Quantitative Variability Modeling and Analysis

Maurice H. ter Beek
ISTI–CNR, Pisa, Italy

Axel Legay
Université Catholique de Louvain, Belgium

ABSTRACT

The explicit management of variability in the development cycle of software-intensive systems has led to a plethora of modeling and analysis techniques tailored to deal with behavioral validation of such configurable systems. Most of the work, however, focuses on qualitative (i.e. functional) requirements. Recently, there is growing interest in variability modeling and analysis techniques that do explicitly consider quantitative (i.e. non-functional) requirements, such as dependability, energy consumption, security, and cost.

Today's software is embedded in a variety of smart and critical systems that run in environments where events occur randomly and affect the system, and to which it needs to adapt. Therefore, quantitative modeling and analysis is currently a hot topic. The panel on Quantitative Variability Modeling and Analysis (QSPL) discusses the latest quantitative techniques and how to apply them to variability modeling and analysis of software-intensive systems.

CCS CONCEPTS

• **Software and its engineering → Formal methods**; **Extra-functional properties**; **Software product lines**.

KEYWORDS

Variability, Quantitative modeling, Quantitative analysis, QSPL

ACM Reference Format:
Maurice H. ter Beek and Axel Legay. 2019. Quantitative Variability Modeling and Analysis. In *13th International Workshop on Variability Modelling of Software-Intensive Systems (VaMoS '19), February 6–8, 2019, Leuven, Belgium.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3302333.3302349

1 BACKGROUND AND MOTIVATION

Software Product Line Engineering (SPLE) concerns engineering, in a *cost*-effective and *time*-efficient manner, *configurable* software-intensive systems. The key issue is to manage *variability* among the products (or variants) of a Software Product Line (SPL), typically expressed in terms of *features*. Products share a set of core features, but differ with respect to product-specific features, meaning that a product can be seen as a set of features. A set of products (or family) is defined by a *feature model*.

The explicit management of variability in software development causes complexity in SPL *modeling and analysis*. An important example concerns *behavioral validation*, i.e., guaranteeing that each product of the family satisfies a series of behavioral requirements. Variants of this problem include computing a set of products that do not satisfy the requirements together with a justification. A common approach to assess such requirements consists of building the system and testing it. However, if the system should fail to meet the requirements, then costly interactions are needed to improve it. This problem is amplified in SPLE, where the number of products typically grows exponentially with the number of features. In fact, it is generally acknowledged that two major challenges with respect to behavioral validation of SPLs are i) to offer a compact and efficient way of modeling the behavior of families of products, and ii) to offer efficient analysis algorithms to exploit such models [44].

Throughout this decade, we witnessed numerous efforts on lifting well-known formal specification languages and formal verification techniques from (single system) software engineering to SPLE (or configurable systems), cf., e.g., [4–6, 9, 12, 19, 21–24, 29, 32, 45, 46] and the more exhaustive references in [7, 51]. However, the vast majority of work on variability modeling and analysis focuses on *qualitative* (i.e., functional) requirements. Only recently, we are witnessing a growing interest in variability modeling and analysis techniques that explicitly consider *quantitative* (i.e., non-functional) requirements, like dependability, which encompasses attributes like availability and reliability, but also energy consumption, security, and cost [8, 10, 11, 16, 17, 26–28, 30, 31, 35, 36, 39, 43, 47, 49].

Today's software is embedded in a wide variety of *smart* and *critical* systems (e.g., aircraft, railways, automotive, and medical devices) that run in environments where events occur *randomly* and affect the system (e.g., think of failures) and to which it needs to *adapt*. For these reasons, quantitative modeling and analysis (e.g., through probabilistic systems and probabilistic or statistical model checking) is nowadays receiving a lot of attention. In the specific setting of configurable software-intensive systems, this requires modeling and analysis techniques able to cope with the complexity of systems stemming from behavior, variability, and randomness.

In [7], we coined the term *Quantitative Variability Modeling and Analysis* (QSPL) with the aim to cover the following kind of topics:

- Quantitative specification and verification techniques for systems with variability;
- Modeling and analysis of real-time, hybrid or probabilistic systems with variability;
- Analysis of safety, security or dependability properties of systems with variability;
- Modeling and analysis of dynamic, adaptive and (runtime) reconfigurable systems.

Many open questions remain regarding the gap between research on verification and that on variability, in particular involving quantities. Hence, the time is ripe for a QSPL panel to address issues such as:

- How to relate quantities with variability?
- How to incorporate quantities in (textual) languages for variability modeling? (cf. [1, 18, 42])
- How do quantities influence the (space-time) trade-off between product-based and family-based analyses (cf. [47]) and can (qualitative) abstractions be of help? (cf. [3, 13, 25])
- How about the robustness and sensitivity of quantitative variability modeling and analysis? (cf. [15, 37])

VAMOS '19, February 6–8, 2019, Leuven, Belgium

ACM ISBN 978-1-4503-6648-9/19/02.
https://doi.org/10.1145/3302333.3302349

2 PANELISTS

The following gender-balanced set of panelists, most of whom never participated at VaMoS before, guarantees interesting discussions with our community on how to handle quantitative notions in SPLE.

Christel Baier (TU Dresden, Germany) is an expert in probabilistic model checking and quantitative analysis of stochastic systems and she has recently started applying this to SPLs [16, 17, 30, 31, 40].

Maxime Cordy (U Luxembourg) is an expert on behavioral modeling and analysis of SPLs, in particular based on featured transition systems, with affinity to quantitative approaches [19–21, 24–28, 49].

Uli Fahrenberg (École polytechnique, France) is an expert in quantitative extensions of transition systems, in particular weighted transition systems, and quantitative logics, including recent applications to behavioral SPL models [2, 14, 34–36, 47].

Mariëlle Stoelinga (U Twente, Netherlands) is an expert on quantitative modeling and analysis, in particular based on probabilistic, stochastic or timed automata models, and quantitative logics, but she has not yet been active in SPLE [33, 38, 41, 48, 50].

REFERENCES

[1] M. Acher, P. Collet, P. Lahire, and R.B. France. 2013. FAMILIAR: A domain-specific language for large scale management of feature models. *Sci. Comput. Program.* 78, 6 (2013), 657–681.

[2] S.S. Bauer, U. Fahrenberg, L. Juhl, K.G. Larsen, A. Legay, and C.R. Thrane. 2013. Weighted modal transition systems. *Form. Method. Sys. Design* 42 (2013), 193–220.

[3] M.H. ter Beek and E.P. de Vink. 2014. Towards Modular Verification of Software Product Lines with mCRL2. In *Proc. ISoLA 2014 (LNCS)*, Vol. 8802. 368–385.

[4] M.H. ter Beek and E.P. de Vink. 2014. Using mCRL2 for the Analysis of Software Product Lines. In *Proc. FormaliSE@ICSE 2014*. ACM, 31–37.

[5] M.H. ter Beek, E.P. de Vink, and T.A.C. Willemse. 2017. Family-Based Model Checking with mCRL2. In *Proc. FASE 2017 (LNCS)*, Vol. 10202. 387–405.

[6] M.H. ter Beek, A. Fantechi, S. Gnesi, and F. Mazzanti. 2016. Modeling and analysing variability in product families: Model checking of modal transition systems with variability constraints. *J. Log. Algebr. Meth. Program.* 85, 2 (2016), 287–315.

[7] M.H. ter Beek and A. Legay (Eds.). 2019. Special Section: Quantitative Variability Modelling and Analysis. *LNCS Trans. Found. Mastering Change* 2 (2019).

[8] M.H. ter Beek, A. Legay, A. Lluch Lafuente, and A. Vandin. 2015. Statistical Analysis of Probabilistic Models of Software Product Lines with Quantitative Constraints. In *Proc. SPLC 2015*. ACM, 11–15.

[9] M.H. ter Beek, A. Legay, A. Lluch Lafuente, and A. Vandin. 2016. Statistical Model Checking for Product Lines. In *Proc. ISoLA 2016 (LNCS)*, Vol. 9952. 114–133.

[10] M.H. ter Beek, A. Legay, A. Lluch Lafuente, and A. Vandin. 2018. A framework for quantitative modeling and analysis of highly (re)configurable systems. *IEEE Trans. Softw. Eng.* (2018).

[11] M.H. ter Beek, A. Legay, A. Lluch Lafuente, and A. Vandin. 2018. QFLan: A Tool for the Quantitative Analysis of Highly Reconfigurable Systems. In *Proc. FM 2018 (LNCS)*, Vol. 10951. 329–337.

[12] M.H. ter Beek, F. Mazzanti, and A. Sulova. 2012. VMC: A Tool for Product Variability Analysis. In *Proc. FM 2012 (LNCS)*, Vol. 7436. 450–454.

[13] T. Belder, M.H. ter Beek, and E.P. de Vink. 2015. Coherent branching feature bisimulation. *Electron. Proc. Theor. Comput. Sci.* 220, 3 (2015), 14–30.

[14] P. Bouyer, U. Fahrenberg, K.G. Larsen, N. Markey, J. Ouaknine, and J. Worrell. 2018. Model checking real-time systems. In *Handbook of Model Checking*, E.M. Clarke, T.A. Henzinger, H. Veith, and R. Bloem (Eds.). Chapter 29, 1001–1046.

[15] R. Calinescu, M. Češka, S. Gerasimou, M. Kwiatkowska, and N. Paoletti. 2018. Efficient synthesis of robust models for stochastic systems. *J. Sys. Softw.* 143 (2018), 140–158.

[16] P. Chrszon, C. Dubslaff, S. Klüppelholz, and C. Baier. 2016. Family-Based Modeling and Analysis for Probabilistic Systems: Featuring ProFeat. In *Proc. FASE 2016 (LNCS)*, Vol. 9633. 287–304.

[17] P. Chrszon, C. Dubslaff, S. Klüppelholz, and C. Baier. 2018. ProFeat: feature-oriented engineering for family-based probabilistic model checking. *Form. Asp. Comp.* 30, 1 (2018), 45–75.

[18] A. Classen, Q. Boucher, and P. Heymans. 2011. A text-based approach to feature modelling: Syntax and semantics of TVL. *Sci. Comput. Program.* 76, 12 (2011), 1130–1143.

[19] A. Classen, M. Cordy, P. Heymans, A. Legay, and P.-Y. Schobbens. 2012. Model checking software product lines with SNIP. *Int. J. Softw. Tools Technol. Transf.* 14, 5 (2012), 589–612.

[20] A. Classen, M. Cordy, P. Heymans, A. Legay, and P.-Y. Schobbens. 2014. Formal semantics, modular specification, and symbolic verification of product-line behaviour. *Sci. Comput. Program.* 80, B (2014), 416–439.

[21] A. Classen, M. Cordy, P.-Y. Schobbens, P. Heymans, A. Legay, and J.-F. Raskin. 2013. Featured Transition Systems: Foundations for Verifying Variability-Intensive Systems and Their Application to LTL Model Checking. *IEEE Trans. Softw. Eng.* 39, 8 (2013), 1069–1089.

[22] A. Classen, P. Heymans, P.-Y. Schobbens, and A. Legay. 2011. Symbolic model checking of software product lines. In *Proc. ICSE 2011*. ACM, 321–330.

[23] A. Classen, P. Heymans, P.-Y. Schobbens, A. Legay, and J.-F. Raskin. 2010. Model Checking Lots of Systems: Efficient Verification of Temporal Properties in Software Product Lines. In *Proc. ICSE 2010*. ACM, 335–344.

[24] M. Cordy, A. Classen, P. Heymans, P.-Y. Schobbens, and A. Legay. 2013. ProVeLines: a product line of verifiers for software product lines. In *Proc. SPLC 2013*, Vol. 2. ACM, 141–146.

[25] M. Cordy, A. Classen, G. Perrouin, P.-Y. Schobbens, P. Heymans, and A. Legay. 2012. Simulation-Based Abstractions for Software Product-Line Model Checking. In *Proc. ICSE 2012*. IEEE, 672–682.

[26] M. Cordy and A. Legay. 2019. Verification and Abstraction of Real-Time Variability-Intensive Systems. *Trans. Found. Mastering Change* 2 (2019).

[27] M. Cordy, P.-Y. Schobbens, P. Heymans, and A. Legay. 2012. Behavioural Modelling and Verification of Real-Time Software Product Lines. In *Proc. SPLC 2012*. ACM, 66–75.

[28] M. Cordy, P.-Y. Schobbens, P. Heymans, and A. Legay. 2013. Beyond Boolean Product-Line Model Checking: Dealing with Feature Attributes and Multi-features. In *Proc. ICSE 2013*. IEEE, 472–481.

[29] A. Dimovski, A. Al-Sibahi, C. Brabrand, and A. Wąsowski. 2015. Family-Based Model Checking using Off-the-Shelf Model Checkers. In *Proc. SPLC 2015*. ACM, 397.

[30] C. Dubslaff, C. Baier, and S. Klüppelholz. 2015. Probabilistic Model Checking for Feature-Oriented Systems. In *Trans. AOSD XII*. LNCS, Vol. 8989. 180–220.

[31] C. Dubslaff, S. Klüppelholz, and C. Baier. 2014. Probabilistic Model Checking for Energy Analysis in Software Product Lines. In *Proc. MODULARITY 2014*. ACM, 169–180.

[32] M. Erwig and E. Walkingshaw. 2011. The Choice Calculus: A Representation for Software Variation. *ACM Trans. Softw. Eng. Methodol.* 21, 1 (2011).

[33] M. Faella, A. Legay, and M. Stoelinga. 2008. Model checking quantitative linear time logic. *Electron. Notes Theor. Comput. Sci.* 220, 3 (2008), 61–77.

[34] U. Fahrenberg and A. Legay. 2014. The quantitative linear-time–branching-time spectrum. *Theoret. Comput. Sci.* 538 (2014), 54–69.

[35] U. Fahrenberg and A. Legay. 2017. Featured Weighted Automata. In *Proc. FormaliSE@ICSE 2017*. IEEE, 51–57.

[36] U. Fahrenberg and A. Legay. 2019. Quantitative Properties of Featured Automata. *Trans. Found. Mastering Change* 2 (2019).

[37] A. Filieri, G. Tamburrelli, and C. Ghezzi. 2016. Supporting Self-Adaptation via Quantitative Verification and Sensitivity Analysis at Run Time. *IEEE Trans. Softw. Eng.* 42, 1 (2016), 75–99.

[38] M. Gerhold and M. Stoelinga. 2018. Model-based testing of probabilistic systems. *Form. Asp. Comp.* 30, 1 (2018), 77–106.

[39] C. Ghezzi and A. Molzam Sharifloo. 2013. Model-based verification of quantitative non-functional properties for software product lines. *Inform. Softw. Technol.* 55, 3 (2013), 508–524.

[40] L. Herrmann, M. Küttler, T. Stumpf, C. Baier, H. Härtig, and S. Klüppelholz. 2019. Configuration of Inter-Process Communication with Probabilistic Model Checking. *Trans. Found. Mastering Change* 2 (2019).

[41] T. Hune, J. Romijn, M. Stoelinga, and F. Vaandrager. 2002. Linear parametric model checking of timed automata. *J. Log. Algebr. Program.* 52–53 (2002), 183–220.

[42] P. Juodisius, A. Sarkar, R.R. Mukkamala, M. Antkiewicz, K. Czarnecki, and A. Wąsowski. 2019. Clafer: Lightweight Modeling of Structure, Behaviour, and Variability. *Programming* 3, 1 (2019), 657–681.

[43] M. Kowal, I. Schaefer, and M. Tribastone. 2014. Family-Based Performance Analysis of Variant-Rich Software Systems. In *Proc. FASE 2014 (LNCS)*, Vol. 8411. 94–108.

[44] A. Legay and G. Perrouin. 2017. On Quantitative Requirements for Product Lines. In *Proc. VaMoS 2017*. ACM, 2–4.

[45] M. Lochau, S. Mennicke, H. Baller, and L. Ribbeck. 2014. DeltaCCS: A Core Calculus for Behavioral Change. In *Proc. ISoLA 2014 (LNCS)*, Vol. 8802. 320–335.

[46] R. Muschevici, J. Proença, and D. Clarke. 2016. Feature Nets: behavioural modelling of software product lines. *Softw. Sys. Model.* 15, 4 (2016), 1181–1206.

[47] R. Olaechea, U. Fahrenberg, J.M. Atlee, and A. Legay. 2016. Long-term Average Cost in Featured Transition Systems. In *Proc. SPLC 2016*. ACM, 109–118.

[48] A. Remke and M. Stoelinga. 2014. *Stochastic Model Checking*. LNCS, Vol. 8453.

[49] G.N. Rodrigues, V. Alves, V. Nunes, A. Lanna, M. Cordy, P.-Y. Schobbens, A. Molzam Sharifloo, and A. Legay. 2015. Modeling and Verification for Probabilistic Properties in Software Product Lines. In *Proc. HASE 2015*. IEEE, 173–180.

[50] M. Stoelinga. 2002. An introduction to probabilistic automata. *Bulletin EATCS* 78 (2002), 176–198.

[51] T. Thüm, S. Apel, C. Kästner, I. Schaefer, and G. Saake. 2014. Classification and survey of analysis strategies for software product lines. *ACM Comp. Surv.* 47, 1 (2014).