Doctoral Thesis ETH No. 16268

Liveness Checking as Safety Checking to Find Shortest Counterexamples to Linear Time Properties

A dissertation submitted to the
SWISS FEDERAL INSTITUTE OF TECHNOLOGY ZURICH
(ETH ZÜRICH)

for the degree of
Doctor of Technical Sciences

presented by
Viktor Schuppan
Dipl.-Inf. Univ. TU München
born September 5, 1973
citizen of Germany

accepted on the recommendation of
Prof. Dr. Armin Biere, examiner
Prof. Dr. Daniel Kröning, co-examiner

2005

Abstract

Temporal logic is widely used for specifying hardware and software systems. Typically two types of properties are distinguished, safety and liveness properties. While safety can easily be checked by reachability analysis, and many efficient checkers for safety properties exist, more sophisticated algorithms have always been considered to be necessary for checking liveness. In this dissertation we describe an *efficient translation of liveness checking problems into safety checking problems* for finite state systems. More precisely, fair repeated reachability in a fair Kripke structure K is formulated as reachability in a transformed Kripke structure $K^{\mathbf{S}}$. A fair loop in K is detected in $K^{\mathbf{S}}$ by saving a previously visited state in an additional state-recording component, waiting until a fair state has been seen, and checking a loop closing condition. The approach extends to all ω-regular properties. We show that the size of the state space, the reachable state space, the transition relation, and its transitive closure grow by a factor of $|S|$ in the transformed model, where $|S|$ is the size of the state space in the original model. Radius and diameter increase by a small, constant factor. We discuss optimizations that limit the overhead of our translation. We have implemented the approach for BDD-based model checkers of the SMV family. Experimental results show not only that the approach is feasible for complex examples, but that it may lead to faster verification if the property turns out to be false. For one example even an exponential speed-up can be observed. We finally show that a similar reduction can be applied to a number of infinite state systems, namely, (ω-) regular model checking, pushdown systems, and timed automata.

Counterexamples as produced by a model checker for a failing property help developers to understand the problem in a faulty design. The shorter a counterexample, the easier it is typically to understand. The length of a counterexample, as reported by a model checker, depends on both the algorithm used for state space exploration and the way the property is encoded. We provide *necessary and sufficient criteria for a Büchi automaton to accept shortest counterexamples*. Extending a notion introduced by Kupferman and Vardi we call a Büchi automaton that accepts shortest counterexamples tight. We prove that Büchi automata constructed using the approach of Kesten et al. (KPR), which is essentially the same as the construction by Lichtenstein and Pnueli, are tight for future time LTL formulae, while an automaton generated with the algorithm of Gerth et al. (GPVW) may lead to unnecessary long counterexamples. Optimality is lost in the first case as soon as past time operators are included. We show that potential excess length is in both cases at most linear in the length of the specification. Using a recently proposed encoding for bounded model checking of LTL with past by Latvala et al., we construct a Büchi automaton that accepts shortest counterexamples for full LTL. The construction adapts the idea of virtual unrolling by Benedetti and Cimatti to Büchi automata. Its generalization gives a method to make an arbitrary Büchi automaton accept shortest counterexamples. We use our method of translating liveness into safety to find shortest counterexamples with a BDD-based symbolic model checker without modifying the model checker itself. Though

our method involves a quadratic blowup of the state space, it proves to be competitive with SAT-based bounded model checking. Experimental results show that using a model checking algorithm that finds shortest cycles contributes much more to a reduction in counterexample length than using an automaton that accepts shortest counterexamples when compared with the automaton by Kesten et al.

Zusammenfassung

Temporale Logik wird häufig für die Spezifikation von Hardware- und Softwaresystemen eingesetzt. Dabei wird oft zwischen Sicherheits- und Lebendigkeitseigenschaften unterschieden. Während Sicherheitseigenschaften einfach mittels Erreichbarkeitsanalyse überprüft werden können, wird Verifikation von Lebendigkeitseigenschaften meist mit spezialisierten Algorithmen in Verbindung gebracht. Dementsprechend sind mehr effiziente Werkzeuge zur Überprüfung von Sicherheitseigenschaften verfügbar als zur Überprüfung von Lebendigkeitseigenschaften. In der vorliegenden Dissertation wird eine Übersetzung entwickelt, die das Problem der Verifikation einer Lebendigkeitseigenschaft für Systeme mit endlich vielen Zuständen in das der Verifikation einer Sicherheitseigenschaft überführt. Genauer gesagt wird das Problem der Erreichbarkeit eines Zustandes s von sich selbst über einen fairen Pfad (faire wiederholte Erreichbarkeit) in einer fairen Kripke Struktur K in das Problem der (einfachen) Erreichbarkeit eines Zustandes $s^{\mathbf{S}}$ in einer anderen Kripke Struktur $K^{\mathbf{S}}$ übersetzt. Ein fairer Zyklus in K kann in $K^{\mathbf{S}}$ gefunden werden, indem zunächst nicht-deterministisch eine Kopie des momentanen Zustands s in einer separaten Version der Zustandsvariablen abgelegt wird. Sobald zunächst ein fairer und dann ein zu s identischer Zustand besucht worden sind, liegt ein fairer Zyklus vor. Die Übersetzung kann zur Überprüfung beliebiger ω-regulärer Eigenschaften verwendet werden. Die Größe des Zustandsraumes, des erreichbaren Zustandsraumes, der Übergangsrelation sowie ihrer transitiven Hülle im transformierten System wachsen proportional zur Anzahl Zustände im Originalsystem. Radius und Durchmesser ändern sich nur um einen kleinen, konstanten Faktor. Die Übersetzung wurde für Modellprüfer der SMV Familie implementiert. Es werden außerdem Optimierungen vorgestellt, die den Anstieg der Komplexität begrenzen helfen. Experimente zeigen, daß sich die transformierten Systeme nicht nur mit akzeptablem Aufwand verifizieren lassen, sondern daß sich Gegenbeispiele im transformierten System sogar manchmal schneller finden lassen als im Originalmodell. Ein Beispiel kann dabei sogar exponentiell schneller überprüft werden. Schließlich wird die Übersetzung auf einige Systeme mit unendlichem Zustandsraum erweitert, im einzelnen reguläre Modellprüfung, Kellerautomaten und Realzeitautomaten.

Die Gegenbeispiele, die ein Modellprüfer bei Fehlschlagen einer Spezifikation ausgibt, haben sich als sehr hilfreich bei der Fehlersuche erwiesen. Ein Gegenbeispiel ist umso einfacher zu verstehen, je kürzer es ist. Die Länge des ausgegebenen Gegenbeispiels hängt dabei sowohl vom Modellprüfungsalgorithmus als auch von der Repräsentation der Spezifikation als Büchi Automat ab. Es werden notwendige und hinreichende Kriterien entwickelt, die gewährleisten, daß ein Büchi Automat kürzeste Gegenbeispiele erkennt. Der Begriff der tightness von Kupferman und Vardi wird auf Büchi Automaten erweitert. Es wird gezeigt, daß ein Büchi Automat, der mit dem Algorithmus von Kesten et al. (KPR) konstruiert wird, kürzeste Gegenbeispiele für LTL eingeschränkt auf Zukunft erkennt. Im Gegensatz dazu führt der Algorithmus von Gerth et al. (GPVW) zu unnötig langen Gegenbeispielen. Die Optimalität bei KPR geht verloren, sobald

die Einschränkung auf Zukunft aufgehoben wird. Es konnte gezeigt werden, daß in beiden Fällen eine mögliche Überlänge höchstens linear in der Länge der Spezifikation ist. Durch Anpassung einer kürzlich vorgestellten Kodierung für Bounded Model Checking mit Vergangenheit von Latvala et al. wird ein Büchi Automat konstruiert, der kürzeste Gegenbeispiele für ganz LTL erkennt. Die Konstruktion überträgt die Idee des virtuellen Ausrollens von Benedetti und Cimatti auf Büchi Automaten. Ihre Verallgemeinerung ermöglicht es, einen beliebigen Büchi Automaten so zu verändern, daß er kürzeste Gegenbeispiele akzeptiert. Die oben beschriebene Übersetzung von Lebendigkeits- in Sicherheitseigenschaften kann nun benutzt werden, um kürzeste Gegenbeispiele mit einem BDD-basierten symbolischen Modellprüfer zu finden, ohne dabei den Programmkode des Modellprüfers selbst zu verändern. Trotz des quadratischen Wachstums im Zustandsraum werden Gegenbeispiele ähnlich schnell gefunden wie von einem SAT-basierten Pfadlängen-begrenzten Modellprüfer. Experimente zeigen, daß die Verkürzung in der Länge der Gegenbeispiele, die durch den Einsatz eines Modellprüfungsalgorithmus zum Finden kürzester Zyklen erreicht wird, weit größer ist als die, die aus dem Einsatz eines Büchi Automaten für kürzeste Gegenbeispiele resultiert.