Formal Verification of Fault-Tolerant Systems

Diss. ETH No. 24152

Formal Verification of Fault-Tolerant Systems

A thesis submitted to attain the degree of

DOCTOR OF SCIENCES of ETH ZURICH

(Dr. sc. ETH Zurich)

presented by

Ognjen Marić

MSc in Computer Science, University of Belgrade
born 29.04.1981
citizen of Bosnia and Herzegovina

accepted on the recommendation of

Prof. Dr. David Basin, examiner
Dr. Stephan Merz, co-examiner
Dr. Christoph Sprenger, co-examiner

2017

Abstract

The goal of formal methods is to enhance our confidence in the correctness of computer systems, by proving that a model of a system obeys a given specification. For software systems, the models are typically built with the assumption that the underlying hardware is perfect. This assumption greatly simplifies both the model and the reasoning, but it is not valid for software designed with the explicit goal of fault-tolerance. In this thesis, we examine the verification problem for such software, in different fault scenarios.

In the first part of the thesis, we focus on crashes (for example, due to power failures) and persistent storage faults (for example, random bit flips) within a single computer system. We formally verify the correctness of the persistent memory manager in IBM's 4765 secure coprocessor, which provides a transactional semantics of persistent memory updates. The inclusion of storage faults is novel in this area and incurs a significant jump in complexity of both the system and its verification. We tackle the resulting verification challenge by a combination of a monad-based model, an abstraction that reduces the system's non-determinism, and stepwise refinement. We derive novel proof rules for handling repeated system restarts and for compositional reasoning that exploits the system's layered structure. The entire development is formalized in the theorem prover Isabelle/HOL.

The second part of the thesis deals with faults in a distributed setting. We consider benign (non-Byzantine) faults: crashes and recoveries of entire nodes, as well as network faults such as message delay and loss. A standard approach to distributed fault-tolerance is node replication. The central issue with replication is ensuring state consistency across replicas. A standard approach exists again: make replicas deterministic, and have them always agree on the next operation to be performed. Achieving such agreement is known as the consensus problem. The problem is typically parameterized in the size of the system, that is, the number of replicas; the solution should work for all values of the parameter. Past research has yielded a large number of algorithms that solve this problem. However, precisely understanding how these algorithms work and proving their correctness is non-trivial due to the complex and failure-prone environment they operate in. In fact, due to their parameterized nature, no fully automated procedure for verifying their correctness exists in the literature. Moreover, many of the algorithms

evidently share similar basic constructs, but it is unclear whether and which underlying ideas are also shared.

Our first main contribution in this area is a development, based on stepwise refinement, which provides an abstract and unified view of a sizable family of consensus algorithms. The models we build provide insights into the main ideas and design choices underlying the different algorithms, and classify them based on those choices. The results guide us in constructing a new consensus algorithm that answers an open theoretical question. All our results are formalized and verified in Isabelle/HOL, yielding precision and strong correctness guarantees.

For our second main contribution in this setting, we again exploit the observation that many algorithms share the same basic constructs. We design a simple but expressive language based on these constructs, and prove a cutoff theorem for consensus algorithms written in the language. Given an algorithm $\mathcal{A}$, the theorem gives a number B (the cutoff bound) and states that the correctness of $\mathcal{A}$ when applied to a system with B processes implies the correctness of $\mathcal{A}$ for any system with any process count. The proof relies on a zero-one principle for the language, an analogue of the same principle for sorting networks. We encode a number of real-world algorithms in the language; for these, the theorem yields small bounds, either 5 or 7. This formalizes the so-called small scope hypothesis for the language. The hypothesis is a well-known informal observation that most bugs in parameterized systems manifest themselves already for small parameter values. It implicitly underlies approaches such as testing and model checking. As a consequence of the main theorem, we show that model checking can be leveraged to provide the first complete and fully automated verification method applicable to consensus algorithms.

Zusammenfassung

Das Ziel formaler Methoden ist es, das Vertrauen in die Korrektheit von Rechnersystemen zu erhöhen, indem bewiesen wird, dass ein Modell des Systems eine vorgegebene Spezifikation erfüllt. Für Softwaresysteme sind die Modelle üblicherweise mit der Annahme gebaut, dass die Hardware fehlerfrei ist. Diese Annahme vereinfacht das Modell und die Beweisführung stark, aber sie gilt nicht für Software, die Fehlertoleranz als ausdrückliches Entwicklungsziel hat. In dieser Dissertation untersuchen wir das Verifikationsproblem für solche Software in verschiedenen Fehlerszenarien.

Im ersten Teil der Dissertation konzentrieren wir uns auf Abstürze (zum Beispiel, durch Stromausfälle) und Fehler des persistenten Speichers (zum Beispiel, durch zufällige Bitveränderungen, sogenannte "bit flips") innerhalb von einzelnen Rechnersystemen. Wir beweisen formell die Korrektheit der Steuerungssoftware des persistenten Speichers in einem IBM 4765 Coprozessor, der über Speicheroperationen mit atomarer Semantik verfügt. Die Beurteilung der Speicherfehler ist neuartig auf diesem Gebiet, und bringt einen wesentlichen Sprung in der Komplexität des Systems und dessen Verifikation mit sich. Wir gehen die dadurch entstandene Herausforderung für die Verifikation an, indem wir die folgenden Elemente kombinieren: ein Modell, das auf Monaden basiert, eine Abstraktion die den Nichtdeterminismus des Systems reduziert, und schrittweise Verfeinerung. Wir leiten neuartige Beweisregeln ab, um mit mehrmaligen Neustarten umzugehen, und um kompositionelle Beweisführung zu ermöglichen, die die geschichtete Struktur des Systems ausnutzt. Die gesamte Entwicklung wurde im Theorembeweiser Isabelle/HOL formalisiert.

Der zweite Teil der Dissertation untersucht Fehler in einem verteilten Szenario. Wir untersuchen nicht-bösartige Fehler, wie Abstürze und Neustarts ganzer Knoten, sowie Nachrichtenverluste oder Nachrichtenverzögerungen im Netzwerk. Ein üblicher Ansatz für verteilte Fehlertoleranz ist die Replikation der Knoten. Das Hauptproblem dabei ist die Konsistenz des Zustands aller Replikate. Auch hier existiert ein üblicher Ansatz: wenn alle Replikate deterministisch sind, reicht es wenn sie in jedem Schritt übereinstimmen was die nächste Operation ist. Das Problem dieser Übereinstimmung wird das Konsensproblem genannt. Das Problem wird typischerweise mit der Systemgrösse, das heisst, der Anzahl der Replikate, parametrisiert; eine

Lösung soll für einen beliebigen Parameterwert funktionieren. Die bisherige Forschung hat eine grosse Anzahl Algorithmen ergeben, die dieses Problem lösen. Allerdings ist die Komplexität dieser Algorithmen und deren Beweisführung, aufgrund ihrer komplexen und fehleranfälligen Szenarien, nicht trivial. Begründet durch die oben genannte Parametrisierung gibt es keine vollständig automatisierte Prozedur, um die Korrektheit dieser Algorithmen zu beweisen. Es ist ausserdem offensichtlich, dass viele solche Algorithmen ähnliche Grundkonstrukte benützen, allerdings ist es unklar ob und welche gemeinsame Grundideen vorliegen.

Unser erster Beitrag auf diesem Gebiet ist eine Entwicklung, die auf schrittweiser Verfeinerung basiert ist, und einen abstrakten und einheitlichen Blick auf eine beträchtliche Anzahl von Konsensalgorithmen liefert. Die Modelle, die wir bauen, gewähren Einblick in die Grundideen und die Entwurfsentscheidungen der verschiedenen Algorithmen, und klassifizieren sie entsprechend. Die Resultate leiten uns zur Konstruktion eines neuen Konsensalgorithmus, der eine offene theoretische Frage beantwortet. All unsere Resultate sind formalisiert und verifiziert in Isabelle/HOL, sodass Präzision und Korrektheit garantiert sind.

Für unseren zweiten Hauptbeitrag in diesem Gebiet nutzen wir wieder die Tatsache aus, dass viele Algorithmen die gleichen Grundkonstrukte benützen. Wir gestalten eine einfache, aber ausdrucksstarke Sprache basierend auf diesen Konstrukten, und beweisen ein sogenanntes “Cutoff” Theorem für Algorithmen, die in dieser Sprache geschrieben sind. Für einen vorgegebenen Algorithmus $\mathcal{A}$ ergibt das Theorem eine Zahl B (die “cutoff” Schranke) und besagt, dass die Korrektheit $\mathcal{A}$s, angewendet auf ein System mit B Knoten, auch die Korrektheit $\mathcal{A}$s impliziert, wenn $\mathcal{A}$ auf ein beliebiges System mit einer beliebigen Anzahl Knoten angewendet wird. Der Beweis baut auf einem Null-Eins Prinzip, welches analog zum selben Prinzip für Sortiernetze ist, auf. Wir modellieren mehrere bereits bekannte Algorithmen mit der neu eingeführten Sprache; für diese ergibt das Theorem kleine Schranken, entweder 5 oder 7. Damit wurde die sogenannte “Small Scope” Hypothese für diese Sprache bewiesen. Die Hypothese ist eine informelle Bemerkung und sagt aus, dass viele Fehler parametrisierter Systeme schon bei kleinen Parameterwerten auftreten. Sie schafft eine implizite Grundlage für Ansätze wie Testen und Model Checking. Als eine Folge des Haupttheorems zeigen wir, dass Model Checking eingesetzt werden kann, um die erste, komplette und völlig automatisierte Verifikationsmethode zu kreieren, die auf Konsensalgorithmen anwendbar ist.