

Title	形式検証ツールの並列化
Author(s)	DO, MINH CANH
Citation	
Issue Date	2022-09
Type	Thesis or Dissertation
Text version	ETD
URL	http://hdl.handle.net/10119/18129
Rights	
Description	Supervisor:緒方 和博, 先端科学技術研究科, 博士

氏名	DO, Minh Canh		
学位の種類	博士 (情報科学)		
学位記番号	博情第 473 号		
学位授与年月日	令和 4 年 9 月 22 日		
論文題目	Parallelization of Formal Verification Tools		
論文審査委員	主査	緒方和博	北陸先端科学技術大学院大学 教授
		小川瑞史	同 教授
		青木利晃	同 教授
		劉少英	広島大学 教授
		土屋達弘	大阪大学 教授

論文の内容の要旨

Today, software systems are used in various applications where failure is unacceptable. Among them are airplanes, utilities, telephones, banking & financial systems, commerce, logistics, appliances, houses, and securities. Very important systems, such as operating systems and the Internet that have been used as infrastructures, are typically in the form of concurrent/distributed programs. We are undeniable that the quality of software systems will affect the quality of our life more and more considerably. Therefore, the need for reliable software systems is critical. Model checking is one of the most successful achievements in computer science for hardware and software verification. However, there are still some challenges to tackle. One of them is the state space explosion problem, which can make it impossible to conduct model checking experiments. Many techniques have been proposed to alleviate the problem to some extent, but the problem still remains when dealing with large systems and often prevents model checking experiments from being carried out. Another challenge is to increase the running performance of model checking. One promising approach to this challenge is to parallelize model checking, which can make the best use of multicore architectures. In this thesis, we propose some techniques to mitigate the state space explosion problem (space challenge) and improve the running performance of model checking (time challenge) by parallelization for some formal verification tools. In summary, the thesis describes three non-trivial cases to demonstrate the proposed techniques: (1) parallelization of Java Pathfinder, a software model checker, for testing concurrent programs, (2) parallelization of Maude LTL model checker for checking leads-to properties, and (3) parallelization of Maude-NPA, a logical model checker, for cryptographic protocol analysis. Besides, we describe some shared techniques used for parallelization in this thesis and a generic approach to parallelizing tools used for formal methods.

Studies on testing concurrent programs have been conducted for nearly 40 years or even more. Compared to testing techniques for sequential programs, however, any testing techniques for concurrent programs do not seem mature enough. Moreover, many important software systems, such as operating systems, are in the form of concurrent programs. Therefore, testing techniques for concurrent programs must be worth studying so that they can be matured enough. We propose a specification-based testing technique for concurrent programs. For a formal specification S and a concurrent program P , state sequences are generated from P and checked to be accepted by S . We suppose that S is specified in Maude and P is implemented in Java. Java Pathfinder (JPF) and Maude are then used to generate state sequences from P and to check if such state sequences are accepted by S , respectively. Even without checking any property violations with JPF, JPF often encounters the notorious state space explosion while only generating state sequences. Thus, we propose a technique to generate state sequences from P and check if such state sequences are accepted by S in a stratified way. A tool is developed to support the

proposed technique that can be processed naturally in parallel. Some experiments demonstrate that the proposed technique mitigates the state space explosion and improves the verification time, which cannot be achieved with the straightforward use of JPF.

Our research group has proposed the $L + 1$ -layer divide & conquer approach to leads-to model checking ($L + 1$ -DCA2L2MC), which is a new technique to mitigate the state space explosion in model checking. As shown by the name, $L + 1$ -DCA2L2MC is dedicated to leads-to properties. This thesis describes a parallel version of $L + 1$ -DCA2L2MC and a tool that supports it. In a temporal logic called UNITY designed by Chandy and Misra, the leads-to temporal connective plays an important role and many case studies have been conducted in UNITY, demonstrating that many systems requirements can be expressed as leads-to properties. Hence, it is worth dedicating to the properties. This thesis also reports on some experiments that demonstrate that the tool can increase the running performance of model checking. Counterexample generation is one of the main tasks in the tool that can be optimized to improve the running performance of the tool to some extent. This thesis then proposes a technique to generate all counterexamples at once that is based on the Tarjan algorithm, implemented in C++, and integrated into Maude, a programming/specification language based on rewriting logic, so that users can use it easily. Some experiments are conducted to demonstrate the power of the technique that can improve the running performance of the tool. Furthermore, layer configuration selection affects the running performance of the tool. Therefore, this thesis then proposes an approach to finding good layer configurations for the tool with an analysis tool that supports the approach. Some experiments are conducted to demonstrate the usefulness of the analysis tool as well as the approach for layer configuration selection.

With the emergence of the Internet and network-based services, many cryptographic protocols, also called security protocols, have been developed over decades to provide information security in an insecure network, such as confidentiality and authentication. The design of cryptographic protocols, such as authentication protocols, is difficult, error-prone, and hard to detect bugs. Therefore, it is important to have automated tools to verify some desired properties of cryptographic protocols. Maude-NPA is a formal verification tool for analyzing cryptographic protocols in the Dolev-Yao strand space model modulo an equational theory defining the cryptographic primitives. It starts from an attack state to find counterexamples or conclude that the attack concerned cannot be conducted by performing a backward narrowing reachability analysis. Although Maude-NPA is a powerful analyzer, its running performance can be improved by taking advantage of parallel and/or distributed computing when dealing with non-trivial protocols whose state space is huge. This thesis describes a parallel version of Maude-NPA in which the backward narrowing and the transition subsumption are parallelized at each layer. The tool supporting the parallel version has been implemented in Maude with a master-worker model. We report on some experiments of various kinds of protocols that demonstrate that the tool can increase the running performance of Maude-NPA by 44% on average for all non-trivial case studies experimented in which the number of states located at each layer is considerably large.

Keywords: testing concurrent programs, LTL model checking, cryptographic protocol analysis, state space explosion, parallelization.

論文審査の結果の要旨

(1) Java用のソフトウェアモデル検査器 Java Pathfinder (JPF)、(2) Maude 線形時相論理モデル検査器、及び(3) セキュリティプロトコル解析・検証用の論理モデル検査器 Maude-NPA を並列化し、28コア・1.5TBメモリを搭載するアップル社製 MacPro 上での実験により十分な実行性能の改善を達成できたことを確認している。(1)と(3)で扱える性質は不変性、(2)で扱える性質は leads-to 性である。不変性と leads-to 性で多くのシステム要求を表現できることは、時相論理 UNITY の設計者で

ある Chandy と Misra 等により実証されており、扱える性質に限りがあったとしても (1) – (3) は十分に有効であることの証左となっている。(1) と (2) の扱えるシステムモデルは有限状態であり、(3) は無限状態を扱える。パターンマッチの代わりにユニフィケーションを用いる、簡約を一般化したナローイングを基本機能として用い、状態を表す式 (項) に変数を含むことを可能とすることで無限状態を扱うことを可能としている。(1) – (3) に共通するものとし、Google MapReduce においても採用されているマスタ・ワーカモデル (パターン)、到達可能状態を複数の小さな部分空間に分割し複数の部分空間を並列に処理すること、ソフトウェアキャッシュの使用により無駄な処理を抑制すること、マスタ・ワーカ間のコミュニケーションコスト等の並列化に伴うオーバーヘッドに見合う速度向上を得るためワーカに割り当てるジョブの負荷を十分に大きくすることなどを抜き出し、上述した 3 つのツール以外に対しても応用可能な並列化技法を提案している。(2) に関しては、(2・1) 一度のモデル検査ですべての反例を発見可能なモデル検査器を、グラフの強連結コンポーネントを効率よく探すことが可能な Tarjan アルゴリズムを採用し、Maude 線形時相論理モデル検査器を改変することで開発することで更なる実行性能の改善を達成している。(2) に関しては更に、到達可能状態空間の層の数と各層の深さのことをレイヤーコンフィギュレーションと呼んでいるが、実行性能がレイヤーコンフィギュレーションに大きく依存するため、(2・2) 良いレイヤーコンフィギュレーションを探すための方法論とそれを支援するツールも開発されている。実験により、上述した MacPro 上で実験を行うことで、(2・1) と (2・2) の有効性を確認している。

以上、本論文は、形式検証ツールの並列化について非自明な 3 つの事例研究を実施し、実験により実行性能の向上を確認し、形式検証ツール (主にモデル検査器) の並列化技法を提案したものであり、学術的に貢献するところが大きい。よって博士 (情報科学) の学位論文として十分価値あるものと認めた。