Available online at www.sciencedirect.com
ScienceDirect
ELSEVIER
Electronic Notes in Theoretical Computer Science 174 (2007) 71–93
Electronic Notes in Theoretical Computer Science
www.elsevier.com/locate/entcs

Proving Approximate Implementations for Probabilistic I/O Automata

Sayan Mitra[1] Nancy Lynch[2]

Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, USA

Abstract

In this paper we introduce the notion of approximate implementations for Probabilistic I/O Automata (PIOA) and develop methods for proving such relationships. We employ a *task structure* on the locally controlled actions and a *task scheduler* to resolve nondeterminism. The interaction between a scheduler and an automaton gives rise to a *trace distribution*—a probability distribution over the set of traces. We define a PIOA to be a (discounted) *approximate implementation* of another PIOA if the set of trace distributions produced by the first is close to that of the latter, where closeness is measured by the (resp. discounted) uniform metric over trace distributions. We propose simulation functions for proving approximate implementations corresponding to each of the above types of approximate implementation relations. Since our notion of similarity of traces is based on a metric on trace distributions, we do not require the state spaces nor the space of external actions of the automata to be metric spaces. We discuss applications of approximate implementations to verification of probabilistic safety and termination.

Keywords: Approximate implementation, equivalence, Approximate simulation, Abstraction, Probabilistic I/O Automata.

1 Introduction

Implementation relations play a fundamental role in the study of complex interacting systems because they allow us to prove that a given concrete system implements an abstract specification. Formally, an automaton is said to implement another automaton if the set of traces or the observable behavior of the first is subsumed by that of the latter. Many different kinds of implementation or abstraction relations and their corresponding proof methods have been developed for timed [1], hybrid [17,30,29] and probabilistic automata [19,20,5,2,28,4].

These traditional notions of implementation rely on equality of traces. That is, every trace of the concrete system must be exactly equal to some trace of the

[1] Email: mitras@theory.csail.mit.edu
[2] Email: lynch@theory.csail.mit.edu

1571-0661
doi:10.1016/j.entcs.2006.11.040

abstract specification. It is well known from [16,10,15] that such strict equality based implementation relations are not robust. Small perturbations to the parameters of the system produces traces with slightly different numbers (representing say, timing or probability information), and thus breaks the equality between traces. One way to overcome this problem is to relax the notion of implementation by taking into consideration the "similarity" of traces that are not exactly equal. In [16] Jou and Smolka formalized "similarity" of traces using a metric and developed the corresponding notion of approximate equivalence for probabilistic automata. Based on similar ideas, there is now a growing body of work on developing robust notions of approximate implementations; in Section 1.1, we briefly describe previous contributions in this area that are related to our work. Apart from providing robust implementation relations, notions of approximate implementation also enable us to create abstract models without introducing extra nondeterminism.

In this paper we introduce the notion of approximate implementations for the *Probabilistic Input/Output Automaton (PIOA)* [27,6] and develop simulation based methods for proving such relationships. A PIOA is a nondeterministic automaton with a countable state space. Transitions are labelled by *actions*. Many transitions may be possible from a given state. Each transition gives a discrete probability distribution over the state space. We use a *task structure* [5]—an equivalence relation on the set of locally controlled actions—as a means for restricting the nondeterminism in a PIOA. The resulting automaton model is called *task-PIOA*. A task-PIOA interacts with a *task scheduler* to give rise to a probability distribution over its executions. For every such distribution there exists a corresponding distribution over its set of traces, which is called a *trace distribution*. Visible behavior of a task-PIOA is the set of *trace distributions* that it can produce. A task-PIOA is said to (exactly) implement another task-PIOA if the set of trace distributions of the first is a subset of the trace distributions of the latter. Implementations, simulation relations for proving implementations, and compositionality results for task-PIOAs are presented in [5]. A special kind of approximate implementation relation that tolerates small differences in the probability of occurrence of a particular action is used in [6] to verify a security protocol. In contrast, the notions of approximation introduced here are more general because they are based on metrics on trace distributions. We define two kinds of approximate implementations of task-PIOAs: (1) *uniform approximate implementation* is based on the uniform metric of trace distributions [23], and (2) *discounted approximate implementation* is based on the discounted uniform metric.

A PTIOA $\mathcal{A}$ is a δ-approximate implementation of another PTIOA $\mathcal{B}$, for a positive δ, if the for any trace distribution of $\mathcal{A}$, there exists a trace distribution of $\mathcal{B}$ such that their discrepancy over any measurable set of traces is at most δ. We present *Expanded Approximate Simulations (EAS)* for proving uniform approximate implementations. EAS is a natural generalization of the simulation relation presented in [6]. Let μ_1 and μ_2 be probability distributions over executions of task-PIOAs $\mathcal{A}$ and $\mathcal{B}$. An EAS from $\mathcal{A}$ to $\mathcal{B}$ is a function ϕ mapping each μ_1, μ_2 pair to a nonnegative real. The number $\phi(\mu_1, \mu_2)$, is a measure of how similar μ_1 and μ_2 are