Tseitin or not Tseitin? The Impact of CNF Transformations on Feature-Model Analyses

Elias Kuiter[1] Sebastian Krieter[2] Chico Sundermann[3] Thomas Thüm[4] Gunter Saake[5]

Abstract: This work was published at the 37th IEEE/ACM International Conference on Automated Software Engineering (ASE) 2022 [Ku22].

Feature modeling is widely used to systematically model features of variant-rich software systems and their dependencies. By translating feature models into propositional formulas and analyzing them with solvers, a wide range of automated analyses across all phases of the software development process become possible. Most solvers only accept formulas in conjunctive normal form (CNF), so an additional transformation of feature models is often necessary. However, it is unclear whether this transformation has a noticeable impact on analyses. We compare three transformations for bringing feature-model formulas into CNF. We analyze which transformation can be used to correctly perform feature-model analyses and evaluate three CNF transformation tools on a corpus of 22 real-world feature models. Our empirical evaluation illustrates that some CNF transformations do not scale to complex feature models or even lead to wrong results for model-counting analyses. Further, the choice of the CNF transformation can substantially influence the performance of subsequent analyses.

Keywords: Feature Modeling; Automated Reasoning; Conjunctive Normal Form

Many software systems in today's industry can be diversely configured to serve specific customer needs, making it necessary to systematically model their features and dependencies. To this end, *feature models* are widely used. *Satisfiability (SAT) solvers* search for satisfying assignments of propositional formulas and are routinely used for the automated analysis of feature models. Similarly, *model-counting (#SAT) solvers* count satisfying assignments of propositional formulas and also empower numerous feature-model analyses.

To analyze feature models using solvers, they must be translated into propositional formulas. For automated analysis using solvers, these formulas typically must be supplied in *conjunctive normal form (CNF)*, which necessitates a transformation into CNF. However, in many papers on feature-model and variability analysis, this step (although necessary) is not mentioned or discussed only superficially; and evaluations using tools for feature-model extraction or analysis typically do not document the chosen CNF transformation. Moreover, we repeatedly observed in industry collaborations that using different CNF transformations may affect the efficiency and results of analyses.

[1] Otto-von-Guericke University Magdeburg kuiter@ovgu.de
[2] University of Ulm sebastian.krieter@uni-ulm.de
[3] University of Ulm chico.sundermann@uni-ulm.de
[4] University of Ulm thomas.thuem@uni-ulm.de
[5] Otto-von-Guericke University Magdeburg saake@ovgu.de

To assess the impact of CNF transformations on SAT- and #SAT-based feature-model analyses, we describe and compare three state-of-the-art techniques for transforming feature-model formulas into CNF: the distributive [BL99], Tseitin [Ts83], and Plaisted-Greenbaum [PG86] transformation. As a tool for comparison, we propose a taxonomy of five properties (two of which have not been considered in the literature before) by which we classify these three transformations. We characterize how our taxonomy relates to selected feature-model analyses, finding that the distributive and Tseitin transformations are suitable for for model-counting analyses, while the Plaisted-Greenbaum transformation is not.

In addition, we empirically evaluate the efficiency of three CNF transformation tools commonly used for feature-model analyses on a corpus of 22 real-world feature models. We find that the selection of a CNF transformation has a substantial impact not only on the performance of the transformation itself, but also on the efficiency and even the correctness of subsequent analyses.

In summary, both our theoretical analysis and empirical evaluation show that the selection of CNF transformations is highly relevant for practitioners and researchers, especially when using performance-critical analyses, and has to be considered carefully.

Data Availability To ensure reproducibility, we disclose our fully automated evaluation pipeline[6] and all feature models, solvers, and results as a replication package.[7]

Bibliography

[BL99] Büning, Hans Kleine; Lettmann, Theodor: Propositional logic: deduction and algorithms, volume 48. Cambridge University Press, 1999.

[Ku22] Kuiter, Elias; Krieter, Sebastian; Sundermann, Chico; Thüm, Thomas; Saake, Gunter: Tseitin or not Tseitin? The Impact of CNF Transformations on Feature-Model Analyses. In: Proc. Int'l Conf. on Automated Software Engineering (ASE). ACM, October 2022.

[PG86] Plaisted, David A; Greenbaum, Steven: A Structure-Preserving Clause Form Translation. J. Symbolic Computation, 2(3):293–304, 1986.

[Ts83] Tseitin, Grigori S.: On the Complexity of Derivation in Propositional Calculus. In: Automation of Reasoning: 2: Classical Papers on Computational Logic 1967–1970. Springer, pp. 466–483, 1983.

[6] Automation scripts available at: `https://doi.org/10.5281/zenodo.6922807`
[7] Replication package available at: `https://doi.org/10.5281/zenodo.6525375`