Towards efficient model checking Statecharts: A Statecharts to Promela Complier

Erich Mikk, Yassine Lakhnech, Michael Siegel[1]

Harel proposes the graphical language statecharts [Har87] for specifying behavior of reactive systems. Statecharts extend Mealy machines by parallelism and hierarchy. The communication mechanism in statecharts is instantaneous broadcast and the concurrency model is maximal parallelism.

Statecharts has been supported by the commercial STATEMATE tool for about 12 years now. There, statecharts are used to describe the behavior of the system and are complemented with other languages for describing the system architecture and functionality (behavioral, structural and functional view [HP96].) STATEMATE supports simulation of statecharts specifications, provides a so called dynamic test tool for checking safety properties and code generation facilities from specifications. The semantics of statecharts as implemented in STATEMATE is described in [HN]. Our research deals with this version of statecharts.

According to the STATEMATE MAGNUM manual (the most recent version of STATEMATE), the dynamic test tool uses explicit state space enumeration and implements a depth-first search of the state space. The manual does not report any state space reduction techniques implemented in the tool. Our research concerns a compiler from statecharts to Promela/SPIN in order to exploit state space reduction techniques implemented in SPIN (e.g., partial order reduction [HP94]) for statecharts.

We are interested in verifiable correctness of tools that we are going to develop for statecharts. Therefore we must have a formal syntax and semantics definition. We formalized the informal description [HN] in [MLPS97]. In [MLPS97] a transition system is associated to each statechart.

In terms of compiler construction our semantics definition [MLPS97] may be seen as a compiling specification where the source is a statechart specification and the target is a finite state machine. Unfortunately this is not suitable for compiling statecharts to Promela/SPIN for efficient model checking. Rather, we want to build a compiler that preserves the parallelism of a statechart during the compilation process. In order to establish compiling specification between statecharts and Promela in a syntax-driven way we must have a structural semantics definition for statecharts. Additionally, the structural approach seems to be inevitable for using state space reduction techniques (this applies for partial order reduction of SPIN but also to symbolic model checking using BDDs). The pay-off of these techniques comes from the fact that they often lead to a compact representation of object compositions.

For the definition of structural semantics we use an inference-rules based style as proposed by Plotkin [Plo81] (SOS style). Unfortunately, so called inter-level transitions in statecharts spoil the SOS-style semantics for statecharts. Structural decomposition of a statechart with inter-level transitions along the state borders possibly destroys the syntactical well-formedness of the original chart: dangling transitions would be encountered. We propose the notion of hierarchical automata (HA) to remedy this situation. In a first step a statecharts is transformed to HA by eliminating inter-level transitions while preserving the semantics of the statecharts. Then, a SOS-style semantics is defined for the class of HA , which serves as our starting point for tool construction.

An intriguing problem is now how to implement the parallel construct of statecharts (AND composition of states) in Promela. The semantics of AND is characterized by the following two observations:

[1]Christian-Albrechts-Universität zu Kiel, Institut für Informatik und Praktische Mathematik, Preußerstr. 1-9, D24105 Kiel, Germany, e-mail: {erm,yl,mis}@informatik.uni-kiel.de

- A maximal non-conflicting set of enabled transitions (they are pairwise orthogonal) is executed in one step.
- There is no communication during such a step: events generated by transitions are sensed in the next step only.

The question is now how to translate the stated pattern of parallel execution into Promela, which uses an interleaving model of parallelism. There are two choices:

- Transitions, formerly locating in parallel statecharts, are now executed in an interleaved fashion: AND-states of statecharts are mapped to Promela processes. This solution requires scheduler to implement the "lock-step" of statecharts.
- Arbitrary interleaving of parallel transitions is determined at the compile-time yielding a fully sequential implementation. This can be done since these transitions do not communicate with each other.

In the parallel solution different interleavings lead to the same result. Hence we can expect that the partial order reduction techniques of SPIN are able to find it out and reduce the number of possible interleavings to one. Since it is know which types of Promela statements can be exploited by the reduction algorithm Holzmann and Peled encourage the protocol designer to choose the structure of the model appropriate ([HP94]).

On the other hand the sequential implementation guarantees that exactly one interleaving is chosen. We are aware of the fact that with the sequential solution the current partial order reduction algorithm does not apply. Nethertheless the sequential solution is at least as good as the parallel one.

The compiler from statecharts to Promela based on these ideas has been implemented. The compiler can generate optionally parallel or sequential code.

References

[Har87] D. Harel. Statecharts: A Visual Formalism for Complex Systems. *Science of Computer Programming*, 8:231–274, 1987.

[HN] D. Harel and A. Naamad. The STATEMATE Semantics of Statecharts. *ACM Trans. Soft. Eng. Method., to appear (Also available as technical report of Weizmann Institute of Science, CS95-31).*

[HP94] G.J. Holzmann and Doron Peled. An improvement in formal verification. October 1994.

[HP96] D. Harel and M. Polity. *Modelling Reactive Systems with Statecharts: the Statemate Approach.* i-Logix, Inc, 1996.

[MLPS97] E. Mikk, Y. Lakhnech, C. Petersohn, and M. Siegel. On formal semantics of Statecharts as supported by STATEMATE. *Submitted for publication*, 97.

[Plo81] G.D. Plotkin. A structural approach to operational semantics. Technical report, University of Aarhus, 1981.