Jan Jürjens, Kurt Schneider (Hrsg.): Software Engineering 2017,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2017 125

On Automated Anomaly Detection for Potentially Unbounded Cardinality-based Feature Models

Markus Weckesser[1], Malte Lochau[1], Thomas Schnabel[1], Björn Richerzhagen[2], Andy Schürr[1]

Abstract: In this work, we report about our research results on analysis of cardinality-based feature models with potentially unbounded feature multiplicities, initially published in [We16]. Feature models are frequently used for specifying variability of user-configurable software systems, e.g., software product lines. Numerous approaches have been developed for automating feature model validation concerning constraint consistency and absence of anomalies. As a crucial extension to feature models, cardinality annotations allow for multiple, and even potentially unbounded occurrences of feature instances within configurations. This is of particular relevance for user-adjustable application resources as prevalent, e.g., in cloud-based systems where not only the type, but also the amount of available resources is explicitly configurable. However, a precise semantic characterization and tool support for automated and scalable validation of cardinality-based feature models is still an open issue. We present a comprehensive formalization of cardinality-based feature models with potentially unbounded feature multiplicities. We apply a combination of ILP and SMT solvers to automate consistency checking and anomaly detection, including novel anomalies, e.g., interval gaps. Furthermore, we show evaluation results gained from our tool implementation showing applicability and scalability of our approach to larger-scale models.

Keywords: Software Product Lines, Cloud-based Systems, Cardinality-based Feature Models, Integer Linear Programming (ILP)

1 Summary

Feature models become more and more established to specify *variability* of highly-configurable software, e.g., software product lines [CHE05]. During domain engineering, feature models are used to capture valid configuration spaces of product lines, by means of configuration parameters which are denoted as *features*, and *constraints* further restricting combinations of parameters to constitute valid configurations. A feature represents a user-visible (Boolean) configuration option from the problem domain. In addition, features are mapped onto variable implementation artifacts within the solution space to derive customer tailored products from a common code base during application engineering. The FODA feature diagram notation is a frequently used graphical representation for feature models [Ka90] in which features are depicted as nodes and organized in a tree-like layout

[1] Technische Universität Darmstadt, FG Echtzeitsysteme, Merckstr. 25, 64283 Darmstadt,
Email: {markus.weckesser,lochau,schuerr}@es.tu-darmstadt.de
[2] Technische Universität Darmstadt, FG Multimedia Kommunikation, Rundeturmstr. 10, 64283 Darmstadt,
Email: bjoern.richerzhagen@kom.tu-darmstadt.de

Acknowledgements: This work was partially supported by the DFG (German Research Foundation) as part of projects B01 and C02 within CRC 1053 – MAKI and under SPP 1593: Design For Future – Managed Software Evolution.

to denote a parent-child hierarchy. The feature tree is enriched with constraints to describe logical dependencies among features. A well established approach to formalizing feature model semantics uses a transformation into equivalent constraint satisfaction problems. This allows for applying off-the-shelf constraint-solvers for automatically validating desired properties such as constraint consistency and absence of anomalies.

FODA feature diagram notation is, in many cases, not expressive enough for capturing all user-configurable properties of real-world applications. As a crucial extension to feature models, *Cardinality-based Feature Models* (CFM) [CHE05] with UML-like feature multiplicities in terms of cardinality annotations allow the selection of multiple feature instances including clones of their corresponding sub-tree. As a result, the required extension complicates feature model semantics and makes consistency checking and anomaly detection crucial in practice. For consistency checking and anomaly detection of CFMs, only preliminary attempts exist so far, although cardinality-based variability modeling is emerging in nowadays applications. As a prominent example, for cloud-based systems, not only the *type*, but also the *amount* of available resources is explicitly configurable by the user, especially including (virtually) *unrestricted* resources. The resulting *compound* cardinality intervals lead to novel kinds of anomalies which yield information about possible flaws in variablity specifications. The main contributions of [We16] are a comprehensive formalization and a automated validation technique for CFMs. Our approach is motivated by a real-world cloud-based application. We support cardinality annotations including compound cardinality intervals and unbounded cardinality for singleton features, feature groups, as well as cross-tree constraints. We further introduce a *normal form* for cardinality constraints and enhance established notions of feature model consistency and anomaly to explicitly take feature cardinality constraints into account. Our tool implementation, presented in full detail in [Sc16], combines solving of problem formulations based on *Integer Linear Programming* (ILP) for interval-bound analysis and SMT solvers for interval-gap analysis to automate validation of CFMs. We have evaluated the applicability and demonstrated scalability of our validation approach for input models of varying sizes and complexity. For future work we plan to extend our automated validation technique to support analysis for mapping of feature instances to solution space artifacts.

References

[CHE05] Czarnecki, Krzysztof; Helsen, Simon; Eisenecker, Ulrich W.: Formalizing Cardinality-based Feature Models and Their Specialization. Software Process: Improvement and Practice, 10(1):7–29, 2005.

[Ka90] Kang, Kyo C.; Cohen, Sholom G.; Hess, James A.; Novak, William E.; Peterson, Spencer A.: Feature Oriented Domain Analysis (FODA). Technical report, CMU, 1990.

[Sc16] Schnabel, Thomas; Weckesser, Markus; Kluge, Roland; Lochau, Malte; Schürr, Andy: CardyGAn: Tool Support for Cardinality-based Feature Models. In: 10th Workshop on Variability Modeling of Software-Intensive Systems (VaMoS). pp. 33–40, 2016.

[We16] Weckesser, Markus; Lochau, Malte; Schnabel, Thomas; Richerzhagen, Björn; Schürr, Andy: Mind the Gap! Automated Anomaly Detection for Potentially Unbounded Cardinality-based Feature Models. In: 19th Int. Conference on Fundamental Approaches to Software Engineering (FASE). pp. 158–175, 2016.