The Journal of Logic and Algebraic Programming 83 (2014) 1–19

Contents lists available at ScienceDirect

The Journal of Logic and Algebraic Programming

www.elsevier.com/locate/jlap

Stuttering for Abstract Probabilistic Automata ☆

Benoît Delahaye [a], Kim G. Larsen [b], Axel Legay [a,*]

[a] *INRIA/IRISA, Rennes, France*
[b] *Aalborg University, Denmark*

ARTICLE INFO

Article history:
Received 10 November 2012
Received in revised form 22 May 2013
Accepted 23 May 2013
Available online 13 June 2013

Keywords:
Specification theory
Probabilistic Automata
Stuttering
Logical characterization
Refinement

ABSTRACT

Probabilistic Automata (PAs) are a widely-recognized mathematical framework for the specification and analysis of systems with non-deterministic and stochastic behaviors. In a series of recent papers, we proposed Abstract Probabilistic Automata (APAs), a new abstraction framework for representing possibly infinite sets of PAs. We have developed a complete abstraction theory for APAs, and also proposed the first specification theory for them. APAs support both satisfaction and refinement operators, together with classical stepwise design operators.

One of the major drawbacks of APAs is that the formalism cannot capture PAs with hidden actions – such actions are however necessary to describe behaviors that shall not be visible to a third party. In this paper, we revisit and extend the theory of APAs to such context. Our first main result takes the form of proposal for a new probabilistic satisfaction relation that captures several definitions of PAs with hidden actions. Our second main contribution is to revisit all the operations and properties defined on APAs for such notions of PAs. Finally, we also establish the first link between stochastic modal logic and APAs, hence linking an automata-based specification theory to a logical one.

1. Introduction

Nowadays, systems are tremendously big and complex and mostly result from the assembling of several components. These components are usually designed by teams working *independently* but with a common agreement on what the interface of each component should be. These interfaces, also called specifications, precise the behaviors expected from each component as well as the environment in which they can be used, but do not impose any constraint on how the components are implemented.

Instead of relying on Word/Excel text documents or modeling languages such as UML/XML, as is usually done in practice, a series of recent works recommend relying most possibly on mathematically sound formalisms. Mathematical foundations that allow to reason at the abstract level of interfaces, in order to infer properties of the global implementation, and to design or to advisedly (re)use components is a very active research area, known as *compositional reasoning* [18]. Any good specification theory shall be equipped with a *satisfaction relation* (to decide whether an implementation satisfies a specification), a *refinement relation* (to compare sets of implementations), a *logical conjunction* (to compute intersection of sets of implementations), and a *structural composition* (to combine specifications). Additionally, properties such as precongruence of composition with respect to refinement [18] shall also be satisfied.

☆ This paper is an extended version of a conference paper presented at LFCS 2013 [7]. The main differences are in the presentation of the theory, the addition of proofs for main theorems, and new examples and definitions.

* Corresponding author.
E-mail address: axel.legay@inria.fr (A. Legay).

1567-8326/$ – see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jlap.2013.05.006

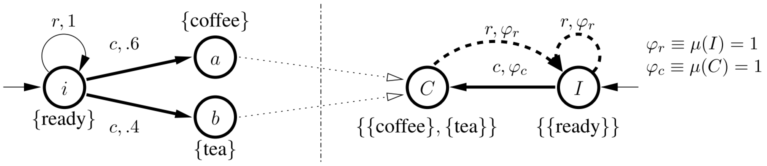

Fig. 1. Implementation PA (left) and specification APA (right) of a coffee machine.

Building good specification theories has been the subject of intensive studies among which one finds classical logical specifications, various process algebrae such as CSP, or Input/Output automata/interfaces (see [19,6,24]). Recently, a new series of works has concentrated on *modal specifications* [20], a language theoretic account of a fragment of the modal mu-calculus logic which is known to admit a more flexible and easy-to-use compositional refinement method than those carried out in CSP [20,29,4].

As soon as systems include randomized algorithms, probabilistic protocols, or interact with physical environment, probabilistic models are required to reason about them. This is exacerbated by requirements for fault tolerance, when systems need to be analyzed quantitatively for the amount of failure they can tolerate, or for the delays that may appear. As Henzinger and Sifakis [18] point out, introducing probabilities into design theories allows assessing dependability of IT systems in the same manner as commonly practiced in other engineering disciplines.

In recent works [5,10], we proposed Constraint Markov Chains (CMCs), a complete specification theory for pure stochastic systems, namely Markov Chains (MCs). Roughly speaking, a CMC is an MC equipped with a constraint on the next-state probabilities from any state. An implementation for a CMC is thus a MC, whose next-state probability distribution satisfies the constraint associated with each state. Contrary to Interval Markov Chains where sets of distributions are represented by intervals, CMCs are closed under both composition and conjunction. Later, in [8], the CMC approach was extended to handle those systems that combine both stochastic and non-deterministic behaviors, i.e., Probabilistic Automata (PA). APAs, is the result of combining Modal Automata and CMCs – the abstractions for labeled transition systems and Markov Chains, respectively. Like other modal-based specification theories, our formalism can be used in various areas, including abstract model checking and compositional reasoning.

The specification theory induced by APAs is more expressive than any classical specification theories where both implementations and specifications are represented by the same object. As an example, Segala's theory assumes that both specifications and implementations are represented with PAs [33,26]. Such an approach does not permit to represent an infinite set of non-deterministic behaviors in a finite way. On the other hand, while satisfaction relation between PAs [25] can be expressed with classical notions of (stochastic) simulations [33], ours requires the use of a rather more complex definition of equivalence relation. Consider the implementation (left) and specification (right) of a coffee machine given in Fig. 1. The specification specifies that there are two possible transitions from initial state I: a may transition labeled with action r (reset) and a must transition labeled with action c (coin). May transitions, which may not be implemented, are represented with dashed arrows. Must transitions, which shall be present in any implementation of the specification, are represented with plain arrows. The probability distributions associated with these actions are specified by the constraints φ_r and φ_c, respectively. One can see that the implementation gives a more precise behavior of the coffee machine: action r loops back to initial state i with probability 1, while coin leads to state a (coffee) with probability .6 and to state b (tea) with probability .4. Satisfaction between implementation and specification lifts the classical notion of simulation for PAs to APAs as follows: (1) all must transitions of the specification must be matched with transitions in the implementations, and (2) all transitions in the implementation must be matched with may transitions in the specification. Additionally, we have to check that the probability distributions in the implementation are matched with probability distributions in the specification that satisfy the given constraints.

1.1. Contribution

In the process of incremental design (as well as for other applications), it may be necessary to incrementally widen the scope of implementations. Usually, the latter is done by permitting the addition of hidden actions also called stutter steps [33,3] in the implementation. In some cases, such stutter steps are even considered at the specification level [3]. Introducing such actions is known to complicate the definition and the computation of operations such as bisimulation/simulation [33]. Moreover, it may break up some properties such as precongruence of refinement with respect to composition [33]. The objective of this paper is to extend the APA specification theory by considering implementations with stuttering steps. Our first contribution is the definition of a new stochastic satisfaction relation for APAs. This relation generalizes stochastic simulation to the APA level. We then study various notions of stuttering and compare their expressivity. We also study the impact of adding stuttering on various properties such as precongruence of refinement with respect to composition. Finally, we define and study ML-(A)PA that is a new modal logic for APAs and stuttering PAs. ML-(A)PA generalizes the PML logic [21,22] of Larsen et al. from PAs to APAs and stuttering PAs.