氏名	MALEEHUAN, Pattaravut		
学位の種類	博士(情報科学)		
学位記番号	博情第 400 号		
学位授与年月日	平成 30 年 9 月 21 日		
論文題目	Program Verification for Multiprocessors with Relaxed Memory Models using an SMT Solver		
論文審査委員	主査	青木 利晃	北陸先端科学技術大学院大学 教授
		平石 邦彦	同 教授
		鈴木 正人	同 准教授
		田中 清史	同 准教授
		高田 広章	名古屋大学 教授

論文の内容の要旨

In modern multiprocessors, the consistency of shared memory would be relaxed to increase the computing power; hence, the value of a memory location could be observed as different values at the same time on each execution unit. Note that, term memory model is usually used to determine the semantics of the memory system. In particular, the memory model that relaxes the consistency of the shared memory is usually called relaxed memory model. Consequently, an anomalous result of the concurrent programs could occur on relaxed memory models. Therefore, relaxed memory model is the primary concern to ensure the program correctness.

For ensuring program correctness, the program property is defined as the invariant of the concurrent programs. Due to the relaxed memory models, this research provides an abstraction, called operation structures, of the concurrent programs. The targets of this abstraction are (1) to be sufficient for program verification, and (2) can describe the essence of assembly programs to be verified. Consequently, the program verification approach should be introduced to prove the program property on target relaxed memory model. In particular, this research uses SMT-based program verification approach to ensure the program correctness automatically.

This thesis shows two program verification methods for relaxed memory models. Mainly, the methods rely on the SMT-based program verification approach. In both methods, the behavior of program execution and the program property are encoded into a verification condition represented by a first-order formula; the formula is then used to check every execution satisfies the program property. The primary difference between the proposed methods is the way to abstract the behavior of program executions into the verification condition.

In both methods, the program executions are abstracted symbolically. In particular, the computation of program execution is considered in SMT-based program verification. The first method uses the bounded loop unwinding technique to abstract the symbolic executions. In the bounded method, the loop iterations are unwound systematically within a bound. For the second method, the inductive invariant approach is used instead of loop unwinding. However, the proposed inductive invariant method has seemed to be sound for partial store

ordering (PSO) and stronger memory models. For SMT-based program verification, the abstraction of program execution and the program property are encoded regarding the relaxed memory model into a first-order formula. Primarily, the encoded formula is a decidable formula to be solved by an SMT solver automatically. Consequently, the program correctness can be ensured automatically.

In the experiment, an experiment tool was developed, and the Z3 solver is adopted to solve the first-order formula. As a result, the tool can automatically verify the property of the abstraction of concurrent programs on a relaxed memory model. In particular, the abstraction of concurrent programs can represent some essential behaviors of assembly programs. Besides, the bounded method is an under-approximation approach, while the inductive invariant method is an over-approximation approach.

In summary, concurrent assembly programs can be abstracted for ensuring the correctness by our methods. For the bounded method, the program correctness on a relaxed memory model can be ensured if there is no loop. Otherwise, the method can at least disprove the program property on a relaxed memory model. As for inductive invariant method, the correctness of concurrent program contains loop can be ensured on partial store ordering (PSO).

Keywords: Concurrent Program Verification, SMT-based Program Verification, Multi- processors, Relaxed Memory Model, and Automated Program Verification.

論文審査の結果の要旨

本博士論文では，弱いメモリ一貫性モデルに基づいてマルチコア・マルチプロセッサ向けの並列プログラムを検証する手法を提案している．近年の共有メモリマルチプロセッサシステムは，性能向上のために，様々な弱いメモリ一貫性モデルを採用している．しかしながら，その元では，書き込み，読み出し命令の実行順序が，プログラムに記述された順番と異なる場合があるため，正しいプログラムを作成することは容易ではない．そこで，本博士論文では，アセンブリ言語で記述されたプログラムを，弱いメモリ一貫性モデルに基づいて，検証する手法を提案している．

提案手法では，複数種類のプロセッサ向けアセンブリ言語を取り扱うため，Operation Structure と呼ばれる抽象表現を導入している．Operation Structure における読み出し，書き込み命令の実行順序は，弱いメモリ一貫性モデルの影響を受けて決まる．そこで，Operation Structure の振る舞いにおいて，弱いメモリ一貫性モデルを満たす実行パスを見つけるため，SMT ソルバを用いている．Operation Structure は，自動的に，決定可能な論理式に変換され，弱いメモリ一貫性モデルの論理式表現，表明と組み合わせて自動的に検証が実施される．弱いメモリ一貫性モデルには，様々なクラスがあるが，TSO(Total Store Order)，PSO(Partial Store Order)，ARM とシーケンシャルな実行である SC(Sequential Consistency)を取り扱っている．検証法は，ループを有限回展開して検証する手法とループ不変表明を導入して検証する手法を提案している．前者は，効率的に検証が実施できるが，

ループを展開する回数を事前に指定する必要があり，実際のループ回数が，指定した回数より多い場合は，正しく検証が実施されない．後者は，任意回のループの実行においてもプログラムの正しさを保証することができる．弱いメモリ一貫性モデルに基づいた検証法はいくらか提案されているが，カバーしている弱いメモリ一貫性モデルのクラス，不変表明を使用する方法に関して新規性が認められる．実験では，バグの存在が知られている SPARC プロセッサ向け Linux Kernel のスピンロックプログラムにおいて，そのバグの自動的な検出に成功している．また，ARM 向け TOPPERS/FMP カーネルのスピンロックプログラムの自動検証も成功し，その正しさを確認できている．これらのことから，提案手法の有効性が確認されている．

以上，本論文は，形式検証を実践するための理論と手法を提案しており，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に価値があるものと認めた．