Automata and Logic for Concurrent Systems

Benedikt Bollig

Laboratoire Spécification et Vérification, École Normale Supérieure de Cachan & Centre National de la Recherche Scientifique, France

Automata are a popular model of computer systems, making them accessible to formal methods and, in particular, synthesis and model checking. While classical finite-state automata are suitable to model *sequential* boolean programs, models of *concurrent* systems, involving several interacting processes, extend finite-state machines in several respects. Roughly, we may classify a system (or a system model) according to the following characteristics:

Form of Communication. Inter-process communication may be achieved, e.g., via shared variables or message passing. While boolean shared-variable programs usually give rise to finite-state systems so that classical methods are applicable for their analysis, message passing via a priori unbounded channels leads to undecidability of basic verification questions. However, putting some restrictions on the system (e.g., imposing a channel bound or restricting the system architecture) will allow us to infer positive results for both system synthesis and model checking.

System Architecture. The system architecture, connecting processes and arranging them in a certain way (e.g., as a pipeline or as a tree), may be static and known, or static but unknown, or it may change dynamically during a system execution. In the particular case where the topology is static but unknown, we deal with a parameterized setting. So, one will be interested in questions such as "Is the system correct no matter what the system architecture or the number of processes is?" or "Can one transform a specification into a system that is equivalent to the specification over all system architectures?". There has been a wide range of techniques for the verification of parameterized and dynamic systems. In the dynamic case, there are also close connections with the theory of words over infinite alphabets, where the alphabet may represent an unbounded supply of process identifiers.

Finite-State vs. Recursive Processes. Processes themselves can have finite state space (i.e., be modeled as finite-state automata) or recursive (i.e., be modeled as pushdown automata). Like processes communicating via message passing through unbounded channels, shared-variable recursive processes have an undecidable control-state reachability problem. However, under- and overapproximating the behavior of a system will still allow us to check certain system requirements.

In this talk, we survey automata models for some combinations of the above-mentioned features. We also present suitable specification formalisms (such as monadic second-order logic, temporal logic, and high-level expressions). In particular, we will compare the expressive power of automata and logic, give translations of specifications into automata, and show, for some cases, how to solve the model-checking problem: “Does a given automaton satisfy its specification?”.