様式 C-19

科学研究費補助金研究成果報告書

平成２３年　５月２０日現在

機関番号：１２５０１
研究種目：基盤研究(B)
研究期間：2008～2010
課題番号：２０３００００６
研究課題名（和文）　仮想計算機によるコミュニケーションバックトラッキングとモデル検査への応用
研究課題名（英文）　Communication Backtracking by Virtual Machines and Applications to Model Checking
研究代表者
山本 光晴（YAMAMOTO MITSUHARU）
千葉大学・大学院理学研究科・准教授
研究者番号：００２９１２９５

研究成果の概要（和文）：

状態の系統的･網羅的探索でプログラムの正当性を検証するモデル検査という手法がある。これをネットワークアプリケーションに適用するため、通信を含んだ形でプログラムの実行を遡るバックトラッキングを実装し､Java プログラムのモデル検査器 Java PathFinder の拡張の形で組み込んだ。通信を含むバックトラッキングは、通信相手のプログラムを実行状態の保存・復元が可能なチェックポインティング環境で動作させ、さらに通信接続の保存・復元のために独自の伝達層を用いて通信をすることで実現されている。

研究成果の概要（英文）：

Model checking verifies program correctness using systematic/exhaustive search on the state space. In order to apply this technique to network applications, we implemented "communication backtracking" that rewinds the program execution including its communication states, and also included it as a module for Java PathFinder, a model checker for Java programs. Communication backtracking is implemented by running a communication peer program under a checkpointing environment, which enables us to save/restore program states, and providing an original transport layer in order to save/restore connections.

交付決定額

（金額単位：円）

	直接経費	間接経費	合　計
２００８年度	3,100,000	930,000	4,030,000
２００９年度	2,800,000	840,000	3,640,000
２０１０年度	2,800,000	840,000	3,640,000
年度			
年度			
総　計	8,700,000	2,610,000	11,310,000

研究分野：情報数理学
科研費の分科・細目：情報学・ソフトウェア
キーワード：モデル検査・仮想計算機・チェックポインティング

１．研究開始当初の背景

実際の計算機や OS 上で、仮想的に計算機や OS を動作させるための技術をここでは仮想計算機技術と呼ぶことにする。実際の計算機で動作している側の OS をホスト OS と呼ぶ。近年のプロセッサの高速化やメモリの大容量化により、仮想計算機技術の実際への応用

は現実的なものになっている。
仮想計算機の中にはスナップショットを撮る機能を持つものがある。これは動作中の仮想計算機の状態を(スナップ写真を撮るように)切り出してファイルの形で保存する機能であり、後にその実行を再開させることが可能である。
スナップショットはラップトップパソコンなどに見られるようなサスペンド機構の提供にとどまらず、適切なタイミングでスナップショットを撮っていくことにより、OS 全体の実行を巻き戻すバックトラッキングを行うことを可能にする。これにより、何か異常が発見された時点から、過去に遡ってデバッギングや解析を行うことが可能となる。
Linux 上のユーザプロセスとして Linux OS を動作させる仮想計算機技術の実装として User-mode Linux (UML)がある。また、UML に対し、スナップショットを撮ってそれをスクラップブックに保存する機能を追加した ScrapBook for UML(SBUML)がある。我々は以前の研究において、SBUML を拡張し、バックトラッキングによるマルチプロセスアプリケーションの実環境ソフトウェアモデル検査を行った。モデル検査とは、システムの状態空間を系統的・網羅的に探索することによって検証を行う手法である。マルチプロセスアプリケーションに対するモデル検査においては、スケジューリングが持つ非決定性を網羅的に探索し、どのようなスケジューリングを行ってもシステムが不適切な状態に陥らないことを検証することがその目的となる。
旧来のモデル検査の対象は専用の言語で記述された擬似的なプログラムであったが、近年は実際のプログラムを検査対象とするソフトウェアモデル検査が研究対象となってきている。さらに実環境ソフトウェアモデル検査においては、検査対象のプログラムを、それが実際に使用される状況に近い環境で実行しながら検査を行うことにより、実際に起こりうる不具合を早期に発見することを目的としている。このとき、実行状態の網羅的な探索を実環境に近い環境で行うために、仮想計算機のスナップショット機能が利用される。すなわち、非決定性が生じる時点で OS 全体のスナップショットを作成し、後の探索において別の枝を辿る際にそのスナップショットから実行を再開するのである。
このような実環境ソフトウェアモデル検査手法をクライアント・サーバに代表されるネットワークアプリケーションにそのまま適用しようとすると、スナップショット作成・再開のオーバーヘッドにより、検証が非常に非効率的になるという問題点がある。これは探索を行う際のバックトラッキングを可能にするために、クライアント・サーバ双方を同一の仮想計算機に収め、全てをモデル検査の対象としてしまっていることに起因する。この問題を解決するため、我々は最近の研究において、ネットワークアプリケーションを効率的に検証するための新たな枠組を提案した。この枠組ではクライアントかサーバの一方をモデル検査の対象とし、網羅的探索のためにバックトラックを随時行いながら動作する。この研究においてはバックトラックのために検査対象のプログラムを Java で実装したものに制限し、実行の際には Java PathFinder (JPF)と呼ばれるモデル検査用の特殊な Java VM を使用している。他方、すなわち検査対象の通信の相手となる方は、通常のプログラムとして動作し、実装言語・OS は問わない。後者は検査対象に影響を与える動作環境という意味で「環境」と呼ばれる。
ここで、通信を行う検査対象がバックトラックを起こすと、同じメッセージを環境に対して再送信しようとすることがある。環境側は通常のプログラムであってバックトラックには対応していないから、そのまま送信してしまっては不整合を起こす。そこで、モデル検査器と環境の間に過去の通信履歴を保存する「キャッシュ層」を追加し、過去に送信したことのある内容が再送信されようとした際には、キャッシュ層が環境に成り代わって返答する。もちろん、この仕組みが矛盾なく動作するためには、検査対象内でのスケジューリングの非決定性に関係なく、通信におけるメッセージ列が一定であることが大前提となる。しかしながら、このことは検査対象の種類によっては大きな制約となる。

２．研究の目的

本研究課題の目的は、以下のように要約される。
(1) 仮想計算機技術を拡張・発展させることにより、プログラムの実行を遡る「バックトラッキング」を、通信を含むネットワークアプリケーションにおいても可能にすること。
(2) 上記のバックトラッキングをソフトウェアモデル検査に応用し、ネットワークアプリケーションの効率的な検証を可能にすること。
「研究開始当初の背景」で述べたように、仮想計算機のスナップショットのみを用いた手法では効率の問題が、JPF とキャッシュ層を用いた手法ではメッセージ列に対する制約が問題となる。本研究課題の狙いは、両者の技術を組み合わせ、補完しあうことにより、効率的で、かつメッセージ列に制約のない形でソフトウェアモデル検査が行えるような枠組を提案し、実装することである。

３．研究の方法

当初の計画に従い、まずは各種仮想計算機実装の調査を行った。特に、バックトラッキングに必要なサスペンド・レジューム・マイグレーション機能について調査した。その際、単純なプロトタイプ実装による予備実験においてパフォーマンス測定を行ったのだが、既存の仮想計算機実装ではスナップショットの保存や復元に時間がかかり過ぎ、実用的なモデル検査に用いるには困難であることが判明した。
一方で、仮想計算機の周辺技術の調査の過程で、より軽量なチェックポインティング機構を用いても我々の目的を達成することができる目処がついたことから、2 年度目以降からは仮想計算機ではなくチェックポインティング機構を用いた実装に軸足を移した。具体的には、Linux オペレーティングシステム上でマルチスレッドプログラムのチェックポインティングが行える、MTCP というシステムを利用している。
ネットワークアプリケーションにバックトラッキングを適用するには、通信相手の復元時に通信接続を回復させる必要があるが、我々がチェックポインティングに使用した MTCP は通信接続の回復に対応していない。そこで、いくつかの解決策をテスト実装を含めた形で検討し、独自にトランスポート層を実装することでこの問題を解決した。
さらにコミュニケーションバックトラッキングの実装と並行して、モデル検査に応用した場合の評価実験と、この枠組みで扱うことのできるアプリケーションの特徴付けに関する考察を行った。

４．研究成果

(1) コミュニケーションバックトラッキングの実現: 通信を含む形でプログラムの実行を遡る「コミュニケーションバックトラッキング」を実現した。「研究方法」で述べた通り、当初の計画ではバックトラッキングに仮想計算機機構を用いる予定であったが、初年度における調査の結果を反映させ、チェックポインティングを利用するように設計を変更した。
我々がチェックポインティングに利用した MTCP というシステムは、Linux オペレーティングシステム上でマルチスレッドプログラムのチェックポインティングを行うものである。しかし、MTCP はプログラム実行状態の保存・復元の際に通信接続を回復させる機能を持っていない。そこで、復元時に通信接続を回復させるため、独自にトランスポート層を実装した。アプリケーションからは通常の TCP として見えるが、実際にはライブラリの注入により、OS が提供する TCP とは別のトランスポート層が用いられる
(2) モデル検査への応用: コミュニケーションバックトラッキングを Java プログラムのモデル検査器である Java PathFinder のモジュールとして組込み、ネットワークアプリケーションのモデル検査を可能にした。この枠組みでは、検証対象の Java プログラムはモデル検査器 Java PathFinder の制御下で動作し、通信相手のプログラムはチェックポインティング環境 MTCP のもとで動作する。通信相手については Java プログラムに限る必要はない。
(3) 提案した枠組みで扱うことのできるアプリケーションの特徴付け: 研究当初におけるバックトラッキングを用いる第一義的な動機は、従来手法のキャッシュ層を用いたネットワークアプリケーションのモデル検査における、通信におけるメッセージ列が一定でなければならないという制約を回避するというものであった。しかし、初年度においてキャッシュ層自体を拡張することにより、仮想計算機を用いずにこの制限を緩和することに成功するという、当初想定していない成果が得られてしまった。
この成果により、仮想計算機なりチェックポインティング環境なりを用いる意義付けが曖昧になったため、その意義を明確にする必要が生じた。このため、ネットワークアプリケーションのモデル検査に対する各種アプローチの特徴付けとその考察を行い、どのような種類のアプリケーションでどのようなアプローチが必要となるかを明確にした。
具体的には、通信相手の出力が入力に対して非決定的であるような場合に、キャッシュ層だけではモデル検査を続行させることができない。これは、通信相手を最初から再実行してそれまでの入力列を再度処理させたとしても、もはや同じ出力が得られるとは限らないためである。例えば、乱数によりセッション鍵を生成し、出力するような通信相手が出力が入力に対して非決定的であるような例である。
チェックポインティングを利用することでこの制約を回避できる。通信相手側が乱数生成デバイスからの読み込みや時刻参照などの出力の非決定性に影響する操作を行った場合、モデル検査器側にその情報を伝える。これにより、モデル検査器側は通信相手側に対してチェックポインティングを用いて状態を保存するように指示でき、後でその非決定的な選択を行った直後から再開できるようにするのである。
(4) 成果の公表: 研究成果は国際会議やワークショップ等での発表に加え、実装を NASA で開発されているモデル検査器 Java Path Finder の拡張として公開しており、Java

archiveファイルとしてNASAのページからダウンロード可能である(「主な発表論文等」における「その他」の項目参照)。
(5) 今後の展望: 本課題の成果では、検証対象のネットワークアプリケーションの通信相手として単一プロセスのアプリケーションしか利用できなかったが、平成 23 年度からの新課題において、本課題の成果をより発展させ、プロセスを動的に生成する通信相手や、複数プロセスからなる通信相手にも対応させる予定である。

５．主な発表論文等
（研究代表者、研究分担者及び連携研究者には下線）

〔雑誌論文〕（計５件）
(1) Cyrille Valentin Artho, Run-Time Verification of Networked Software, Proceedings of the First international conference on Runtime verification, 査読有, 2010, pp. 59-73.
(2) Cyrille Artho, Masami Hagiya, Watcharin Leungwattanakit, Yoshinori Tanabe, and Mitsuharu Yamamoto, Model Checking of Concurrent Algorithms: From Java to C, IFIP Advances in Information and Communication Technology, 査読有, Vol. 329, 2010, pp. 90-101.
(3) Cyrille Artho, Watcharin Leungwattanakit, Masami Hagiya, Yoshinori Tanabe, and Mitsuharu Yamamoto, Cache-based Model Checking of Networked Applications: From Linear to Branching Time, 24th IEEE/ACM International Conference on Automated Software Engineering, 査読有, 2009, pp. 447-458.
(4) Watcharin Leungwattanakit, Cyrille Artho, Masami Hagiya, Yoshinori Tanabe, and Mitsuharu Yamamoto, Verifying networked programs using a model checker extension, 31st International Conference on Software Engineering, Companion Volume, 査読有, 2009, pp. 409-410.
(5) Watcharin Leungwattanakit, Cyrille Artho, Masami Hagiya, Yoshinori Tanabe, and Mitsuharu Yamamoto, Introduction of virtualization technology to multi-process model checking, First NASA Formal Methods Symposium, NASA Conference Publication, 査読有, 2009, pp. 106-110.

〔学会発表〕（計４件）
(1) Watcharin Leungwattanakit, Distributed Software Model Checking Using I/O Cache and Process Checkpointing, Osaka workshop for Verification and Validation, 2011年2月28日, 産業技術総合研究所 関西センター 尼崎事業所.
(2) Watcharin Leungwattanakit, Cache-based Model Checking of Networked Software, ICNC 2010, DNSA workshop, 2010年11月18日, 広島大学 東広島キャンパス.
(3) 田辺 良則, ネットワークアプリケーションのマスタースレーブ方式によるソフトウェアモデル検査, 第８回ディペンダブルシステムワークショップ, 日本ソフトウェア科学会ディペンダブルシステム研究会, 2010年７月20日, 函館大沼プリンスホテル.
(4) Cyrille Artho, Model Checking Networked Software using I/O caching, 第６回ディペンダブルシステムシンポジウム(DSS2009), 2009年12月15日, 大阪大学コンベンションセンター.

〔図書〕（計０件）
〔産業財産権〕
○出願状況（計０件）
○取得状況（計０件）

〔その他〕
ホームページ等
(1) A JPF extension for model checking networked programs, by Watcharin Leungwattanakit and Cyrille Artho. http://babelfish.arc.nasa.gov/trac/jpf/wiki/projects/net-iocache

６．研究組織
(1)研究代表者
山本 光晴（YAMAMOTO MITSUHARU）
千葉大学・大学院理学研究科・准教授
研究者番号：00291295
(2)研究分担者
萩谷 昌己（HAGIYA MASAMI）
東京大学・大学院情報理工学系研究科・教授
研究者番号：30156252
アルト シリル（ARTHO CYRILLE）
独立行政法人産業技術総合研究所・情報セキュリティ研究センター・研究員
研究者番号：30462831
田辺 良則（TANABE YOSHINORI）
国立情報学研究所・アーキテクチャ科学研究系・特任准教授
研究者番号：60443199
(H20～H21:連携研究者)
高橋 孝一（TAKAHASHI KOICHI）
独立行政法人産業技術総合研究所・情報技術研究部門・情報戦略グループ長
研究者番号：40357372
(H21:連携研究者, H22:不参画)