Code Generation and Model Driven Development for Constrained Embedded Sotware

Andrzej Wąsowski

Department of Innovation
IT University of Copenhagen

Abstract

We consider statechart models of discrete control embedded programs operating under severe memory constraints. There has been very few results in code generation for such systems. We analyze code generation methods based on runtime interpreters. We choose a suitable subset of hierarchical statecharts and engineer an efficient interpreter for it. An algorithm is provided that simplifies general models to our sublanguage removing *dynamic scoping* and *transition conflicts*. The resulting code generator improves over an industrial implementation provided by IAR A/S.

The interpreter for hierarchical statecharts is complex. We define *flattening* as a process of transforming hierarchical models into their hierarchy-less counterparts. We prove that even with a simulation-based correctness criterion any flattening algorithm would cause a *super polynomial growth of models*, if it does not exploit message passing. Then we devise a *polynomial flattening algorithm* based on internal asynchronous communication in the model. The implementation of this algorithm beats our earlier hierarchical code generator by 20–30% on realistic examples.

In the second part of the thesis we develop a unified theory for specifying correctness of model transformations and modeling software product lines. Our framework is based on a novel notion of *color-blindness*: a dynamically changing inability of the environment to observe differences in system outputs. Being safe approximations of all possible usage scenarios the environments can be used to specify specialized versions of the product. We propose a correctness criterion for specialization algorithms based on Larsen's relativized bisimulation extended with color-blindness.

Any good modeling formalism for software product lines supports composition and step-wise modeling, so that families can be organized in hierarchies or even more flexible structures. To serve this purpose we introduce an *information ordering* on our models of environments. Crucially, we show that the abstract information preorder can be *characterized operationally* by means of simulation. Then we use the information preorder to define intuitive *composition operators* as meets and joins in the respective quotient lattice. We demonstrate an extended example of a hierarchical family of alarm clocks, specified by means of color-blind environments.