Improving Explicit Model Checking for Petri Nets

Dissertation

zur

Erlangung des akademischen Grades

Doktor-Ingenieur (Dr.-Ing.)

der Fakultät für Informatik und Elektrotechnik

der Universität Rostock

vorgelegt von

Dipl.-Inf. Torsten Liebke

geboren am 09.10.1985 in Rostock

aus Rostock

Rostock, 11. Dezember 2020

Abstract

Model checking is the automated verification that systematically checks if a given behavioral property holds for a given model of a system. For modeling, analyzing, and verifying systems with mathematical rigor, formal methods are used, which can be considered as applied mathematics. We use Petri nets and temporal logic as formalisms to describe a system and its behavior in a mathematically precise and unambiguous manner.
Explicit state space verification explores all possible states of a system one by one to prove or disprove the given property. The problem is that the state space usually grows exponentially fast with the size of the model. This is called the state explosion problem and handling it, is a main challenge in model checking today. Although the number of states can be enormous, modern techniques have accomplished the successful verification of large industrial systems due to the greatly reduced size of state spaces. Nevertheless, these techniques are often not powerful enough for some very large and complex systems that we see today. Thus, it is still important to come up with refined or even new methods to reduce the number of states that have to be explored.
The contributions of this thesis are concerned with the improvement of model checking efficiency both in theory and in practice.
At first, we present two new reduction techniques that can be used in a portfolio setting, which means that several methods run in parallel and the fastest successful method determines the runtime. In the category of scalable models, there is usually a parameter to scale over the structure or over the resources of the model. To verify a property in the latter category, it is sometimes sufficient to consider the model with very limited resources. We propose a technique utilizing this idea that considerably reduces the state space. The second reduction technique that we introduce is based on the counterexample guided abstraction refinement (CEGAR) method, which, so far, could only be used for reachability problems. We demonstrate that our CEGAR approach works for a whole class of temporal logic formulas.
For our second contribution, we provide certain supplementary strength re-

ductions, that is, techniques that complement the model checking process. More precisely, we introduce formula simplifications based on structural methods and propose quick checks for the fast verification of certain necessary or sufficient conditions of the input formula. Furthermore, we introduce a new and faster algorithm to compute conflicting transitions.
The third and final contribution of this thesis is all about stubborn sets, a dialect of partial order reduction. In fact, in LTL model checking, the formula is represented by an automaton. We use the rather unexplored idea to utilize all available information from this automaton in order to compute smaller stubborn sets. In addition, we propose specialized stubborn sets for simple and frequently occurring CTL formulas.
When possible, we implemented the proposed algorithms as a proof-of-concept in our explicit model checker LoLA and validated them with the benchmark provided by the annual model checking contest. This includes all but two methods, when implementation was hampered by an internal incompatibility with the architecture of LoLA. Our experiments show that all implemented techniques increase the model checking efficiency.

Zusammenfassung

Model Checking ist die automatisierte systematische Überprüfung, ob eine gegebene Verhaltenseigenschaft für ein gegebenes Modell eines Systems erfüllt ist. Zur Modellierung, Analyse und Verifikation von Systemen mit mathematischer Rigorosität werden formale Methoden verwendet, die als angewandte Mathematik betrachtet werden können. Wir verwenden Petrinetze und temporale Logik als Formalismen, um ein System und sein Verhalten mathematisch präzise und eindeutig zu beschreiben.
Explizite Zustandsraumverifikation untersucht nacheinander alle möglichen Zustände eines Systems, um die gegebene Eigenschaft zu beweisen oder zu widerlegen. Das Problem ist, dass der Zustandsraum in der Regel exponentiell schnell mit der GröSSe des Modells wächst. Dies wird als das Zustandsexplosionsproblem bezeichnet und der Umgang damit ist heutzutage eine der gröSSten Herausforderungen beim Model Checking. Obwohl die Anzahl der Zustände riesig sein kann, haben moderne Techniken die Verifikation groSSer industrieller Systeme ermöglicht. Diese Techniken reduzieren die GröSSe des Zustandsraums erheblich und ermöglichen so die erfolgreiche Verifikation gröSSerer Systeme. Dennoch sind sie für einige sehr groSSe und komplexe Systeme, die wir heutzutage sehen, oft nicht leistungsfähig genug. Daher ist es nach wie vor wichtig, verbesserte oder sogar neue Methoden zu entwickeln, um die Anzahl der zu untersuchenden Zustände zu reduzieren.
Die Beiträge dieser Arbeit beschäftigten sich mit der Verbesserung der Effizienz des Model Checkings sowohl in der Theorie als auch in der Praxis. Zunächst stellen wir zwei neue Reduktionstechniken vor, die in einer Portfolioumgebung eingesetzt werden können, d.h. dass mehrere Methoden parallel laufen und die schnellste erfolgreiche Methode die Laufzeit bestimmt. In der Kategorie der skalierbaren Modelle gibt es normalerweise einen Parameter, der über die Struktur oder über die Ressourcen des Modells skaliert werden kann. Um eine Eigenschaft in dieser Kategorie zu verifizieren, ist es manchmal ausreichend, das Modell mit sehr begrenzten Ressourcen zu betrachten. Wir schlagen eine Technik vor, die diese Idee nutzt und den Zustandsraum erheblich reduziert. Die zweite Reduktionstechnik, die wir

vorstellen, basiert auf der CEGAR-Methode (Counterexample Guided Abstraction Refinement), die bisher nur bei Erreichbarkeitsproblemen eingesetzt werden konnte. Wir zeigen, dass unser CEGAR-Ansatz für eine ganze Klasse von Formeln der temporalen Logik funktioniert.
Unser zweiter Beitrag stellt zusätzliche Stärkereduzierung zur Verfügung, d.h. Techniken, die das Model Checking ergänzen. Genauer gesagt führen wir Formelvereinfachungen ein, die auf strukturellen Methoden basieren, und schlagen Quick Checks zur schnellen Überprüfung bestimmter notwendiger oder hinreichender Bedingungen der Eingabeformel vor. Darüber hinaus führen wir einen neuen und schnelleren Algorithmus zur Berechnung von Konflikt-Transitionen ein.
Im dritten und letzten Beitrag dieser Arbeit geht es um sture Mengen, einem Dialekt der Partial Order Reduction. Beim LTL Model Checking wird die Eingabeformel durch einen Automaten dargestellt. Wir verwenden die eher unerforschte Idee, alle verfügbaren Informationen aus diesem Automaten zu nutzen, um kleinere sture Mengen zu berechnen. Darüber hinaus schlagen wir spezielle sture Mengen für einfache und häufig vorkommende CTL Formeln vor.
Wenn möglich, haben wir die vorgestellten Algorithmen als Proof-of-Concept in unserem expliziten Model Checker LoLA implementiert und dann mit dem Benchmark des jährlichen Model Checking Contests validiert. Dies schlieSSt alle Methoden bis auf zwei ein, bei denen die Implementierung durch eine interne Inkompatibilität mit der Architektur von LoLA erschwert wurde. Unsere Experimente zeigen, dass alle implementierten Techniken die Effizienz des Model Checkings erhöhen.