Incrementally Synthesizing Controllers from Scenario-Based Product Line Specifications (Extended Abstract)

Joel Greenyer[1], Christian Brenner[2], Maxime Cordy[3], Patrick Heymans[3], Erika Gressi[4]
[1]Software Engineering Group, Leibniz Universitt Hannover, Germany
greenyer@inf.uni-hannover.de
[2]Software Engineering Group, Heinz Nixdorf Institute, University of Paderborn, Germany
cbr@uni-paderborn.de
[3]PReCISE Research Center, University of Namur, Belgium
{mcr|phe}@info.fundp.ac.be
[4]DEEPSE Group, DEIB, Politecnico di Milano, Italy
erika.gressi@mail.polimi.it

Abstract: Many software-intensive systems consist of components that interact to fulfill complex functionality. Moreover, often many variants of such systems have to be designed at once. This adds complexity to the design task. Recently, we proposed a scenario-based approach to specify product lines, which combines feature diagrams and Modal Sequence Diagrams. We propose a new game-based technique for checking which product specifications are realizable and which ones are not, due to inconsistencies in the specification. In addition, our technique automatically synthesizes controllers for the realizable product specifications. To increase efficiency, we exploit the fact that many product variants are similar and synthesize product controllers incrementally. We provide a prototype tool and evaluate the efficiency of the approach.

Nowadays, software is part of a wide variety of safety-critical systems, including cars, planes, and medical devices, which typically consist of many components that provide functionality through their interaction. These interactions must often satisfy complex specifications, and the failure to comply may have tragic consequences.

Moreover, often today engineers do not only develop a single system, but rather build several *variants* (also called *products*) of the same system—a *Product Line* (PL). The variability is commonly organized in terms of *features*, which are functions or components that may or may not be present in a given variant. As more features are added to the PL, the number of possible variants grows exponentially, in the worst case.

Recently, we proposed to specify PLs using feature diagrams (FDs) to model variability, and Modal Sequence Diagrams (MSDs) to specify the behavior of features. The use of MSDs has several advantages. First, it is an intuitive, yet precise visual formalism that allows engineers to specify what may, must, or must not happen in a system. This fits most development processes that propose a scenario-based approach during the early design. Second, with MSDs engineers can describe behavioral aspects of features that can extend or constrain the behavior specified for other features. Third, product specifications can be easily composed by forming the union of the MSDs of its constituent features.

A specification of allowed and forbidden behavior, however, can be *unrealizable*, i.e., it may be impossible to build a product that is able to react to all possible sequences of environment events in a way that its specification is satisfied. This can require costly iterations or in the end many desired products may be unrealizable or intrinsically flawed.

In this paper [GBC+13], we propose a new realizability-checking technique. The technique is based on an *on-the-fly game-solving algorithm* for *synthesizing controllers* from the MSD specifications of each product in a PL. This algorithm will not only compute a yes/no answer; instead the synthesis algorithm will produce for each realizable product specification a *controller* that implements this specification. For unrealizable product specification, it will produce a *counter-strategy* that shows how the environment can always force the system to violate the specification. The algorithm operates on a *game graph* that is induced by the MSD specification and represents the possible interactions of the environment and the system. Being on-the-fly, the game-solving algorithm can often find a controller or counter-strategy before the complete game graph is explored. This avoids parts of the effort of constructing the game graph and exploring/backtracking.

We apply the synthesis *incrementally* and *specifically optimized for PLs described by FDs*. Typically many products in a PL are very similar, i.e., they share many common features and consequently share many common MSDs. Therefore, if a controller for one product could be successfully synthesized, we can more efficiently, based on this controller, synthesize a controller for a similar product. The efficiency gain can obtained because, by looking at a controller for a similar product, the on-the-fly synthesis algorithm can follow actions that led to a successful execution w.r.t. a similar specification. There exists a chance that these actions will lead to a behavior that satisfies also the MSD specification for the current product. This way, the synthesis algorithm can often more quickly find an admissible behavior and avoid building and exploring large parts of the game graph.

Our technique not only supports MSD specification that formalize requirements for the system; MSD specifications can also describe environment assumptions. We implemented this technique in ScenarioTools, an Eclipse/UML-based modeling, simulation, and synthesis tool suite for MSD specifications. Evaluations show that the incremental variant of the algorithm could outperform the non-incremental variant in many cases. Altogether, we pave the way for the intuitive and rigorous design of PLs of reactive systems.

In ongoing work, we are exploring an approach where, instead of synthesizing controllers in a sequential-incremental way, we perform the synthesis for all products at once [Gre14].

References

[GBC+13] Joel Greenyer, Christian Brenner, Maxime Cordy, Patrick Heymans, and Erika Gressi. Incrementally Synthesizing Controllers from Scenario-based Product Line Specifications. In *Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering*, ESEC/FSE 2013, pages 433–443, New York, NY, USA, 2013. ACM.

[Gre14] Erika Gressi. Featured Synthesis of Scenario-Based Software Product Line Specifications. Master's thesis, Politecnico di Milano, 2014.