Learning From Families: Inferring Behavioral Variability From Software Product Lines

Carlos Diego Nascimento Damasceno

University of Sao Paulo (BR) and University of Leicester (UK)
damascenodiego@usp.br

Abstract

Family models are behavioral specifications extended with variability constraints that enable efficient model-based analysis of software product lines (SPL). Albeit reasonably efficient, the creation and maintenance of family models are time-consuming and error-prone, especially if there are large models or crosscutting features. In this PhD project, we investigate the problem of learning family models from SPLs. Our initial contributions are two-fold: (1) `partial-Dynamic` $\mathtt{L}^*_{\mathtt{M}}$, a novel adaptive algorithm to speed up automata learning by exploring models from alternative software versions *on-the-fly*; and (2) $\mathtt{FFSM}_{\mathtt{Diff}}$, a fully automated technique to learn family models by comparing, merging and annotating finite state machines with variability constraints. Our experiments have shown that our techniques are more efficient than the state-of-the-art of adaptive learning in terms of queries and that succinct family models with fewer states can be learnt, especially if there is high feature reuse. We envisage that our studies can leverage model-based techniques to cases where models are non-existent or outdated and will scale better than independently exploring several versions of evolving systems or product models from configurable systems.

1 Introduction

The modeling and analysis of software product lines (SPL) are known to be challenging; they should incorporate variability to express product-specific behavior to avoid/minimize redundant computations of shared assets and cater for feature interactions [8]. Thus, substantial effort has been spent for developing analysis techniques specifically tailored to product families.

Family-based analysis operates on a single artifact, referred to as *family model*, that is annotated with variability constraints to express variability in terms of states and transitions specific to product configurations. This modeling approach paves the way for efficient model-based testing and verification of SPLs. Nevertheless, the creation and maintenance of test models are known to be time consuming and error-prone, especially if there are large SPLs or crosscutting features; and the traceability between the family- and variability models can be complex due to crosscutting features [3]. Added to this, as requirements change and product instances evolve, the lack of maintenance may render models outdated [2]. To tackle these issues, we proposed this PhD project to investigate how automata model learning [9] can be lifted to the family-based level to support the extraction of family models from SPLs.

Model learning has been a popular approach to automatically derive behavioral models from a system under learning (SUL) by posing tests as queries, i.e., *transfer* and *separating* sequences, to reach and distinguish states [9]. It has been harnessed for a wide range of problems [1]. However, there is a lack of studies about how to cope with variability in time and space [6].

2 Approach

Applying model learning to real systems can be hampered by constant changes along their life-cycle [5], as it may require *learning from scratch*. Adaptive learning attempts to speed up

learning by reusing knowledge (i.e., sequences) from alternative/previous versions of a SUL. Studies have shown that reusing sequences from pre-existing models can reduce the cost for learning models from updates. However, after several changes, old separating and transfer sequences may render *redundant* and *deprecated* queries, respectively [2].

2.1 Learning to Reuse: Adaptive Learning for Evolving Systems [2]

We improve upon the state-of-the-art by introducing `partial-Dynamic` $\mathtt{L}^*_\mathtt{M}$ ($\partial\mathtt{L}^*_\mathtt{M}$), an adaptive algorithm that runs an *on-the-fly* exploration of reused models to avoid irrelevant queries [2].

To achieve this, our algorithm explores *transfer sequences* to find redundancy. Then, given a subset of "useful" transfer sequences, we designed an optimization technique to find deprecated *separating sequences* and hence, the smallest subset with equivalent separating capability, named *experiment cover*. These subsets of transfer and separating sequences initialize the $\mathtt{L}^*_\mathtt{M}$ algorithm for learning Mealy machines [7]. Our experiments showed that $\partial\mathtt{L}^*_\mathtt{M}$ is less sensitive to evolution and more efficient (i.e., requires less queries) than the state-of-the-art for adaptive learning. The paper has been published at the iFM'19 [2].

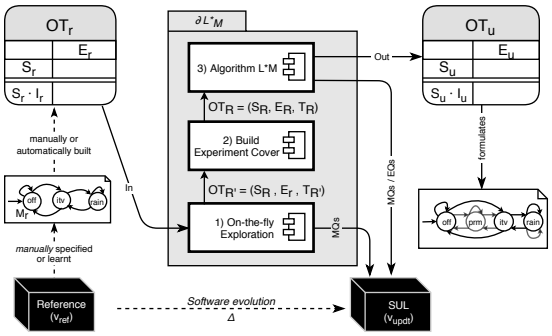

Figure 1: `partial-Dynamic` $\mathtt{L}^*_\mathtt{M}$

2.2 Learning from Difference: An Automated Approach for Learning Family Models [3]

Within SPLs, similar challenges may emerge and hamper family-based testing and verification. Families of software products share a common and managed set of features and hence, their behavior tend to have commonalities and variabilities [8]. Thus, model learning for SPLs should avoid redundant effort, and at the same time cater for variability and feature interaction.

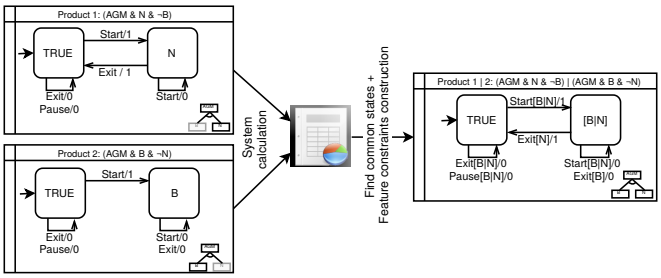

Figure 2: $FFSM_{Diff}$

We have designed $FFSM_{Diff}$, an automated technique to learn family models by comparing, merging and annotating product models. Our technique is presented in terms of featured finite state machines (FFSM) [4], a family-based notation that unifies Mealy Machines from SPLs by annotating states and transitions with variability constraints. A schematic representation of our approach is depicted in Figure 2.

Our technique allows to (i) learn succinct FFSMs from two product models, and (ii) include novel product-specific behavior into an existing FFSM. Our results support the hypothesis that family models can be effectively merged into succinct FFSMs with fewer states, especially if there is high feature sharing among products. These findings indicate that $FFSM_{Diff}$ paves the way for family model learning techniques, which are still understudied; and efficient family-based analysis, even if there are no models specified *a priori*. The full paper has been published at the 23rd International Systems and Software Product Line Conference (SPLC'19) [3].

3 Final Remarks and Next Steps

Real systems pass through many changes along their life-cycle and, as we often do not know how states may have changed, their models tend to become outdated. To deal with these issues, we have designed two techniques for learning models from evolving systems [2] and product families [3]. Our techniques improve upon the state-of-the-art and are complementary to each other in the sense that they pave the way for an *active family model learning* framework.

As the next step of this PhD project, we will propose the concept of active family model learning. In Figure 3, we show our vision of active family model learning. Our vision of *active family model learning* stands for a framework where SPLs can have their family models harvested by re-using *partial family models*, i.e., family models describing subsets of valid product instances from SPLs, to steer an active model learning process [9]. We envisage that such variability-aware model learning framework will scale better than exhaustively and independently applying adaptive learning to product instances or software releases.

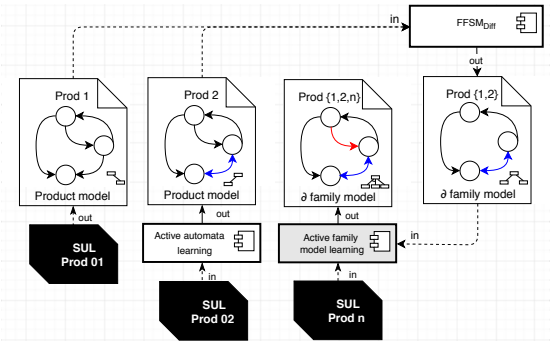

Figure 3: Active family model learning

References

[1] Bernhard K. Aichernig, Wojciech Mostowski, Mohammad Reza Mousavi, Martin Tappler, and Masoumeh Taromirad. Model learning and model-based testing. In Amel Bennaceur, Reiner Hähnle, and Karl Meinke, editors, *Machine Learning for Dynamic Software Analysis: Potentials and Limits: International Dagstuhl Seminar 16172*, pages 74–100, Cham, 2018. Springer.

[2] Carlos Diego N. Damasceno, Mohammad Reza Mousavi, and Adenilso Simao. Learning to reuse: Adaptive model learning for evolving systems. In *Integrated Formal Methods - 15th International Conference, IFM 2019, Bergen, Norway, December 2-6, 2019, Proceedings*. Springer, 2019.

[3] Carlos Diego Nascimento Damasceno, Mohammad Reza Mousavi, and Adenilso da Silva Simao. Learning from difference: An automated approach for learning family models from software product lines. In *Proceeedings of the 23rd International Systems and Software Product Line Conference - Volume 1, SPLC 2019*, Paris, France, 2019. ACM Press.

[4] Vanderson Hafemann Fragal, Adenilso Simao, and Mohammad Reza Mousavi. *Validated Test Models for Software Product Lines: Featured Finite State Machines*, pages 210–227. Springer International Publishing, Cham, 2017.

[5] David Huistra, Jeroen Meijer, and Jaco van de Pol. Adaptive learning for learn-based regression testing. In Falk Howar and Jiri Barnat, editors, *Formal Methods for Industrial Critical Systems*, Lecture Notes in Computer Science, pages 162–177, Switzerland, 9 2018. Springer.

[6] Klaus Pohl, Günter Böckle, and Frank J. van der Linden. *Software Product Line Engineering: Foundations, Principles and Techniques*. Springer-Verlag, Secaucus, NJ, USA, 2005.

[7] Muzammil Shahbaz and Roland Groz. Inferring mealy machines. In Ana Cavalcanti and Dennis R. Dams, editors, *FM 2009: Formal Methods*, pages 207–222, Berlin, Heidelberg, 2009. Springer.

[8] Thomas Thüm, Sven Apel, Christian Kästner, Ina Schaefer, and Gunter Saake. A classification and survey of analysis strategies for software product lines. *ACM Comput. Surv.*, 47, June 2014.

[9] Frits Vaandrager. Model learning. *Commun. ACM*, 60(2):86–95, January 2017.