficiently. In this example, we have the additional problem of stiffness and the abstraction method proposed here can tackle this by choosing high values for k. Then one step in the Erlang-k interval process happens after a large number of arrivals in the underlying Poisson process and the self-loop probability in the abstract model is much smaller than in the concrete one. We chose $k \in \{2^{10}, 2^{11}, 2^{12}\}$ for the construction of the Erlang-k interval process $\alpha_k(\mathcal{M})$ and calculate the transition probability intervals by taking the k-th matrix power of $\mathbf{P}$. The choice for k is reasonable, since for a given error bound $\varepsilon = 10^{-10}$, $s_S = 300$ and $t = 10000$, a transient analysis of the concrete model via uniformization would require around $6 \cdot 10^7$ steps. By contrast, our method considers k steps in the concrete model and around $(6 \cdot 10^7)/k$ steps in the smaller abstract model. Thus, although the construction of the Erlang-k interval process is expensive, the total time savings are enormous. We used the MATLAB software for our prototypical implementation and the calculation of $\mathbf{P}^k$ could be performed efficiently because $\mathbf{P}^{2^j}$ can be computed

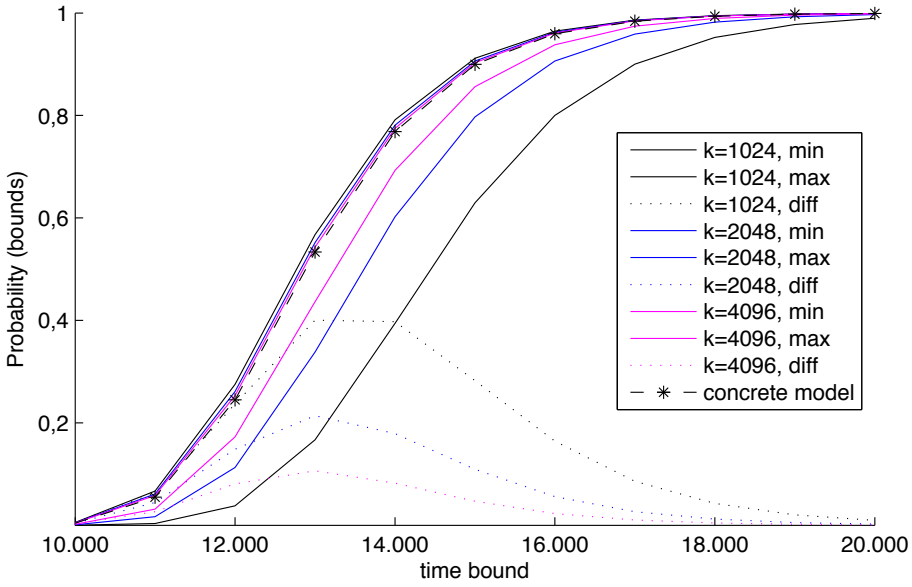
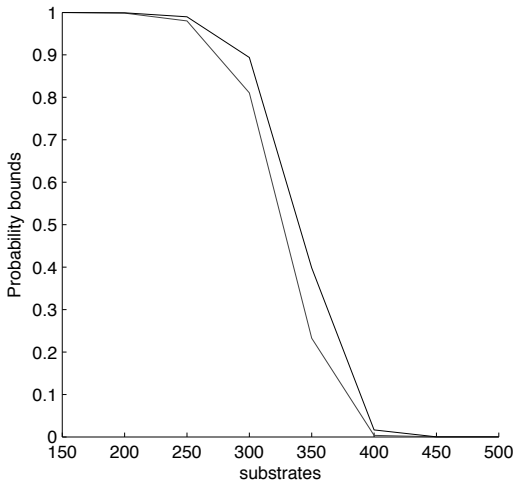


Figure 5.9: Time-bounded reachability

using j matrix multiplications. For non-stiff models, a smaller value for k may be appropriate; in that case, it is obvious that upper and lower bounds for the k-step transition probabilities can be obtained in a local fashion, i.e. by computing the k-th matrix power of submatrices of $\mathbf{P}$. Therefore, we expect our method to perform well even if $|S|$ is large. However, for stiff *and* large concrete models more sophisticated techniques for the construction of the abstract model must be applied that exploit the fact that only upper and lower bounds are needed.

Experimental results. For $s_S = 200$ we compared the results of our abstraction method for the probability to reach A_n within time bound t with results for the concrete model that were obtained using PRISM. While it took more than one day to generate the plot for the concrete model in Figure 5.9, right, our MATLAB implementation took less than one hour for all three pairs of upper and lower probability bounds and different values of t.



$\|S'\|$	$\|S\|$	time
50	861	$0m\ 5s$
300	6111	$37m\ 36s$
500	10311	$70m\ 39s$
1000	20811	$144m\ 49s$
1500	31311	$214m\ 2s$
2000	41811	$322m\ 50s$

Figure 5.10: Time-bounded reachability and computation times

Both jobs were run on the same desktop computer (Athlon 64 X2 3800+, 2GB RAM). Figure 5.9, left, shows the lower and upper probability bounds using $k = 2^{12}$, $t = 20000$ and varying s_S. For high values of s_S, e.g., $s_S = 500$ the construction of the Erlang-k interval process took more than 99% of the total computation time as the sparsity of the matrix is lost during matrix multiplication.

We conclude this section with the additional experimental details on computation times (run on a workstation; Xeon 5140 – 2.33 GHz, 32GB RAM), given in Figure 5.10, using $k = 2^{12}$, $t = 50000$ (and $s_S = 200$).

Note that for this case study exact abstraction techniques such as lumping do not yield any state-space reduction. Also note that unbounded reachability is not of interest in this setting as it follows quite trivially from the model that on the long run, the probability to end up in the goal state is 1 regardless of the initial distribution.

5.6 Summary and discussion

In this chapter, an abstraction technique for CTMCs was presented that allows for collapsing sequences of transitions. In contrast to the techniques presented before, this allows for collapsing states that do not have the same successor states – more precisely, the same successor partitions – and to still obtain reasonable abstractions. We showed how to compute lower and upper bounds for step- and time-bounded reachability probabilities for the resulting abstract models; further, we identified classes of schedulers that are well suitable for obtaining lower and upper bounds that provide safe bounds for the reachability probabilities in the original model. As has been shown in the previous chapter, it is straightforward to lift reachability analysis to CSL model checking, thus we did not elaborate on model checking E_kIPs. However, we showed the viability of the approach in terms of the *enzyme catalyzed substrate conversion* case study.

Further research efforts in this direction should point at the following questions:

- How to obtain an adequate value for k automatically? Can we use graph analysis, at least for small values of k?
- How to compute (lower and upper bounds) for powers of a matrix efficiently? Can we counteract the denseness of the obtained results?

6

Compositional abstraction for interactive Markov chains

To overcome the absence of hierarchical, compositional facilities in modeling stochastic (continuous-time) systems a number of approaches have been proposed that basically distinguish themselves in two major respects (cf. Figure 6.1). The first is on the generality of distributions that are allowed for describing timed behavior. Stochastic process algebras such as Modest and TIPP allow for arbitrary distributions, whereas, e.g., PEPA and interactive Markov chains (IMCs) restrict to Markovian, that is exponentially distributed, delays in order to obtain analytically tractable models. The second major difference is the way time is incorporated. PEPA and TIPP associate delays to each action, a design decision that requires special semantics for cooperative behavior: which cooperative partner determines the delay, are delays cumulative or are delays reduced on cooperation? In contrast to that, IMCs and Modest keep functional and timed behavior separate: actions are performed instantaneously and before or after certain actions are performed, delays may be *executed*. We decided to build our theory of compositional abstraction [KKN09] on interactive Markov chains as this way of modeling timed behavior yields clear and elegant semantics, also for cooperative behavior; a further argument is, that in contrast to models with arbitrary residence time distributions, existing analysis techniques can be exploited for the computation of time-bounded reachability probabilities [Joh07].

	Markovian	General
integrated time $(a, \lambda).P$	PEPA [GH94]	TIPP [HHM98]
orthogonal time $\lambda.a.P$	IMC [Her02]	Modest [BDHK06]

Table 6.1: The zoo of stochastic process algebras (a selection).

Interactive Markov chains basically combine labeled transition systems and continuous-time Markov chains. For continuous-time Markov chains, we already investigated possible ways to perform abstraction. In the following we extend upon interval abstraction. When abstracting labeled transition systems in a similar way as we did for CTMCs, one obtains *modal transition systems* [HJS01]; we thus start this chapter by briefly introducing labeled and modal transition systems as well as parallel composition as a tool for compositional modeling. Then we will introduce interactive Markov chains and show how abstraction for this class of models yields an abstract model that combines modal transition systems with ACTMCs. The main part of this chapter will discuss how to perform abstraction and parallel composition on abstract interactive Markov chains and the relations of the resulting models in terms of simulation. A variant of parallel composition called symmetric composition is introduced for abstract IMCs and it is shown that both composition methods, when applicable, yield bisimilar models. We then provide the main results of this chapter: congruence and precongruence results. We conclude with a discussion on some algorithmic issues and a case study showing the potential gains when using this technique, especially when abstraction introduces symmetries that have not been present in the original model.

For simplicity, in this chapter we restrict to finite state spaces and initial states instead of initial distributions. Further, we do not introduce state labels for interactive Markov chains.

6.1 Labeled (modal) transition systems

In the previous chapters we dealt with probabilistic and timed systems, however, we did not concern ourselves with the functional behavior of a system. By functional behavior, we refer to the capability of performing *actions* in a defined way. For example, we want to be able to express that a client first sends a request before entering into a receiving mode.

For modeling such functional behavior, we use the notion of labeled transition systems.

Definition 6.1.1. A labeled transition system (LTS) $\mathcal{M} = (S, A, \mathbf{L}, s_0)$ consists of a state space S, a set of external and internal actions $A = A_e \dot\cup A_i$, a transition relation $\mathbf{L} : S \times A \times S \to \mathbb{B}_2$ and an initial state s_0.

Internal actions are assumed to be unobservable by the environment, whereas external actions may interact with other system components, e.g., a client could have external actions *send* and *receive* and an internal action *switch* for switching between sending and receiving mode.

A generalization of labeled transition systems that is suitable for three-valued abstraction are so-called modal transition systems (MTS) [LT88]. In literature, MTSs are often defined as a pair of LTSs where one of them describes the possible (*may*) behavior of the system and the other describes the required (*must*) behavior. For example, the capability of sending and receiving messages repeatedly could be the required behavior of a client, receiving messages before having sent a message in the first place could be possible but not required behavior. In this thesis, we use another definition for modal transition systems [HJS01] that fits more to the flavor of this thesis.

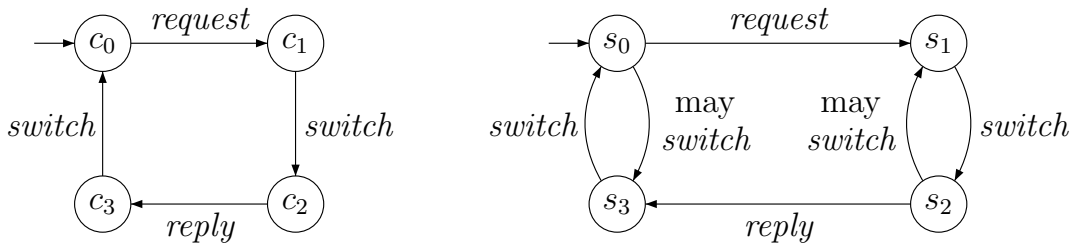


Figure 6.1: Client (left) and server (right) components.

Definition 6.1.2. A modal transition system (MTS) $\mathcal{M} = (S, A, \mathbf{L}, s_0)$ consists of a state space S, a set of external and internal actions $A = A_e \uplus A_i$, a *three-valued* transition relation $\mathbf{L} : S \times A \times S \to \mathbb{B}_3$ and an initial state s_0.

If $\mathbf{L}(s, a, s') = \top$, we say that there definitively exists a transition from s to s' with action a, for $\mathbf{L}(s, a, s') = ?$, we say that there *may* or *may not* exist such a transition and for $\mathbf{L}(s, a, s') = \bot$ there definitively does *not* exist an a-transition from s to s'. May-transitions can be interpreted as uncertainty, underspecification or as abstract behavior. We will not elaborate on performing abstraction for labeled transition systems, however, in the following sections, we introduce abstraction for interactive Markov chains that generalize LTSs conservatively.

Example 6.1.1. Consider the labeled transition system in Figure 6.1 (left), describing the behavior of a client that can send a request to and receive a reply from some other system components; further the client can switch between sending and receiving mode (independently from the rest of the system). Formally, *request*, *reply* $\in A_e$ whereas *switch* $\in A_i$.

The rest of the system, in this example, is the server MTS shown in Figure 6.1 (right) that can accept requests sent by the client and reply to it; as the client, it has to switch between sending and receiving mode internally. In contrast to the client, the server *may* switch into the receiving mode before

having replied a request and *may* switch to the sending mode before having received a request. Formally, $L(s_1, \mathit{switch}, s_2) = \top$, $L(s_2, \mathit{switch}, s_1) = ?$, etc.

In order to obtain the behavior of a system consisting of several components, say, one client and one server, one has to build the monolithic system that represents the exact behavior of the concurrent components. This is done by composing the components with respect to the synchronizing actions, that is, if an action is in the synchronization set, the components have to take the corresponding action simultaneously; non-synchronizing actions may be taken concurrently resulting in interleaving semantics. We now recall the definition of parallel composition for modal transition systems.

Definition 6.1.3. Let $\mathcal{M} = (S, A, \mathbf{L}, s_0)$ and $\mathcal{M}' = (S', A', \mathbf{L}', s_0')$ be modal transition systems. The *parallel composition* of $\mathcal{M}$ and $\mathcal{M}'$ w.r.t. synchronization set $\bar{A} \subseteq A_e \cap A'_e$ is defined by $\mathcal{M}||_{\bar{A}}\mathcal{M}' = (S \times S', A \cup A', \mathbf{L}'', (s_0, s_0'))$ where for $s, u \in S$ and $s', u' \in S'$:

$$\mathbf{L}''((s,s'),a,(u,u')) = \begin{cases} (\mathbf{L}(s,a,u) \sqcap (s' = u')) \sqcup (\mathbf{L}'(s',a,u') \sqcap (s = u)) & \text{if } a \notin \bar{A} \\ \mathbf{L}(s,a,u) \sqcap \mathbf{L}'(s',a,u') & \text{if } a \in \bar{A} \end{cases}$$

Example 6.1.2. Reconsider the client and server components ($\mathcal{M}$ and $\mathcal{M}'$) from Figure 6.1. To force the client and the server to actually communicate via *request* and *reply*, we put them in parallel choosing synchronization set $\bar{A} = \{\mathit{request}, \mathit{reply}\}$. This yields the MTS $\mathcal{M}||_{\bar{A}}\mathcal{M}'$ depicted in Figure 6.2.

Note that for *request* and *reply* actions, both components take the transition simultaneously whereas for *switch* we obtain the typical interleaving diamond as the client and the server switch the transfer mode in an arbitrary order.

A general introduction to compositional modeling can be found, e.g., in [Fok00]. For further information on modal transition systems we refer to [LT88, UC04].

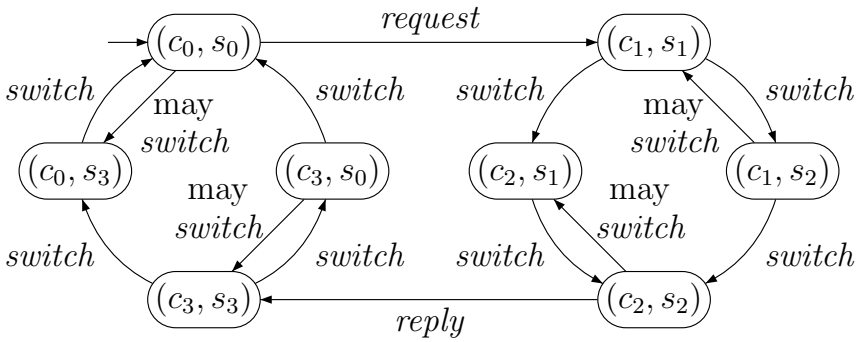


Figure 6.2: Composed system of one client and one server component synchronizing on *request* and *reply*.

6.2 Interactive Markov chains

Interactive Markov chains (IMCs) are a formalism for compositionally modeling systems that embrace nondeterministic and stochastic behavior in a completely orthogonal way. They can be seen as an extension of transition systems with exponentially distributed delays and probabilism. A detailed introduction to interactive Markov chains is given in [Her02]. In the following, we consider a restricted form, where all delays are exponentially distributed with the same exit rate. These *uniform* IMCs have been adopted for the performability analysis of statemate models [HJ07] by specifying random time constraints as uniform CTMCs that are composed with the functional behavior as in [HK00]. Most recently, in [Neu10], some issues have been discussed which are closely related to the question of how to obtain a uniform IMC from a general one.

Definition 6.2.1 (Uniform IMC)**.** A uniform interactive Markov chain is a tuple $(S, A, \mathbf{L}, \mathbf{P}, \lambda, s_0)$ where

- S is a non-empty finite set of states with initial state $s_0 \in S$,
- $A = A_e \uplus A_i$ is a non-empty finite set of *external* and *internal* actions,

- $\mathbf{L} : S \times A \times S \to \mathbb{B}_2$ is a two-valued labeled transition relation,
- $\mathbf{P} : S \times S \to [0, 1]$ is a transition probability function such that for all $s \in S$ it holds $\mathbf{P}(s, S) = 1$,
- $\lambda \in \mathbb{R}_{\geq 0}$ is a uniform exit rate.

A *Markovian transition* leads from state s to state s' (denoted $s \dashrightarrow s'$) iff $\mathbf{P}(s, s') > 0$; intuitively, if $s \dashrightarrow s'$, the probability to take this transition equals $\mathbf{P}(s, s')$ whereas the residence time in state s is exponentially distributed with rate λ. We require $\mathbf{P}(s, S) = 1$ to exclude deadlock states; this can easily be achieved by adding Markovian self-loops to states without Markovian transitions. Similarly, an *interactive transition* leads from s to s' via action a (denoted $s \xrightarrow{a} s'$) iff $\mathbf{L}(s, a, s') = \top$. As for MTSs, *external* actions $a \in A_e$ allow synchronization with the environment whereas *internal* actions $\tau \in A_i$ happen instantaneously and autonomously. The *maximal progress assumption* [Her02] states that as soon as some transition is enabled, it is not allowed to stall until some other transitions get enabled as well. This implies that internal actions cannot be delayed and thus, whenever internal transitions exist in the current state, the system nondeterministically moves along one of these transitions ignoring all other Markovian and external ones.

Example 6.2.1. As a running example, we consider the IMC model of a worker, depicted in Figure 6.3, where $\lambda = 10$. The work cycle starts in s_0 where the quality of a piece of raw material has to be determined. One out of ten pieces is flawed and cannot be used to craft a *premium* product. In that case (s_1) the worker will only be able to make a *value* product, which may take several work steps.

If the raw material is flawless, the worker decides for *value* or *premium*. For a *premium* product (s_3), everything has to be done smoothly in the first attempt, however, if the result is not perfect, with some corrections, a value product will be made. If the worker decides for *value* ($\bar{s}_2$), chances that no corrections are necessary are better than as if the raw material was flawed.

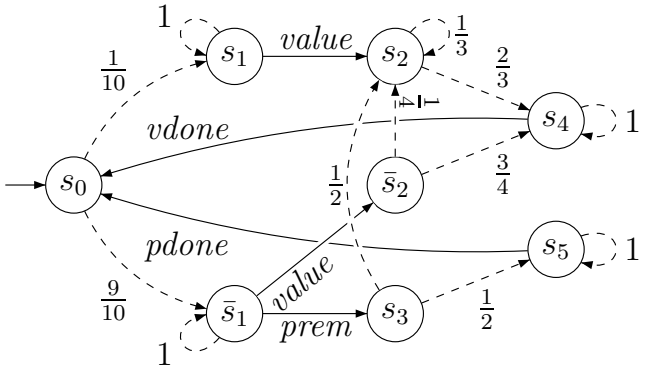


Figure 6.3: An interactive Markov chain.

Analysis. For uniform IMCs, efficient algorithms have been proposed for quantitative analysis, i.e. for the computation of minimal and maximal time-bounded reachability probabilities [Joh07]. It is required to *close* the system by turning all external actions to internal ones, preventing any further interaction with other components. Probability measures for IMCs are defined via *combined transitions* and the core of the algorithm used for that purpose is the same as for the computation of time-bounded reachability in ACTMCs (cf. Section 4.3). We refrain from giving specific details at this point and refer to [Joh07]. Instead, we concentrate on the formal relations between models rather than their corresponding probability measures.

6.3 Abstract interactive Markov chains

In this section, we develop an abstraction technique for IMCs similar to interval abstraction. We abstract an IMC along two lines: We use must- and may-transitions as introduced for modal transition systems to abstract from differences in the states' available nondeterministic choices. Further, we abstract from precise transition probabilities by taking intervals of probabilities as for AMCs. The combination of these two ingredients yields abstract interactive Markov chains.

Definition 6.3.1 (AIMC)**.** An abstract IMC is a tuple $(S, A, \mathbf{L}, \mathbf{P}^l, \mathbf{P}^u, \lambda, s_0)$ where S, A, λ and s_0 are as before, and

- $\mathbf{L} : S \times A \times S \to \mathbb{B}_3$ is a *three-valued* labeled transition relation, and
- $\mathbf{P}^l, \mathbf{P}^u : S \times S \to [0, 1]$ are transition probability bound functions such that $\mathbf{P}^l(s, S) \leq 1 \leq \mathbf{P}^u(s, S)$ for all $s \in S$.

Note that any IMC is an AIMC without may-transitions for which $\mathbf{P}^l = \mathbf{P}^u = \mathbf{P}$. Further, any interval Markov chain is an AIMC without must- and may-transitions. Disregarding $\mathbf{P}^l$, $\mathbf{P}^u$ and λ yields a modal transition system.

The requirement $\mathbf{P}^l(s, S) \leq 1 \leq \mathbf{P}^u(s, S)$ ensures that in every state s, a distribution μ over successor states can be chosen such that $\mathbf{P}^l(s, s') \leq \mu(s') \leq \mathbf{P}^u(s, s')$ for all $s' \in S$. This can be achieved by equipping such states with a Markovian $[1, 1]$ self-loop, without altering the model's behavior: if state s has an outgoing internal interactive transition, the maximal progress assumption guarantees that it still takes priority; otherwise, the self-loop neither alters its synchronization capabilities nor its sojourn time.

Example 6.3.1. Figure 6.4 (middle) depicts an example abstract model (AIMC) of a worker, similar to the one in Figure 6.4 (top). It abstracts from the difference between the raw material quality represented by the states s_1 and $\bar{s}_1$ in Figure 6.4 (top). Instead, the *premium* choice is modeled as a *may*-transition, i.e., it is possible to decide for *premium* in state s_1' but this possibility may be omitted. In state s_2', the probability that no further working step is necessary varies from $\frac{2}{3}$ to $\frac{3}{4}$.

Closing. AIMCs are (like IMCs and transition systems) subject to interaction. In order to carry out a quantitative analysis of such "open" models, one typically considers a closed variant, i.e., a variant that is behaviorally the same, but can no longer interact with some environment. This corresponds to the hiding operation in process algebras where external actions

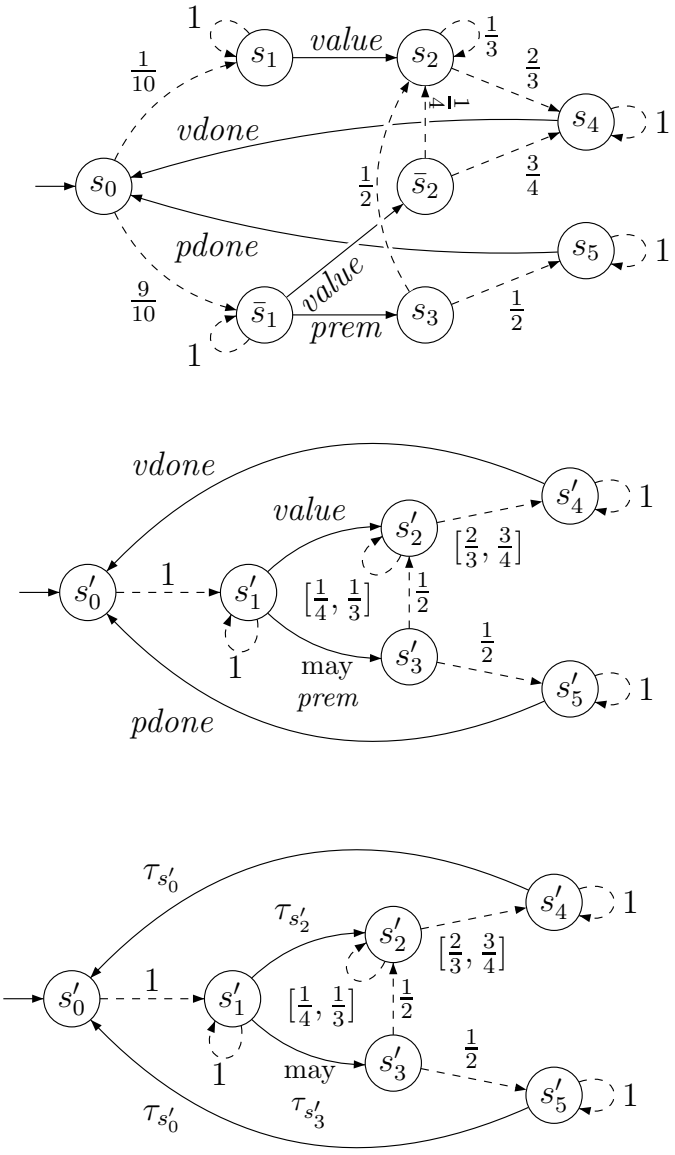


Figure 6.4: An open IMC (top), an open AIMC (middle) and its closed version (bottom).

are turned into internal (τ)-actions. We keep slightly more information: the distributions in case of a Markovian transition, and the target state name for interactive transitions. This facilitates a transformation of an AIMC into a CTMDP as described later on.

Definition 6.3.2 (Closed AIMC)**.** An AIMC $\mathcal{M} = (S, A, \mathbf{L}, \mathbf{P}^l, \mathbf{P}^u, \lambda, s_0)$ induces the closed AIMC $\mathcal{M}_\tau = (S, A_\tau, \mathbf{L}_\tau, \mathbf{P}^l, \mathbf{P}^u, \lambda, s_0)$ where

$$
\begin{aligned}
A_\tau &= \textstyle\bigcup_{s \in S} A_s^I \cup A_s^M, \\
A_s^I &= \{\tau_{s'} \mid \exists s' \in S\ \exists a \in A : \mathbf{L}(s,a,s') \neq \bot\}, \\
A_s^M &= \left\{\tau_\mu \mid \exists \mu \in \mathit{distr}(S)\ \forall s' \in S : \mathbf{P}^l(s,s') \leq \mu(s') \leq \mathbf{P}^u(s,s')\right\}, \\
\mathbf{L}_\tau(s,\tau,s') &= \begin{cases} \bigsqcup_{a \in A} \mathbf{L}(s,a,s') & \text{if } \tau = \tau_{s'} \\ \bot & \text{otherwise.} \end{cases}
\end{aligned}
$$

Note that in general, the sets A_s^M are uncountable as the range $[\mathbf{P}^l(s,s'), \mathbf{P}^u(s,s')]$ is dense. However, by replacing the set of all possible distributions by the set of extreme distributions as introduced in Section 4.1, the analysis of time-bounded reachability probabilities for such models will be made possible.

A *timed path* in a closed AIMC $\mathcal{M}_\tau$ is an infinite alternating sequence $\sigma = s_0 \xrightarrow{\tau_0,t_0} s_1 \xrightarrow{\tau_1,t_1} \ldots$ of states, internal actions and the states' residence times. A *path fragment* is a finite alternating sequence $\varsigma = s_0 \xrightarrow{\tau_0,t_0} s_1 \ldots \xrightarrow{\tau_n,t_n} s_{n+1}$. Time-abstract paths (path fragments) are alternating sequences of states and actions only. We adopt notions for paths (and path fragments) in CTMCs, e.g., $\sigma[i], \sigma@t, \mathit{Path}_{\mathcal{M}_\tau}, \mathit{Pathf}_{\mathcal{M}_\tau}$ and so forth; further, we denote the sets of time-abstract paths and path fragments by adding subscript *abs*.

Example 6.3.2. Figure 6.4 (bottom) illustrates the closed induced AIMC of the AIMC given in Figure 6.4 (middle).

In a closed AIMC, we classify states according to the type of outgoing transitions: the state space S is partitioned into the sets of *Markovian states*

S_M, *hybrid states* S_H and *may states* S_{MH}. A state is *Markovian* iff only Markovian transitions leave that state; a state is *hybrid* iff it has emanating Markovian and must-transitions. Further, states in S_{MH} only have outgoing Markovian and may-transitions but no must-transitions. By assumption, any state has at least one outgoing Markovian transition; hence, deadlock states and purely interactive states do not exist.

Nondeterminism. According to this state classification, three sources of nondeterminism occur in AIMCs: If multiple must-transitions exist in a state $s \in S_H$, that is, if $\mathbf{L}(s, a, s') = \mathbf{L}(s, b, s'') = \top$ for some $a, b \in A_s^I$ and $s' \neq s''$, the decision which state is selected as the successor state is nondeterministic.

May-transitions induce the second indefinite behavior: If $\mathbf{L}(s, a, s') = ?$ for some $a \in A_s^I$ and $s, s' \in S$, the existence of the may-transition to s' is nondeterministically resolved: In the positive case, the behavior is that of a hybrid state (i.e. the may-transition is treated as a must-transition). Otherwise, the may-transition will considered to be missing; if further must-transitions exist, the state is treated as a hybrid state, otherwise, it becomes a Markovian state.

The third type of nondeterminism occurs in Markovian states $s \in S_M$ of an AIMC: The abstraction yields transition probability intervals (formalized by $\mathbf{P}^l$ and $\mathbf{P}^u$) which induce a generally uncountable set of distributions that conform to these intervals. Selecting one of these distributions is nondeterministic. Note that in the special case of IMCs, the successor-state distribution is uniquely determined as $\mathbf{P}^l = \mathbf{P}^u$. Hence, IMCs do not exhibit this type of nondeterminism.

To formalize this intuition, let $A(s)$ be the set of *enabled actions* in state s. Formally, define $A(s) = A_s^I$ if $s \in S_H$, $A(s) = A_s^M$ if $s \in S_M$ and if $s \in S_{MH}$ let $A(s) = A_s^I \cup A_s^M$. Each action $\tau \in A(s)$ represents a distribution over the successors of state s. We define (for arbitrary $\tau \in A_\tau$) the distribution $\mathbf{T}(\tau)$ such that $\mathbf{T}(\tau_\mu) = \mu$ if $\tau = \tau_\mu$ is a Markovian transition and $\mathbf{T}(\tau_s) = \{s \mapsto 1\}$

if $\tau = \tau_s$ is an internal action; further, we extend this notion to sets of actions: for $B \subseteq A_\tau$ let $\mathbf{T}(B) = \bigcup_{\tau \in B} \mathbf{T}(\tau)$. We use normalization as for AMCs to restrict the intervals such that only valid probability distributions arise.

Schedulers. In order to maximize (or minimize) the probability to reach a set of goal states B within a given time bound t (denoted $Reach_{\leq t}(B)$), as for AMCs, schedulers are used for resolving the nondeterministic choices in the underlying AIMC. If the AIMC is in state $s \in S$, a scheduler selects an enabled action $\tau \in A(s)$ to continue with. For IMCs, we have basically the same scheduler classes as for CTMDPs, namely timed, history dependent and stationary schedulers that make decisions in a deterministic or randomized fashion (cf. Section 2.4.2).

Note that for Markovian states $s \in S_M$, the set A_s^M is generally uncountable as it consists of all distributions μ that obey the transition probability intervals of Markovian transitions emanating from state s. Therefore, we reduce A_s^M to the finitely many corresponding extreme distributions:

As discussed in Section 4.1, by taking convex combinations of extreme distributions one obtains the set of all possible distributions respecting the intervals. Further, as deterministic history-dependent schedulers and randomized history-dependent schedulers yield the same infimum and supremum of reachability probabilities for CTMDPs [BHKH05], as we will see, it suffices to consider deterministic ones for the analysis of AIMC. We thus may replace the uncountable sets A_s^M in the induced closed AIMC by finite sets $A_s^{M,extr} = \{\tau_\mu \mid \mu \in extr(\mathbf{P}^l, \mathbf{P}^u, S, s)\}$. We use A_s^{extr} to denote the set $A_s^{M,extr} \cup A_s^I$; further, let $A^{extr} = \bigcup_{s \in S} A_s^{extr}$.

We consider history-dependent randomized schedulers that choose from the set of extreme distributions and from interactive transitions: Let $\mathcal{M}_\tau$ be a closed AIMC. Then, an *extreme scheduler* on $\mathcal{M}_\tau$ is a history dependent scheduler $E : Path_{abs}^{\mathcal{M}_\tau} \to distr(A^{extr})$ with $supp(E(\varrho)) \subseteq A_{\lfloor\varrho\rfloor}^{extr}$ for all time-abstract path fragments (histories) $\varrho \in Path_{abs}^{\mathcal{M}_\tau}$.

Let $\mathcal{E}^{\mathcal{M}_\tau}$ denote the set of extreme schedulers for $\mathcal{M}_\tau$. For $E \in \mathcal{E}^{\mathcal{M}_\tau}$ and history $\varsigma \in \mathit{Pathf}_{abs}$, let the distribution over all successor states be given by $\sum_{\tau \in A^{extr}} E(\varsigma)(\tau) \cdot \mathbf{T}(\tau)(s)$ for all $s \in S$.

We are interested in the infimum and supremum of probability measures on measurable sets of paths over all schedulers in $\mathcal{E}^{\mathcal{M}_\tau}$. In the same fashion as for IMCs [Joh07, p.53], for AIMCs the probability measure $\Pr_E$ w.r.t. $E \in \mathcal{E}^{\mathcal{M}_\tau}$ can be inductively defined via *combined transitions* and *measurable schedulers*.

6.4 Compositional modeling

We consider *parallel* and *symmetric* composition of AIMCs and show that the latter typically yields more compact models which are bisimilar to the parallel composition of identical components. These operators are defined in a TCSP-like manner, i.e., they are parameterized with a set of external actions that need to be performed simultaneously by all involved components.

Definition 6.4.1 (Parallel composition)**.** Let $\mathcal{M} = (S, A, \mathbf{L}, \mathbf{P}^l, \mathbf{P}^u, \lambda, s_0)$ and $\mathcal{M}' = (S', A', \mathbf{L}', \mathbf{P}^{l'}, \mathbf{P}^{u'}, \lambda', s'_0)$ be AIMCs. The *parallel composition* of $\mathcal{M}$ and $\mathcal{M}'$ w.r.t. synchronization set $\bar{A} \subseteq A_e \cap A'_e$ is defined by:

$$\mathcal{M} ||_{\bar{A}} \mathcal{M}' = (S \times S', A \cup A', \mathbf{L}'', \mathbf{P}^{l''}, \mathbf{P}^{u''}, \lambda + \lambda', (s_0, s'_0))$$

where for $s, u \in S$ and $s', u' \in S'$:

- $\mathbf{L}''((s,s'),a,(u,u')) = \begin{cases} (\mathbf{L}(s,a,u) \sqcap (s' = u')) \sqcup (\mathbf{L}'(s',a,u') \sqcap (s = u)) & \text{if } a \notin \bar{A} \\ \mathbf{L}(s,a,u) \sqcap \mathbf{L}'(s',a,u') & \text{if } a \in \bar{A} \end{cases}$
- $\mathbf{P}^{l''}((s,s'),(u,u')) = \frac{\lambda}{\lambda+\lambda'} \cdot \mathbf{P}^l(s,u) \cdot \mathbf{1}(s',u') + \frac{\lambda'}{\lambda+\lambda'} \cdot \mathbf{P}^{l'}(s',u') \cdot \mathbf{1}(s,u)$
- $\mathbf{P}^{u''}((s,s'),(u,u')) = \frac{\lambda}{\lambda+\lambda'} \cdot \mathbf{P}^u(s,u) \cdot \mathbf{1}(s',u') + \frac{\lambda'}{\lambda+\lambda'} \cdot \mathbf{P}^{u'}(s',u') \cdot \mathbf{1}(s,u)$

Non-synchronizing actions are interleaved while actions in the set $\bar{A}$ need to be performed simultaneously by the involved components. Due to the memoryless property of exponential distributions, parallelly composed components delay completely independently. This is similar as in Markovian process algebras and for parallel composition of IMCs [Her02, HK00]. The proportion with which one of the components delays, i.e., $\frac{\lambda}{\lambda+\lambda'}$ and $\frac{\lambda'}{\lambda+\lambda'}$ respectively, results from the race between the components, more precisely the exponential distributions that are associated with each component. This justifies the definition of $\mathbf{P}^{l''}$ and $\mathbf{P}^{u''}$.

Composing several instances of the same AIMC by parallel composition may lead to excessive state spaces, in fact, the size of the state space grows exponentially with the number of components. To alleviate this problem, we consider symmetric composition as presented for IMCs in [HR98]. To formally define this notion, we use the concept of multisets (or bags). A multiset M over a finite set S is a function $S \to \mathbb{N}$. The cardinality of s in M is given by $M(s)$. We use common notations such as $s \in M$ for $M(x) > 0$ and, e.g., $M = \{\!|a, a, b|\!\}$ for M over $\{a, b\}$ with $M(a) = 2$ and $M(b) = 1$. For multisets M, M' over S, $M \uplus M' = M''$ is a multiset for which $M''(s) = M(s) + M'(s)$ for all $s \in S$. The same applies to $M \setminus M' = M''$ where $M''(s) = \max(0, M(s) - M'(s))$. A *multiset relation* $R : S \times S \to \mathbb{N}$ is a mapping w.r.t. multisets M, M' over S, iff $R(s, S) = M(s)$ and $R(S, u) = M'(u)$. The set of all *mappings* w.r.t. multisets M, M' is denoted $\Gamma_{M,M'}$.

Definition 6.4.2 (Symmetric composition)**.** For AIMC $\mathcal{M} = (S, A, \mathbf{L}, \mathbf{P}^l, \mathbf{P}^u, \lambda, s_0)$ and $\bar{A} \subseteq A_e$, the *symmetric composition* of $n \in \mathbb{N}_{>0}$ copies of $\mathcal{M}$ is given by:

$$|||^n_{\bar{A}}\mathcal{M} = (S'', A, \mathbf{L}'', \mathbf{P}^{l''}, \mathbf{P}^{u''}, n\lambda, \{\!|\overbrace{s_0, \ldots, s_0}^{n \text{ times}}|\!\})$$

where $S'' = \{M : S \to \mathbb{N} \mid \sum_{s \in S} M(s) = n\}$ and for all $s'', u'' \in S''$:

- $\mathbf{L}''(s'', a, u'') = \begin{cases} \bigsqcup_{s \in s'', u \in u'' : u'' = (s'' \setminus \{\!|s|\!\}) \uplus \{\!|u|\!\}} \mathbf{L}(s, a, u) & \text{if } a \notin \bar{A} \\ \bigsqcup_{R \in \Gamma_{s'', u''}} \bigsqcap_{s,u \in S : R(s,u) > 0} \mathbf{L}(s, a, u) & \text{if } a \in \bar{A} \end{cases}$

- $\mathbf{P}^{l''}(s'', u'') = \begin{cases} \frac{s''(s)}{n} \cdot \mathbf{P}^{l}(s, u) & \text{if } s'' \neq u'' = (s'' \setminus \{\!|s|\!\}) \uplus \{\!|u|\!\} \\ \sum_{s \in S} \frac{s''(s)}{n} \cdot \mathbf{P}^{l}(s, s) & \text{if } s'' = u'' \\ 0 & \text{otherwise} \end{cases}$

- $\mathbf{P}^{u''}(s'', u'') = \begin{cases} \frac{s''(s)}{n} \cdot \mathbf{P}^{u}(s, u) & \text{if } s'' \neq u'' = (s'' \setminus \{\!|s|\!\}) \uplus \{\!|u|\!\} \\ \sum_{s \in S} \frac{s''(s)}{n} \cdot \mathbf{P}^{u}(s, s) & \text{if } s'' = u'' \\ 0 & \text{otherwise} \end{cases}$

While in parallel compositions, states are tuples, in symmetric compositions they are represented by multisets. Transitions, however, are defined in the very same fashion as for parallel composition. Non-synchronized actions of n components are interleaved and in the synchronized case, all components have to simultaneously take the same synchronizing action. For transition probabilities, as all instances of the same component have the same exit rate λ, each component *wins the race* with probability $\frac{1}{n}$.

The application of both composition operators on AIMCs results in another AIMC. Note that this also implies uniformity of the resulting model, cf. [HJ07].

Lemma 6.4.1. *Let $\mathcal{M}$ and $\mathcal{M}'$ be AIMCs, $\bar{A}$ the synchronization set and $n \in \mathbb{N}_{>0}$, then $\mathcal{M}||_{\bar{A}}\mathcal{M}'$ and $|||^{n}_{\bar{A}}\mathcal{M}$ are AIMCs.*

Proof. Let $\mathcal{M} = (S, A, \mathbf{L}, \mathbf{P}^{l}, \mathbf{P}^{u}, \lambda, s_0)$ and $\mathcal{M}' = (S', A', \mathbf{L}', \mathbf{P}^{l'}, \mathbf{P}^{u'}, \lambda', s'_0)$. For $\mathcal{M}||_{\bar{A}}\mathcal{M}'$, $|||^{n}_{\bar{A}}\mathcal{M}$ and (in case of $\lambda = \lambda'$) $\mathcal{M} \cup \mathcal{M}'$ to be AIMCs, the following conditions have to be fulfilled:

(a) the state space has to be non-empty and finite,

(b) the action set has to be non-empty,

(c) the transition probability bounds have to be consistent,

(d) the set of initial states has to be a non-empty subset of the state space.

For all three operators, conditions (a), (b) and (d) follow from the fact that S, S', A, A', $s_0 \in S$ and $s'_0 \in S'$ are non-empty and all but A and A' are also finite. Condition (c) requires for $\mathcal{M}'' = (S'', A'', \mathbf{L}'', \mathbf{P}^{l''}, \mathbf{P}^{u''}, \lambda'', s''_0) \in \{\mathcal{M}||_{\bar{A}}\mathcal{M}', |||^n_{\bar{A}}\mathcal{M}, \mathcal{M} \cup \mathcal{M}'\}$ that for all $s'' \in S''$ it holds $\mathbf{P}^{l''}(s'', S'') \leq 1 \leq \mathbf{P}^{u''}(s'', S'')$. We will show the left inequation in detail. The proof for $1 \leq \mathbf{P}^{u''}(s'', S'')$ follows along the same lines.

For parallel composition we compute for all $s'' = (s, s') \in S''$:

$$\begin{aligned}
\mathbf{P}^{l''}(s'', S'') &= \textstyle\sum_{u \in S, u' \in S'} \mathbf{P}^{l''}((s, s'), (u, u')) \\
&= \textstyle\sum_{u \in S \setminus \{s\}} \mathbf{P}^{l''}((s, s'), (u, s')) + \sum_{u' \in S' \setminus \{s'\}} \mathbf{P}^{l''}((s, s'), (s, u')) \\
&\quad + \mathbf{P}^{l''}((s, s'), (s, s')) \\
&= \textstyle\sum_{u \in S \setminus \{s\}} \frac{\lambda}{\lambda + \lambda'} \cdot \mathbf{P}^{l}(s, u) + \sum_{u' \in S' \setminus \{s'\}} \frac{\lambda'}{\lambda + \lambda'} \cdot \mathbf{P}^{l'}(s', u') \\
&\quad + \tfrac{\lambda}{\lambda + \lambda'} \cdot \mathbf{P}^{l}(s, s') + \tfrac{\lambda'}{\lambda + \lambda'} \cdot \mathbf{P}^{l}(s, s') \\
&= \tfrac{\lambda}{\lambda + \lambda'} \cdot \mathbf{P}^{l}(s, S) + \tfrac{\lambda'}{\lambda + \lambda'} \cdot \mathbf{P}^{l'}(s', S') \\
&\leq \tfrac{\lambda}{\lambda + \lambda'} + \tfrac{\lambda'}{\lambda + \lambda'} \\
&= 1
\end{aligned}$$

For symmetric composition we compute for all $s'' \in S''$:

$$\begin{aligned}
\mathbf{P}^{l''}(s'', S'') &= \textstyle\sum_{u'' \in S''} \mathbf{P}^{l''}(s'', u'') \\
&= \textstyle\sum_{s \in s'', u \neq s} \mathbf{P}^{l''}(s'', s'' \setminus \{s\} \uplus \{u\}) + \mathbf{P}^{l''}(s'', s'') \\
&= \textstyle\sum_{s \in s'', u \neq s} \frac{s''(s)}{n} \cdot \mathbf{P}^{l}(s, u) + \sum_{s \in S} \frac{s''(s)}{n} \cdot \mathbf{P}^{l}(s, s) \\
&= \textstyle\sum_{s \in s''} \frac{s''(s)}{n} \cdot \mathbf{P}^{l}(s, S) \\
&\leq \textstyle\sum_{s \in s''} \frac{s''(s)}{n} \\
&= 1
\end{aligned}$$

Altogether we showed that conditions (a)-(d) hold for parallel and symmetric composition, implying that the result of those compositional operations on AIMCs are also AIMCs. □

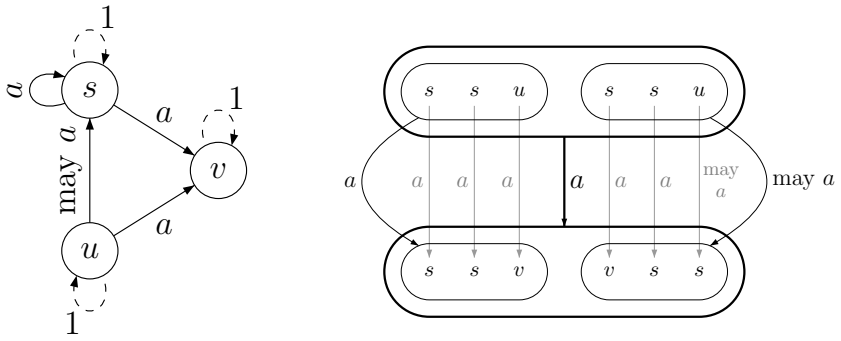


Figure 6.5: AIMC $\mathcal{M}$ (left) and how to derive modalities for synchronized transitions in $\mathcal{M}||_{\{a\}}\mathcal{M}||_{\{a\}}\mathcal{M}$ (right).

Example 6.4.1. Consider AIMC $\mathcal{M}$ in Figure 6.5, left. For state $\{\!|s, s, u|\!\}$ in $|||^3_{\{a\}}\mathcal{M}$, the states reachable with a synchronized must a-transition are $\{\!|s, s, v|\!\}$, $\{\!|s, v, v|\!\}$, $\{\!|v, v, v|\!\}$ and the states reachable with a synchronized may-transition are $\{\!|s, s, s|\!\}$, $\{\!|s, s, v|\!\}$, $\{\!|s, v, v|\!\}$. Note that there are several ways for the system to move to states $\{\!|s, s, v|\!\}$ and $\{\!|s, v, v|\!\}$. In both cases, there exists a must-transition and thus a must a-transition leads from $\{\!|s, s, u|\!\}$ to $\{\!|s, s, v|\!\}$ and $\{\!|s, v, v|\!\}$ respectively.

Let us have a closer look at the transition to $\{\!|s, s, v|\!\}$. As depicted in Figure 6.5 (right), from $\{\!|s, s, u|\!\}$ there are two possible ways to take a synchronized transition, either the component in state u takes a transition to v and the s-components take a self loop, or the u-component takes the may transition to s and one of the former s-components takes the transition to v while the other s-component takes the self-loop. In the former case, all involved transitions are must transitions, therefore yielding a must transition, in the latter case, as the transition from the u-component to s is a may transition, this yields a may transition at the system level. In the definition of symmetric composition, this is achieved by taking the *meet* over the involved component-level transitions.

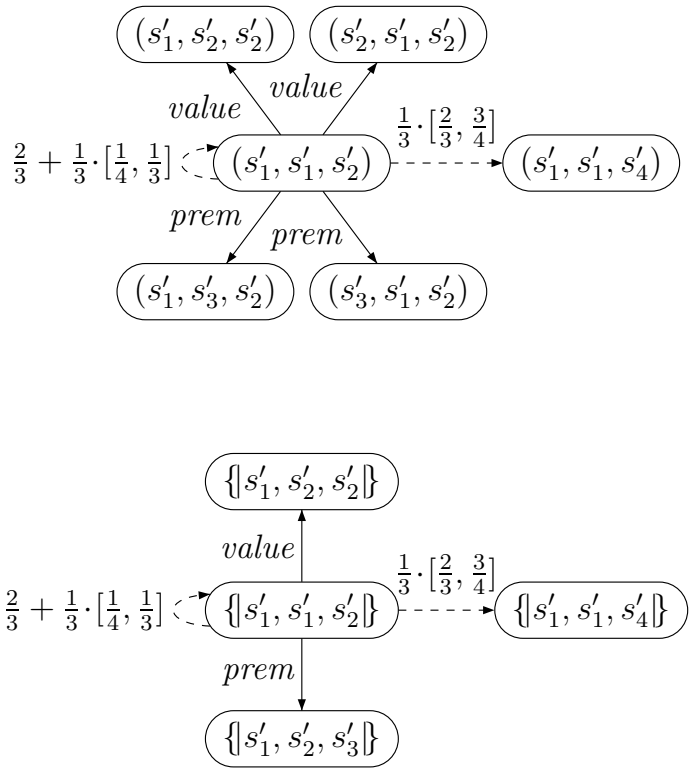


Figure 6.6: Fragment of the parallel composition $\mathcal{M}||_{\varnothing}\mathcal{M}||_{\varnothing}\mathcal{M}$ (top) and the symmetric composition $|||^3_{\varnothing}\mathcal{M}$ (bottom) for open AIMC $\mathcal{M}$ from Figure 6.4.

As we know of the existence of at least one must transition leading from $\{\!|s, s, u|\!\}$ to $\{\!|s, s, v|\!\}$, we conclude that we have indeed a must transition in the composed model. In the definition, this is achieved by taking the *join* over all possible system-level transitions.

Example 6.4.2. Modeling three independent (abstract) workers as given in Figure 6.4 can be done by both parallel and symmetric composition with an empty synchronization set. As shown in Table 6.2, differences in the sizes of the resulting models are significant. Figure 6.6 depicts the outgoing transitions of states (s'_1, s'_1, s'_2) and $\{\!|s'_1, s'_1, s'_2|\!\}$ that result from parallel and symmetric composition of three *abstract* workers.

states	IMC	AIMC
1 worker	8	6
3, par. comp.	512	216
3, sym. comp.	120	56

Table 6.2: Reductions in the size of the state space.

As suggested by Example 6.4.2, symmetric composition is a more space-efficient way to compose a component several times with itself. While for parallel composition of n identical components the size of the state space is in $\mathcal{O}(|S|^n)$, with symmetric composition, it is in $\mathcal{O}\left(\binom{n-1+|S|}{n}\right)$. The following result shows that symmetric composition is sound, i.e. it yields models that are bisimilar to parallel composition of a component with itself. This generalizes a similar result for IMCs, cf. [HR98].

Definition 6.4.3 (Bisimulation)**.** Let $\mathcal{M} = (S, A, \mathbf{L}, \mathbf{P}^l, \mathbf{P}^u, \lambda, s_0)$ be an AIMC. An equivalence $R \subseteq S \times S$ is a *bisimulation* on $\mathcal{M}$, iff for any sRs':

1. for all $a \in A$ and $u \in S$ with $\mathbf{L}(s, a, u) \neq \bot$, there exists $u' \in S$ with

$$\mathbf{L}(s, a, u) = \mathbf{L}(s', a, u') \text{ and } uRu'$$

2. if for all $a \in A_i$ and all $u \in S$ it holds $\mathbf{L}(s, a, u) \neq \top$, then for all $C \in S/R$:

$$\mathbf{P}^l(s, C) = \mathbf{P}^l(s', C) \text{ and } \mathbf{P}^u(s, C) = \mathbf{P}^u(s', C)$$

We write $s \approx s'$ if sRs' for some bisimulation R on $\mathcal{M}$ and we write $\mathcal{M} \approx \mathcal{M}'$ for IMCs $\mathcal{M}$ and $\mathcal{M}'$ with initial states s_0 and s'_0, iff $s_0 \approx s'_0$ holds for the disjoint union of $\mathcal{M}$ and $\mathcal{M}'$. Note that the union is only defined for two uniform AIMCs with the same exit rate as for different exit rates, the result is not uniform.

The first condition on may- and must-transitions is standard. The second condition asserts that for state s without outgoing internal must-transitions —which would have priority over Markovian transitions according to the maximal progress assumption— the probability to directly move to an equivalence class (under R) coincides with that of s'. The condition on probabilities is standard, whereas the exception of outgoing internal must-transition originates from IMCs [Her02, HK00]. We now present a result on the interchangeability of parallel and symmetric composition. We show that (a) the resulting AIMCs are bisimilar and (b) bisimulation is a congruence relation:

Theorem 6.4.2 (Symmetric composition)**.** *Let $\mathcal{M}$ be an AIMC, $\bar{A}$ a synchronization set and $n \in \mathbb{N}_{>0}$, then:*

$$|||^n_{\bar{A}}\mathcal{M} \approx \overbrace{\mathcal{M}||_{\bar{A}} \ldots ||_{\bar{A}}\mathcal{M}}^{n \text{ times}}$$

Proof. Let $\mathcal{M} = (S, A, \mathbf{L}, \mathbf{P}^l, \mathbf{P}^u, \lambda, s_0)$ be an AIMC, $\bar{A} \subseteq A_e$ and $n \in \mathbb{N}_{>0}$. Let

$$|||^n_{\bar{A}}\mathcal{M} = \mathcal{M}' = (S', A', \mathbf{L}', \mathbf{P}^{l'}, \mathbf{P}^{u'}, \lambda', s'_0)$$

and

$$\underbrace{\mathcal{M}||_{\bar{A}} \ldots ||_{\bar{A}}\mathcal{M}}_{n \text{ times}} = \tilde{\mathcal{M}}' = (\tilde{S}', \tilde{A}', \tilde{\mathbf{L}}', \tilde{\mathbf{P}}^{l'}, \tilde{\mathbf{P}}^{u'}, \tilde{\lambda}', \tilde{s}'_0).$$

We define $R_n \subseteq (S' \cup \tilde{S}') \times (S' \cup \tilde{S}')$ on $\mathcal{M}^{\cup}$ as the coarsest reflexive and symmetric relation with

$$s' R_n \tilde{s}' \text{ if } s' = \{s_1, \ldots, s_n\} \text{ and } \tilde{s}' = (s_1, \ldots, s_n) \text{ for } s_1, \ldots, s_n \in S.$$

Then, as we show in the following, (a) R_n is a bisimulation relation and (b) the initial states are R_n related, i.e. $s'_0 R_n \tilde{s}'_0$.

a)

We prove that R_n is a bisimulation relation: Therefore, assume $s' R_n \tilde{s}'$ for $s' = \{s_1, \ldots, s_n\}$ and $\tilde{s}' = (s_1, \ldots, s_n)$ with $s_1, \ldots, s_n \in S$.

1. If $\mathbf{L}^{\cup}(s', a, u') = x$ with $x \in \{\top, ?\}$, then we consider two cases:

   - Assume $a \notin \bar{A}$: As $\mathbf{L}^{\cup}(s', a, u') = x$, $(*)$ there exist $s \in s'$, $u \in u'$ such that $\mathbf{L}(s, a, u) = x$ and there are no $\hat{s} \in s'$, $\hat{u} \in u'$ such that $\mathbf{L}(\hat{s}, a, \hat{u}) > x$. By the definition of symmetric composition, there exists $k \in \{1, \ldots, n\}$ such that

   $$s' = \{s_1, \ldots, s_{k-1}, s, s_{k+1}, \ldots, s_n\} \text{ and}$$
   $$u' = \{s_1, \ldots, s_{k-1}, u, s_{k+1}, \ldots, s_n\}.$$

   As $s' R_n \tilde{s}'$, there exists a permutation $\pi \in Perm(\{1, \ldots, n\})$ such that

   $$\tilde{s}' = \left(s_{\pi(1)}, \ldots, s_{\pi(j-1)}, s_{\pi(j)}, s_{\pi(j+1)}, \ldots, s_{\pi(n)}\right).$$

   Then $\pi(j) = k$ for some j and $s_{\pi(j)} = s_k = s$. Applying π to u' yields

   $$\tilde{u}' = \left(s_{\pi(1)}, \ldots, s_{\pi(j-1)}, u, s_{\pi(j+1)}, \ldots, s_{\pi(n)}\right).$$

   From $(*)$ we obtain $\mathbf{L}^{\cup}(\tilde{s}', a, \tilde{u}') = x$. Further, from the definition of relation R_n, we conclude $\tilde{s}' R_n \tilde{u}'$.

   - Assume $a \in \bar{A}$: We first consider $x = \top$. For u' with

   $$\mathbf{L}'(s', a, u') = \bigsqcup_{\bar{R} \in \Gamma_{s',u'}} \bigsqcap_{s,u \in S^{\cup} : \bar{R}(s,u) > 0} \mathbf{L}(s, a, u) = \top$$

   it holds

   $$\exists \bar{R} \in \Gamma_{s',u'} : \forall s, u \in S^{\cup} : (\bar{R}(s, u) > 0 \implies \mathbf{L}(s, a, u) = \top).$$

   As $s' R_n \tilde{s}'$ we have $s' = \{s_1, \ldots, s_n\}$ and $\tilde{s}' = (s_1, \ldots, s_n)$ for some $s_1, \ldots, s_n \in S$. We will define $\tilde{u}'$ inductively based on some $\bar{R} \in \Gamma_{s',u'}$ with $\forall s, u \in S^{\cup} : (\bar{R}(s, u) > 0 \implies \mathbf{L}(s, a, u) = \top)$: let $\bar{R}_n = \bar{R}$ and for $i \in \{1, \ldots, n\}$, let

   $$\bar{R}_{i-1} = \bar{R}_i[(s_i, u_i) \mapsto \bar{R}_i(s_i, u_i) - 1] \text{ for some } u_i : \bar{R}_i(s_i, u_i) > 0.$$

Then for all $i \in \{1, \ldots, n\} : \mathbf{L}(s_i, a, u_i) = \top$ as $\bar{R}(s_i, u_i) > 0$. For $\tilde{u}' = (u_1, \ldots, u_n)$, this implies $\tilde{\mathbf{L}}'(\tilde{s}', a, \tilde{u}') = \prod_{i=1}^{n} \mathbf{L}(s_i, a, u_i) = \top$. Moreover, as by the definition of $\Gamma_{s',u'}$ it holds $\bar{R}(S', u) = u'(u)$ for all $u \in S$, it follows that $u' = \{u_1, \ldots, u_n\}$, i.e. $u' R_n \tilde{u}'$.

For $x \neq \bot$, it can be argued analogously to the $x = \top$ case, that for u' with $\mathbf{L}'(s', a, u') \neq \bot$ there exists $\tilde{u}'$ with $u' R_n \tilde{u}'$ and $\tilde{\mathbf{L}}'(\tilde{s}', a, \tilde{u}') \neq \bot$. Together with the result from $x = \top$ it follows that for $x \in \{?, \top\}$ and u' with $\mathbf{L}'(s', a, u') = x$ there exists $\tilde{u}'$ with $u' R_n \tilde{u}'$ and $\tilde{\mathbf{L}}'(\tilde{s}', a, \tilde{u}') = x = \mathbf{L}'(s', a, u')$.

2. For Markovian transitions we show for $s' = \{s_1, \ldots, s_n\}$ and $\tilde{s}' = (s_1, \ldots, s_n)$ with $s_1, \ldots, s_n \in S$ that for all $C \in S^{\cup}/R_n$:

$$\mathbf{P}^{l\cup}(s', C) = \mathbf{P}^{l\cup}(\tilde{s}', C) \text{ and } \mathbf{P}^{u\cup}(s', C) = \mathbf{P}^{u\cup}(\tilde{s}', C).$$

Note that the condition "$\mathbf{L}^{\cup}(s', a, u') \neq \top$ for all $a \in A_i$ and $u' \in S'$" from the definition of bisimulation is not used for this proof. If $C = \varnothing$, then $\mathbf{P}^{l\cup}(s', C) = \mathbf{P}^{l\cup}(\tilde{s}', C) = 0$; hence, in the following we assume $C \neq \varnothing$.

First, we lift the definition of parallel composition to the n-ary case for the lower bound probability matrix:

$$\begin{aligned}
&\tilde{\mathbf{P}}^{l'}((s_1, \ldots, s_n), (u_1, \ldots, u_n)) \\
&= \textstyle\sum_{i=1}^{n} \frac{1}{n} \cdot \mathbf{P}^{l}(s_i, u_i) \cdot \prod_{j \neq i} \mathbf{1}(s_j, u_j) \\
&= \begin{cases} \frac{1}{n} \cdot \mathbf{P}^{l}(s_i, u_i) & \text{if } \exists i.\ s_i \neq u_i \ \wedge\ \forall j \neq i.\ s_j = u_j \\ \sum_{i=1}^{n} \frac{1}{n} \cdot \mathbf{P}^{l}(s_i, u_i) & \text{if } \forall j.\ s_j = u_j \\ 0 & \text{otherwise.} \end{cases}
\end{aligned}$$

Next, we observe for $S^{\cup}/R_n$ that:

(a) every non-empty $C \in S^{\cup}/R_n$ contains exactly one $s' \in S'$

(b) $\{s_1, \ldots, s_n\} \in C \in S^{\cup}/R_n$
$\iff$ for all $\pi \in Perm(\{1, \ldots, n\}) : (s_{\pi(1)}, \ldots, s_{\pi(n)}) \in C$

From (a) it follows directly that for $s' \in S'$ and nonempty $C \in S^{\cup}/R_n$:

$$\mathbf{P}^{l\cup}(s', C) = \mathbf{P}^{l'}(s', C \cap S') = \mathbf{P}^{l'}(s', u') \text{ with } \{u'\} = C \cap S'$$

Let $\tilde{s}' = (s_1, \ldots, s_n)$, $s' = \{s_1, \ldots, s_n\}$, $s = s_i$ and $u' \neq s' \setminus \{s\} \uplus \{u\}$ for all $u \in S$. Then $\mathbf{P}^{\cup}_l(s', [u']_{R_n}) = \mathbf{P}^{\cup}_l(\tilde{s}', [u']_{R_n}) = 0$. Otherwise, i.e. $u' = s' \setminus \{s\} \uplus \{u\}$ for some $u \in S$, we set $C = [u']_{R_n}$ and obtain

$$\begin{aligned}
\mathbf{P}^{l\cup}(\tilde{s}', C) &= \tilde{\mathbf{P}}'_l(\tilde{s}', C \cap \tilde{S}') = \textstyle\sum_{\tilde{u}' \in C \cap \tilde{S}'} \tilde{\mathbf{P}}'_l(\tilde{s}', \tilde{u}') \\
&= \begin{cases} \sum_{\tilde{u}' \in C \cap \tilde{S}' : \tilde{u}' = (s_1, \ldots, s_{k-1}, u, s_{k+1}, s_n)} \frac{1}{n} \cdot \mathbf{P}^l(s, u) & \text{if } s \neq u \\ \sum_{i=1}^{n} \frac{1}{n} \cdot \mathbf{P}^l(s_i, u_i) & \text{if } s = u \end{cases} \\
&\overset{(*)}{=} \begin{cases} \frac{s'(s)}{n} \cdot \mathbf{P}^l(s, u) & \text{if } s' \neq u' \\ \sum_{i=1}^{n} \frac{1}{n} \cdot \mathbf{P}^l(s_i, u_i) & \text{if } s' = u' \end{cases} \\
&= \mathbf{P}^{l'}(s', u') = \mathbf{P}^{l\cup}(s', C)
\end{aligned}$$

where at $(*)$ we use the fact that there are $s'(s)$ positions k in $\tilde{s}'$ where s can be replaced by u such that $\tilde{u}' = (s_1, \ldots, s_{k-1}, u, s_{k+1}, \ldots, s_n)$.

For the upper bound probability matrix, the proof can be done analogously.

Altogether, for all $n \in \mathbb{N}_{>0}$, we showed that R_n as defined earlier is a bisimulation relation.

b)

Now, we show that the initial states are bisimilar. By definition:

$$s'_0 = \{\!|s_0, \ldots, s_0|\!\} \quad \text{and} \quad \tilde{s}'_0 = (s_0, \ldots, s_0)$$

For all $s_1, \ldots, s_n \in S_0$, it holds $\{s_1, \ldots, s_n\} R_n (s_1, \ldots, s_n)$ and thus, obviously,

$$\{\!|s_0, \ldots, s_0|\!\} R_n (s_0, \ldots, s_0).$$

Now, from (a) and (b), it follows $|||^n_{\bar{A}}\mathcal{M} \approx \underbrace{\mathcal{M}||_{\bar{A}} \ldots ||_{\bar{A}}\mathcal{M}}_{n \text{ times}}$. □

The following congruence result for bisimulation allows to interchange parallel composition of identical components by symmetric composition deep down in a system's description. E.g., for AIMCs $\mathcal{M}$, $\mathcal{N}$ and synchronization set $\bar{A}$, the congruence result, together with Theorem 6.4.2, implies $(\mathcal{M}||_{\varnothing}\mathcal{M})||_{\bar{A}}\mathcal{N} \approx (|||^2_{\varnothing}\mathcal{M})||_{\bar{A}}\mathcal{N}$.

Lemma 6.4.3. *Bisimulation $\approx$ is a congruence w.r.t. $||_{\bar{A}}$ and $|||_{\bar{A}}$.*

Proof. Let $\mathcal{M} = (S, A, \mathbf{L}, \mathbf{P}^l, \mathbf{P}^u, \lambda, S_0)$, $\mathcal{N} = (S, A, \mathbf{L}, \mathbf{P}^l, \mathbf{P}^u, \lambda, S'_0)$ and $\mathbf{P} = (S, A, \mathbf{L}, \mathbf{P}^l, \mathbf{P}^u, \lambda, S''_0)$. Reflexivity and symmetry are derived by trivial deduction from the definition of bisimulation.

For proving transitivity, for given bisimulation equivalences $R, R' : S \times S$ for $\mathcal{M} \approx \mathcal{N}$, $\mathcal{N} \approx \mathcal{P}$ respectively, we define $R'' : S \times S$ with

$$R'' = \{(s, s'') \mid \exists s' \in S : (s, s') \in R, (s', s'') \in R'\}$$

(note that $R'' \supseteq R \cup R'$ due to the reflexivity of equivalence R) and show that it is a bisimulation equivalence. For any $s, s'' \in S$ for which there exists $s' \in S$ with sRs' and $s'R's''$, it holds that:

- for all $a \in A$ and $u \in S$, if $\mathbf{L}(s, a, u) \neq \bot$ then, as R is a bisimulation equivalence, there exists $u' \in S$ with $\mathbf{L}(s', a, u') = \mathbf{L}(s, a, u) \neq \bot$ and uRu'; then, as R' is a bisimulation equivalence and $\mathbf{L}(s', a, u') \neq \bot$, there exists $u'' \in S$ with $\mathbf{L}(s'', a, u'') = \mathbf{L}(s', a, u')$ and $u'R'u''$; thus, for all $a \in A$ and $u \in S$, if $\mathbf{L}(s, a, u) \neq \bot$ then there exists $u'' \in S$ with $\mathbf{L}(s'', a, u'') = \mathbf{L}(s, a, u)$ and $uR''u''$.

- if $\mathbf{L}(s, a, u) \neq \top$ for *all* $a \in A$ and *all* $u \in S$, then due to symmetry of equivalence R, $\mathbf{L}(s', a, u') \neq \top$ for all $a \in A$ and all $u' \in S$; from the definition of R'' it follows that S/R and S/R' can be derived from S/R'' by splitting its elements such that, e.g., $C'' \in S/R''$ is

given by $\bigcup_{i=0}^{n} C_i$ for some $n \in \mathbb{N}$ and $C_i \in S/R$ for all $i \in \{0, \ldots, n\}$. Thus for all $s \in S$ it holds $\mathbf{P}^l(s, C'') = \sum_{C \in S/R : C \subseteq C''} \mathbf{P}^l(s, C)$ and $\mathbf{P}^l(s', C'') = \sum_{C' \in S/R' : C' \subseteq C''} \mathbf{P}^l(s', C')$ (similarly for $\mathbf{P}^u$); this yields that, if $\mathbf{L}(s, a, u) \neq \top$ for all $a \in A$ and all $u \in S$, then for all $C'' \in S/R''$:

$$\begin{aligned}\mathbf{P}^l(s, C'') &= \textstyle\sum_{C \in S/R : C \subseteq C''} \mathbf{P}^l(s, C) = \sum_{C \in S/R : C \subseteq C''} \mathbf{P}^l(s', C) \\ &= \mathbf{P}^l(s', C'') \\ &= \textstyle\sum_{C' \in S/R' : C \subseteq C''} \mathbf{P}^l(s', C') = \sum_{C' \in S/R' : C \subseteq C''} \mathbf{P}^l(s'', C') \\ &= \mathbf{P}^l(s'', C'')\end{aligned}$$

and $\mathbf{P}^u(s, C'') = \mathbf{P}^u(s'', C'')$ (analogously).

Thus R'' is a bisimulation equivalence. To show that $\mathcal{M} \approx \mathcal{P}$ we just have to prove that for every initial state $s_0 \in S_0$ there is an initial state $s_0'' \in S_0''$ in $\mathcal{P}$ with $s_0 R'' s_0''$. This follows trivially from $\mathcal{M} \approx \mathcal{N}$ and $\mathcal{N} \approx \mathcal{P}$ as by definition, for s_0 there exists initial state $s_0' \in S_0'$ with $s_0 R s_0'$ and for s_0' there exists initial state $s_0'' \in S_0''$ with $s_0' R' s_0''$ implying that $s_0 R'' s_0''$. □

6.5 Abstraction of components

This section describes the process of abstracting (A)IMCs by partitioning the state space, i.e., by grouping sets of concrete states to abstract ones. For state space S and partitioning S' of S, let $\alpha : S \to S'$ map states to their corresponding abstract one, i.e., $\alpha(s)$ denotes the abstract state of s, and $\gamma(s')$ is the set of concrete states that map to s'. Abstraction yields an AIMC that covers at least all possible behaviors of the concrete model, but perhaps more. The relationship between the abstraction and its concrete model is formalized by a *simulation*. We will define this notion and show that it is a precongruence with respect to parallel and symmetric composition. This result enables a *compositional* abstraction of AIMCs.

Definition 6.5.1 (Abstraction)**.** For an AIMC $\mathcal{M} = (S, A, \mathbf{L}, \mathbf{P}^l, \mathbf{P}^u, \lambda, s_0)$ and partitioning S' of S, the abstraction function $\alpha : S \to S'$ induces the AIMC $(S', A, \mathbf{L}', \mathbf{P}^{l'}, \mathbf{P}^{u'}, \lambda, \alpha(s_0))$, denoted by $\alpha(\mathcal{M})$, where:

- $\mathbf{L}'(s', a, u') = \begin{cases} \top & \text{if } \bigsqcap_{u \in \gamma(u')} \mathbf{L}(s, a, u) = \top \text{ for all } s \in \gamma(s') \\ \bot & \text{if } \bigsqcup_{u \in \gamma(u')} \mathbf{L}(s, a, u) = \bot \text{ for all } s \in \gamma(s') \\ ? & \text{otherwise} \end{cases}$
- $\mathbf{P}^{l'}(s', u') = \min_{s \in \gamma(s')} \sum_{u \in \gamma(u')} \mathbf{P}^l(s, u)$
- $\mathbf{P}^{u'}(s', u') = \min(1, \max_{s \in \gamma(s')} \sum_{u \in \gamma(u')} \mathbf{P}^u(s, u))$

Lemma 6.5.1. *For any AIMC $\mathcal{M}$ and abstraction function α, $\alpha(\mathcal{M})$ is an AIMC.*

Proof. The proof follows along the lines of the proof for Lemma 3.2.1. □

Example 6.5.1. Let $\mathcal{M}$ be the IMC in Figure 6.4 (left) and $\mathcal{N}$ be the AIMC in Figure 6.4 (middle). Then, $\mathcal{N} = \alpha(\mathcal{M})$ with $\alpha(s_i) = u_i$ for $i \in \{0, \ldots, 5\}$ and $\alpha(s'_i) = u_i$ for $i \in \{1, 2\}$. Consider a worker $\mathcal{M}'$ that is a variant of the one in Figure 6.4 (left), say, whose judgment on the quality of raw material is different, i.e. whose $\mathbf{P}(s_0, s_1)$ and $\mathbf{P}(s_0, s'_1)$ differ. For such a worker, we also get $\mathcal{N} = \alpha(\mathcal{M}')$. Symmetric composition of two different workers $\mathcal{M}$ and $\mathcal{M}'$ is not possible. However, replacing both $\mathcal{M}$ and $\mathcal{M}'$ by abstract worker $\mathcal{N}$ enables symmetric composition and yields a compact representation of an abstraction of $\mathcal{M}||_{\bar{A}}\mathcal{M}'$.

The formal relationship between an AIMC and its abstraction is defined in terms of a simulation. In fact, the notion defined below combines the concepts of refinement for modal transition systems [LT88] (items 1a and 1b) with that of probabilistic simulation [JL91] (item 2).

Definition 6.5.2 (Simulation)**.** For AIMC $\mathcal{M} = (S, A, \mathbf{L}, \mathbf{P}^l, \mathbf{P}^u, \lambda, s_0)$, $R \subseteq S \times S$ is a *simulation relation*, iff for all sRs' the following holds:

1a. for all $a \in A$ and $u \in S$ with $\mathbf{L}(s, a, u) \neq \bot$ there exists $u' \in S$ with $\mathbf{L}(s', a, u') \neq \bot$ and uRu',

1b. for all $a \in A$ and $u' \in S$ with $\mathbf{L}(s', a, u') = \top$ there exists $u \in S$ with $\mathbf{L}(s, a, u) = \top$ and uRu', and

2. if for all $a \in A_i$ and all $u \in S$ it holds $\mathbf{L}(s, a, u) \neq \top$, then for all $\mu \in \mathbf{T}(s)$ there exists $\mu' \in \mathbf{T}(s')$ with $\mu \preceq_R \mu'$.

We write $s \preceq s'$ if sRs' for some simulation R and $\mathcal{M} \preceq \mathcal{M}'$ for AIMCs $\mathcal{M}$ and $\mathcal{M}'$ with initial states $s_0 \preceq s_0'$ in the disjoint union of $\mathcal{M}$ and $\mathcal{M}'$.

Let us briefly explain this definition. Item 1a requires that any may- or must-transition of s must be reflected in s'. Item 1b requires that any must-transition of s' must match some must-transition of s, i.e., all required behavior of s' stems from s. Note that this allows a must-transition of s to be mimicked by a may-transition of s'. Finally, condition 2 requires that whenever there is not definitively an internal transition that prevents Markovian transitions to occur, then the stochastic behavior in s can be mimicked in s'.

Theorem 6.5.2. *For any AIMC $\mathcal{M}$ and abstraction function α:*

$$\mathcal{M} \preceq \alpha(\mathcal{M})$$

Proof. The claim follows from the proofs of the corresponding results for MTSs and ACTMCs in a straightforward manner. First, conditions 1a and 1b of Definition 6.5.2 have to be shown; this can be done just as for pure modal transition systems (cf. [LT88]). Second, condition 2 has to be shown; this part of the proof follows along the lines of the proof for Theorem 3.4.1. We thus omit the technical details. □

Example 6.5.2. Consider AIMCs $\mathcal{M}$ and $\mathcal{N}$ given in Example 6.5.1. As $\mathcal{N}$ is an abstraction of $\mathcal{M}$, it follows $\mathcal{M} \preceq \mathcal{N}$.

To be able to compose abstractions while preserving this formal relation, the following result is of interest. It allows to abstract parallel and symmetric compositions of AIMCs in a component-wise manner to avoid the need for generating the entire state space prior to abstraction.

Theorem 6.5.3. *Simulation $\preceq$ is a precongruence w.r.t. $||_{\bar{A}}$ and $|||_{\bar{A}}$.*

Proof. Reflexivity of $\preceq$ follows trivially from the definition. Consider AIMCs $\mathcal{M} = (S, A, \mathbf{L}, \mathbf{P}^l, \mathbf{P}^u, \lambda, s_0)$, $\mathcal{N} = (S', A', \mathbf{L}', \mathbf{P}^{l'}, \mathbf{P}^{u'}, \lambda', s_0')$ and $\mathcal{P} = (S'', A'', \mathbf{L}'', \mathbf{P}^{l''}, \mathbf{P}^{u''}, \lambda'', s_0'')$. To argue about simulation of states in different models we have to analyze the disjoint union. For simplicity, in this proof we refrain from explicitly composing the disjoint unions, however, we stress the necessity for all AIMCs involved in a union to have the same exit rates. This will be ensured in the following, as for $\mathcal{M} \preceq \mathcal{N}$ (and $\mathcal{N} \preceq \mathcal{P}$) it follows that $\lambda = \lambda'$ (and $\lambda' = \lambda''$ respectively).

Transitivity: Let $R : S \times S'$ and $R' : S' \times S''$ be simulation relations with $s_0 R s_0'$ and $s_0' R' s_0''$ respectively. We define $R'' : S \times S''$ with

$$R'' = \{(s, s'') \mid \exists s' \in S : (s, s') \in R, (s', s'') \in R'\}$$

and prove that it is a simulation relation (note that $R'' \supseteq R \cup R'$ due to reflexivity of R and R').

We show that conditions (1a), (1b) and (2) of Definition 6.5.2 are fulfilled: for all $s \in S$, $s'' \in S''$ for which there exists $s' \in S'$ with sRs' and $s'R's''$ it holds

1a. By the definition of simulation it holds that

$$\forall a \in A\, \forall u \in S : \mathbf{L}(s, a, u) \neq \bot \implies \exists u' \in S : \mathbf{L}'(s', a, u') \neq \bot \wedge uRu'$$

and

$$\forall a \in A\, \forall u' \in S : \mathbf{L}'(s', a, u') \neq \bot \implies \exists u'' \in S : \mathbf{L}''(s'', a, u'') \neq \bot \wedge u'R'u''.$$

Thus, it follows directly:

$$\forall a \in A\, \forall u \in S : \\ \mathbf{L}''(s'', a, u'') \neq \bot \implies \exists u'' \in S : \mathbf{L}(s, a, u) \neq \bot \wedge u R'' u''.$$

1b. As for (1a), by the definition of simulation it holds that

$$\forall a \in A\, \forall u'' \in S : \\ \mathbf{L}''(s'', a, u'') = \top \implies \exists u' \in S : \mathbf{L}'(s', a, u') = \top \wedge u' R' u''$$

and

$$\forall a \in A\, \forall u' \in S : \\ \mathbf{L}'(s', a, u') = \top \implies \exists u \in S : \mathbf{L}(s, a, u) = \top \wedge u R u'.$$

Thus, it follows directly:

$$\forall a \in A\, \forall u'' \in S : \\ \mathbf{L}''(s'', a, u'') = \top \implies \exists u \in S : \mathbf{L}(s, a, u) = \top \wedge u R'' u''$$

2. We show that if for all $u \in S$ and all $a \in A_i$ it holds $\mathbf{L}(s, a, u) \neq \top$, then for all $\mu \in \mathbf{T}(s)$ there exists $\mu'' \in \mathbf{T}(s'')$ and $\Delta'' : S \times S'' \to [0, 1]$ such that for all $u \in S$ and $u'' \in S''$: (cf. Figure 6.7)

   (a) $\Delta''(u, u'') > 0 \implies u R'' u''$

   (b) $\Delta''(u, S'') = \mu(u)$

   (c) $\Delta''(S, u'') = \mu''(u'')$

   First, note that if for all $u \in S$ it holds $\mathbf{L}(s, a, u) \neq \top$ for all $a \in A_i$, then for all $\mu \in \mathbf{T}(s)$ there exists $\mu' \in \mathbf{T}(s')$ and $\Delta : S \times S' \to [0, 1]$ such that for all $u \in S$ and $u' \in S'$:

   (a) $\Delta(u, u') > 0 \implies u R u'$

   (b) $\Delta(u, S') = \mu(u)$

   (c) $\Delta(S, u') = \mu'(u')$

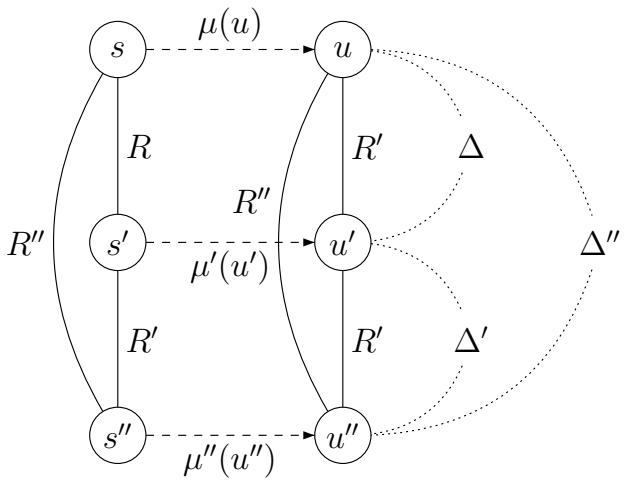


Figure 6.7: Transitivity

Second, we observe that for any $s \in S$ and $u \in S$ with $\mathbf{L}(s, a, u) \neq \top$ for some $a \in A_i$, all $s' \in S$ with sRs' may not have a successor $u' \in S$ with $\mathbf{L}(s', a, u') = \top$. Otherwise, if there was some $u' \in S$ with $\mathbf{L}(s', a, u') = \top$, due to condition (1b) in Definition 6.5.2 there would exist $u \in S$ with $\mathbf{L}(s, a, u) = \top$ and uRu', leading to a contradiction.

Hence, for all $s' \in S'$ with sRs', $u' \in S'$ and $a \in A_i$ it holds $\mathbf{L}(s', a, u') \neq \top$ and, as R' is a simulation relation, for all $\mu' \in \mathbf{T}(s')$ there exists $\mu'' \in \mathbf{T}(s'')$ and $\Delta' : S' \times S'' \to [0, 1]$ such that for all $u' \in S'$ and $u'' \in S''$:

(a) $\Delta'(u', u'') > 0 \implies u'R'u''$

(b) $\Delta'(u', S'') = \mu'(u')$

(c) $\Delta'(S', u'') = \mu''(u'')$

We define $\Delta'' : S \times S'' \to [0, 1]$ such that

$$\Delta''(u, u'') = \textstyle\sum_{u' \in S' : \mu'(u') > 0} \frac{\Delta(u,u') \cdot \Delta'(u',u'')}{\mu'(u')}$$

for Δ, Δ' and μ' satisfying the above constraints. For condition (2a), we observe that if $\Delta''(u, u'') > 0$ there exists u' such that $\Delta(u, u') > 0$ and $\Delta'(u', u'')$. Thus, uRu' and $u'R'u''$ imply $uR''u''$.

Further, we show conditions (2b) and (2c) by proving that for all $\mu \in \mathbf{T}(s)$ there exists $\mu'' \in \mathbf{T}(s'')$, such that $\Delta''(u, S'') = \mu(u)$ and $\Delta''(S, u'') = \mu''(u'')$ for all $u \in S$ and $u'' \in S''$:

$$\begin{aligned}
\Delta''(u, S'') &= \textstyle\sum_{u' \in S', u'' \in S'' : \mu'(u') > 0} \frac{\Delta(u,u') \cdot \Delta'(u',u'')}{\mu'(u')} \\
&= \textstyle\sum_{u' \in S' : \Delta'(u', S'') > 0} \frac{\Delta(u,u') \cdot \Delta'(u',S'')}{\Delta'(u',S'')} \\
&= \textstyle\sum_{u' \in S' : \Delta'(u', S'') > 0} \Delta(u, u') \\
&\overset{(*)}{=} \textstyle\sum_{u' \in S' : \Delta(S,u') > 0} \Delta(u, u') \\
&= \Delta(u, S') \\
&= \mu(u)
\end{aligned}$$

Equation $(*)$ follows from $\Delta'(u', S'') = \mu'(u') = \Delta(S, u')$ for all $u' \in S'$.

$$\begin{aligned}
\Delta''(S, u'') &= \textstyle\sum_{u \in S, u' \in S' : \mu'(u') > 0} \frac{\Delta(u,u') \cdot \Delta'(u',u'')}{\mu'(u')} \\
&= \textstyle\sum_{u' \in S' : \Delta(S,u') > 0} \frac{\Delta(S,u') \cdot \Delta'(u',u'')}{\Delta(S,u')} \\
&= \textstyle\sum_{u' \in S' : \Delta(S,u') > 0} \Delta'(u', u'') \\
&\overset{(*)}{=} \textstyle\sum_{u' \in S' : \Delta'(u', S'') > 0} \Delta'(u', u'') \\
&= \Delta'(S', u'') \\
&= \mu''(u'')
\end{aligned}$$

This concludes the proof of transitivity.

Now we show that parallel composition does not destroy simulation relations. Let $\mathcal{M}||_{\bar{A}}\mathcal{P} = (S \times S'', A \cup A'', \tilde{\mathbf{L}}, \tilde{\mathbf{P}}^l, \tilde{\mathbf{P}}^u, \lambda + \lambda'', (s_0, s_0''))$ and $\mathcal{N}||_{\bar{A}}\mathcal{P} = (S' \times S'', A' \cup A'', \tilde{\mathbf{L}}', \tilde{\mathbf{P}}^{l'}, \tilde{\mathbf{P}}^{u'}, \lambda' + \lambda'', (s_0', s_0''))$. Recall that from $\mathcal{M} \preceq \mathcal{N}$ it follows $\lambda = \lambda'$ and therefore the union of $\mathcal{M}||_{\bar{A}}\mathcal{P}$ and $\mathcal{N}||_{\bar{A}}\mathcal{P}$ that will implicitly be used is valid.

We show that $\mathcal{M} \preceq \mathcal{N}$ implies $\mathcal{M}||_{\bar{A}}\mathcal{P} \preceq \mathcal{N}||_{\bar{A}}\mathcal{P}$ for synchronization set $\bar{A}$, i.e. that for initial state (s_0, s_0'') there exists (s_0', s_0'') with $(s_0, s_0'') \preceq (s_0', s_0'')$. Let $\tilde{R} : (S \times S'') \times (S' \times S'')$ such that $(s, s'')\tilde{R}(s', s''')$ iff $s \preceq s'$ and $s'' \preceq s'''$. From $\mathcal{M} \preceq \mathcal{N}$ we know that $s_0 \preceq s_0'$ and due to reflexivity, $s_0'' \preceq s_0''$. Thus, $(s_0, s_0'')\tilde{R}(s_0', s_0'')$.

In the following, we show that for all $(s, s'') \in S \times S''$ and $(s', s''') \in S' \times S''$ with sRs' and $s''R's'''$ for simulation relations R and R', conditions (1a), (1b) and (2) in Definition 6.5.2 are fulfilled, i.e. that $\tilde{R}$ is a simulation relation.

1a. This can be shown in a similar fashion as for (1b).

1b. For $a \in \bar{A}$ we compute:

$$\begin{aligned}
&\forall(u', u''') \; \exists(u, u'') : \tilde{\mathbf{L}}'((s', s'''), a, (u', u''')) = \top \\
&\qquad \Longrightarrow \; \tilde{\mathbf{L}}((s, s''), a, (u, u'')) = \top \wedge (u, u'')\tilde{R}(u', u''') \\
\Longleftrightarrow \\
&\forall(u', u''') \; \exists(u, u'') : \mathbf{L}'(s', a, u') \sqcap \mathbf{L}''(s''', a, u''') = \top \\
&\qquad \Longrightarrow \; \mathbf{L}(s, a, u) \sqcap \mathbf{L}''(s'', a, u'') = \top \wedge (u, u'')\tilde{R}(u', u''') \\
\Longleftrightarrow \\
&\forall(u', u''') \; \exists(u, u'') : \mathbf{L}'(s', a, u') = \top \wedge \mathbf{L}''(s''', a, u''') = \top \\
&\qquad \Longrightarrow \; \mathbf{L}(s, a, u) = \top \wedge \mathbf{L}''(s'', a, u'') = \top \wedge (u, u'')\tilde{R}(u', u''')
\end{aligned}$$

This follows directly from sRs' and $s''R's'''$ as

$$\forall u' \; \exists u : \mathbf{L}'(s', a, u') = \top \implies \mathbf{L}(s, a, u) = \top \wedge uRu'$$

and

$$\forall u''' \; \exists u'' : \mathbf{L}''(s''', a, u''') = \top \implies \mathbf{L}''(s'', a, u'') = \top \wedge u''Ru'''$$

For $a \notin \bar{A}$ we compute:

$$\begin{aligned}
&\forall(u', u''') \; \exists(u, u'') : \tilde{\mathbf{L}}'((s', s'''), a, (u', u''')) = \top \\
&\qquad \Longrightarrow \; \tilde{\mathbf{L}}((s, s''), a, (u, u'')) = \top \wedge (u, u'')\tilde{R}(u', u''').
\end{aligned}$$

$$\Longleftrightarrow$$

$$\begin{aligned}\forall(u', u''')\ \exists(u, u'') :\\ &(\mathbf{L}'(s', a, u') \sqcap (s''' = u''')) \sqcup (\mathbf{L}''(s''', a, u''') \sqcap (s' = u')) = \top\\ &\implies (\mathbf{L}(s, a, u) \sqcap (s'' = u'')) \sqcup (\mathbf{L}''(s'', a, u'') \sqcap (s = u)) = \top\\ &\qquad \wedge (u, u'')\tilde{R}(u', u''')\end{aligned}$$

We investigate the two cases where on the left side of the implication either $(\mathbf{L}'(s', a, u') \sqcap (s''' = u''')) = \top$ or $(\mathbf{L}''(s''', a, u''') \sqcap (s' = u')) = \top$.

If $(\mathbf{L}'(s', a, u') \sqcap (s''' = u'''))$ resolves to $\top$, so does $\mathbf{L}(s, a, u)$ for some $u \in S$. We choose $u'' = s''$, such that $(s'' = u'') = \top$. For the right side of the implication to be fulfilled, it remains to show that $(u, u'')\tilde{R}(u', u''')$. From the satisfaction of $(s''' = u''')$ it follows $u''' = s'''$ and together with $u'' = s''$, from $s''R's'''$ we directly get $u''R'u'''$. Further, sRs' implies

$$\forall u'\ \exists u : \mathbf{L}'(s', a, u') = \top \implies \mathbf{L}(s, a, u) = \top \wedge uRu'.$$

Thus, there exist $u \in S$ and $u'' \in S''$ fulfilling the right side of the implication.

For the case where $(\mathbf{L}(s''', a, u''') \sqcap (s' = u'))$ resolves to $\top$, the proof goes along the same lines.

2. First, note that from sRs' it follows: if for all $u \in S$ it holds $\mathbf{L}(s, a, u) \neq \top$ for all $a \in A_i$, then for all $\mu \in \mathbf{T}(s)$ there exists $\mu' \in \mathbf{T}(s')$ and $\Delta : S \times S' \to [0, 1]$ such that for all $u \in S$ and $u' \in S'$:

   (a) $\Delta(u, u') > 0 \implies uRu'$

   (b) $\Delta(u, S') = \mu(u)$

   (c) $\Delta(S, u') = \mu'(u')$

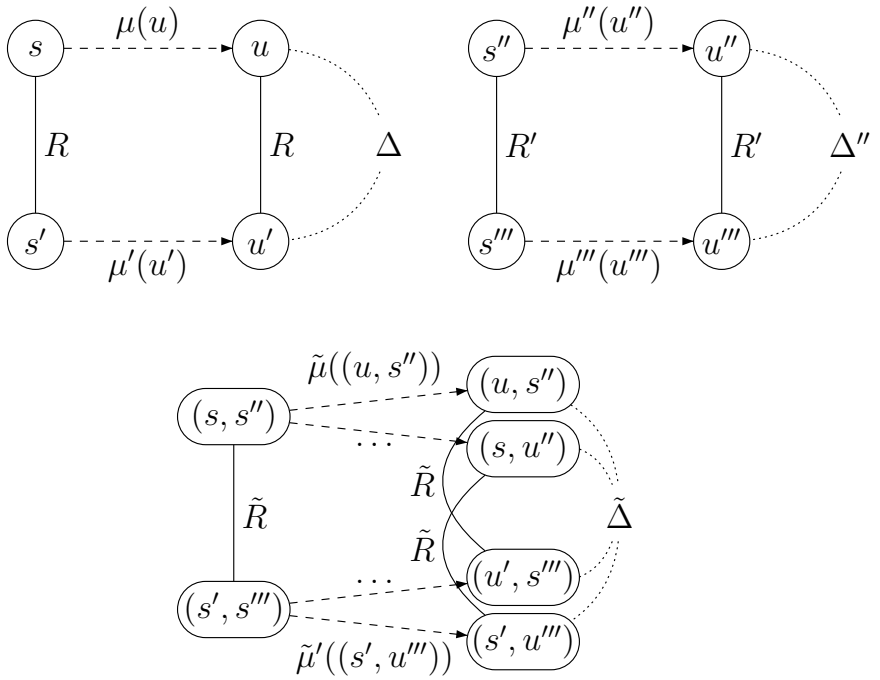


Figure 6.8: Compatibility with parallel composition

Second, from $s''R's'''$ it follows: if for all $u'' \in S''$ it holds $\mathbf{L}(s'', a, u'') \neq \top$ for all $a \in A_i$, then for all $\mu'' \in \mathbf{T}(s'')$ there exists $\mu''' \in \mathbf{T}(s''')$ and $\Delta'' : S'' \times S'' \to [0,1]$ such that for all $u'' \in S''$ and $u''' \in S''$:

(a) $\Delta''(u'', u''') > 0 \implies u''R'u'''$

(b) $\Delta''(u'', S'') = \mu''(u'')$

(c) $\Delta''(S'', u''') = \mu'''(u''')$

We show that, if for all $(u, u'') \in S \times S''$ it holds $\tilde{\mathbf{L}}((s, s''), a, (u, u'')) \neq \top$ for all $a \in A_i \cup A_i'$, then for all $\tilde{\mu} \in \mathbf{T}((s, s''))$ there exists $\tilde{\mu}' \in \mathbf{T}((s', s'''))$ and $\tilde{\Delta} : (S \times S'') \times (S' \times S'') \to [0,1]$ such that for all $(u, u'') \in S \times S''$ and $(u', u''') \in S' \times S''$: (cf. Figure 6.8)

(a) $\tilde{\Delta}((u, u''), (u', u''')) > 0 \implies (u, u'')\tilde{R}(u', u''')$

(b) $\tilde{\Delta}((u, u''), S' \times S'') = \tilde{\mu}((u, u''))$

(c) $\tilde{\Delta}(S \times S'', (u', u''')) = \tilde{\mu}'((u', u'''))$

Given $(s, s'') \in S \times S''$ and $(s', s''') \in S' \times S''$, we define $\tilde{\Delta} : (S \times S'') \times (S' \times S'') \to [0,1]$ such that:

$$\begin{aligned}\tilde{\Delta}((u, u''), (u', u''')) = {} & \tfrac{\lambda}{\lambda+\lambda''} \cdot \Delta(u, u') \cdot \mathbf{1}(s'', u'') \cdot \mathbf{1}(s''', u''') \\ & + \tfrac{\lambda''}{\lambda+\lambda''} \cdot \Delta''(u'', u''') \cdot \mathbf{1}(s, u) \cdot \mathbf{1}(s', u')\end{aligned}$$

Condition (a) follows from this definition as

$$\begin{aligned}\tilde{\Delta}((u, u''), (u', u''')) > 0 \implies & (\Delta(u, u') > 0 \wedge s'' = u'' \wedge s''' = u''') \\ & \vee (\Delta''(u'', u''') > 0 \wedge s = u \wedge s' = u') \\ \implies & (uRu' \wedge s'' = u'' \wedge s''' = u''') \\ & \vee (u''R'u''' \wedge u = u' \wedge s = u \wedge s' = u') \\ \implies & (u, u'')\tilde{R}(u', u''')\end{aligned}$$

where in the last implication we use the fact that sRs' and $s''R's'''$.

Regarding condition (b), for any $\tilde{\mu} \in \mathbf{T}((s, s''))$ we compute:

$$\begin{aligned}
\tilde{\mu}((u, u'')) &= \tfrac{\lambda}{\lambda+\lambda''} \cdot \mu(u) \cdot \mathbf{1}(s'', u'') + \tfrac{\lambda''}{\lambda+\lambda''} \cdot \mu''(u'') \cdot \mathbf{1}(s, u) \\
&= \textstyle\sum_{u' \in S'} \tfrac{\lambda}{\lambda+\lambda''} \cdot \Delta(u, u') \cdot \mathbf{1}(s'', u'') \\
&+ \textstyle\sum_{u''' \in S''} \tfrac{\lambda''}{\lambda+\lambda''} \cdot \Delta''(u'', u''') \cdot \mathbf{1}(s, u) \\
&= \textstyle\sum_{u''' \in S''} \Big(\sum_{u' \in S'} \tfrac{\lambda}{\lambda+\lambda''} \cdot \Delta(u, u') \cdot \mathbf{1}(s'', u'') \cdot \mathbf{1}(s''', u''') \\
&\qquad + \textstyle\sum_{u' \in S'} \tfrac{\lambda''}{\lambda+\lambda''} \cdot \Delta''(u'', u''') \cdot \mathbf{1}(s, u) \cdot \mathbf{1}(s', u') \Big) \\
&= \textstyle\sum_{u' \in S', u''' \in S''} \tfrac{\lambda}{\lambda+\lambda''} \cdot \Delta(u, u') \cdot \mathbf{1}(s'', u'') \cdot \mathbf{1}(s''', u''') \\
&\qquad + \tfrac{\lambda''}{\lambda+\lambda''} \cdot \Delta''(u'', u''') \cdot \mathbf{1}(s, u) \cdot \mathbf{1}(s', u') \\
&= \textstyle\sum_{u' \in S', u''' \in S''} \tilde{\Delta}((u, u''), (u', u''')) \\
&= \tilde{\Delta}((u, u''), S' \times S'')
\end{aligned}$$

Analogously, for condition (c) we compute for any $\tilde{\mu}' \in \mathbf{T}((s', s'''))$:

$$\begin{aligned}
\tilde{\mu}'((u', u''')) &= \tfrac{\lambda}{\lambda+\lambda''} \cdot \mu'(u') \cdot \mathbf{1}(s''', u''') + \tfrac{\lambda''}{\lambda+\lambda''} \cdot \mu'''(u''') \cdot \mathbf{1}(s', u') \\
&= \textstyle\sum_{u \in S} \tfrac{\lambda}{\lambda+\lambda''} \cdot \Delta(u, u') \cdot \mathbf{1}(s''', u''') \\
&+ \textstyle\sum_{u'' \in S''} \tfrac{\lambda''}{\lambda+\lambda''} \cdot \Delta''(u'', u''') \cdot \mathbf{1}(s', u') \\
&= \textstyle\sum_{u'' \in S''} \Big(\sum_{u \in S} \tfrac{\lambda}{\lambda+\lambda''} \cdot \Delta(u, u') \cdot \mathbf{1}(s'', u'') \cdot \mathbf{1}(s''', u''') \\
&\qquad + \textstyle\sum_{u \in S} \tfrac{\lambda''}{\lambda+\lambda''} \cdot \Delta''(u'', u''') \cdot \mathbf{1}(s, u) \cdot \mathbf{1}(s', u') \Big) \\
&= \textstyle\sum_{u \in S, u'' \in S''} \tfrac{\lambda}{\lambda+\lambda''} \cdot \Delta(u, u') \cdot \mathbf{1}(s'', u'') \cdot \mathbf{1}(s''', u''') \\
&\qquad + \tfrac{\lambda''}{\lambda+\lambda''} \cdot \Delta''(u'', u''') \cdot \mathbf{1}(s, u) \cdot \mathbf{1}(s', u') \\
&= \textstyle\sum_{u \in S, u'' \in S''} \tilde{\Delta}((u, u''), (u', u''')) \\
&= \tilde{\Delta}(S \times S'', (u', u'''))
\end{aligned}$$

Thus, conditions (a) to (c) hold.

We complete this proof by observing that $\preceq$ is reflexive, transitive and compatible with parallel composition and as parallel and symmetric composition yield bisimilar AIMCs, the same applies to symmetric composition. □

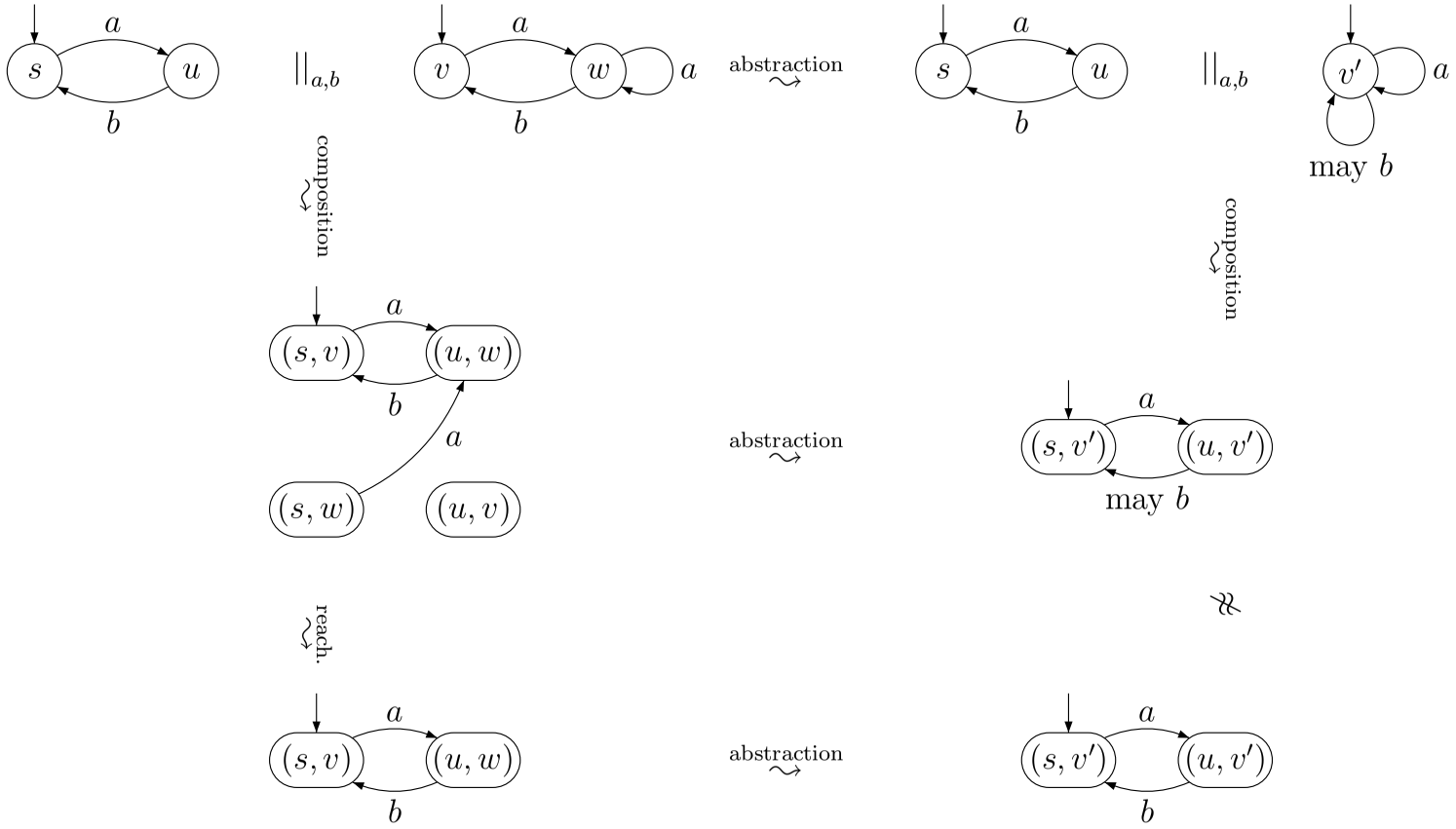


Figure 6.9: Performing abstraction on the component level vs. abstracting the monolithic system.

We conclude this section with an example, showing that performing abstraction on component level in general yields more abstract models than applying abstraction to the underlying monolithic model. In fact, we consider a modal transition system; an AIMC with exactly the same behavior is obtained by adding a Markovian self-loop with probability one to each state.

Example 6.5.3. Consider the MTS in Figure 6.9 (top, left). Deriving the monolithic model and then merging all states (s_0, v), (s_1, v) with $s_0, s_1 \in \{s, u\}$ and $v \in \{v, w\}$ by means of abstraction yields the same MTS (middle, right). However, states (s, w) and (u, v) are not reachable from the initial state (s, v) in the concrete monolithic model (middle, left) and thus can be omitted (bottom, left). Merging all states (s_0, v), (s_1, v) with $s_1, s_2 \in \{s, u\}$ and $v \in \{v, w\}$ in this MTS yields the MTS bottom right, which in fact simulates the MTS obtained by performing abstraction on component level.

6.6 Time-bounded reachability

In this section, we show how to analyze closed AIMCs by reducing them to uniform IMCs. As presented in [Joh07], those can be reduced to uniform continuous-time Markov decision processes (CTMDP) for which an efficient algorithm is implemented in MRMC, a state of the art model checker. We analyze two reachability objectives for the running example and show how abstraction and symmetric composition reduce the maximal size of the state space during the construction of the model.

To obtain the induced IMC for an AIMC, we separate the nondeterministic choice for values from the intervals in Markovian states from the actual Markovian behavior, i.e. the delay and the subsequent probabilistic transitions. This is achieved by adding one intermediate state for each extreme distribution.

Definition 6.6.1 (Induced IMC)**.** For closed AIMC $\mathcal{M} = (S, A, \mathbf{L}, \mathbf{P}^l, \mathbf{P}^u, \lambda, s_0)$, the *induced IMC* is given by $\theta(\mathcal{M}) = (S \uplus S^{extr}, A^{extr}, \mathbf{L}', \mathbf{P}', \lambda, s_0)$,

- $S^{extr} = \{s_\mu \mid \exists s \in S : \mu \in \mathit{extr}(\mathbf{P}^l, \mathbf{P}^u, S, s)\}$
- $\mathbf{L}'(s,a,s') = \begin{cases} \mathbf{L}(s,a,s') & \text{if } s \in S_H \cup S_{MH},\ a = \tau_{s'} \\ \top & \text{if } s \in S_M \cup S_{MH},\ a = \tau_\mu,\ s' = s_\mu \\ & \text{and } \mu \in \mathit{extr}(\mathbf{P}^l, \mathbf{P}^u, S, s) \\ \bot & \text{otherwise} \end{cases}$
- $\mathbf{P}'(s,s') = \begin{cases} \mu(s') & \text{if } s = s_\mu \in S^{extr} \\ \mathbf{1}(s,s') & \text{otherwise} \end{cases}$

Lemma 6.6.1. *For a closed AIMC $\mathcal{M}$ it holds that $\theta(\mathcal{M})$ is a closed uniform IMC.*

Proof. Follows directly from the definition and the fact that the given AIMC is uniform. □

Example 6.6.1. Let $\mathcal{M}$ be the symmetric composition of two independent abstract workers as depicted in Figure 6.4 (middle). We focus on state $\{\!|s_0, s_2|\!\}$ in $\mathcal{M}$, cf. Figure 6.10 (left). In the corresponding induced IMC $\theta(\mathcal{M})$, there are new states s_μ and $s_{\mu'}$ with outgoing Markovian transitions according to the extreme distributions μ and μ' of $\{\!|s_0, s_2|\!\}$ with $\mu = \{\{\!|s_1, s_2|\!\} \mapsto \frac{1}{2}, \{\!|s_0, s_2|\!\} \mapsto \frac{1}{6}, \{\!|s_0, s_4|\!\} \mapsto \frac{2}{6}\}$ and $\mu' = \{\{\!|s_1, s_2|\!\} \mapsto \frac{1}{2}, \{\!|s_0, s_2|\!\} \mapsto \frac{1}{8}, \{\!|s_0, s_4|\!\} \mapsto \frac{3}{8}\}$. Additionally, labeled transitions with internal actions τ_μ ($\tau_{\mu'}$ resp.) leading from $\{\!|s_0, s_2|\!\}$ to the new intermediate states are introduced.

For closed AIMC $\mathcal{M} = (S, A, \mathbf{L}, \mathbf{P}^l, \mathbf{P}^u, \lambda, s_0)$, we define the set of paths starting in initial state s_0 and visiting a state in $B \subseteq S$ within $t \in \mathbb{R}_{\geq 0}$ time units by $\mathit{Reach}_{\leq t}(B) = \{\sigma \in \mathit{Path}_{\mathcal{M}} \mid \sigma[0] = s_0, \exists t' \in [0,t] : \sigma@t' \in B\}$.

Lemma 6.6.2. *Let $\mathcal{M} = (S, A, \mathbf{L}, \mathbf{P}^l, \mathbf{P}^u, \lambda, s_0)$ be a closed AIMC and $\theta(\mathcal{M})$ its induced IMC. For all $B \subseteq S$, $t \in \mathbb{R}_{\geq 0}$ and $D \in \mathcal{E}(\mathcal{M})$ there exists $D' \in \mathcal{D}(\theta(\mathcal{M}))$ with $\Pr_D(\mathit{Reach}_{\leq t}(B)) = \Pr_{D'}(\mathit{Reach}_{\leq t}(B))$.*

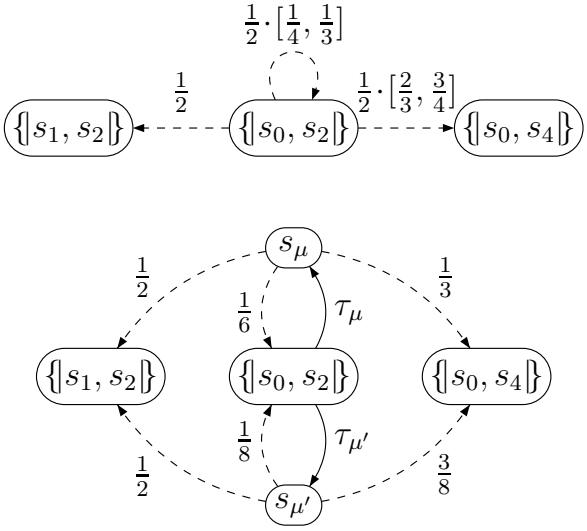


Figure 6.10: Fragment of the parallel composition $\mathcal{M}||_\varnothing\mathcal{M}$ for the AIMC $\mathcal{M}$ from Figure 6.4 (top) and the induced IMC detail (bottom).

Proof. The transformation operator θ does nothing more but separating the nondeterministic choice for values from the intervals in a Markovian state from the actual Markovian behavior. As this obviously preserves reachability probabilities, we only give a sketch of the idea instead of a formal proof:

The claim can be shown by constructing a scheduler $D' \in \mathcal{D}^{\theta(\mathcal{M})}$ that makes exactly the same choices as a given scheduler $D \in \mathcal{D}^{\mathcal{M}}$. As by definition, there is no nondeterminism in the *new* states s_μ, the choice of D' in such states is naturally given. In all other states, that is for the respective histories, the choice of D' simply has to be matched with the choice of D w.r.t. the history modulo *new* states. For example, for history $s\, s_\mu\, s'\, s_{\mu'}\, s''$ the scheduler D' has to make the same choice as D for history $s\, s'\, s''$. If an interactive transition is chosen by D, the corresponding transition is chosen by D'. If D decides for Markovian transitions, that is for a distribution μ over successor states, the scheduler D' has to mimic that decision by taking the corresponding τ_μ transition.

It then can be shown that the sets $Reach_{\leq t}(B)$ w.r.t. the two schedulers D and D' have the same probability measure. □

Corollary 6.6.3. *For a closed AIMC* $\mathcal{M} = (S, A, \mathbf{L}, \mathbf{P}^l, \mathbf{P}^u, \lambda, s_0)$, $B \subseteq S$, $t \in \mathbb{R}_{\geq 0}$*:*

$$\sup_{D \in \mathcal{D}^{\mathcal{M}}} \Pr_{s_0,D}(Reach_{\leq t}(B)) = \sup_{D \in \mathcal{D}(\theta(\mathcal{M}))} \Pr_{s_0,D}(Reach_{\leq t}(B))$$
$$\inf_{D \in \mathcal{D}^{\mathcal{M}}} \Pr_{s_0,D}(Reach_{\leq t}(B)) = \inf_{D \in \mathcal{D}(\theta(\mathcal{M}))} \Pr_{s_0,D}(Reach_{\leq t}(B))$$

Proof. Follows directly from Lemma 6.6.2. □

The analysis of time-bounded reachability probabilities for uniform IMCs is investigated in [Joh07] and the core algorithm [BHKH05] is implemented in MRMC. Basically, a uniform IMC is reduced to a uniform CTMDP by transformations to so-called *Markov alternating* and *strictly alternating* IMCs. This transformation preserves (weak) bisimulation. The following example relies on this results:

Example 6.6.2. Assume the number of machines that are available for crafting value and premium products is limited to two. First, we investigate the probabilities for b out of w workers $\mathcal{M}_1$ to $\mathcal{M}_w$ to be waiting for machines within t time units. Let $\mathcal{P} = (\{m_0, m_1, m_2\}, A, \mathbf{L}, \mathbf{1}, \mathbf{1}, \varepsilon, m_0)$ where in m_i there are i machines in use and let $A = \{\text{value}, \text{prem}, \text{vdone}, \text{pdone}\}$, $\mathbf{L}(m_i, a, m_{i+1}) = \top$ if $a \in \{\text{value}, \text{prem}\}$ for $i \in \{0, 1\}$ and $\mathbf{L}(m_{i+1}, a, m_i) = \top$

	b=1	b=1	b=2				
max. size	w=3	w=4	w=4	w=1	w=2	w=3	w=4
IMC, par.	512	4096	4096	352	2816	22528	180224
IMC, sym.	120	330	330	352	1584	5280	14520
AIMC, par.	216	1296	1296	264	1584	9504	57024
AIMC, sym.	56	126	126	264	924	2464	5544

Table 6.3: Maximal size of the state spaces during construction.

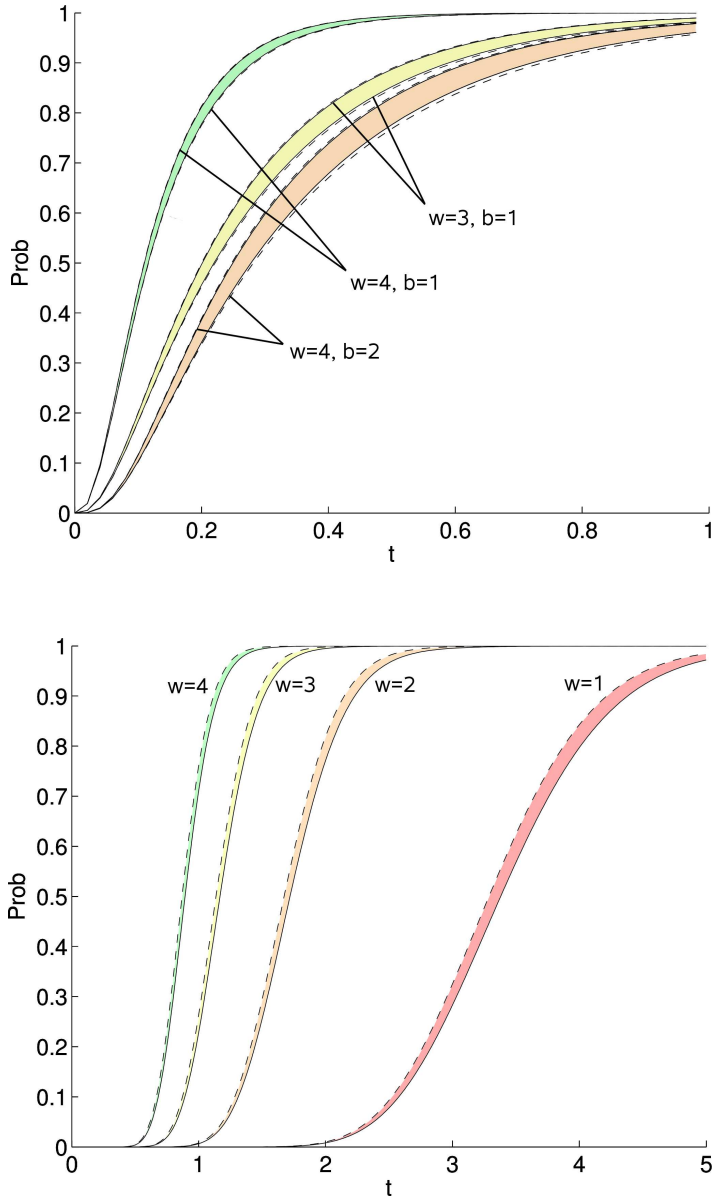


Figure 6.11: Minimal and maximal probabilities for b out of w workers having no access to one of 2 machines in t time units (top). Maximal probabilities for w workers and 2 machines to produce 10 value and 3 premium in t time units (bottom). Curves for concrete workers are solid and dashed for abstract ones.

if $a \in \{\text{vdone}, \text{pdone}\}$ for $i \in \{0, 1\}$, otherwise $\mathbf{L}(m, a, m') = \bot$. Let $\mathcal{M}_i$ be pairwise distinct variants of workers as described in Example 6.5.1. Then, $(\mathcal{M}_1||_{\varnothing} \ldots ||_{\varnothing}\mathcal{M}_w)||_A\mathcal{P}$ yields an IMC where the measure of interest can be derived by computing probabilities for reaching states $(\bar{s}, m_2)$ with at least b components of $\bar{s}$ being s_1 or s_1'. In contrast, when $\mathcal{M}_1 = \ldots = \mathcal{M}_w = \mathcal{M}$ we can instead compute the probabilities in $\left(|||^w_{\varnothing}\mathcal{M}\right)||_A\mathcal{P}$ for reaching states (M, m_2) with $M(s_1) + M(s_1') \geq b$. The maximal sizes of the state spaces obtained during the construction of the models are given in Table 6.3 (left). Let AIMC $\mathcal{N} = \alpha(\mathcal{M}_1) = \ldots = \alpha(\mathcal{M}_w)$ as described in Example 6.5.1. Then, even for pairwise distinct workers, symmetric composition can be used to obtain the abstract system $(|||^w_{\varnothing}\mathcal{N})||_A\mathcal{P}$. While the abstract model of one worker has 6 instead of 8 states, the relative savings during composition are much larger (cf. Table 6.3). But still, the minimal and maximal probabilities (Figure 6.11, top) obtained for w instances of the abstract worker $\mathcal{N}$ (dashed curves) are almost the same as for w copies of the concrete worker $\mathcal{M}$ as shown in Figure 6.4 (top, solid curves).

Secondly, we compute the maximal probabilities for producing 10 value and 3 premium products with w workers within t time units. Note, that minimal probabilities are 0 for all time bounds t, as workers may stall premium production. We define counting AIMC

$$\mathcal{Q} = (\{n_{v,p} \mid v \in \{0, \ldots, 10\}, p \in \{0, \ldots, 3\}\}, A, \mathbf{L}, \mathbf{1}, \mathbf{1}, \varepsilon, n_{0,0})$$

with $A = \{\text{vdone}, \text{pdone}\}$, $\mathbf{L}(n_{v,p}, \text{vdone}, n_{v+1,p}) = \top$ for $v \in \{0, \ldots, 9\}$, $p \in \{0, \ldots, 3\}$ and $\mathbf{L}(n_{v,p}, \text{pdone}, n_{v,p+1}) = \top$ for $v \in \{0, \ldots, 10\}, p \in \{0, \ldots, 2\}$, otherwise $\mathbf{L}(n, a, n') = \bot$. Let concrete and abstract workers $\mathcal{M}$ and $\mathcal{N}$ be given as in Figure 6.4. Then, in $\left(|||^w_{\varnothing}\mathcal{M}\right)||_A\mathcal{Q}$ and $\left(|||^w_{\varnothing}\mathcal{N}\right)||_A\mathcal{Q}$, we compute probabilities to reach any state $(M, n_{10,3})$. As shown in Figure 6.11 (bottom), the maximal probabilities for $w \in \{1, \ldots, 4\}$ abstract workers (dashed curves) are rather close to values derived for concrete workers (solid curves). The maximal sizes of the state spaces during construction are given in Table 6.3 (right).

6.7 Summary and discussion

In this chapter, a compositional abstraction technique for interactive Markov chains has been proposed. It is suggested to perform abstraction on component level to reduce the peak size of the state space while constructing the monolithic version of a compositional model – which has to be done when analyzing the quantitative behavior of the system. Abstract components have been shown to simulate their concrete counterparts and as a main result of the chapter, we showed that simulation is a precongruence w.r.t. composition. From a practical point of view, introducing symmetries by performing abstraction has been shown to be most effective as symmetric composition instead of parallel composition can be applied, resulting in bisimilar models with state spaces whose size can very well differ by several orders of magnitude in favor of symmetric composition. We also showed that bisimulation is a congruence w.r.t. composition for abstract interactive Markov chains.

The analysis of time-bounded reachability probabilities has been discussed and successfully performed on an example. A more systematic treatment – defining weak bisimulation for abstract interactive Markov chains, showing weak bisimilarity between AIMCs and their induced IMCs, as well as providing general preservation results for (weak) bisimulation and simulation – is left for future work. Furthermore, an adaptation of the polynomial time algorithm from Section 4.3 for the analysis of AIMCs is not considered here, however, we do not expect major difficulties there.

Related work. While a number of publications on compositional verification and abstraction are available for classical models [CLM89, HQR98, SG07] and non-probabilistic extensions thereof [BV08], only few results have been reported on probabilistic models and – to our knowledge – non for stochastic systems:

The earliest related work on probabilistic systems we are aware of is the one by Segala and Lynch on simulation preorders for (discrete-time) proba-

bilistic automata [SL95]. Several variants of weak and strong simulations are investigated regarding compositionality and preservation of properties.

Recently, compositional abstraction has been used for language-level abstraction [KKNP08]. The approach taken in this work is to locally peek into the composed concrete model to obtain information for the abstract version of the compositional model (so-called controlled MDPs).

Further, most recent work [CDL+09] is dealing also with logical composition operators such as conjunction. Rather than considering abstraction, the focus is on the compositional specification of properties. Technically speaking, by conjunction of several given CMC (Constraint Markov Chain) specifications, a composed specification is obtained that only contains the common parts of all given subspecifications. While formal properties of the defined operators have been discussed in detail, the authors left the study of efficient algorithms to future work. A further interesting topic for future work might be to bring this approach and our work on interactive Markov chains together.

7 Conclusion

Model checking probabilistic and stochastic systems is a complex and memory intensive task. Often, models with tremendously large to infinite state spaces are obtained from rather small high-level models such as guarded command languages or stochastic Petri nets. This thesis provides a remedy for the state space explosion problem for probabilistic and stochastic systems: three-valued abstraction. We focused on abstractions allowing to check step-bounded and time-bounded until formulas as these are the most intricate ones.

We started investigating abstraction for fully probabilistic models with infinite state space; we discussed how sets of distributions and intervals of probabilities can be utilized for modeling abstract behavior, further we elaborated on step- and time-bounded reachability and discussed unbounded reachability briefly. The presented techniques have been shown to yield abstract models simulating the original ones in a stepwise manner; as a result, safe lower and upper bounds for any actual probability measure can be calculated. Efficient algorithms have been developed for the computation of reachability probabilities and PCTL/CSL model checking. Furthermore, we showed for a case study from queueing theory that quite precise results can be obtained using abstraction when traditional methods do fail.

Subsequently, we isolated a downside of the abstractions presented in Section 3, namely that techniques based on stepwise simulation yield coarse abstractions for a significant class of models. We developed a generalization of the interval abstraction method where several consecutive transitions are

merged into one abstract transition; it turned out that substantially tighter intervals can be obtained by doing so. The technique was successfully applied to a case study from systems biology known as *enzyme catalyzed substrate conversion*; the fully featured model checker PRISM took more than a day to compute exact results whereas our MATLAB prototype took less than half an hour to compute fairly accurate bounds.

Finally, we lifted interval abstraction to concurrent stochastic systems. The compositional modelling framework of interactive Markov chains (IMCs) has been equipped with facilities for abstraction: where IMCs extend labeled transition systems and continuous-time Markov chains (CTMCs), the abstract version of IMCs extend modal transition systems and abstract CTMCs as introduced earlier in the thesis. We propose to perform abstraction at the component level instead of system level to reduce the size of the state space during the construction of a monolithic system model when analyzing such systems. In cases where the abstraction of components introduces symmetries that are not present in the concrete model, this symmetry can be exploited allowing for radical reductions with respect to the peak size of the state space during construction of a monolithic model.

Outlook. In this thesis, we provide some fundamental work on abstraction and model checking for stochastic systems and we also show the feasibility of our approach. Yet, this area is comparatively uncharted and additional effort is required to unfold its potential. Some of the still open questions closely related to this thesis are "How to automate three-valued abstraction refinement for stochastic systems?", " When and how can uniformization be applied to interactive Markov chains?" and "How do these techniques perform in industrial case studies?". Even more pointers to topics that are worth investigating can be found in the summary section of each chapter.

Obviously, while reaching the end of this thesis, we are by no means completing the research on this field. For model checking unbounded until

formulas there have already existed some advanced abstraction techniques for some time [KNP06, dAP07] that are still investigated and extended upon. When model checking step- and especially time-bounded until as done in this thesis, these techniques are often not (or not yet) applicable. However, recently, problems related to the latter are increasingly gaining attention [HMW09, DHMW09, Neu10]; a trend that, due to the importance to performance analysis, bioinformatics and other research areas, will most likely continue in coming years.

Bibliography

[ADD00] Robert B. Ash and Cathrine A. Doléans-Dade. *Probabilty & Measure Theory, 2nd Edition*. Academic Press, 2000.

[ASSB00] Adnan Aziz, Kumud Sanwal, Vigyan Singhal, and Robert Brayton. Model-checking continuous-time Markov chains. *ACM Trans. Comput. Logic*, 1(1):162–170, 2000.

[BB01] James M. Bower and Hamid Bolouri. *Computational Modeling of Genetic and Biochemical Networks*. The MIT Press, 2001.

[BdA95] Andrea Bianco and Luca de Alfaro. Model checking of probabalistic and nondeterministic systems. In *Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, LNCS, pages 499–513. Springer, 1995.

[BDHK06] Henrik Bohnenkamp, Pedro R. D'Argenio, Holger Hermanns, and Joost-Pieter Katoen. MODEST: A compositional modeling formalism for hard and softly timed systems. *Software Engineering, IEEE Transactions on*, 32(10):812–830, 2006.

[Bel57] Richard E. Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[BEMC00] Christel Baier, Bettina Engelen, and Mila E. Majster-Cederbaum. Deciding bisimilarity and similarity for probabilistic processes. *Journal of Computer and System Sciences*, 60(1):187–231, 2000.

[BHH+06] Eckard Böde, Marc Herbstritt, Holger Hermanns, Sven Johr, Thomas Peikenkamp, Reza Pulungan, Ralf Wimmer, and Bernd Becker. Compositional performability evaluation for STATEMATE. In *International Conference on Quantitative Evaluation of Systems (QEST)*, pages 167–178. IEEE CS, 2006.

[BHHK03] Christel Baier, Boudewijn R. Haverkort, Holger Hermanns, and Joost-Pieter Katoen. Model-checking algorithms for continuous-time Markov chains. *IEEE TSE*, 29(6):524–541, 2003.

[BHKH05] Christel Baier, Holger Hermanns, Joost-Pieter Katoen, and Boudewijn R. Haverkort. Efficient computation of time-bounded reachability probabilities in uniform continuous-time Markov decision processes. *Theoretical Computer Science*, 345(1):2–26, 2005.

[BK08] Christel Baier and Joost-Pieter Katoen. *Principles of Model Checking*. The MIT Press, 2008.

[BKHW05] Christel Baier, Joost-Pieter Katoen, Holger Hermanns, and Verena Wolf. Comparative branching-time semantics for Markov chains. *Information and Computation*, 200(2):149–214, 2005.

[BKS05] Tomáš Brázdil, Antonín Kucera, and Oldrich Strazovský. On the decidability of temporal properties of probabilistic pushdown automata. In *International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 3404 of *LNCS*, pages 145–157. Springer, 2005.

[BLM03] Dario A. Bini, Guy Latouche, and Beatrice Meini. Solving nonlinear matrix equations arising in tree-like stochastic processes. *Lin. Alg. and its Applications*, 366:39–64, 2003.

[BSW06] Hauke Busch, Werner Sandmann, and Verena Wolf. A numerical aggregation algorithm for the enzyme-catalyzed substrate conversion. In *Conference on Computational Methods in Systems Biology (CMSB)*, volume 4210 of *LNCS*, pages 298–311. Springer, 2006.

[Buc94] Peter Buchholz. Exact and ordinary lumpability in finite Markov chains. *Journal of Applied Probability*, 31:59–75, 1994.

[BV08] Jasper Berendsen and Frits W. Vaandrager. Compositional abstraction in real-time model checking. In *International Conference on Formal Modelling and Analysis of Timed Systems (FORMATS)*, volume 5215, pages 233–249, 2008.

[CCG+02] Alessandro Cimatti, Edmund M. Clarke, Enrico Giunchiglia, Fausto Giunchiglia, Marco Pistore, Marco Roveri, Roberto Sebastiani, and Armando Tacchella. NuSMV Version 2: An OpenSource Tool for Symbolic Model Checking. In *International Conference on Computer-Aided Verification (CAV)*, volume 2404 of *LNCS*, pages 359–364. Springer, 2002.

[CDL+09] Benoît Caillaud, Benoît Delahaye, Kim G. Larsen, Axel Legay, Mikkel L. Pedersen, and Andrzej Wasowski. Compositional design methodology with constraint Markov chains. Technical report, INRIA, 2009.

[CES83] Edmund M. Clarke, E. Allen Emerson, and A. Prasad Sistla. Automatic verification of finite state concurrent systems using temporal logic specifications: A practical approach. In *Symposium on Principles of Programming Languages (POPL)*, pages 117–126. ACM Press, 1983.

[CGJ+00] Edmund M. Clarke, Orna Grumberg, Somesh Jha, Yuan Lu, and Helmut Veith. Counterexample-guided abstraction refinement. In *International Conference on Computer-Aided Verification (CAV)*, volume 1855, pages 154–169, 2000.

[CGL94] Edmund M. Clarke, Orna Grumberg, and David E. Long. Model checking and abstraction. *ACM Trans. Program. Lang. Syst.*, 16(5):1512–1542, 1994.

[Cia95] Gianfranco Ciardo. Discrete-time Markovian stochastic Petri nets. In *Computations with Markov Chains*, pages 339–358. Raleigh, 1995.

[CKSY05] Edmund M. Clarke, Daniel Kroening, Natasha Sharygina, and Karen Yorav. SATABS: SAT-based predicate abstraction for ANSI-C. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 3440, pages 570–574, 2005.

[CLM89] Edmund M. Clarke, David E. Long, and Kenneth L. McMillan. Compositional model checking. In *IEEE Symposium on Logic in Computer Science (LICS)*, pages 353–362, 1989.

[CVV08] Rohit Chadha, Mahesh Viswanathan, and Ramesh Viswanathan. Least upper bounds for probability measures and their applications to abstractions. In *International Conference on Concurrency Theory (CONCUR)*, volume 5201, pages 264–278, 2008.

[dA97] Luca de Alfaro. *Formal verification of probabilistic systems.* PhD thesis, Stanford University, Stanford, CA, USA, 1997.

[dAP07] Luca de Alfaro and Roy Pritam. Magnifying-lens abstraction for Markov decision processes. In *International Conference on Computer-Aided Verification (CAV)*, volume 4590 of *LNCS*, pages 325–338. Springer, 2007.

[DHMW09] Frederic Didier, Thomas A. Henzinger, Maria Mateescu, and Verena Wolf. Fast adaptive uniformization of the chemical master equation. *International Workshop on High Performance Computational Systems Biology (HiBi)*, pages 118–127, 2009.

[Dij75] Edsger W. Dijkstra. Guarded commands, non-determinancy and a calculus for the derivation of programs. In *Language Hierarchies and Interfaces*, pages 111–124, 1975.

[DJJL01] Pedro R. D'Argenio, Bertrand Jeannet, Henrik E. Jensen, and Kim G. Larsen. Reachability analysis of probabilistic systems by successive refinements. In *Joint International Workshop on Process Algebra and Performance Modelling and Probabilistic Methods in Verification (PAPM-PROBMIV)*, volume 2165 of *LNCS*, pages 39–56. Springer, 2001.

[EWY08] Kousha Etessami, Dominik Wojtczak, and Mihalis Yannakakis. Quasi-birth-death processes, tree-like QBDs, probabilistic 1-counter automata, and pushdown systems. In *International Conference on Quantitative Evaluation of Systems (QEST)*, pages 243–253. IEEE CS Press, 2008.

[Fel68] William Feller. *An Introduction to Probability Theory and its Applications, Vol. I.* John Wiley & Sons, Inc., 1968.

[FG88] Bennett L. Fox and Peter W. Glynn. Computing Poisson probabilities. *Commun. ACM*, 31(4):440–445, 1988.

[FLW06] Harald Fecher, Martin Leucker, and Verena Wolf. Don't know in probabilistic systems. In *Model Checking Software*, volume 3925 of *LNCS*, pages 71–88. Springer, 2006.

[Fok00] Wan Fokkink. *Introduction to Process Algebra.* Texts in Theoretical Computer Science. Springer, 2000.

[GB06] Marcus Größer and Christel Baier. Partial order reduction for Markov decision processes: a survey. In *International Symposia on Formal Methods for Components and Objects (FMCO)*, volume 4111 of *LNCS*, pages 408–427. Springer, 2006.

[GH94] Stephen Gilmore and Jane Hillston. The PEPA workbench: A tool to support a process algebra-based approach to performance modelling. In *Computer Performance Evaluation*, volume 794 of *LNCS*, pages 353–368. Springer, 1994.

[GM84] Donald Gross and Douglas R. Miller. The randomization technique as a modeling tool and solution procedure for transient Markov chains. *Operations Research*, 32(2):343–361, 1984.

[Gra91] Winfried K. Grassmann. Finding transient solutions in Markovian event systems through randomization. In *Numerical Solution of Markov Chains*, pages 411–420. Dekker, 1991.

[HA00] Qi-Ming He and Attahiru S. Alfa. The discrete time MMAP[K]/PH[K]/1/LCFS-GPR queue and its variants. In *International Conference on Matrix Analytic Methods*, pages 167–190, 2000.

[Har87] David Harel. Statecharts: A visual formalism for complex systems. *Sci. Comput. Program.*, 8(3):231–274, 1987.

[Hav98] Boudewijn R. Haverkort. *Performance of Computer Communication Systems: A Model-Based Approach.* John Wiley & Sons, Inc., New York, NY, USA, 1998.

[Her90] Ted Herman. Probabilistic self-stabilization. *Information Processing Letters*, 35(2):63–67, 1990.

[Her02] Holger Hermanns. *Interactive Markov Chains and the Quest for Quantified Quality*, volume 2428 of *LNCS.* Springer, Berlin, 2002.

[HHK02] Holger Hermanns, Ulrich Herzog, and Joost-Pieter Katoen. Process algebra for performance evaluation. *Theoretical Computer Science*, 274(1-2):43–87, 2002.

[HHM98] Holger Hermanns, Ulrich Herzog, and Vassilis Mertsiotakis. Stochastic process algebras - between LOTOS and Markov chains. *Computer Networks*, 30(9-10):901–924, 1998.

[HJ94] Hans Hansson and Bengt Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6(5):512–535, 1994.

[HJ07] Holger Hermanns and Sven Johr. Uniformity by construction in the analysis of nondeterministic stochastic systems. *Dependable Systems and Networks*, pages 718–728, 2007.

[HJS01] Michael Huth, Radha Jagadeesan, and David Schmidt. Modal transition systems: A foundation for three-valued program analysis. In *European Symposium on Programming (ESOP)*, volume 2028 of *LNCS*, pages 155–169. Springer, 2001.

[HK00] Holger Hermanns and Joost-Pieter Katoen. Automated compositional Markov chain generation for a plain-old telephone system. *Sci. Comput. Program.*, 36(1):97–127, 2000.

[HMW09] Thomas A. Henzinger, Maria Mateescu, and Verena Wolf. Sliding window abstraction for infinite Markov chains. In *International Conference on Computer-Aided Verification (CAV)*, pages 337–352, 2009.

[HPW09] Michael Huth, Nir Piterman, and Daniel Wagner. Three-valued abstractions of Markov chains: Completeness for a sizeable fragment of PCTL. In *International Symposium on Fundamentals of Computation Theory (FCT)*, pages 205–216, 2009.

[HQR98] Thomas A. Henzinger, Shaz Qadeer, and Sriram K. Rajamani. You assume, we guarantee: Methodology and case studies.

In *International Conference on Computer-Aided Verification (CAV)*, volume 5699, pages 440–451, 1998.

[HR98] Holger Hermanns and Marina Ribaudo. Exploiting symmetries in stochastic process algebras. In *European Simulation Multiconference*, pages 763–770. SCS Europe, 1998.

[HRTT04] Gábor Horváth, Sándor Rácz, Árpád Tari, and Miklós Telek. Evaluation of reward analysis methods with MRMSolve 2.0. In *International Conference on Quantitative Evaluation of Systems (QEST)*, pages 165–174, 2004.

[Hut05] Michael Huth. On finite-state approximants for probabilistic computation tree logic. *Theoretical Computer Science*, 346(1):113–134, 2005.

[HWZ08] Holger Hermanns, Björn Wachter, and Lijun Zhang. Probabilistic cegar. In *International Conference on Computer-Aided Verification (CAV)*, pages 162–175, 2008.

[JL91] Bengt Jonsson and Kim G. Larsen. Specification and refinement of probabilistic processes. In *IEEE Symposium on Logic in Computer Science (LICS)*, pages 266–277. IEEE Press, 1991.

[Joh07] Sven Johr. *Model Checking Compositional Markov Systems*. PhD thesis, Universität des Saarlandes, Saarbrücken, Germany, 2007.

[Jon83] Cliff B. Jones. Specification and design of (parallel) programs. In *International Federation for Information Processing (IFIP) Congress*, pages 321–332, 1983.

[Kar84] Narendra Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4):373–396, 1984.

[KH09] Mark Kattenbelt and Michael Huth. Verification and refutation of probabilistic specifications via games. In *Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, volume 4 of *LIPIcs*, pages 251–262. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2009.

[KKLW07a] Joost-Pieter Katoen, Daniel Klink, Martin Leucker, and Verena Wolf. Three-valued abstraction for continuous-time Markov chains. In *International Conference on Computer-Aided Verification (CAV)*, volume 4590 of *LNCS*, pages 316–329. Springer, 2007.

[KKLW07b] Joost-Pieter Katoen, Daniel Klink, Martin Leucker, and Verena Wolf. Three-valued abstraction for probabilistic systems. Technical Report AIB-2007-20, RWTH Aachen, December 2007.

[KKLW08] Joost-Pieter Katoen, Daniel Klink, Martin Leucker, and Verena Wolf. Abstraction for stochastic systems by Erlang's method of stages. In *International Conference on Concurrency Theory (CONCUR)*, volume 5201 of *LNCS*, pages 279–294. Springer, 2008.

[KKN09] Joost-Pieter Katoen, Daniel Klink, and Martin R. Neuhäußer. Compositional abstraction for stochastic systems. In *International Conference on Formal Modelling and Analysis of Timed Systems (FORMATS)*, volume 5813 of *LNCS*, pages 195–211. Springer, 2009.

[KKNP08] Mark Kattenbelt, Marta Z. Kwiatkowska, Gethin Norman, and David Parker. Game-based probabilistic predicate abstraction in PRISM. In *Proc. 6th Workshop on Quantitative Aspects of Programming Languages (QAPL'08)*, 2008.

[KKZJ07] Joost-Pieter Katoen, Tim Kemna, Ivan S. Zapreev, and David N. Jansen. Bisimulation minimisation mostly speeds up probabilistic model checking. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 4424, pages 87–101, 2007.

[KNP06] Marta Z. Kwiatkowska, Gethin Norman, and David Parker. Game-based abstraction for Markov decision processes. In *International Conference on Quantitative Evaluation of Systems (QEST)*, pages 157–166. IEEE CS Press, 2006.

[KNP09] Marta Kwiatkowska, Gethin Norman, and David. Parker. Prism: Probabilistic model checking for performance and reliability analysis. *ACM SIGMETRICS Performance Evaluation Review*, 36(4):40–45, 2009.

[KNPS06] Marta Kwiatkowska, Gethin Norman, David Parker, and Jeremy Sproston. Performance analysis of probabilistic timed automata using digital clocks. *Formal Methods in System Design*, 29:33–78, 2006.

[KNS03] Marta Kwiatkowska, Gethin Norman, and Jeremy Sproston. Probabilistic model checking of deadline properties in the IEEE 1394 FireWire root contention protocol. *Formal Aspects of Computing*, 14(3):295–318, 2003.

[KRHK09] Daniel Klink, Anne Remke, Boudewijn R. Haverkort, and Joost-Pieter Katoen. Time-bounded reachability in tree-structured QBDs by abstraction. In *International Conference on Quantitative Evaluation of Systems (QEST)*, pages 133–142. IEEE CS, 2009.

[KU02] Igor Kozine and Lev V. Utkin. Interval-valued finite Markov chains. *Reliable Computing*, 8(2):97–113, 2002.

[KWB03] Anneke G. Kleppe, Jos Warmer, and Wim Bast. *MDA Explained: The Model Driven Architecture: Practice and Promise*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2003.

[KZH+09] Joost-Pieter Katoen, Ivan S. Zapreev, Ernst Moritz Hahn, Holger Hermanns, and David N. Jansen. The ins and outs of the probabilistic model checker MRMC. *International Conference on Quantitative Evaluation of Systems (QEST)*, 0:167–176, 2009.

[LR99] Guy Latouche and Vaidyanathan Ramaswami. *Introduction to Matrix Analytic Methods in Stochastic Modeling*. SIAM, 1999.

[LSS94] Nancy Lynch, Isaac Saias, and Roberto Segala. Proving time bounds for randomized distributed algorithms. In *Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 314–323. ACM Press, 1994.

[LT88] Kim G. Larsen and Bent Thomsen. A modal process logic. In *IEEE Symposium on Logic in Computer Science (LICS)*, pages 203–210. IEEE Computer Society Press, 1988.

[Neu81] Marcel F. Neuts. *Matrix-geometric solutions in stochastic models: an algorithmic approach*. Johns Hopkins University Press, Baltimore, 1981.

[Neu10] Martin Neuhäußer. *Model Checking Nondeterministic and Randomly Timed Systems*. PhD thesis, RWTH Aachen University, 2010.

[NK07] Martin R. Neuhäußer and Joost-Pieter Katoen. Bisimulation and logical preservation for continuous-time Markov decision

processes. In *International Conference on Concurrency Theory (CONCUR)*, pages 412–427, 2007.

[NSK09] Martin R. Neuhäußer, Mariëlle Stoelinga, and Joost-Pieter Katoen. Delayed nondeterminism in continuous-time Markov decision processes. In *International Conference on Foundations of Software Science and Computation Structures (FOSSACS)*, pages 364–379, 2009.

[NZ09] Martin R. Neuhäußer and Lijun Zhang. Time-bounded reachability in continuous-time Markov decision processes. Technical Report AIB-2009-12, RWTH Aachen, 2009.

[Pet62] Carl Adam Petri. *Kommunikation mit Automaten.* PhD thesis, Bonn: Institut für Instrumentelle Mathematik, Schriften des IIM Nr. 2, 1962. Second Edition:, New York: Griffiss Air Force Base, Technical Report RADC-TR-65–377, Vol.1, 1966, Pages: Suppl. 1, English translation.

[PH08a] Reza Pulungan and Holger Hermanns. Effective minimization of acyclic phase-type representations. In *International Conference on Analytical and Stochastic Modeling Techniques and Applications (ASMTA)*, volume 5055 of *LNCS*, pages 128–143. Springer, 2008.

[PH08b] Reza Pulungan and Holger Hermanns. The minimal representation of the maximum of Erlang distributions. In *GI/ITG Konferenz Messung, Modellierung und Bewertung von Rechen- und Kommunikationssystemen (MMB)*, pages 207–222, 2008.

[Rab82] Michael O. Rabin. N-process mutual exclusion with bounded waiting by $4 \log_2 N$-valued shared variable. *Journal of Computer and System Sciences*, 25(1):66–75, 1982.

[RH08] Anne Remke and Boudewijn R. Haverkort. A uniformization-based algorithm for model checking the CSL until operator on labeled queueing networks. In *International Conference on Formal Modelling and Analysis of Timed Systems (FORMATS)*, volume 4763 of *LNCS*, pages 336–351. Springer, 2008.

[RHC07] Anne Remke, Boudewijn R. Haverkort, and Lucia Cloth. CSL model checking algorithms for QBDs. *Theoretical Computer Science*, 382(1):24–41, 2007.

[Seg95] Roberto Segala. *Modeling and Verification of Randomized Distributed Real-Time Systems*. PhD thesis, Massachusetts Institute of Technology, 1995.

[SG07] Sharon Shoham and Orna Grumberg. Compositional verification and 3-valued abstractions join forces. In *Static Analysis, International Symposium (SAS)*, pages 69–86, 2007.

[Sku06] Damjan Skulj. Finite DTMCs with interval probabilities. In *Soft Methods for Integrated Uncertainty Modelling*, volume 37 of *Advances in Soft Computing*, pages 299–306. Springer, 2006.

[SL95] Roberto Segala and Nancy A. Lynch. Probabilistic simulations for probabilistic processes. *Nordic Journal of Computing*, 2(2):250–273, 1995.

[SvHB06] Kathleen Spaey, Benny van Houdt, and Chris Blondia. On the generality of binary tree-like Markov chains. In *Markov Aniversary Meeting*, pages 79–88. Boston Books, 2006.

[TBT06] Axel Thümmler, Peter Buchholz, and Miklós Telek. A novel approach for phase-type fitting with the EM algorithm. *IEEE Trans. Dependable Sec. Comput.*, 3(3):245–258, 2006.

[TSY95] Tetsuya Takine, Bhaskar Sengupta, and Raymond W. Yeung. A generalization of the matrix M/G/1 paradigm for Markov chains with a tree structure. *Stochastic Models*, 11(3):411–421, 1995.

[UC04] Sebastian Uchitel and Marsha Chechik. Merging partial behavioural models. In *International symposium on Foundations of Software Engineering (FSE)*, pages 43–52. ACM, 2004.

[vHB05] Benny van Houdt and Chris Blondia. Throughput of Q-ary splitting algorithms for contention resolution in communication networks. *Commun. Inform. Sys.*, 4(2):135–164, 2005.

[vHB06] Benny van Houdt and Chris Blondia. Analyzing priority queues with 3 classes using tree-like processes. *Queueing Systems*, 54:99–109, 2006.

[vK07] Nico G. van Kampen. *Stochastic Processes in Physics and Chemistry*. Elsevier, 3rd edition, 2007.

[vVvHB06] Jeroen van Velthoven, Benny van Houdt, and Chris Blondia. Transient analysis of tree-like processes and its application to random access systems. *SIGMETRICS/Performance*, 34(1):181–190, 2006.

[Wei01] K. Weichselberger. *Elementare Grundbegriffe einer allgemeineren Wahrscheinlichkeitsrechnung I.* Physica-Verlag Heidelberg, 2001.

[WJ06] Nicolás Wolovick and Sven Johr. A characterization of meaningful schedulers for continuous-time Markov decision processes. In *International Conference on Formal Modelling and Analysis of Timed Systems (FORMATS)*, volume 4202, pages 352–367, 2006.

[YA99] Raymond W. Yeung and Attahiru S. Alfa. The quasi-birth-death type Markov chain with a tree structure. *Stochastic Models*, 15(4):639–659, 1999.

[ZHHW08] Lijun Zhang, Holger Hermanns, E. Moritz Hahn, and Björn Wachter. Time-bounded model checking of infinite-state continuous-time Markov chains. In *ACSD*, pages 98–107. IEEE CS Press, June 2008.

Lebenslauf

Name:	Daniel Klink, geb. Willems
Geburtsdatum:	09.03.1980
Geburtsort:	Daun

Bildungsgang

1990–1999	Thomas Morus Gymnasium Daun Abschluß: Abitur
1999–2000	Ersatzdienst
2000–2006	Studium der Informatik an der RWTH Aachen University Abschluß: Diplom
2006–2009	Stipendiat im DFG Graduiertenkolleg AlgoSyn
2009–2010	Wissenschaftlicher Angestellter am Lehrstuhl für Informatik 2 (Prof. Dr. Ir. Joost-Pieter Katoen), RWTH Aachen University