様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９　（共通）

科学研究費助成事業　　研究成果報告書

令和　元 年　 ６ 月 ２１ 日現在

機関番号：３２７１０
研究種目：基盤研究(C)（一般）
研究期間：2016～2018
課題番号：１６Ｋ００１０９

研究課題名（和文）ソフトウェアモデル検査における活性検証

研究課題名（英文）Liveness verification in software model checking

研究代表者
田辺　良則（Tanabe, Yoshinori）

鶴見大学・文学部・教授

研究者番号：６０４４３１９９

交付決定額（研究期間全体）：（直接経費）　　3,400,000円

研究成果の概要（和文）：モデル検査技術を用いてソフトウェアが正しく動作することを検証する「ソフトウェアモデル検査」において活性性質の検証に関して研究を行った．LTL論理式から作成されるBuchiオートマトンと状態空間の同期積を探索する際に，同一の状態を何度も作成するために遷移に大きなコストがかかるという点に着目し，同一状態を再利用することで平均的な実行時間を短くする方式を考案し，Java Pathfinderの上での実装を得た．

研究成果の学術的意義や社会的意義
ソフトウェアモデル検査は，安全性性質の検証には多くの場面で用いられており，プログラムの誤りを見つけることに役立っている．しかし，活性性質の検証事例は比較的少数である．この理由のひとつは性質記述の難しさ，もうひとつには計算量の大きさである．本研究は，Java Pathfinderというひとつのモデル検査器上での実装ではあるが，問題の後者の側面を改善するものである．

研究成果の概要（英文）：Software model checking is a method to verify that software works properly. When verifying a liveness property, a naive method would search the synchronous product of a Buchi automaton created from a LTL formula and a state space. This method is not very effective, because a same state can be created many times. In this research, we propose a method that reduces the average cost of transition by saving and restoring states, and implemented it on Java PathFinder.

研究分野：ソフトウェア工学

キーワード：model checking　software　liveness

様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９、ＣＫ－１９（共通）

１．研究開始当初の背景

モデル検査技術を用いてソフトウェアが正しく動作することを検証する「ソフトウェアモデル検査」には，さまざまな方式がある．本稿では，Java 言語のプログラムを対象としたモデル検査器である Java PathFinder (JPF) に代表されるような，プログラムを実際に実行しながら（on-the-fly で）状態遷移系を生成しつつ探索する方式を対象とする．この方式を OSMC と呼ぶことにする．

OSMC では，設計やプロトコル仕様を対象とする伝統的なモデル検査と比較して，一般的には非常に状態数が大きくなる．近年の計算機のハードウェア能力の向上や方式の改良等により改善されてはいるが, OSMC で完全な検証ができるソフトウェアの規模はかなり小さい. 一方, OSMC は，テスト手法では検出しにくい誤りを見つけ出す方式としても利用することができ，この観点ならば，より大きなソフトウェアにも適用できる．

しかし，JPF などの多くの現行の OSMC 方式のモデル検査器では，SPIN などの伝統的モデル検査器とは異なり，検出できる誤りが，表明（assertion）違反やデッドロックなどの安全性性質違反に限られている．意図しない無限ループや使用したリソースの回収漏れなどの，活性性質違反に関する誤りは検出できない．我々の研究グループでは，長期にわたり JPF を用いてネットワークソフトウェアの検証に取り組んでいるが，JPF のこの制限のため，検証する性質は安全性性質に限られている．ネットワークソフトウェアでも，要求に対する応答などの活性性質を検証したい場面はよく現れるので，この機能の欠如は実用化に向けての障害となっている．

安全性性質の検証は，単に全状態を探索することで実行可能であるが，活性性質を検証しようとすると，一般には LTL (Linear-time Temporal Logic) などの時相論理に関する検証を行う機構が必要になる．JPF を拡張して LTL のサポートを行う試みはいくつか存在する．これらは伝統的なモデル検査器と同じ手法を用いている．すなわち，LTL 論理式を Buchi オートマトンと呼ばれる構造 B に変換し，元の状態空間 M との同期積と呼ばれる，一種の直積のような構造 B*M を作成し，この同期積を探索するのである．

しかし，この方法を単に実装すると，長い実行時間が必要となる．上述のようにもともと状態空間 M のサイズが大きいのであるが，B*M は，その数倍以上となる．さらに，活性性質検証の際には，しばしば「強公平性」と呼ばれる条件の考慮が必要になるが，これをナイーブな方式 (LTL で記述)で実現すると，簡単に数百倍のオーダーになってしまう．

強公平性を効率的に扱う方式の研究もあるが，OSMC への適用可能性は未知数である．

２．研究の目的

OSMC の特徴を利用した効率的な探索方式を開発することにより，上記の問題を解決し，活性性質等の検証を可能にすることを，目標とした．

本研究でも，従来手法と同様に，同期積 B*M の探索を行う．B*M における 1 つの遷移は，B の遷移と M の遷移の組み合わせでできている．B*M での探索中に，B と M の各々の遷移が多数回実行されることになる．OSMC の特徴として，M の各々の遷移の実行が高コスト（時間を要する）であることがあげられる．M の遷移は，プログラム片の実行に相当する．JPF の場合，M の 1 つの遷移で，数～数百の Java バイトコードを実行する．さらに後処理としてヒープ領域全体のハッシュ値も計算している．

そこで，多数回実行される M の各遷移について，1 回目には実際に遷移を実行するが，その際に実行した結果をキャッシュしておき，2 回目以降は実際に遷移を実行する代わりにキャッシュにある結果を参照するようにすれば，速度向上が得られると考えられる．この考え方に基づいて JPF 上に実装を行い，実際に使用可能なツールを得ることを研究の目的とした．

あわせて，アルゴリズムの形式検証も目標とした．基本探索アルゴリズムは難解なものではないが，特定の条件を導入して最適化を追求して誤りが混入することはあるため，形式検証によって，そのような問題が生じていないことを確認することもスコープに加えた．

３．研究の方法

我々の研究グループが，本研究を開始する以前に実装を行った，JPF 上の LTL 検証エンジンを，ベースとして用いることとし，このエンジンを拡張する形で実装を進めた．

形式検証については，定理証明支援系 Coq およびそのライブラリである SSReflect を用いて検証を進めていくこととした．

本研究の主たるターゲットは OSMC であるが，ソフトウェアの検証においてはモデル検査のような探索系の技術とテスティング技術が相補的に用いられる．特に状態遷移系を使用したモデルベーステストはモデル検査技術と要素技術に重なる点がある．そこで，本研究の遂行では，関連するテスティング技術も積極的に開発していくこととした．

4. 研究成果

(1) 方式

上述のアイディアにもとづいて，具体的なアルゴリズムの検討を行い，これを定式化することができた．JPF のような OSMC ツール上で実現するためには，いくつかの実行時オプションが必要なことが明らかとなった．これらは状態を復元するためのキャッシュ情報が得られない場

合の対処である．対策としては 2 つが考えられ，ひとつは探索を再実行すること，もうひとつは，1 回目の探索の際に，復元用の情報をキャッシュとは別にあらかじめ保存しておき，2 解明校ではこの情報から復元を実施することである．両者の検討および実験の結果論理式や探索空間に依存してどちらも有利となる場合がありうることがわかった．このため，一定の遷移回数毎に復元ポイントを用意し，その時点からの再実行を行うという，両者を融合させた方式にすることで，ひとつのアルゴリズムとして確定させることができた．

(2) 実装

初年度に行った試験的な実装とその評価によって，JPF の状態保存とその復元方式に関する分析を行った結果，当初想定していた形での実装がうまく動作しないということが明らかになった．JPF では，状態保存と復元は，その探索順序に従って発生するという前提で書かれており，我々のアルゴリズムではその前提が満たされないからである．これに対処するため，状態保存・復元がその前提なしでも機能するように，必要な情報を合わせて保存するように変更して実装を行い,正しく機能する（内部利用のための）ライブラリとして作成することができた．

次に，JPF 実装上の最適化によって，同一の遷移であっても，作成される状態が異なりうることへの対処を行った．JPF の作成基準に手を入れることも検討したが，JPF 本体のコードを変更するリスクと実際のコーディングの困難さから別方式に変更した．リスナーを利用することで JPF の状態作成の判断をモニターし，1 回目の記録を参照することで必要な箇所を特定し，遷移を中断させる方式である．以上によって，考案した方式を実現する実装を得ることができた．

なお，ベースとして採用した LTL ライブラリについて，上記以外に弱公平性条件指定の高速化のための実装も行った．条件を LTL で指定する代わりに，専用のアルゴリズムを用いるものである．基本命題として採用する必要のあるスレッドの状態判定は従来のルーチンをそのまま採用した．このアルゴリズム自体は広く知られており，たとえばモデル検査ツール SPIN でも実装されているものであるが，我々のツールの使いやすさを向上させる意味があった．

(3) 形式検証

OSMC が扱う状態遷移系を扱うためには，グラフ構造を扱うための構造およびそれに関する定理を証明するための補題群が必要となる.そのためのライブラリの整備を実行することとした．この段階で，取り組む対象を 2 つ設け，1 つは本アルゴリズムの形式化と証明，もう 1 つは作成したライブラリを用いて取り組める問題の解明とした．前者に関しては形式化までは問題無く取り組めたものの，最終的な証明完成までは至らなかった．後者に関しては以下の成果を得た．

具体的な問題としては，ペトリネット上における Karp-Miller 木の構成をとりあげた．これはペトリネットにおける被覆可能性判定問題を解くために用いられる古典的な構成であり，これを「紙とペン」でなく，証明支援系上に実現し，さらにその正当性証明を支援系上で遂行した．正当性は妥当性と完全性からなるが，これらを直接証明するのでは無く，ペトリネットから導出される遷移系の性質として記述して，その Karp-Miller 木との関係性を示すという方針を用いた．これによってライブラリ化を容易にし，さらには証明が過度に複雑になるのを抑える効果を得た．形式証明構築の上での一つの問題点は，通常の証明では，Karp-Miller 木の構成が終了することを証明する際に排中律を用いていることである．しかし，形式化の過程全般を見直すことによって，Coq の推論法則のみで公理を加えること無く証明が構築できることがわかった．ここで用いた手法を用いることで，ペトリネットに関する他の体系の証明も容易になることがあると考えている．

(4) その他

以上の内容を応用，ないしアイディアを発展させて，以下の研究成果が得られた．

① JPF でも利用されている静的実行を用いることで，プログラムのある時点で指定された条件式を実際に満足するための入力セットを生成することができる．これを利用して，プログラム理解に関する試験問題の半自動作成手法を考案した．具体的には以下のような手順である．

問題作成者は，プログラム，プログラム上の位置，その位置における不変式を与える．不変式は一般にいくつかの単純な式の結合で書けるため，構成する各式の真偽の組合せを生成して各々に対して静的実行を行うことで入力およびその位置における状態を得る．これらを回答者はありうる入力と判定する必要がある．一方で，不変式を満たさない状態を生成する手順も考案した．これは，ダミーの入力を与えて，その位置において状態の一部をダミーと入れ換える操作をすることで得られる．

以上の手法に基づき，対象プログラムと不変式群が与えられたときに，必要な入力群を生成するためのライブラリを，静的実行ツールをベースとして作成し，これを用いて問題生成が可能であることを確認した．さらに，作成した問題を実際に（あらかじめ学習段階を把握している）プログラム学習者に解答させることによって，その時のプログラム作成能力との相関関係が存在することを確認した．

② モデル検査が対象とする状態遷移系と同様にグラフ構造を用いて計算を行うことで,判定器を作成する手法について，2 方面から検討，実験を行った．一つは Coq による形式検証のアプローチであり，状態数についての命題を形式化できることを確認した．もう一つは図書自動分類の試みであり，書誌情報などを入力として NDC 分類を行う判別器の試作を行った．

５．主な発表論文等

〔雑誌論文〕（計 6 件）

① Masami Hagiya, Kosuke Fukuda, Yoshinori Tanabe and Toshinori Saito: Automatically Generating Programming Questions Corresponding to Rubrics Using Assertions and Invariants, Sustainable ICT, Education and Learning - IFIP TC3 WG3.4 in collaboration with other TC3 WGs (SUZA2019), to appear.　査読有．
② 杉山治紀，田辺良則: ニューラルネットワークを用いた図書の自動分類．信学技報，118(425), pp.61-66, 2019．査読無．
③ Mitsuharu Yamamoto, Shogo Sekine, Saki Matsumoto: Formalization of Karp-Miller Tree Construction on Petri Nets.　Proc. 6th ACM SIGPLAN Conference on Certified Programs and Proofs.　pp.66-78, 2017.　査読有．DOI: 10.1145/3018610.3018626
④ 山崎智史，登内敏夫，田辺良則: Datalog を利用したネットワーク設定変更手順生成．信学技報, 116(124), pp.33-38, 2016. 査読無
⑤ 米山惇, Cyrille Artho，田辺良則, 萩谷昌己: MQTT 実装のモデルベーステスト．ソフトウェア工学の基礎, vol.23, pp.249-250, 2016.　査読無
⑥ 坂西一暁, Cyrille Artho，田辺良則, 萩谷昌己: Modbat を用いた Apache ZooKeeper のモデルベーステストにおける探索アルゴリズムの改善．ソフトウェア工学の基礎，vol.23, pp.253-254, 2016.　査読無

〔学会発表〕（計 5　件）

① 井上健太，山本光晴: ニューラルネットワークにおける表現可能なデータ数の SSReflect による形式化. 第21回プログラミングおよびプログラミング言語ワークショップ(PPL2019)（ポスター），2019.
② 杉山治紀，田辺良則: ニューラルネットワークを用いた図書の自動分類. 電子情報通信学会知能ソフトウェア工学研究会，2019.
③ 坂西一暁，Cyrille Artho，田辺良則，萩谷昌己，北村崇師: 分散システムを対象としたモデルベーステストにおけるテストオラクルの高速化. 第 20 回プログラミングおよびプログラミング言語ワークショップ(PPL2018)（ポスター），2018.
④ 米山惇，Cyrille Artho，萩谷昌己，田辺良則: IoT ソフトウェアのための不安定なネットワークとデバイスをシミュレートするモデルベーステスト. 第 20 回プログラミングおよびプログラミング言語ワークショップ(PPL2018)（ポスター），2018.
⑤ 太田十字光，田辺良則，青木利晃: Java Pathfinder における弱公平性条件の実装．日本ソフトウェア科学会第 33 回大会，2016.

６．研究組織

(1)研究分担者

研究分担者氏名：萩谷 昌己

ローマ字氏名：Hagiya Masami

所属研究機関名：東京大学

部局名：大学院情報理工学系研究科

職名：教授

研究者番号（8 桁）：30156252

研究分担者氏名：山本 光晴

ローマ字氏名：Mitsuharu Yamamoto

所属研究機関名：千葉大学

部局名：大学院理学研究院

職名：教授

研究者番号（8 桁）：00291295

(2)研究協力者
なし

※科研費による研究は、研究者の自覚と責任において実施するものです。そのため、研究の実施や研究成果の公表等については、国の要請等に基づくものではなく、その研究成果に関する見解や責任は、研究者個人に帰属されます。