Compositional Reasoning for Probabilistic Finite-State Behaviors

Yuxin Deng[1,⋆], Catuscia Palamidessi[2,⋆⋆], and Jun Pang[2]

[1] INRIA Sophia-Antipolis and Université Paris 7, France
[2] INRIA Futurs and LIX, École Polytechnique, France

Abstract. We study a process algebra which combines both nondeterministic and probabilistic behavior in the style of Segala and Lynch's simple probabilistic automata. We consider strong bisimulation and observational equivalence, and provide complete axiomatizations for a language that includes parallel composition and (guarded) recursion. The presence of the parallel composition introduces various technical difficulties and some restrictions are necessary in order to achieve complete axiomatizations.

1 Introduction

Process algebras, also known as process calculi, are a powerful mathematical model for the specification and verification of concurrent systems. They provide a formal apparatus for representing and reasoning about the behaviors of distributed systems, algorithms and protocols in a compositional way. Some of the most prominent representants of these formalisms are CCS [27], ACP [8, 6], and CSP [21].

The axiomatic theories of process algebra provide an elegant way for proving properties of systems. Both a system and its desired external behavior can be expressed as process terms. The correctness of the system can then be verified by proving that these two terms are equivalent.

In a process algebra typically there are only a few operators, such as action prefix, summation (nondeterministic choice), recursion and parallel composition. The latter is particularly important for concurrency, as it allows to specify the structural properties of systems composed of several interacting parts. For example, a typical communication protocol for data transferring involves two agents S and R, representing the sender and the receiver, and two lossy channels K and L between them (see Figure 1). The behavior of each of these four components can be described as a process term in a chosen process algebra, and then they are all put together in parallel to form the complete view of the protocol. The parallel composition operator captures both the interleaving behaviors and the possible synchronization of the components. The external behavior of the

⋆ Supported by the EU project PROFUNDIS.
⋆⋆ Partially supported by the projet Rossignol of the ACI Sécurité Informatique (Ministère de la recherche et nouvelles technologies).

A. Middeldorp et al. (Eds.): Processes... (Klop Festschrift), LNCS 3838, pp. 309–337, 2005.

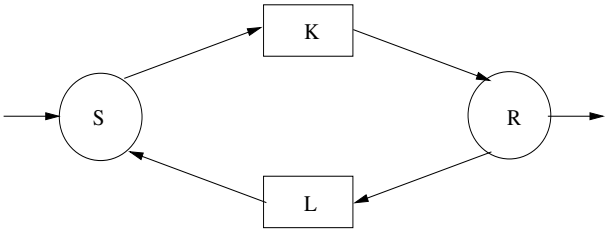

Fig. 1. A communication protocol

protocol can be specified as a FIFO queue. The equivalence proof between the protocol and its external behavior is established by equational reasoning based on axiomatization, hiding internal behavior, using fairness assumption, and the other feasible methods (see e.g. [9, 17]).

Developing a both complete and sound axiomatization for a chosen bisimulation relation over a process algebra expressing finite-state processes has been a research focus for the process algebra community. This led to a wealth of classical results in the literature. Milner [26, 28] gave complete axiomatizations of both strong bisimilarity and observational equivalence for a core CCS (not containing the parallel composition operator) with both unguarded and guarded recursion. Bergstra and Klop [10] axiomatized observational equivalence in an alternative way by using an interesting graph rewriting technique. Hennessy and Milner [20] offered a complete equational axiomatization of strong bisimulation over the recursion free fragment of CCS. To deal with parallel composition, they used the so-called *expansion law*, which is an equation schema with a countably infinite number of instances. Bergstra and Klop [8] gave a finite equational axiomatization of the merge operator (as the parallel composition in CCS) using the auxiliary left merge and communication merge operators. An interesting essay on equational axiomatizations of parallel composition can be found in [2].

Having both recursion and parallel composition in a process algebra complicates the matters to establish a complete axiomatization, mostly because this can give rise to infinite-state systems even with the guardedness condition. For example, let E be the expression $\mu_X(a.(X \mid b))$, then we have the infinite transition graph starting from E in Figure 2. Milner pointed out in [28] that in order to have a complete axiomatization for CCS with both recursion and parallel composition, a sufficient condition is that the parallel composition does not occur in the body of any recursive expression.

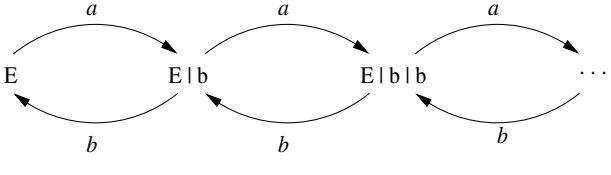

Fig. 2. The transition graph of E