Project no.: **ICT-FP7-STREP-214755**

Project full title: **Quantitative System Properties in Model-Driven Design**

Project Acronym: **QUASIMODO**

Deliverable no.: D2.3

Title of Deliverable: Abstraction

Contractual Date of Delivery to the CEC:	**Month 24**
Actual Date of Delivery to the CEC:	**Month 24 (01. February 2010)**
Organisation name of lead contractor for this deliverable:	**P04 RWTH Aachen**
Author(s):	**Henrik Bohnenkamp, Ernst Moritz Hahn, Joost-Pieter Katoen, Daniel Klink, Kim Larsen, Thomas Noll, Jean-Froncois Raskin, Mariëlle Stoelinga, Frits Vaandrager, Björn Wachter**
Participants(s):	**P01, P02, P04, P05, P06**
Work package contributing to the deliverable:	**WP2**
Nature:	**R**
Version:	**1.1**
Total number of pages:	**27**
Start date of project:	1 Jan. 2008 **Duration:** 36 month

Project co-funded by the European Commission within the Seventh Framework Programme (2007-2013)
Dissemination Level

PU Public

Abstract:

This deliverable reports on the results in the area of *abstraction of quantitive system*, produced in the QUASIMODO project.

Keyword list: Abstraction

Contents

Abbreviations

AAU: Aalborg University, DK (P01)

ESI: Embedded Systems Institute, NL (P02)

ESI/RU: Radboud University Nijmegen, NL (under the auspices of P02)

ESI/UT: University Twente, Enschede , NL (under the auspices of P02)

CNRS: National Center for Scientific Research, FR (P03)

RWTH: RWTH Aachen University, D (P04)

SU: Saarland University, D (P05)

CFV: Centre Fèdèrè en Vèrification, B (P06)

CHESS: CHESS (P08)

HYDAC: Hydac (P10)

1 Introduction

In the modelling of systems, abstraction is inherently of great importance. The act of modelling is itself already an act of abstraction, in the sense that only relevant information of the modelled system finds entry into the model. Frequently, abstraction is an indispensable means to make the analysis of systems feasible, usually because the state-space of the system model to be analysed is either too large, or even infinite. On the other hand, abstraction, the word which basically is synonymous with "*throwing information away*", might introduce imprecision in the results obtainable from the analysis of the abstract system: the obtained analysis result might be inconclusive or plain wrong.

Research on abstraction addressed several issues, of which a few, relevant for this deliverable, are introduced in the following.

How can abstractions be described formally? Seminal work on abstraction has been done in [36], where the concept of *abstract interpretation* is introduced. Various abstraction approaches for a variety of model classes can be described using abstract interpretation. An application is for example [37], where abstraction in the context of reactive systems is defined using abstract interpretation. It is shown there that safety as well as liveness properties in the modal μ-calculus that hold in the abstract model, hold also in the concrete model. However, there might be properties that do not hold in the abstract model, and neither do their negations. This is an example of the imprecision introduced by abstraction, mentioned above.

How can abstract and concrete models be related? Usually the abstract model should be formally related to the concrete model. This question fits in the area of semantics, i.e. equivalences and preorders/precongruences. The semantics of a process and the comparison of process behaviour is directly related, and the amount of information used to compare behaviour is a (relative) measure of the abstractness of the semantics. This has been thoroughly investigated in [73, 72].

Usually, abstract and concrete models should be related by some refinement relation as a correctness criterion of the abstraction, which ensures that (some) properties satisfied for abstract model are still satisfied on the concrete level. The notion of refinement has an influence on which properties are still satisfied: does the satisfaction of properties in both the abstract and concrete model coincide (like for bisimulation), or is it the case that the satisfaction of safety properties on the abstract model implies their validity on the concrete model? Investigations into the field of comparative semantics thus has a direct relation to abstraction.

How to refine too coarse abstractions? Since abstraction means removing information, there are different levels of abstractness, from coarse (much information removed) to fine (little information removed). Coarse abstractions might be small, and analysis might be efficient, and a good strategy is to start with a coarse abstraction. However, the abstract model might be too coarse to be useful. In that case the abstraction should be refined, i.e. relevant information thrown away

previously should be re-introduced [35]. A fashionable approach to do this automatically is CEGAR: *Counterexample-Guided Abstraction Refinement* [34, 50]. The idea is to generate an initial abstraction and use spurious counterexamples obtained through model-checking (i.e. counterexamples of properties that are violated in the abstract model due to a too coarse abstraction) to guide the refinement of the abstract model.

How can abstractions be obtained efficiently? Obtaining abstractions for large systems can be a difficult task. When using a compositional modelling formalism like process algebras or networks of processes, exploiting the compositional structure of the model for component-wise abstraction may yield some insight into how an appropriate abstraction can be obtained. Other approaches exploit the kind of property that one wants to check so as to guide the abstraction. This applies to e.g., predicate abstraction. It is also possible to combine these approaches and apply compositional abstraction that is tailored to the properties of interest.

A more extensive treatment of abstraction in a qualitative setting is given by Grumberg [42]. Within the context of the Quasimodo project, abstraction of *quantitative systems* such as probabilistic and timed systems is a topic of intensive research. The aforementioned issues are highly relevant in this setting as well.

Overview

The Quasimodo contributions on abstraction are ordered roughly in 5 categories. Section 2 is about compositional abstraction; Section 3 is on refinement relations and equivalences; Section 4 covers approaches based on abstract interpretation; Section 5 is on the abstraction of infinite systems. Finally, Section 6 is on abstraction by state-aggregation.

The work described in [50] on probabilistic CEGAR is a Quasimodo contribution on abstraction, but left out here, since it is already described in Deliverable 2.1.

2 Compositional Abstraction

2.1 Compositional Abstraction for Stochastic Systems

Participants

- Daniel Klink, Martin Neuhäußer, Joost-Pieter Katoen (RWTH)
- Anne Remke, Boudewijn Haverkort (ESI/UT)
- Verena Wolf (SU)
- Martin Leucker, TU Munich, Germany (EXT)

Challenge

To overcome the absence of hierarchical, compositional facilities in performance modelling, several efforts have been undertaken to integrate performance aspects, most notably probability distributions, into compositional modelling formalisms. Resulting formalisms are, among others, extensions of the Petri box calculus [65], Statecharts [30], and process algebras [51, 47]. To bridge the gap towards classical performance and dependability analysis, compositional formalisms for continuous-time Markov chains (CTMCs) have received quite some attention. Nowadays, these formalisms are also used intensively in, e.g., the area of systems biology [32].

An elegant and prominent semantic model in this context are interactive Markov chains (IMCs) [46, 48]. They extend CTMCs with nondeterminism, or viewed differently, enrich labelled transition systems with exponential sojourn times in a fully orthogonal and simple manner. They naturally support the specification of phase-type distributions, i.e., sojourn times that are non-exponential, and facilitate the compositional integration of random timing constraints in purely functional models [48]. In addition, bisimulation minimisation can be done in a compositional fashion reducing the peak memory consumption during minimisation. While this has been applied to several examples yielding substantial state-space reductions, and allowing the analysis of CTMCs that could not be analysed without compositional minimisation [48, 40, 41], with increasingly complex systems under consideration, more radical reduction techniques are needed.

Results

In [23] we propose a framework to perform aggressive abstraction of IMCs in a compositional manner. Our abstraction technique is a natural mixture of abstraction of labelled transition systems by modal transition systems [61, 64] and abstraction of probabilities by intervals [39, 57] which we recently applied in the area of queueing theory [24] and adapted for the analysis of a well-known but hard-to-solve case study in systems biology: enzyme-catalysed substrate conversion [22].

Abstraction is shown to preserve simulation, that is to say, abstract models simulate concrete ones. Here, simulation is a simple combination of refinement of modal transition systems [64] and probabilistic simulation [55]. By abstraction lower bounds for minimal and upper bounds for maximal timed reachability probabilities are obtained.

Compositional aggregation is facilitated by the fact that simulation is a precongruence with respect to TCSP-like parallel composition and symmetric composition [49] on our abstract model. Accordingly, components can be abstracted prior to composing them. As this abstraction is coarser than bisimulation, a significantly larger state-space reduction may be achieved and peak memory consumption is reduced. This becomes even more advantageous when components that differ only marginally are abstracted by the same abstract model. In this case, the symmetric composition of these abstract components may yield huge reductions compared to the parallel composition of the slightly differing concrete ones.

Perspective

Future work includes the application of this technique to realistic applications, counterexample-guided abstraction refinement [50, 59], and the treatment of non-uniform IMCs.

2.2 Compositional Abstraction of Timed Automata

Participants Jasper Berendsen and Frits Vaandrager (ESI/RU)

Challenge When researchers apply model checking technology to analyse communication protocols, they typically construct models that already abstract drastically from the official standards in which these protocols are described. Otherwise, the models would become intractable due to state-space explosion problems. As a consequence, the relationship between the protocol standard and the model becomes problematic. Of course, when model checking reveals an error in the model then often this can be traced back to an error in the protocol standard. But often it is not clear whether quantitative properties of models will also hold for protocol implementations. And although for academics it is challenging to search for subtle bugs in abstract models (that only manifest themselves after thousands of transitions), experience shows that in practice most of the ambiguities and flaws in protocol standards can be found by construction and inspection of detailed, concrete models. An important research challenge therefore is to device abstraction techniques that allow one to link detailed models of protocols that are close to the standard to abstract models that are amenable to formal analysis.

Results Within Quasimodo, we finalised two articles, that have been accepted for publication in ACM TECS and JAL [3, 4], that improve and clarify our earlier results on the analysis of the Zeroconf protocol.

Within this work, the model checker Uppaal is used to formally model and analyse parts of Zeroconf, a protocol for dynamic configuration of IPv4 link-local addresses that has been defined in RFC 3927 of the IETF [33]. Our goal has been to construct a model that (a) is easy to understand by engineers, (b) comes as close as possible to the informal text (for each transition in the model there should be a corresponding piece of text in the RFC), and (c) may serve as a basis for formal verification. Our modelling efforts revealed several errors (or at least ambiguities) in the RFC that no one else spotted before. We presented two proofs of the mutual exclusion property for Zeroconf (for an arbitrary number of hosts and IP addresses): a manual, operational proof, and a proof that combines model checking with the application of a new abstraction relation that is compositional with respect to committed locations. The model checking problem has been solved using Uppaal and the abstractions have been checked by hand.

Figure 1 gives a schematic overview of the abstractions that we needed to go from our concrete model of Zeroconf to a model that we could analyse using Uppaal. We used several different types of abstractions, for instance weakening of guards and dead variable reduction. A key abstraction was to overapproximate all nodes in the network, except for two, by a chaos automaton that can display arbitrary behaviour, thus putting a “spotlight” on the two hosts for which we are trying to prove mutual exclusion.

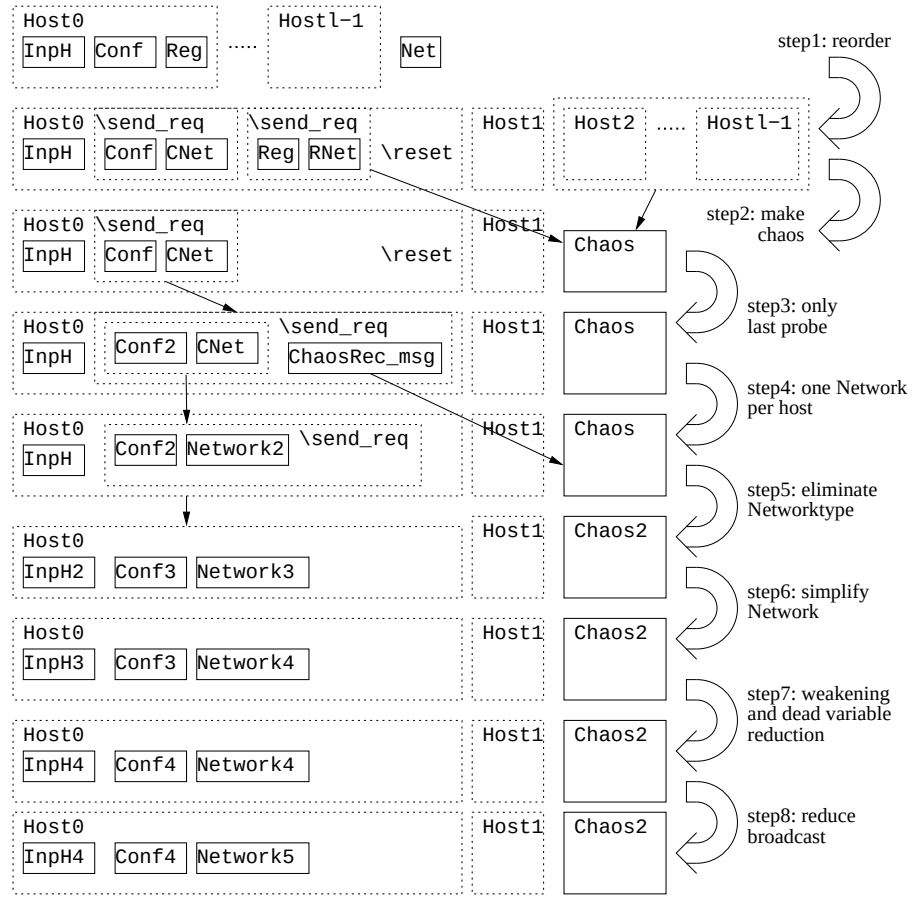

Figure 1: Overview of abstractions

We developed the compositional abstraction technique of [28] (which is based on simulation relations) because we needed it for this case study, but clearly it has a much broader range of applicability. Numerous papers have been written before on compositionality of simulation relations, but the specific challenge that we faced is that it is hard to obtain a sound theory in a setting with both shared variables and shared actions. In [29],[5], for instance, we have shown that the composition operators defined in two published papers [54, 38] is not associative. In order to prove associativity of the composition operator of [28] (which essentially is the composition operator of Uppaal), we needed a series of laws for override and update.

This triggered the work of [4], in which we provide the first sound and complete axiomatisation of overriding and update. There are only very few natural ways in which arbitrary functions can be combined. One composition operator is *override*: for arbitrary functions f and g, $f \rhd g$ is the function with domain $\mathsf{dom}(f) \cup \mathsf{dom}(g)$ that behaves like f on $\mathsf{dom}(f)$ and like g on $\mathsf{dom}(g) \setminus \mathsf{dom}(f)$. Another operator is *update*: $f[g]$ has the same domain as f, behaves like f on $\mathsf{dom}(f) \setminus \mathsf{dom}(g)$, and like g on $\mathsf{dom}(f) \cap \mathsf{dom}(g)$. These operators are widely used, especially within computer science, where for instance $f[g]$ may denote the new state that results when in

state f the updates given as g are applied. It is therefore surprising that thus far no axiomatisation of these operators has been proposed in the literature. As an auxiliary operator we consider the *minus* operator: $f - g$ is the restriction of f to the domain $\mathsf{dom}(f) \setminus \mathsf{dom}(g)$. The update operator can be defined in terms of override and minus. We present five equations that together constitute a sound and complete axiomatisation of override and minus. As part of our completeness proof, we infer a large number of useful derived laws using the proof assistant ISABELLE. With the help of the SMT solver YICES, we establish independence of the axioms. Thus, our axiomatisation is also minimal. Finally, we establish that override and minus are functionally complete in the sense that any operation on general functions that corresponds to a valid colouring of a Venn diagram can be described using just these two operations.

Perspective Much further work is needed to fully mechanise the type of reasoning that we carried out in the Zeroconf case study. In particular, Uppaal needs to be extended with compositional abstraction as in [28]. Whereas some of the abstractions can be proved fully automatically using the algorithms that have been developed by David et al, we expect that also theorem proving support will be required for other abstractions.

2.3 A Complete Specification Theory for Real-Time Systems

Participants

- Alexandre David, Kim G. Larsen, Ulrik Nyman (AAU)
- Axel Legay; INRIA/IRISA, France (EXT)
- Andrzej Wasowski; IT University, Copenhagen, Denmark (EXT)

Challenge

Many modern systems are big and complex assemblies of numerous components. The components are often designed by independent teams, working under a common agreement on what the interface of each component should be. Consequently, compositional reasoning, a mathematical foundations of reasoning about interfaces, is an active research area. It supports inferring properties of the global implementation, or designing and advisedly reusing components. In a logical interpretation, interfaces are specfications and components that implement an interface are understood as models/implementations. Specification theories should support various features including (1) refinement, which allows to compare specfications as well as to replace a specification by another one in a larger design, (2) logical conjunction expressing the intersection of the set of requirements expressed by two or more specifications, (3) structural composition, which allows to combine specifications, and (4) last but not least, a quotient operator that is dual to structural composition. The latter is crucial to perform incremental design. Also, the operations have to be related by compositional reasoning theorems, guaranteeing both incremental design and independent implementability.

Results

In [10] we develop a complete specification framework for real-time systems using Timed I/O Automata as the specfication formalism, with the semantics expressed in terms of Timed I/O Transition Systems. We provide constructs for refinement, consistency checking, logical and structural composition, and quotient of specifications, all indispensable ingredients of a compositional design methodology.

In [9] the theory is implemented on top of the engine for timed games, Uppaal-Tiga, supporting the operations of composition, conjunction, and refinement. Algorithms to perform these operations have been based on a game theoretical setting that permits, for example, to capture the real-time constraints on communication events between components. In particular the algorithms applied for refinement checking and consistency checking are variants of the algorithms for alternating simulation between timed game automata presented in [8, 7].

2.4 Compositional Design Methodology with Constraint Markov Chains

Participants

- Benoit Caillaud, Benoit Delahay, Axel Legay; INRIA/IRISA, France (EXT)
- Kim G. Larsen, Mikkel Larsen Pedersen (AAU)
- Andrzej Wasowski; IT University, Copenhagen, Denmark (EXT)

Challenge

Over the years process algebraic frameworks have been proposed for describing and analyzing probabilistic systems based on Markov Chains (MCs) and Markov Decision Processes. Also a variety of probabilistic logics have been developed for expressing properties of such systems, e.g., PCTL. Both traditions support refinement between specifications using various notions of probabilistic simulation [55, 39] and, respectively, logical entailment [50]. Whereas the process algebraic approach favors structural composition (parallel composition), the logical approach favors logical composition (conjunction). Neither of the two supports both structural and logical composition. For functional analysis of discrete-time non-probabilistic systems, the theory of Modal Transition Systems (MTS) [62] provides a specification formalism supporting refinement as well as conjunction and parallel composition. It has been recently applied to construct interface theories [63, 70]. Generalizing the notion of MTSs to the nonfunctional analysis of probabilistic systems, the formalism of Interval Markov Chains (IMCs[1]) [55] was introduced; with notions of satisfaction and refinement generalizing probabilistic bisimulation. Informally, IMCs extend Markov Chains by labeling transitions with intervals of allowed probabilities rather than concrete probability values. However, the expressive power of IMCs is inadequate as it supports neither logical nor structural composition.

[1] Note that the IMCs here are not to be confused with the *Interactive Markov Chains* of Section 2.1.

Results

In [6], we introduce *Constraint Markov Chains* (CMCs) as a foundation for component-based design of probabilistic systems. CMCs are a further extension of MCs allowing rich constraints on the next-state probabilities from any state. Whereas linear constraints suffice for closure under conjunction, polynomial constraints are necessary for closure under parallel composition. We provide constructs for refinement, consistency checking, logical and structural composition of CMC specifications – all indispensable ingredients of a compositional design methodology.

3 Equivalences and Refinement relations

3.1 Logics and Equivalences for Metric Transition Systems

Participants Mariëlle Stoelinga, Luca de Alfaro, Marco Faella, A. Legay (UT)

In [11] and its predecessors [1, 13], we extend the classical system relations of trace inclusion, trace equivalence, simulation, and bisimulation to a quantitative setting in which propositions are interpreted not as boolean values, but as elements of arbitrary metric spaces. Trace inclusion and equivalence give rise to asymmetrical and symmetrical linear distances, while simulation and bisimulation give rise to asymmetrical and symmetrical branching distances. We study the relationships among these distances, and we provide a full logical characterisation of the distances in terms of quantitative versions of LTL and μ-calculus. We show that, while trace inclusion (resp. equivalence) coincides with simulation (resp. bisimulation) for deterministic boolean transition systems, linear and branching distances do not coincide for deterministic metric transition systems. Finally, we provide algorithms for computing the distances over finite systems, together with a matching lower complexity bound, and algorithms for model checking quantitative LTL over labelled transition systems and Markov Chains.

3.2 Abstraction for Microcontroller Systems

Participants Thomas Noll, Bastian Schlich, Lucas Brutschy, Gerlind Herberich, Carsten Weise, Jörg Brauer (RWTH)

Challenge

Embedded systems usually operate in uncertain environments, giving rise to a high degree of nondeterminism in the corresponding formal models. This, together with other effects, leads to the well-known state-space explosion problem, meaning that the models of those systems grow exponentially in size as the number of components increases. Careful handling of nondeterminism is therefore crucial for obtaining efficient tools for analysis and verification. This requires the development of formal computation models and state-space reduction techniques, and associated correctness proofs.

Results

A general automata-based model for microcontrollers has been developed, taking into account both the hardware, the software, and the environment of the system. This model was used to prove the correctness of a particular abstraction method, called *delayed nondeterminism*, which resolves the uncertainties caused by undetermined input values only if and when this is required by the application code [67]. More concretely, a simulation relation between the concrete and the abstract state space was established, thus showing the soundness of delayed nondeterminism with respect to "path-universal" verification logics such as ACTL and LTL.

Another source of nondeterminism is the potential occurrence of interrupts that can be triggered, e.g., by timers or external events. Aiming at reducing the number of program locations where interrupt handlers have to be taken into account, a new abstraction technique based on partial-order reduction has been developed [25, 21]. This significantly reduces state spaces while the validity of the verification results is preserved. The abstraction is based on an underlying static analysis which annotates the programs before verification, indicating those locations where interrupts can safely be ignored. Moreover, the abstraction method has been proved correct by showing that it preserves the validity of the branching-time logic CTL*-X by establishing a stutter bisimulation equivalence between the abstract and the concrete transition system. Finally, the effectiveness of this abstraction was demonstrated in a larger case study.

Perspective

Current efforts concentrate on refining the delayed nondeterminism technique in two directions. The first observation is that the present version constitutes a (safe) over-approximation, as only a simulation (and not a bisimulation) relation can be established between the concrete and the abstract state space. The underlying reason is that copying of values does not preserve the connection between different instances of the same nondeterministic value, and therefore destroys the bisimulation relation. Here, making the instantiation relation explicit will also ensure the completeness of delayed nondeterminism. Second, we are planning to improve the static analysis that is used for both delayed nondeterminism and interrupt reduction. Currently, we rely on a coarse analysis of pointer variables, meaning that the set of possible address values is over-approximated in many cases. A more precise pointer analysis would definitely improve the results of the static analyses.

3.3 Equivalences for Labelled Markov Chains

Participants Laurent Doyen, Thomas Henzinger, Jean-Francois Raskin (CFV)

Challenge

Simulation relations (as defined by Milner) and trace pre-orders play fundamental roles in the theory underlying program refinement. When a (more concrete) program P_2 is simulated by a (more abstract) program P_1, we know that all the universal CTL* properties that are true for the

program P_1 are also true for the program P_2. When the traces of a (more concrete) program P_2 are included into the traces of a (more abstract) program P_1, we know that all the LTL properties that are true for the program P_1 are also true for the program P_2.

This makes possible the development of programs in a systematic way: important properties are proved on high level descriptions programs, those high level programs are refined into implementations and simulations or trace pre-orders are used to prove that properties proved on the high level programs are valid on the low level program.

Our objective is to define the necessary theory for the application of the above methodology to probabilistic programs. There are currently few results on the trace-based relation between probabilistic models. Our objective is to define the adequate notions for that context and the algorithms necessary to support their practical application.

Results

In [12], we consider the equivalence problem for labelled Markov chains (LMCs), where each state is labelled with an observation. Two LMCs are equivalent if every finite sequence of observations has the same probability of occurrence in the two LMCs. We show that equivalence can be decided in polynomial time, using a reduction to the equivalence problem for probabilistic automata, which is known to be solvable in polynomial time. We provide an alternative algorithm to solve the equivalence problem, which is based on a new definition of bisimulation for probabilistic automata. We also extend the technique to decide the equivalence of weighted probabilistic automata.

Then, we consider the equivalence problem for labelled Markov decision processes (LMDPs), which asks given two LMDPs whether for every scheduler (*i.e.* way of resolving the nondeterministic decisions) for each of the processes, there exists a scheduler for the other process such that the resulting LMCs are equivalent. The decidability of this problem remains open. We show that the schedulers can be restricted to be observation-based, but may require infinite memory.

Perspective

The equivalence problem between labelled Markov decision processes remains open. An intermediary step would be to consider the relation between LMDPs and LMCs: given a LMDP A and a LMC B, we ask if there exists a scheduler S such that the resulting LMC A(S) is equivalent to the LMC B.

3.4 Quantitative Analysis and Logical Characterization of Weighted Systems

Participants Uli Fahrenberg, Kim G. Larsen, Claus Thrane (AAU)

Challenge

The research presented in this work is motivated by the "The Embedded Systems Design Challenge", posed by Henzinger and Sifakis in [45]. Henzinger and Sifakis express the need for a coherent theory of embedded systems design, where concern for physical constraints is supported by the computational models used to model software, thus achieving a more heterogeneous approach to design. Highly distilled, Henzinger and Sifakis call for a new mathematical basis for systems modeling which facilitates modeling of behavioural properties as well as environmental constraints.

Analysis and verification of concurrent and reactive systems is a well established research field, a branch of which is referred to as implementation verification: verification of systems design based on behavioural equivalence checking. This approach requires a model of the system and specification, as well as a procedure for checking whether the two are related with respect to some equivalence. The choice of this equivalence relation reflects what one wants to observe and how. Classical examples of such relations include trace inclusion and various types of simulation. Correspondingly, the models which are analyzed must encompass all the relevant information to facilitate the analysis. Specifically, the formalism used to model the system must be rich enough to express the characteristics of the system, in order for the analysis to prove or refute the proposed equivalence.

In a quantitative setting, equivalences are replaced by real-valued distances; intuitively the problem is lifted from a decision problem to a search problem, i.e. from deciding on $\{true, false\}$ to computing a distance $\epsilon \in R$. A distance of 0 (zero) is given to instances which are accepted by the binary decision procedure, and the meaning of values $\epsilon > 0$ is that the instance is not equal to the specification, yet related up to some error margin given by the distance ϵ.

Results

In [15] we present a general framework for the analysis of quantitative and qualitative properties of reactive systems, based on a notion of weighted transition systems. We introduce and analyze three different types of distances on weighted transition systems, both in a linear and a branching version. Our quantitative notions appear to be reasonable extensions of the standard qualitative concepts, and the three different types introduced are shown to measure inequivalent properties. When applied to the formalism of weighted timed automata, we show that some standard decidability and undecidability results for timed automata extend to our quantitative setting.

This contribution is followed by [14], where we extend the usual notion of Kripke Structures with a weighted transition relation, and generalize the usual Boolean satisfaction relation of CTL to a map which assigns to states and temporal formulae a real-valued distance describing the degree of satisfaction. The work describes a general approach to obtaining quantitative interpretations for a generic extension of the CTL syntax, and show that, for one such interpretation, the logic is both *adequate* and *expressive* with respect to quantitative bisimulation. Here adequacy means that the bisimulation distance between two systems is identical to the distance in satisfaction for all formulas. Expressivity means that for any system s we are able to find a single formula ϕ_s, such that for any other system the bisimulation distance to s is identical to the degree

of satisfaction of ϕ_s.

4 Abstract Interpretation

4.1 Best Probabilistic Transformers

Participants Ernst Moritz Hahn, Holger Hermanns, Björn Wachter, Lijun Zhang (SU)

Challenge

Markov decision processes (MDPs) [69] play a crucial role as a semantic model in the analysis of systems with random phenomena like network protocols and randomised algorithms. MDPs feature non-determinism and probabilistic choice. Typically one is interested in computing (maximal or minimal) reachability probabilities, e.g., the probability of delivering three messages after ten transmission attempts. Recently predicate-abstraction techniques [50], [59] have evolved that scale to realistic programs which map to infinite MDPs. However, fundamental questions remain open, e.g. for given predicates, what is the most precise abstract program that is still a valid abstraction?

The theory of abstract interpretation [36] has provided answers to such questions in the non-probabilistic case and has served as a foundation and design paradigm for a wide range of program analyses. In abstract interpretation, program analyses are expressed in terms of non-standard abstract semantics obtained by replacing the actual domain of computation (also called *concrete domain*) by an *abstract domain*. Concrete and abstract domain are partially ordered sets where ordering describes relative precision of the denotations.

A specification of the *most precise* analysis is given by the composition of the concretisation function, the functional f characterising the program semantics and the abstraction function. Being the limit on the best achievable precision for *any* valid abstraction, the resulting functional is called *best transformer*. These concepts are the starting point of our work.

Results

Our major theoretical contribution is the first abstract-interpretation framework for MDPs which admits to compute both lower and upper bounds on reachability probabilities. This provides a solid basis to reason about the relative precision and optimality of abstract transformers. Further, we prove that game-based abstraction [60], a pre-existing construction by Kwiatkowska et al., corresponds to best transformers in our framework. Crucial differences to a previous abstract-interpretation framework for MDPs by Monniaux [66] are: we consider not only upper but also lower bounds, study best transformers, and target predicate abstraction not classical domains from static analysis.

Our second contribution is the first abstraction-refinement technique for concurrent probabilistic programs that yields both lower and upper bounds. Previous analysis techniques for such programs were also based on predicate abstraction. However they either only yield effective upper bounds [50] or come without refinement [58]. The basis of our refinement technique is

parallel abstraction, a novel abstraction. Parallel abstraction yields effective lower and upper bounds and combines well with refinement. We have implemented our ideas in the PASS tool and report on experimental results.

Perspective

As future work, we would like to extend our abstract-interpretation framework [26] and our tool [19] to more complex temporal properties, like PCTL, and rewards.

4.2 Guided Abstraction for Alternating Automata

Participants Pierre Ganty, Nicolas Maquet, Jean-Francois Raskin (CFV)

In [16], we develop and evaluate two new algorithms for checking emptiness of alternating automata. These algorithms build on previous works. First, they rely on antichains to efficiently manipulate the state-spaces underlying the analysis of alternating automata. Second, they are abstract algorithms with built-in refinement operators based on techniques that exploit information computed by abstract fixed points (and not counter-examples as it is usually the case). The efficiency of our new algorithms is illustrated by experimental results.

5 Abstraction of Infinite Systems

5.1 Time-Bounded Model Checking of Infinite-State Continuous-Time Markov Chains

Participants Lijun Zhang, Ernst Moritz Hahn, Holger Hermanns, Björn Wachter (SU)

Challenge

The design of complex concurrent systems often involves intricate performance and dependability considerations. Continuous time Markov (reward) models are a widely used modelling formalism that captures such performance and dependability properties, and makes them analysable by model checking. Models with infinite state space show up as abstractions of finite systems, when a certain resource is virtually unrestricted.

A great research interest lies in the study of time bounded properties. These subsume time bounded probabilistic reachability, performability, survivability, and various availability measures like instantaneous, conditional instantaneous and interval availabilities.

For the acceptance of formal methods in practice, the convenient expression of such properties is important. A well-known formalism for the expression of properties of finite Markov models is the the continuous stochastic logic (CSL).

Results

In the papers [17, 18, 27], we introduced time bounded model checking for the time bounded subset of CSL for infinite CTMCs and infinite Markov reward models. For the analysis, we only store a finite fraction of the infinite state space, guaranteeing results up to a certain precision. A CSL formula consists of a set of nested subformulas, each of which may contain different time bounds and Boolean connectors. Our method works by descending into subformulas while extracting a sufficiently large fraction of the state space. Afterward, usual model checking methods are applied on the finite sub-CTMC obtained this way.

Depending on the model under analysis, we can choose between several methods. We have methods which are fast but store a large subset of the infinite state space. In addition, we also have methods which take more time, as they choose this subset in a more elaborate way, leading to smaller memory consumption.

To evaluate our methods, we developed the tool INFAMY. We showed its practical usability on models from various domains, including systems biology, queueing theory as well as performance evaluation.

- Random Walk: we consider a standard random walk model as an introductory example to our method
- Jackson Queueing Networks [53]: we consider a number of Jackson queueing networks. Even though there are specialised methods for these kinds of models, they would not be applicable for slight extensions of this model class, whereas INFAMY is. Because of this, we do consider it interesting to take a look at this model class.
- Quasi-Birth-Death Process [56]: we consider a case study that describes a system consisting of a number processors and an infinite queue for storing job requests.
- Protein synthesis [6]: we consider a CTMC model of a protein synthesis of a cell.
- Workstation Cluster [43]: this model is a standard model in performance evaluation. While this model is finite, INFAMY was still of use, because we only had to explore a small subset of the large state space.
- Grid-World Robot [74]: We consider a grid world in which a robot moves around in an infinite area and may be subject to disturbances by the environment.

In all of the above case studies, we were able to analyse the properties we wanted to consider. For most of them, it was crucial for the analysis to give an adequate choice of the trade-off between speed and memory consumption. We consider the fact that the models under consideration were taken from diverse areas and are very different from each other an indication of the general applicability of our method.

Perspective

We plan to extend our approach into various directions. At first place, we plan to consider further case studies to further evaluate our approach. We also target at improving performance in speed and memory usage. Another main interest is exploring the applicability of more expressive logics than CSL and to extend the set of properties we can analyse using INFAMY.

5.2 Graph Abstraction and Transformation

Participants Jörg Bauer, Iovka Boneva, Marcos E. Kurbán, Arend Rensink (UT)

Infinite or very large state spaces often prohibit the successful verification of graph transformation systems. Abstract graph transformation is an approach that tackles this problem by abstracting graphs to abstract graphs of bounded size and by lifting application of productions to abstract graphs. The paper [2], which received the best paper award at the 4th International Conference of Graph Transformation in 2008, presents a new framework of abstractions unifying and generalising existing takes on abstract graph transformation. The precision of the abstraction can be adjusted according to the properties to be verified facilitating abstraction refinement. We present a modal logic defined on graphs, which is preserved and reflected by our abstractions. Finally, we demonstrate the usability of the framework by verifying a graph transformation model of a firewall.

6 Aggregation

6.1 Probabilistic Reachability for Parametric Markov Models

Participants Lijun Zhang, Ernst Moritz Hahn and Holger Hermanns (SU)

Note

This contribution is also part of Deliverable 2.2 on “Symbolic data structures and analysis of models with multiple quantitative aspects”, Section 1.1.

Challenge

Discrete time Markov chains (DTMCs) have been applied successfully to reason about quantitative properties in a large number of areas such as computer science, engineering, mathematics, and biological systems. Often, several variants of a probabilistic model are of interest. For example, it would be interesting to evaluate several variants of sensor networks with different reliabilities of the wireless connection, without doing a complete analysis for each instance.

We call a DTMC in which certain probabilities or other properties are not fixed but given as parameters of the model a *parametric* DTMC (PDTMC). An analysis of a PDTMC results then in a closed-form solution in form of a function in the parameters. Given such a function f, we

could also analyse properties of the function itself. If f represents the probability of a certain set of goal states, we could find the parameter values which maximise f to obtain the optimal parameters, without having to do large numbers of costly analyses to estimate this point.

The efficient analysis of PDTMCs is involved and different approaches than the well-known ones for the analysis of DTMC have to be taken. Our goal is to nevertheless develop an efficient and effective algorithm for PDTMCs and related models.

Results

In the paper [20], we have presented algorithms for PDTMCs. Our method is based on a variant of the classical state elimination algorithm, used in classical Automata Theory to derive regular expressions from finite automata. It computes the parametric unbounded reachability probability from the initial state of the PDTMC to a set of target states. The state elimination algorithms is a standard means to derive a regular expression from a finite automaton, by eliminating its states except the initial and final one, while relabelling its transitions by regular expressions instead of just elements of the alphabet. In our adaption, instead of having transitions labelled with regular expressions, we label them with functions of the model parameters into probabilities. Finally, we can obtain the function we wanted to obtain from the only transition remaining in the PDTMC.

We also have an initial approach for models involving nondeterminism. There, we replace nondeterminism by parametric probabilistic choice. This method works well for special cases, as seen in the paper.

We further extend our method to compute the expected parametric reward till a set of target states is reached. Rewards are costs or bonuses (depending on the interpretation) obtained from entering a state of the PDTMC or taking a transition from a state to another state. Such reward properties play a crucial role for the estimation of performance properties of probabilistic systems.

The analysis of PDTMCs is more expensive than the analysis of usual DTMCs. Therefore, we use a precomputation to reduce the number of states. This has a great impact of the overall performance of the method.

The algorithms described here have been implemented in the tool PARAM. Using a number of case studies, we have shown the feasibility of our approach.

- Crowds Protocol [71]: an information exchange protocol with aims at protecting the anonymity of its users. We considered the degree of anonymity guarantees possible to users parametric in the number of attackers.

- Zeroconf [31]: a self-configuring network protocol. We considered a variant parametric in the number of possible network addresses. The property under consideration is the probability of duplicate choice of the same address.

- Cyclic Polling Server [52]: This model consists of a number of stations which are handled by a polling server. We considered the probability that a certain station is served first, parametric in the speed with which the server works and the rate with which requests are generated.

- Randomised Mutual Exclusion [68]: a variant of the well-known mutual exclusion protocol where processes decide probabilistically whether they will try to enter the critical section in their next step. We compute the expected number of times the processes try to enter the critical section, parametric in the probability that they try it.
- Bounded Retransmission Protocol [44]: a message transfer protocol to transfer data over unreliable channels. Our variant is parametric in the reliabilities of the channels. The property we consider is the maximal probability that the sender of data in this protocol does not finally finish the transmission.

In all of the above case studies, we were able to analyse the properties we wanted to consider. For most of them, it was crucial to use preprocessing for state space reduction. We consider the fact that the models under consideration were taken from diverse areas and are very different from each other an indication of the general applicability of our method.

Perspective

As future work, we are investigating improvements of the implementation with respect to performance, especially for the setting with nondeterminism. Additionally, we plan to look into continuous time models with clocks and rewards. Other possible directions include the use of symbolic model representations, such as advanced representations of state spaces. We also want to explore model checking for interval Markov chains.

References to Quasimodo Contributions

[1] L. De Alfaro, R. Majumdar, V. Raman, and M.I.A. Stoelinga. Game refinement relations and metrics. *Logical Methods in Computer Science*, 4, 2008.

[2] J. Bauer, I. B. Boneva, M. E. Kurban, and A. Rensink. A modal-logic based graph abstraction. In H. Ehrig, R. Heckel, G. Rozenberg, and G. Taentzer, editors, *International Conference on Graph Transformations (ICGT), Leicester, UK*, volume 5214 of *LNCS*, pages 321–335, Berlin, 2008. Springer Verlag.

[3] J. Berendsen, B. Gebremichael, F.W. Vaandrager, and M. Zhang. Formal specification and analysis of zeroconf using Uppaal. *ACM Transactions on Embedded Computing Systems*, 2010. To appear.

[4] J. Berendsen, D.N. Jansen, J. Schmaltz, and F.W. Vaandrager. The axiomatization of override and update. *Journal of Applied Logic*, 2009. Available online 18 November 2009.

[5] J. Berendsen and F.W. Vaandrager. Parallel composition in a paper by De Alfaro e.a. is not associative. Technical note available at `http://www.ita.cs.ru.nl/publications/papers/fvaan/BV07.html`, May 2008.

[6] B. Caillaud, B. Delahaye, K. G. Larsen, M. Larsen Pedersen, and A. Wasowski. Compositional design methodology with constraint markov chains. under submission.

[7] A. David, K. G. L., and T. Chatain. Playing games with timed games. In *Proc. of 3rd IFAC Conference on analysis and Design of Hybrid Systems*, 2009.

[8] A. David, K. G. Larsen, T. Chatain, and P. Bulychev. Efficient on-the-fly algorithm for checking alternating timed simulation. In *Proc. FORMATS '09*, volume 5813 of *LNCS*, pages 73–87, 2009.

[9] A. David, K. G. Larsen, A. Legay, U. Nyman, and A. Wasowski. An environment for compositional design and analysis of real time systems. under submission.

[10] A. David, K. G. Larsen, A. Legay, U. Nyman, and A. Wasowski. Timed i/o automata: A complete specification theory for real-time systems. In *Proc. HSCC '10*. ACM, 2010. To appear.

[11] L. de Alfaro, M. Faella, and M.I.A. Stoelinga. Linear and branching system metrics. *IEEE Transactions on Software Engineering*, 35(2), 2008.

[12] L. Doyen, T. A. Henzinger, and J.-F. Raskin. Equivalence of labeled markov chains. *Int. J. Found. Comput. Sci.*, 19(3):549–563, 2008.

[13] M. Faella, A. Legay, and M.I.A. Stoelinga. Model checking quantitative linear time logic. In A. Aldini and C.Baier, editors, *Proceedings of the Sixth Workshop on Quantitative Aspects of Programming Languages (QAPL'08)*, ENTCS, 2007.

[14] U. Fahrenberg, K. G. Larsen, and C. Thrane. A quantitative characterization of weighted kripke structures in temporal logic. In P. Hlineny, V. Katyas, and T. Vojnar, editors, *Proc. MEMICS '09*, 2009. Best Paper Award.

[15] U. Fahrenberg, K. G. Larsen, and C. Thrane. Quantitative analysis of weighted transition systems. *J. Logic and Algebraic Programming*, 2010. Special Issue of NWPT08, to appear.

[16] P. Ganty, N. Maquet, and J.-F. Raskin. Fixpoint guided abstraction for alternating automata. In *CIAA'09*, number 5642 in LNCS, pages 155–164. Springer, 2009.

[17] E. M. Hahn, H. Hermanns, B. Wachter, and L. Zhang. Infamy: An infinite-state markov model checker. In *CAV*, volume 5643 of *LNCS*, pages 641–647, 2009.

[18] E. M. Hahn, H. Hermanns, B. Wachter, and L. Zhang. Time-bounded model checking of infinite-state continuous-time Markov chains. *Fundamenta Informaticae*, 95:129–155, 2009.

[19] E. M. Hahn, H. Hermanns, B. Wachter, and L. Zhang. PASS: Abstraction Refinement for Infinite Probabilistic Models. In *Proc. TACAS '10*, 2010. to appear.

[20] E. M. Hahn, H. Hermanns, and L. Zhang. Probabilistic reachability for parametric markov models. In *Proc. SPIN*, volume 5578 of *LNCS*, pages 88–106, 2009.

[21] Gerlind Herberich, Thomas Noll, Bastian Schlich, and Carsten Weise. Proving correctness of an efficient abstraction for interrupt handling. In *Proceedings 3rd International Workshop on Systems Software Verification (SSV)*, volume 217 of *ENTCS*, pages 133–150. Elsevier, 2008.

[22] J.-P. Katoen, D. Klink, M. Leucker, and V. Wolf. Abstraction for stochastic systems by Erlang's method of stages. In *CONCUR*, volume 5201 of *LNCS*, pages 279–294, 2008.

[23] J.-P. Katoen, D. Klink, and M. R. Neuhäußer. Compositional abstraction for stochastic systems. In *Proc. FORMATS '09*, volume 5813 of *LNCS*, pages 195–211. Springer, 2009.

[24] D. Klink, A. Remke, B. R. Haverkort, and J.-P. Katoen. Time-bounded reachability in tree-structured QBDs by abstraction. In *Proc. QEST '09*, pages 133–142. IEEE CS, 2009.

[25] B. Schlich, T. Noll, J. Brauer, and L. Brutschy. Reduction of interrupt handler executions for model checking embedded software. In *Proc. of Haifa Verification Conference 2009 (HVC 2009)*, LNCS. Springer, 2009.

[26] B. Wachter and L. Zhang. Best Probabilistic Transformers. In *VMCAI*, volume 5944 of *LNCS*, pages 362–379, 2010.

[27] L. Zhang, H. Hermanns, E. M. Hahn, and B. Wachter. Time-bounded model checking of infinite-state continuous-time markov chains. In J. Billington, Z. Duan, and M. Koutny, editors, *Proc. ACSD '08*, pages 98–107. IEEE, 2008.

References to Related and General Literature

[28] J. Berendsen and F.W. Vaandrager. Compositional abstraction in real-time model checking. Technical Report ICIS–R07027, Radboud University Nijmegen, 2007. An extended abstract appeared in *Proc. FORMATS'08*, LNCS 5215.

[29] J. Berendsen and F.W. Vaandrager. Parallel composition in a paper of Jensen, Larsen and Skou is not associative. Technical note available at `http://www.ita.cs.ru.nl/publications/papers/fvaan/BV07.html`, September 2007.

[30] E. Bode, M. Herbstritt, H. Hermanns, S. Johr, T. Peikenkamp, R. Pulungan, R. Wimmer, and B. Becker. Compositional performability evaluation for statemate. In *Proc. QEST '06*, pages 167–178. IEEE Computer Society, 2006.

[31] H. Bohnenkamp, P. van der Stok, H. Hermanns, and F. Vaandrager. Cost-optimization of the IPv4 zeroconf protocol. In *Proc. DSN '03*, pages 531–540. IEEE Computer Society, IEEE Computer Society Press, June 2003.

[32] L. Cardelli. On process rate semantics. *Theor. Comput. Sci.*, 391(3):190–215, 2008.

[33] S. Cheshire, B. Aboba, and E. Guttman. Dynamic configuration of ipv4 link-local addresses. Published Online, http://files.zeroconf.org/rfc3927.txt, 2005. IETF Standard, RFC 3927.

[34] E. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith. Counterexample-guided abstraction refinement. In *Proc. CAV '00*, volume 1855 of *LNCS*, pages 154–169. Springer, 2000.

[35] E. M. Clarke, O. Grumberg, and D. E. Long. Model checking and abstraction. *ACM Trans. Program. Lang. Syst.*, 16(5):1512–1542, 1994.

[36] P. Cousot and R. Cousot. Abstract Interpretation: A Unified Lattice Model for Static Analysis of Programs by Construction or Approximation of Fixpoints. In *Proc. POPL '77*, pages 238–252, 1977.

[37] D. Dams, R. Gerth, and O. Grumberg. Abstract interpretation of reactive systems. *ACM Transactions on Programming Languages and Systems*, 19(2):253–291, March 1997.

[38] L. de Alfaro, L. Dias da Silva, M. Faella, A. Legay, P. Roy, and M. Sorea. Sociable interfaces. In B. Gramlich, editor, *Frontiers of Combining Systems, 5th International Workshop, FroCoS 2005, Vienna, Austria, September 19-21, 2005, Proceedings*, volume 3717 of *LNCS*, pages 81–105. Springer, 2005.

[39] H. Fecher, M. Leucker, and V. Wolf. Don't know in probabilistic systems. In *Proc. SPIN '06*, volume 3925 of *LNCS*, pages 71–88, 2006.

[40] H. Garavel and H. Hermanns. On combining functional verification and performance evaluation using CADP. In *FME*, volume 2391 of *LNCS*, pages 410–429, 2002.

[41] S. Gilmore, J. Hillston, and M. Ribaudo. An efficient algorithm for aggregating PEPA models. *IEEE Trans. Software Eng.*, 27(5):449–464, 2001.

[42] O. Grumberg. Abstraction and refinement in model checking. In *Proc. FMCO '05*, volume 4111 of *LNCS*, pages 219–242. Springer, 2005. Revised Lectures.

[43] B. R. Haverkort, H. Hermanns, and J.-P. Katoen. On the use of model checking techniques for dependability evaluation. In *Proc. SRDS '00*, pages 228–237. IEEE, 2000.

[44] L. Helmink, M. Sellink, and F. Vaandrager. Proof-checking a data link protocol. In H. Barendregt and T. Nipkow, editors, *TYPES*, volume 806 of *LNCS*, pages 127–165, 1994.

[45] T. A. Henzinger and J. Sifakis. The embedded systems design challenge. In *Proc FM '06*, volume 4085 of *LNCS*, pages 1–15, 2006.

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

[47] H. Hermanns, U. Herzog, and J.-P. Katoen. Process algebra for performance evaluation. *Theor. Comput. Sci.*, 274(1-2):43–87, 2002.

[48] H. Hermanns and J.-P. Katoen. Automated compositional Markov chain generation for a plain-old telephone system. *Sci. Comput. Program.*, 36(1):97–127, 2000.

[49] H. Hermanns and M. Ribaudo. Exploiting symmetries in stochastic process algebras. In *European Simulation Multiconference*, pages 763–770. SCS Europe, 1998.

[50] H. Hermanns, B. Wachter, and L. Zhang. Probabilistic CEGAR. In *Proc. CAV '08*, pages 162–175, 2008.

[51] J. Hillston. *A Compositional Approach to Performance Modelling*. Cambridge University Press, 1996.

[52] O. Ibe and K. Trivedi. Stochastic Petri net models of polling systems. *IEEE Journal on Selected Areas in Communications*, 8(9):1649–1657, 1990.

[53] J. Jackson. Networks of waiting lines. *Operations Research*, 5:518–521, 1957.

[54] H.E. Jensen, K.G. Larsen, and A. Skou. Scaling up Uppaal: Automatic verification of real-time systems using compositionality and abstraction. In M. Joseph, editor, *Formal Techniques in Real-Time and Fault-Tolerant Systems, 6th International Symposium, FTRTFT 2000*, Pune, India, September 20-22, *Proceedings*, volume 1926 of *LNCS*, pages 19–30. Springer, 2000.

[55] B. Jonsson and K. G. Larsen. Specification and refinement of probabilistic processes. In *Proc. LICS '91*, pages 266–277. IEEE Computer Society, 1991.

[56] J.-P. Katoen, D. Klink, M. Leucker, and V. Wolf. Three-valued abstraction for continuous-time Markov chains. In *CAV*, volume 4590 of *LNCS*, pages 311–324. Springer, 2007.

[57] J.-Pieter Katoen, D. Klink, M. Leucker, and V. Wolf. Three-valued abstraction for continuous-time Markov chains. In *CAV*, volume 4590 of *LNCS*, pages 316–329, 2007.

[58] M. Kattenbelt, M. Z. Kwiatkowska, G. Norman, and D. Parker. Game-Based Probabilistic Predicate Abstraction in PRISM. In *Proc. QAPL '08*, volume 220 of *ENTCS*, pages 5–21, 2008.

[59] M. Kattenbelt, M. Z. Kwiatkowska, G. Norman, and D. Parker. Abstraction Refinement for Probabilistic Software. In *Proc. VMCAI '09*, pages 182–197, 2009.

[60] M. Z. Kwiatkowska, G. Norman, and D. Parker. Game-based Abstraction for Markov Decision Processes. In *Proc. QEST '06*, pages 157–166. IEEE CS Press, 2006.

[61] K. G. Larsen. Modal specifications. In *Automatic Verification Methods for Finite State Systems*, volume 407 of *LNCS*, pages 232–246, 1989.

[62] K. G. Larsen. Modal specifications. In *Proc. AVMS '89*, volume 407 of *LNCS*, 1989.

[63] K. G. Larsen, U. Nyman, and A. Wasowski. Modal i/o automata for interface and product line theories. In *Proc. ESOP*, volume 4421 of *LNCS*, 2007.

[64] Kim G. Larsen and B. Thomsen. A modal process logic. In *LICS*, pages 203–210. IEEE Computer Society, 1988.

[65] H. Macià, V. Valero, and D. de Frutos-Escrig. sPBC: A Markovian extension of finite Petri box calculus. *Petri Nets and Performance Models*, pages 207–216, 2001.

[66] D. Monniaux. Abstract Interpretation of Programs as Markov Decision Processes. *Sci. Comput. Program.*, 58(1-2):179–205, 2005.

[67] T. Noll and B. Schlich. Delayed nondeterminism in model checking embedded systems assembly code. In *Hardware and Software: Verification and Testing (Haifa Verification Conference, HVC)*, volume 4899 of *LNCS*, pages 185–201. Springer, 2008.

[68] A. Pnueli and L. Zuck. Verification of multiprocess probabilistic protocols. *Distrib. Comput.*, 1(1):53–72, 1986.

[69] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 1994.

[70] J-B Raclet, E. Badouel, A. Benveniste, B. Caillaud, A. Legau, and R. Passerone. Modal interfaces: Unifying interface automata and modal cifications. In *Proc. EMSOFT '09*, 2009.

[71] M. K. Reiter and A. D. Rubin. Crowds: anonymity for web transactions. *ACM Trans. Inf. Syst. Secur.*, 1(1):66–92, 1998.

[72] R. van Glabbeek. The linear time - branching time spectrum II. In *CONCUR '93*, pages 66–81, London, UK, 1993. Springer-Verlag.

[73] R. van Glabbeek. The linear time – branching time spectrum I. In J.A. Bergstra, A. Ponse, and S.A. Smolka, editors, *Handbook of Process Algebra*, chapter 1, pages 3–99. Elsevier, 2001.

[74] H. L. S. Younes. Ymer: A statistical model checker. In *Proc. CAV '05*, volume 3576 of *LNCS*, pages 429–433, 2005.