Framework for Inter-Model Analysis of Cyber-Physical Systems

Ivan Ruchkin

Designing a successful cyber-physical system (CPS) like an autonomous car requires tacking its cyber and physical aspects: control stability, planning, schedulability, protocol correctness, thermal safety, and energy efficiency, to name a few. Formalization of these aspects relies upon various domains of engineering expertise for appropriate system models and analytic operations on these models. For example, signal-flow graphs and simulation are appropriate to analyze a controller's stability, while state machines and reachability analysis can be used to ensure protocol correctness. Thus, growing diversity of CPS applications involves new domains of expertise, which consequently bring in diverse models and analytic tools.

Any CPS engineering method faces the challenge of using multiple models and analyses together. First, proper model relationships are crucial to CPS design: missing important ones may create implicit inter-model conflicts, which cause failures, and considering too many relationships leads to low scalability and, ultimately, impracticality of rigorous design. For example, soundness of control modeling depends on network data exchange delays, but not necessarily on data structures. Second, analyses may deliver unsound results if their often-implicit assumptions are compromised. Such assumptions may concern complicated behaviors: for instance, the bin packing algorithm for thread-to-processor allocation is only applicable if the scheduling policy is equivalent to deadline-monotonic. So, major research questions of multi-model CPS design are: what parts of models should be related to other models, and how to do so effectively? How to specify and verify assumptions of analyses for these models? And finally, given many possible ways to relate models, how to evaluate each of these ways, in particular its usefulness and overhead?

One obstacle to answering these questions is profound heterogeneity of models and analyses: their notions of computation, time, and state may be very different (cf., timed automata vs. signal-flow models). Another obstacle is that people with different background comprehend and create models differently: one model aspect may at the same time seem important to a software programmer, self-evident to a control engineer, and irrelevant to a timing engineer. Yet another obstacle is that answers to the questions above are not universal: in some cases, battery lifetime analysis should take computing load and timing into account, while in other cases, when energy is abundant, this complexity is better avoided.

My research focuses on creating a CPS multi-modeling framework that facilitates creating and maintaining model and analysis relationships. The framework's core concepts are models, analyses, and architectural views. A model is a representation of a system part, expressed in a formalism specific to the model's domain of expertise. For example, a control model may be specified in Simulink, and a hardware model – in Verilog. An analysis is an operation that changes or uses models. The framework accepts specifications of analyses dependencies and

assumptions to ensure correct analysis application using existing models. An architectural view for a model is a typed graph of components and connectors that specifies the model's version of system's architecture. Thus, a relationship between models can be broken down into a relationship between each model and its view and a relationship between views. The semantics of this relationship – refinement, composition, or some other – can be given flexibly in terms of the existing models.

One research problem is the relationship between a model and its view. Direct transformation is not an acceptable solution: deep technical details in a model cannot be fully abstracted to a view – otherwise an analysis would not have enough information to be executable. Similarly, deriving a view fully from a model is not possible because models do not generally carry information about architectural types and patterns. My current research investigates usage of model annotations to maintain consistency between KeYmaera hybrid programs and their architectural descriptions. These annotations would promote program reuse and detect modeling mistakes such as using data about an obstacle that is not being sensed. However, the question of model-to-view relationships is open for other kinds of models.

Another research problem is specifying and verifying analysis assumptions. Some assumptions cannot be satisfied by the model on which the analysis runs: for example, selecting a reconfigurable battery scheduler based on battery life simulations assumes that the scheduler does not trigger a chain heat reaction in battery cells known as thermal runaway. To discharge this assumption, I use a Promela model of a battery scheduler to explore possible cell interconnections and detect if connection patterns leading to thermal runaway are reached. The next step is creating a mechanism to match analysis assumptions to models that can discharge them.

My future work includes developing a method for evaluating the engineering value of formal relationships between models. A hypothesis is that this value highly depends on how likely the engineers are to miss or misunderstand the relationship because of their expertise boundaries. I plan on using behavioral research methods like interviews and experiments to test and refine this hypothesis.

I would like to present my research during the plenary session.