氏　　名	TRAN, Nhat Hoa		
学位の種類	博士(情報科学)		
学位記番号	博情第 398 号		
学位授与年月日	平成 30 年 9 月 21 日		
論文題目	Study on Facilitating the Variation of Schedulers in Model Checking		
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
	丹　康雄	同	教授
	平石 邦彦	同	教授
	鈴木 正人	同	准教授
	岡野 浩三	信州大学	准教授

論文の内容の要旨

Software applications play an important role in our lives. The failure of the applications may harm people or equipment. Therefore, the correctness of the software is important. In fact, an application may consist of multiple processes, which are developed based on programming languages and operating systems (OSs). Under the mechanisms provided by these languages and environments, the processes can run simultaneously to increase the scalability. The applications are called concurrent systems. In fact, these systems are error-prone; for example, deadlock, livelock, or violations of constraints may occur in them. Because the processes of a concurrent system can be executed in di erent orders, their behaviors are di cult to verify.

As an exhaustive and automatic technique, model checking explores every execution of a system and automatically nd possible errors. In comparison with other techniques, such as testing and simulation, model checking is more suitable to verify the concurrent systems. To model checking a system, we need to specify its behaviors (usually in a modeling language); then travel all the states of the system (called the state space) represented by its model using a search algorithm to check the corresponding property.

With the increasing of the complexness of a concurrent system, there is a need to schedule the execution of the processes. There are several scheduling strategies applied by real systems. For instance, in OSEK OS for automotive devices, an application can have multiple tasks executed under the priority and mixed preemption strategy. In model checking, the behaviors of a scheduler associate with the algorithm that explores the state space. However, verifying a concurrent system with considering all possible executions (interleaving behaviors) is an over-approximation approach and can produce spurious counterexamples because the errors may occur outside the executions indicated by the scheduler. Therefore, to accurately verify the systems, we need to take the scheduler into account in the veri cation.

Current methods in model checking to deal with sequential/concurrent systems are di cult to apply to verify with scheduling policies because these methods consider a di erent kind of behaviors and can cause spurious counterexamples. To deal with the scheduling policies, existing approaches try to limit the executions of the systems by encoding both of the processes and the scheduler into a model using a modeling language (e.g. Promela). In this case, the scheduling policy needs to be speci ed from scratch. This approach is hard to model interesting schedulers, error-prone, and time-consuming. This means that an approach to easily and exibly describe the scheduling policies is needed.

In reality, the OSs use di erent policies to control the executions of the processes. For example, Linux OS can support several policies for its tasks based on their priorities (e.g. round-robin and rst-in- rst-out). However, the existing approaches cannot deal with the variation of the schedulers because the policy is xed in the model of a system. That means to ensure the accuracy of the concurrent systems, a study on facilitating the variation of schedulers in model checking is necessary and important.

To overcome the problems above, in this research, we propose a method to analyze and verify concurrent systems executed under di erent scheduling policies using model checking techniques. Our method contains three main parts: 1) a language for modeling the processes, 2) a domain-speci c language (DSL) to describe the scheduling policies, and 3) an algorithm to search all of the states of the system.

The originality of this research is proposing a DSL to specify the scheduling policies used in model checking techniques. In this approach, our language aims to provide a high-level support for specifying di erent policies easily. All the information necessary to analyze the system are automatically generated. From the speci cation of the scheduling policy in the DSL, a search algorithm is realized to explore the state space. Following this approach, we implemented a tool named SSpinJa, which is extended from SpinJa, a model checker implemented in Java. Our experiments indicate that the method is practical; it is easy to describe di erent scheduling policies and accurately verify the behaviors of the systems. In addition, in this research, we apply model-based testing techniques to generate the tests to check the correspondence between the policy in our DSL and the real scheduler in an OS; it helps us to increase the con dence of the policy in the DSL and accurately verify the systems.

The impact of this research is that we can easily apply model checking techniques to verify the concurrent systems with the di erent scheduling policies. The state space to be searched is now limited because the scheduler is taken into account in the veri cation. Therefore, we can verify systems more accurately. In addition, with our method, we can reuse the speci cations of the processes and the scheduling policy. It helps to decrease the time necessary for designing and developing a concurrent system.

Key words: concurrent systems, model checking, scheduler, domain-speci c language, model-based testing

論文審査の結果の要旨

この博士論文では，並行システムを対象としたモデル検査において，スケジューラを取り扱う手法について提案している．モデル検査では，並行プロセスの可能な振る舞いの組み合わせを網羅的に探索するが，実際の並行システムでは，並行プロセスの実行はスケジューラにより制御されている．よって，実際の並行システムにおいて，発生しうる正確な振る舞いを探索するためには，モデル検査を実施する際に，スケジューラの振る舞いを含めてモデル化を行う必要がある．しかしながら，スケジューラを含めたモデル化では，モデルの記述が必要以上に大きくなり，モデル化に手間がかかる，スケジューラの状態も探索の対象になり探索状態空間が大きくなる，といった問題を引き起こす．さらに，実際の並行システムでは，様々なスケジューラが用いられており，同一のポリシーで実現されているスケジューラでも，実際の振る舞いは，スケジューラ毎に異なる場合が多い．よって，個々のスケジューラに関する記述を，検証対象が変わる度に準備するのは，コストが高い．そこで，本博士論文では，スケジューラの振る舞いを Domain Specific Language (DSL)により記述し，それに基づいて状態を探索するモデル検査手法を提案している．

提案手法は，1)スケジューラを記述する DSL, 2)提案した DSL 記述に基づいて状態を探索するモデル検査アルゴリズム，3)DSL 記述が実際のスケジューラの挙動にあっていることを確認するテストケースの自動生成手法，により構成されており，さらに，1～3 を実現したツールの実装も行っている．DSL に基づいたモデル検査を中心に，それに関連する一連の技術を実現していると言える．既存研究においては，特定のスケジューラに基づいてモデル検査を実施する，優先度や時間スロットなどのパラメータによりスケジューラを設定してモデル検査を実施する研究はあるが，スケジューラを DSL により記述するのは，本研究が唯一であり，それが新規な点であり，独創的である．既存研究のモデル検査アルゴリズムを拡張し，DSL により記述したスケジューラの挙動に基づいて状態を探索するアルゴリズムを提案しており，モデル検査アルゴリズム自体にも，一定の新規性がある．また，DSL によりスケジューラ記述することにより，既存研究では，効率的に扱えなかった現実的なスケジューラも，効率的に取り扱うことが可能になっており，有効性も確認できている．

以上，本論文は，モデル検査を実践応用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．