

モデルカウンターを使った FM の特徴把握方法の提案

A method for capturing feature model characteristics

氏名 YANG YUXIN†

北村 崇師‡

岸 知二†

所属 †早稲田大学 ‡産業技術総合研究所

1. はじめに

近年、ソフトウェア開発の分野では、開発量の増加や開発期間の短縮化が一層進んでいる。そうした中、ソフトウェアプロダクトライン (Software Product Lines, SPL) 開発が大きく注目を集めている。SPL 開発とは、同系列の製品群のソフトウェアを効率よく再利用開発するための手法である。従来の開発が「製品のソフトウェア開発」であったのに対し、SPL 開発では「製品群のソフトウェア開発」というアプローチを採る。つまり、個別最適から全体最適へと、ソフトウェア開発を改善するものである。

SPL 開発では SPL (製品群) 全体を扱うため可変性の管理が重要となる。可変性とは SPL 中の各製品で変わりうる特徴のことである。可変性をモデル化する手法のひとつとしてフィーチャモデルがある。当初は小規模なフィーチャモデルが主流だったが、現在では例えば自動車産業などにおいて、設計や実装に関わる詳細なモデルが書かれるようになり、そのフィーチャ数も多くなり、フィーチャモデルから SPL 全体の特徴を捉えることが困難になっている。

そうした中、大規模なフィーチャモデルの特徴を捉える研究がある。例えば Ruben は全体の特徴を捉える指標として FIP (Feature Inclusion Probability) と PD (Product Distribution) を提案している [2]。ここで FIP は、フィーチャが製品に出現する確率を、PD は、製品が含むフィーチャの個数の分布をそれぞれ表す。

2. 関連技術

2.1 フィーチャモデル

フィーチャモデルとは Kang らにより、1990 年代に提案されたモデル化手法であり [1]、木構造を用いて可変性を製品のフィーチャ (機能的あるいは非機能的な特徴) により階層的に表現する手法である。

2.2 フィーチャモデルの命題論理による解析

フィーチャモデルのマシン解析を行う典型的な

手法は、フィーチャモデルを命題論理式に変換し、フィーチャモデルの選択・非選択を命題論理の命題変数の真偽と対応つけて論理で扱う方法である。こうすることで例えばフィーチャ構成を求める問題は SAT 問題として扱うことができる。

2.3 モデルカウンター

モデルカウンターは与えられた命題論理式を満たす充足例の数をカウントする技術である。例えば Ganak, sharpSAT などのツールがある。これらのツールでは命題論理式を連言標準語 (Conjunctive normal form, CNF) で表現し、それを入力とすることで充足例の数を求めることができる。これを応用することで、フィーチャモデルから導出できる製品数をカウントすることができる。

3. 先行研究

Ruben らは、大規模なフィーチャモデルの特徴を捉える手法として FIP と PD を提案した [2]。FIP を求めることで、各フィーチャが製品に出現する確率をヒストグラムで表すことができる。図 1 に FIP の例を示す。

Ruben らは FIP と PD を求めるために BDD (Binary Decision Diagram, BDD) [3] を活用したアルゴリズムを提案している。しかしながら各指標に特化した複雑なアルトリズムになっているため、柔軟性や汎用性が低いというデメリットがある。

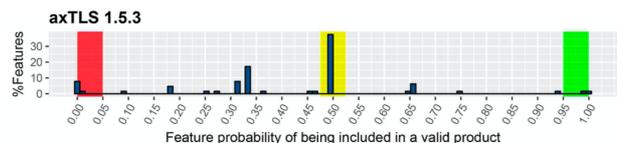


図 1 FIP の例 [2]

4. 研究目的

命題論理の解析を行うための汎用技術を活用することで、フィーチャモデルの特徴を捉える指

標を計算する手法を提案する. 本稿ではモデルカウンターを使った FIP の計算手法を示す.

SAT ソルバやモデルカウンターなどの技術は汎用性が高く, またその効率も飛躍的に改善している. 従ってこれらの技術を活用することで, 特定の指標への依存性を低く抑えつつ, 実用的な時間で指標を計算することを狙うものである.

5. 提案手法

モデルカウンターでフィーチャ構成数を求めることができるが, フィーチャモデルを表す命題論理式に特定のフィーチャに対応する命題変数を真にする条件を付け加えることでそのフィーチャを含む製品数が求まるため, それを全製品数で割ることで, そのフィーチャの出現確率を求めることができる.

具代的には以下の手順で計算する

1. 全製品数を求める
2. すべてのフィーチャについて以下を繰り返す
 - 2.1 そのフィーチャを選択するという条件を付け加える
 - 2.2 そのフィーチャを含む製品数を求める
 - 2.3 そのフィーチャを含む製品数/全製品数で出現確率を得る

6. 評価結果

Ruben らが用いた 5 つのベンチマーク (axTLS, fiasco, toybox, busybox, embtoolkit) を用いて FIP を求めるとともに, 計算時間を計測したなお, モデルカウンターには Ganak を利用した.

図 2 に本手法で得られた 5 つのベンチマークの FIP を, 表 1 に実行時間を示す.

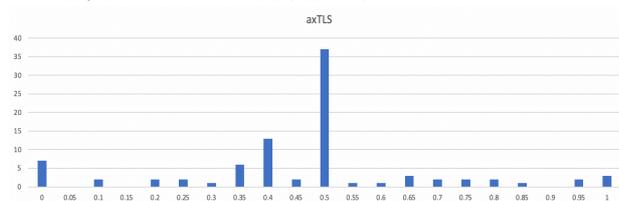


図 2 axTLS の FIP

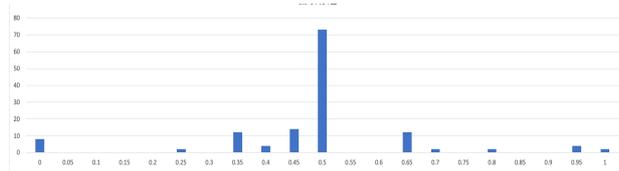


図 3 Toybox の FIP

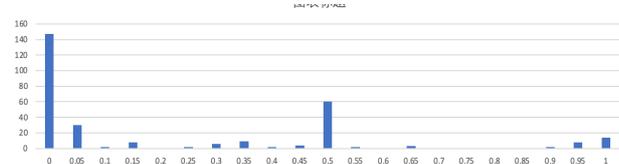


図 4 Fiasco の FIP

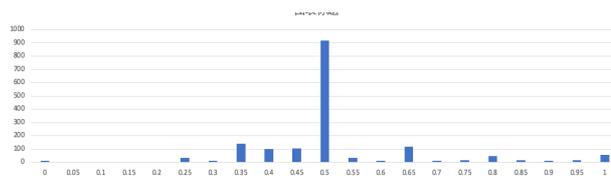


図 5 Busybox の FIP

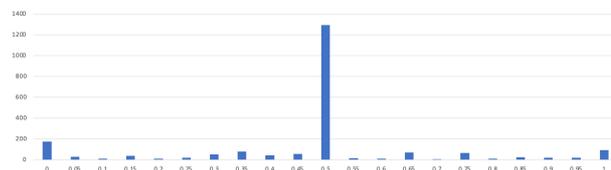


図 6 Embtoolkit の FIP

Benchmark	FeatureNumber	Time
axTLS	180	1.080s
Toybox	136	1.088s
Fiasco	300	4.800s
Busybox	1690	53.38s
Embtoolkit	2136	14659.368s

表 1 実行時間

上記の実験により, 本手法によって FIP を実用的な時間で求めることができることを確認した.

7. おわりに

本稿ではモデルカウンターを用いてフィーチャモデルの特徴を把握する方法を提案した. 汎用ツールを用いたため, 指標に依存する部分が少ない方法で FIP を求めることができた. また計算時間も実用的な範囲であることが確認できた. なお FIP は全体の特徴を捉える目的だけでなく, この確率をテストに応用したり, デッドフィーチャや共通フィーチャの識別などに利用できるため有用性が高い.

今後, より効率的な計算ができるように改善するとともに, 他の指標の手法についても検討する.

参考文献

[1] K. C. Kang, "Feature-oriented development of applications for a domain," Proceedings. Fifth International Conference on Software Reuse (Cat. No.98TB100203), Victoria, BC, Canada, 1998, pp. 354-355, doi: 10.1109/ICSR.1998.685763.
 [2] Ruben Heradio, Supporting the Statistical Analysis of Variability Models, 41st International Conference on Software Engineering (ICSE), pp.843-853, 2019.
 [3] Akers, "Binary Decision Diagrams," in IEEE Transactions on Computers, vol. C-27, no. 6, pp. 509-516, June 1978, doi: 10.1109/TC.1978.1675141.