Symbolische Methoden für die probabilistische Verifikation
– Zustandsraumreduktion und Gegenbeispiele –

Ralf Wimmer
Lehrstuhl für Rechnerarchitektur, Technische Fakultät
Albert-Ludwigs-Universität Freiburg
Georges-Köhler-Allee 51, 79110 Freiburg im Breisgau
wimmer@informatik.uni-freiburg.de

Abstract: Ein bekanntes Hindernis für die formale Verifikation von Systemen bildet die potentiell stark anwachsende Größe des Zustandsraums, genannt „Zustandsraumexplosion“. Dieses Problem konnte für digitale Schaltungen durch den Einsatz symbolischer Methoden zufriedenstellend gelöst oder zumindest entscheidend entschärft werden. Für probabilistische Systeme, die als Markow-Kette oder Markow-Entscheidungsprozess modelliert sind, brachte die direkte Übertragung dieser symbolischen Methoden bisher keinen Durchbruch.

In dieser Arbeit stellen wir zwei neue Ansätze vor, mit denen Markow-Modelle mit sehr großen Zustandsräumen verifiziert werden können. Die erste Methode ist ein symbolisches Verfahren zur Vorverarbeitung: Zu jedem Markow-Modell berechnen wir mit rein symbolischen Verfahren das kleinste Modell, das in den interessierenden Eigenschaften mit dem Original-Modell übereinstimmt. Die Verifikation kann danach auf dem minimierten Modell durchgeführt werden. Das zweite offene Problem, das in der Dissertation gelöst wird, ist die symbolische Berechnung von Gegenbeispielen, wenn Sicherheitseigenschaften von Markow-Ketten mit diskreter Zeit verletzt sind. Anhand von Experimenten wird gezeigt, dass die neu entwickelten Verfahren den bisher verfügbaren Verfahren hinsichtlich der Laufzeit bzw. der Größe der handhabbaren Systeme deutlich überlegen sind.

1 Einführung

Der Einsatz von computergesteuerten Systemen in sicherheitskritischen Umgebungen (wie beispielsweise in Fahrzeugen oder für medizinische Zwecke) und ihre gleichzeitig immer größer werdende Komplexität machen es unerlässlich, die zuverlässige und korrekte Funktionsweise des jeweiligen Systems sicherzustellen. Allerdings haben praktisch relevante Systeme längst eine Komplexität erreicht, die es unmöglich macht, mehr als einen verschwindend geringen Anteil aller Ablaufmöglichkeiten durch Simulation zu überprüfen. Deshalb wurden formale Methoden, sogenannte Model-Checking-Algorithmen, entwickelt, mit denen man Systeme automatisch auf vorgegebene Eigenschaften prüfen kann. Dabei wird auf clevere Weise der gesamte Zustandsraum eines Systems systematisch durchsucht, so dass damit – im Gegensatz zur Simulation – auch die Fehlerfreiheit bewiesen werden kann. Während erste Verfahren noch auf kleine Systeme mit relativ wenigen Zuständen beschränkt waren, verhalfen symbolische Methoden dem Model Checking für digitale Schaltungen zum Durchbruch [BCM$^+$92]. Symbolische Methoden stellen den Zustandsraum eines Systems nicht durch Aufzählen der Zustände und der Übergänge dazwischen dar, sondern beispielsweise als Lösungen einer Formel oder als Pfade in einem Entschei-

dungsdiagramm (engl. Ordered Binary Decision Diagram, OBDD). Dieser sind in vielen Fällen deutlich kompakter als die Aufzählung aller Zustände und Zustandsübergänge.

Die Herausforderungen der neuesten Zeit ergeben sich durch den Einsatz von Mikroprozessoren in eingebetteten Systemen, die kontinuierliche Größen messen und diese digital verarbeiten (sogenannte hybride Systeme) oder die in unsicheren Umgebungen arbeiten: So werden z. B. Nachrichten über unzuverlässige Kanäle verschickt und können verloren gehen, randomisierte Protokolle werden eingesetzt und Daten kommen mit zeitlichen Schwankungen an. Solche stochastischen Systeme werden oft als Markow-Ketten oder Markow-Entscheidungsprozesse modelliert. Um für derartige Systeme Fragen nach der Zuverlässigkeit oder Verfügbarkeit beantworten zu können, bildete in den letzten zwei Jahrzehnten die Entwicklung von Model-Checking-Algorithmen für stochastische Systeme (siehe z. B. [HJ94, BHHK03]) einen Schwerpunkt in der Verifikationsforschung.

Ein Problem, das die Anwendung dieser Algorithmen auf reale Systeme schwierig macht, ist deren Anzahl an Zuständen, die im Allgemeinen exponentiell in der Zahl der Systemkomponenten wächst. Die symbolischen Methoden, die OBDDs zur Darstellung der Zustandsräume verwenden, wurden zum Teil auf Markow-Modelle verallgemeinert, skalierten jedoch nicht im selben Maße für Markow-Ketten wie für Schaltkreise. Für einige Modellklassen wie beispielsweise interaktive Markow-Ketten (IMCs) oder Markow-Entscheidungsprozesse mit kontinuierlicher Zeit (CTMDPs) sind bis heute keine symbolischen Verfahren verfügbar. Darüber hinaus ist es generell nicht möglich, mit den üblichen Verfahren für Markow-Ketten gleichzeitig zum Ergebnis der Eigenschaftsprüfung ein Gegenbeispiel zu erhalten, wenn eine gewünschte Eigenschaft verletzt ist. Zwar wurden im Wesentlichen seit 2003 verschiedene Verfahren vorgeschlagen, mit denen für Markow-Ketten Gegenbeispiele erzeugt werden können (beispielsweise [HKD09, AL10]), allerdings beruhen all diese Verfahren auf einer expliziten Darstellung des Zustandsraums und sind daher auf verhältnismäßig kleine Systeme beschränkt; gut skalierende symbolische Verfahren waren bis jetzt nicht verfügbar.

Das Ziel der Dissertation war folglich, symbolische Verfahren zu entwickeln, die zum einen die Eigenschaftsprüfung für größere Systeme ermöglichen und es zum anderen gestatten, für Markow-Modelle mit sehr großen Zustandsräumen Gegenbeispiele zu erzeugen. Entsprechend ist die Dissertation in zwei Teile gegliedert: Im ersten Teil wird ein Verfahren zur *symbolischen Minimierung* einer ganzen Reihe von Markow-Modellen vorgestellt. Dabei können verschiedene Klassen von interessierenden Eigenschaften im minimierten System erhalten bleiben: Beispielsweise erhält man unterschiedliche minimale Systeme abhängig davon, ob die Wahrscheinlichkeit, innerhalb einer vorgegebenen Anzahl von Schritten eine Menge von Zuständen zu erreichen, erhalten bleiben soll oder lediglich die Wahrscheinlichkeit, irgendwann eine solche Zustandsmenge zu erreichen (unabhängig von der Anzahl der Schritte). Das formale Mittel für die Minimierung sind Äquivalenzrelationen auf dem Zustandsraum eines Systems, die *Bisimulationen* genannt werden. Dabei werden Zustände, die schrittweise dasselbe beobachtbare Verhalten zeigen, als äquivalent angesehen. Indem man die gröbste Bisimulation berechnet und dann zum Quotientensystem übergeht, dessen Zustände gerade den Äquivalenzklassen der Bisimulation entsprechen, erhält man das kleinste System, das dasselbe beobachtbare Verhalten wie das Originalsystem zeigt. Nach der Minimierung kann deshalb die vorgegebene Eigenschaft auf dem minimierten System überprüft werden, das in vielen Fällen deutlich kleiner als das Originalsystem ist. Der

vorgestellte Algorithmus bildet ein einheitliches Framework, das nicht nur verschiedene Systemklassen – beschriftete Transitionssysteme, Markow-Ketten mit diskreter und kontinuierlicher Zeit sowie interaktive Markow-Ketten – minimieren, sondern auch diverse Minimierungskriterien berücksichtigen kann und leicht auf weitere Kriterien erweiterbar ist. Es wird sowohl die Laufzeit als auch der Speicherplatzbedarf optimiert, und es wird eine umfangreiche Anwendung auf die Analyse sicherheitskritischer Systeme vorgestellt, für die der entwickelte Minimierungsalgorithmus eine erfolgreiche Analyse ermöglicht.

Im zweiten Teil der Arbeit wird gezeigt, wie mit symbolischen Methoden *Gegenbeispiele* für Markow-Ketten mit diskreter Zeit (DTMCs) berechnet werden können. Wir erweitern dazu eine Methode namens *Bounded Model Checking* (BMC) [BCC$^+$03], die bereits industriell sehr erfolgreich zur Fehlersuche in digitalen Schaltungen eingesetzt wird. Dabei wird die Existenz von Pfaden vorgegebener Länge, die eine Sicherheitseigenschaft verletzen, als propositionales Erfüllbarkeitsproblem formuliert. Jede Lösung des Problems entspricht genau einem Systemablauf, der zu einem Fehler führt. Während für digitale Schaltungen in der Regel ein einzelner solcher Ablauf ausreicht, um eine Sicherheitseigenschaft („Es wird nie ein sicherheitskritischer Zustand erreicht“) zu widerlegen, sind bei Sicherheitseigenschaften für DTMCs („Die Wahrscheinlichkeit, einen sicherheitskritischen Zustand zu erreichen, ist höchstens λ“) Mengen von Abläufen nötig, deren gemeinsame Wahrscheinlichkeitsmasse die vorgegebene Schranke λ überschreitet. Es wird beschrieben, wie man BMC auf DTMCs anwenden kann, um effizient ein kompaktes Gegenbeispiel zu erzeugen. Experimente zeigen, dass das resultierende Verfahren in der Lage ist, deutlich größere Systeme zu verarbeiten als die bisherigen Verfahren.

Im Folgenden werden kurz die wichtigsten Verfahren und Ergebnisse, die im Rahmen dieser Dissertation entwickelt wurden, zusammengefasst. Für die Details wird auf die Dissertation und die Konferenz- und Zeitschriftenbeiträge, in denen die Ergebnisse publiziert wurden, verwiesen.

2 Symbolische Zustandsraumreduktion

In diesem Abschnitt stellen wir das neu entwickelte symbolische Minimierungsverfahren vor, das den ersten Teil der Dissertation bildet. Grundlage dafür bildet ein Algorithmus von Blom und Orzan [BO05a, BO05b], der die Minimierung von beschrifteten Transitionssystemen (LTSs) bezüglich starker und branching Bisimulation gestattet. Man beginnt mit einer initialen Partition des Zustandsraums, die entweder durch Zustandsbeschriftungen vorgegeben sein kann oder andernfalls einen einzelnen Block mit allen Zuständen enthält. Der Algorithmus beruht darauf, alle Zustände anhand einer Signatur so zu charakterisieren, dass Zustände mit verschiedenen Signaturen nicht äquivalent sein können. Eine neue Einteilung der Zustände in Klassen erfolgt durch Aufspalten der Blöcke der aktuellen Partition gemäß den Signaturen. Man iteriert diesen Vorgang so lange, bis ein Fixpunkt erreicht ist. Das Ergebnis ist die gröbste starke bzw. branching Bisimulation, welche die Anfangspartition verfeinert. Dieser Algorithmus wurde für den Einsatz in einer verteilten Umgebung entwickelt und verwendet explizite Darstellungen des Zustandsraums.

Zunächst wird in Kapitel 3 der ursprüngliche Algorithmus dahingehend erweitert, dass er nicht nur in der Lage ist, die starke und die branching Bisimulation für LTSs zu ver-

wenden, sondern im Wesentlichen alle Minimierungskriterien, die in der Literatur eine Rolle spielen, nämlich zusätzlich die schwache, orthogonale, safety-, η-, delay- und progressing Bisimulation [WHH$^+$06]. Außerdem kann bei der Minimierung unterschiedliches Divergenzverhalten der Zustände berücksichtigt werden, sofern dies nicht bereits durch die Definition der Bisimulation erfolgt. Die Dissertation liefert einen Beweis für die Korrektheit des Verfahrens für all diese Bisimulationstypen. Das Verfahren kann bei Bedarf leicht um weitere Bisimulationen erweitert werden, indem jeweils eine geeignete Signatur formuliert wird.

In dieser Form verwendet der Minimierungsalgorithmus immer noch explizite Darstellungen der Zustandsräume und ist dadurch auf Systeme beschränkt, die klein genug sind, um in den Hauptspeicher zu passen. Er wird nun so modifiziert, dass er ausschließlich mit symbolischen Datenstrukturen in Form von OBDDs arbeitet. Dazu werden Methoden entwickelt, bei denen die Laufzeit der Berechnungen nicht mehr direkt von der Größe des dargestellten Systems abhängt, sondern nur noch von der Größe der Darstellung. Letztere kann – und ist es auch in vielen praktischen Fällen – sehr viel kleiner sein als eine explizite Darstellung. Den Kern des symbolischen Algorithmus bildet dabei eine geschickte Partitionsdarstellung, die eine effiziente Berechnung der Signaturen und der Verfeinerung der aktuellen Partition gestattet. Eine ganze Reihe von Optimierungen – zum Beispiel auf die Partitionsdarstellung angepasste BDD-Operationen zum Zugriff auf die Blöcke der aktuellen Partition; das Auslassen von Blöcken der Partition bei der Verfeinerung, von denen festgestellt werden kann, dass sie in der aktuellen Iteration nicht aufgespaltet werden können; und die Anpassung der Signaturen für eine effizientere Berechnung – reduzieren die Laufzeit beträchtlich. Dadurch ergibt sich ein flexibles Verfahren zur Minimierung von LTSs, das durch den Einsatz symbolischer Datenstrukturen mit deutlich größeren Systemen umgehen kann als Konkurrenzverfahren und das hinsichtlich der Laufzeit konkurrierenden symbolischen Verfahren wie dem von Bouali und de Simone [Bd92] weit überlegen ist.

Während in Kapitel 3 Verfahren für beschriftete Transitionssysteme entwickelt wurden, die keine stochastischen Transitionen besitzen, folgt in Kapitel 4 die Erweiterung der Techniken auf Markow-Ketten mit diskreter (DTMCs) und mit kontinuierlicher Zeit (CTMCs) sowie interaktive Markow-Ketten (IMCs), welche die stochastischen Übergänge der CTMCs mit den interaktiven Transitionen der LTSs kombinieren. Es werden geeignete Signaturen definiert für alle drei stochastischen Systemklassen, und die symbolische Verfeinerung wird so angepasst, dass sie im Falle von IMCs mit Paaren von Signaturen arbeitet. Schwierigkeiten bereiten dabei numerische Instabilitäten [WB10]. Aufgrund von Rundungsfehlern, die bei Verwendung der üblichen Gleitkommaarithmetik auftreten, ergeben sich in den Signaturen von eigentlich äquivalenten Zuständen kleine Unterschiede, die dazu führen, dass sie in unterschiedliche Äquivalenzklassen gelangen. Als Lösungsmöglichkeiten entwickelten wir neben einer Implementierung, die mit rationaler Arithmetik arbeitet, eine weitere Version, bei der die Übergangsraten zwischen Zuständen als reine Symbole behandelt, d. h. nicht als Zahlen interpretiert werden. Dadurch werden nur noch natürliche Zahlen zum Zählen der Kanten mit gleichem Symbol verwendet, was ohne Rundungsprobleme erfolgen kann. Dadurch erhält man als Ergebnis die kleinste Markow-Kette, deren Verhalten für alle möglichen Wahlen der Ratensymbole mit der ursprünglichen Markow-Kette übereinstimmt. Der große Vorteil dabei ist, dass nach der Minimierung die konkreten Werte der Symbole beliebig angepasst werden können, ohne den Quotienten neu berechnen zu müssen. Benötigt man für die konkreten Werte das minimale System, kann dies durch erneutes Minimieren

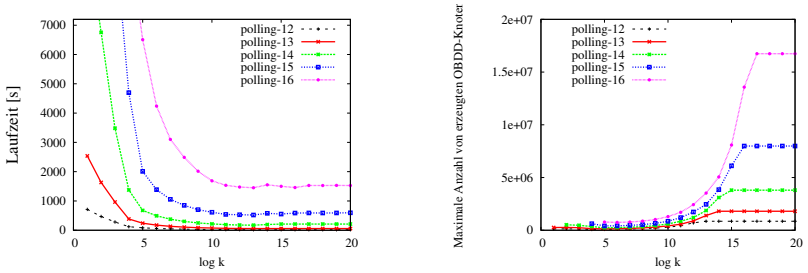

Abbildung 1: Abhängigkeit von Laufzeit (links) und Speicherplatz (rechts) des hybriden Verfahrens von der Zahl k der gleichzeitig verfeinerten Blöcke

des interpretationsunabhängigen Quotientensystems erhalten werden.

Experimente mit unserem Tool SIGREF und verschiedenen Fallstudien zeigten Folgendes: (1) Bei Verwendung von Gleitkommaarithmetik werden in vielen Fällen zu feine Quotientensysteme berechnet und in einem Fall terminierte sogar die Minimierung wegen der numerischen Probleme nicht. (2) Die Verwendung rationaler Arithmetik ist nur unwesentlich teurer als Gleitkommaarithmetik, da der Hauptteil der Rechenzeit nicht für arithmetische Operationen verwendet wird, sondern beispielsweise für Cache-Lookups, auf welche die rationale Arithmetik keinen Einfluss hat. (3) Die Behandlung der Raten als reine Symbole erzeugte in den untersuchten Fällen interpretationsunabhängige Quotienten, die nicht wesentlich größer sind als die mit festen Werten berechneten Systeme. Der Größenunterschied betrug in der Regel weniger als 10 %. Auch die Laufzeit ist mit den Zeiten der beiden anderen Varianten vergleichbar. Damit sind die numerischen Probleme ohne wesentliche Laufzeit- und Speicherplatzeinbußen gelöst.

Kapitel 5 optimiert den Speicherverbrauch bei der Minimierung. Derisavi stellte einen alternativen symbolischen Algorithmus zur Minimierung von CTMCs vor, der eine andere Partitionsdarstellung verwendet [Der07]. Diese besteht aus maximal $\lceil \log_2 n \rceil$ OBDDs, wobei n die Zahl der Zustände der CTMC ist. Experimente haben gezeigt, dass der Speicherverbrauch dieses Algorithmus in vielen Fällen deutlich geringer ist der als von SIGREF; dafür ist SIGREFs Laufzeit um Größenordnungen kleiner als die von Derisavis Verfahren. Aus diesem Grund haben wir in Kooperation mit Derisavi ein hybrides Verfahren entwickelt, das die Vorteile beider Algorithmen vereint [WDH10]. Die Idee ist, die gesamte Partition in der kompakten Darstellung von Derisavi zu halten. Daraus werden Gruppen von k Blöcken – k ist ein Parameter – extrahiert, in SIGREFs effizient zu verfeinerndes Format gebracht, verfeinert und zurück in die kompakte Darstellung gebracht. Die verfeinerten Blöcke werden durch das Ergebnis der Verfeinerung ersetzt. Dies wird wiederholt, bis sich keine Änderungen der Partition mehr ergeben. Die Wahl von k bestimmt, wie viel Speicherplatz und Laufzeit benötigt werden (siehe Abbildung 1). Je kleiner k ist, d. h. je weniger Blöcke in einem Schritt verfeinert werden, desto geringer ist der Platzbedarf und desto größer die Laufzeit. Für $k = 1$ enthält man im Wesentlichen Derisavis Verfahren, für $k = n$ unser signaturbasiertes Verfahren von SIGREF. Damit kann der schon den Speicherplatz reduziert werden, aber das Ziel ist, den Wert von k automatisch so einzustellen, dass die Laufzeit minimal ist, ohne dass der verfügbare Speicherplatz überschritten wird. Dazu wird folgende Strategie verwendet: Man beginnt mit dem SIGREF-Verfeinerungsverfahren, bis der zur

Verfügung stehende Speicherplatz nicht mehr ausreicht. Dann wird die zuletzt erfolgreich verfeinerte Partition gesichert und die aktuelle Iteration abgebrochen. Der Wert von k wird nun so gewählt, dass man zwei Gruppen von Blöcken erhält. Die Verfeinerung wird mit dem hybriden Verfahren und der zuletzt erfolgreich berechneten Partition fortgesetzt. Jedesmal, wenn jetzt zu viel Speicherplatz benötigt wird, wird die aktuelle Iteration abgebrochen, der Wert von k halbiert und die Berechnung mit der zuletzt abgespeicherten Partition fortgesetzt. Falls bei $k = 1$ der verfügbare Speicher nicht ausreicht, kann die Minimierung mit dem zur Verfügung stehenden Speicher nicht durchgeführt werden. Die Ergebnisse zeigen, dass für kleine CTMCs, bei denen der Speicherverbrauch keine Einschränkung darstellt, die Effizienz von SIGREF erreicht wird. Bei großen Modellen jedoch, bei denen die Minimierung mit dem SIGREF-Algorithmus scheitert, ist der hybride Algorithmus in der Lage, die Minimierung durchzuführen. Dabei ist er z. T. um Größenordnungen schneller als der Algorithmus von Derisavi. Damit ist das Ziel erreicht, die Vorteile beider Algorithmen – die Laufzeiteffizienz von SIGREF und die Speichereffizienz von Derisavis Algorithmus – zu kombinieren.

Der erste Teil der Arbeit endet in Kapitel 6 mit einer Anwendung der symbolischen Bisimulationsminimierung, in der für industrielle Statechart-Modelle Wahrscheinlichkeiten für die zeitbeschränkte Erreichbarkeit von sicherheitskritischen Zuständen [BHH$^+$09] berechnet werden. Statecharts sind ein weit verbreiterter Formalismus zur Systemmodellierung und als Teil von UML 2 standardisiert. Da Statecharts an sich keine stochastischen Informationen enthalten, wird es dem Benutzer ermöglicht, Ereignisse zu markieren, die mit einer zeitlichen Verteilung auftreten. Im nächsten Schritt wird aus dem Statechart-Modell ein symbolisch dargestelltes LTS erzeugt, dessen Transitionsbeschriftungen gerade den markierten Ereignissen entsprechen. Die übrigen Ereignisse werden als nicht beobachtbar angesehen. Dieses Transitionssystem wird mit Hilfe unseres symbolischen Algorithmus verkleinert, indem wir den Quotienten der branching Bisimulation berechnen. Nun reichert man das minimierte Transitionssystem um die Zeitinformation an. Das Ergebnis ist eine interaktive Markow-Kette. Im Allgemeinen ist sie zu groß für die Weiterverarbeitung. Deshalb folgt im Anschluss nochmals eine Minimierungsphase – diesmal mit der branching Bisimulation für IMCs. Zum Zeitpunkt, als diese Arbeit entstanden ist, war noch kein Verfahren – weder ein symbolisches noch ein explizites – verfügbar, um direkt auf IMCs Erreichbarkeitswahrscheinlichkeiten zu berechnen (dieses Problem wurde 2010 von Zhang und Neuhäußer [ZN10] gelöst). Deshalb verwenden wir eine Transformation in uniforme Markow-Entscheidungsprozesse mit kontinuierlicher Zeit (CTMDPs), welche die Erreichbarkeitswahrscheinlichkeiten erhält, und berechnen diese anschließend mit dem Algorithmus von Baier et al. [BHKH05]. Durch diesen Ablauf gelingt es, Fragen wie z. B. „Wie groß ist die Wahrscheinlichkeit, innerhalb von 2 Stunden in einen sicherheitskritischen Zustand zu gelangen?“ für industriell relevante Modelle effizient zu beantworten.

3 Gegenbeispiele für Markow-Ketten

Im zweiten Teil der Dissertation wird das Problem gelöst, Gegenbeispiele für DTMCs mit sehr vielen Zuständen mit symbolischen Methoden zu berechnen. Es wird dabei angenommen, dass eine Sicherheitseigenschaft der Form „Die Wahrscheinlichkeit, einen sicherheitskritischen Zustand zu erreichen, ist höchstens λ” verletzt ist. Model Checking für

DTMCs wird in der Regel auf das Lösen eines linearen Gleichungssystems zurückgeführt, das gerade die gesuchte Wahrscheinlichkeit ergibt. Deshalb erhält man bei DTMCs nicht automatisch ein Gegenbeispiel, wenn eine Sicherheitseigenschaft verletzt ist. Gegenbeispiele sind jedoch von zentraler Bedeutung für die Korrektur fehlerhafter Systeme oder für die abstraktionsbasierte Verifikation, bei der eine zu grobe Abstraktion mit Hilfe von Gegenbeispielen an geeigneten Stellen verfeinert wird. Bisherige Verfahren zur Erzeugung von Gegenbeispielen für DTMCs beruhten auf Algorithmen zur Berechnung kürzester Wege in Graphen [HKD09] oder heuristischen Suchverfahren [AL10]. Diese setzen jedoch alle explizit dargestellte Zustandsräume voraus und sind daher auf verhältnismäßig kleine Systeme beschränkt.

Wir entwickeln ein symbolisches Verfahren zur Berechnung von Gegenbeispielen. Grundlage dafür ist das aus der Verifikation von Schaltkreisen bekannte Bounded Model Checking (BMC) [BCC$^+$03]. Dabei wird die Existenz von Pfaden einer festen Länge, die eine Sicherheitseigenschaft verletzen, als logische Formel beschrieben. Deren Erfüllbarkeit wird mit Hilfe eines geeigneten Solvers geprüft. Jede erfüllende Belegung der Formel entspricht genau einem Ablauf, der vom Anfangszustand des Systems zu einem sicherheitskritischen Zustand führt. Während ein einzelner solcher Pfad bei Schaltkreisen als Gegenbeispiel ausreicht, ist bei einer DTMC eine Menge von derartigen Pfaden notwendig, so dass ihre gemeinsame Wahrscheinlichkeitsmasse die Schranke λ überschreitet.

In Kapitel 9 wird gezeigt, wie das klassische BMC-Verfahren für Schaltkreise erweitert werden kann, so dass Pfadmengen als Gegenbeispiele für DTMCs effizient berechnet werden können [WBB09]. Zunächst muss eine Darstellung der Transitionsrelation als Erfüllbarkeitsproblem erzeugt werden. Da die Entscheidung, ob eine solche Formel erfüllbar ist, ein NP-hartes Problem ist, ist es von zentraler Bedeutung, eine möglichst kompakte Formel zu erzeugen. Dazu gehen wir von einer symbolischen Darstellung des Zustandsraums aus, die in Form eines Entscheidungsdiagramms gegeben ist, wie sie beispielsweise der Model Checker PRISM [HKNP06] erzeugt. Um die Darstellung zu verkleinern, werden zunächst die genauen Transitionswahrscheinlichkeiten ignoriert, und es werden Techniken zur Verkleinerung von OBDDs wie Sifting und Don't-Care-Minimierung angewendet. Mit Hilfe der Tseitin-Transformation erhält man aus dem OBDD eine Formel für die Transitionsrelation, deren Länge linear in der Größe des OBDDs ist. Durch k-faches Abrollen des Systems und Erweiterung um Formeln, die den Anfangszustand bzw. die sicherheitskritischen Zustände beschreiben, erzeugen wir eine Formel, deren erfüllende Belegungen genau den Pfaden der Länge k entsprechen, die vom Anfangszustand zu einem sicherheitskritischen Zustand führen.

Ein Gegenbeispiel wird nun folgendermaßen erzeugt: Man beginnt mit Pfadlänge $k = 0$ und wiederholt die folgenden Schritte solange, bis die Wahrscheinlichkeitsmasse der gefundenen Pfade die Grenze λ übersteigt. Man prüft die BMC-Formel auf Erfüllbarkeit; ist sie unerfüllbar, erhöht man die Pfadlänge um eins. Gibt es eine erfüllende Belegung, entspricht diese einem neuen Pfad, den man der Pfadmenge hinzufügt. Man schließt den abgearbeiteten Pfad aus dem Suchraum des Solvers aus und startet den Suchprozess erneut. Falls die Sicherheitseigenschaft verletzt ist, terminiert diese Prozedur nach endlich vielen Schritten. Eine Verbesserungsmöglichkeit ergibt sich aus folgender Beobachtung: Enthält das System Schleifen, können diese beliebig oft durchlaufen werden. Indem Gegenbeispiele nicht als einfache lineare Pfade dargestellt werden, sondern als azyklische Pfade, deren

Zustände annotiert sind mit Schleifen, kann man die Zahl der Pfade, die nötig sind, um genügend Wahrscheinlichkeitsmasse zu erhalten, in vielen Fällen stark verringern und gleichzeitig die Laufzeit verkleinern, da wir Pfade, die durch mehrfaches Abwickeln von Schleifen entstehen, von vorn herein aus dem Suchraum ausschließen können.

In Kapitel 10 evaluieren wir das Tool SBMC, das das beschriebene Verfahren implementiert, anhand einiger Fallstudien und vergleichen es mit dem expliziten Verfahren von Han et al. [HKD09]. Es zeigte sich, dass die Laufzeiten für alle Benchmarks, die beide Verfahren verarbeiten konnten, vergleichbar sind. Jedoch kann SBMC auf deutlich größere Systeme angewendet werden, als dies für das Vergleichsverfahren aufgrund seines Speicherverbrauchs möglich war.

Zum Abschluss der Arbeit folgt in Kapitel 11 eine Übersicht über Ideen, die in Zukunft weiter verfolgt werden und zum Teil inzwischen wurden. Zum einen handelt es sich dabei um weitere Optimierungen des BMC-Verfahrens für DTMCs, zum anderen um Erweiterungen auf allgemeinere Systemtypen wie Markow-Reward-Modelle.

Zu den Optimierungen gehört die Erzeugung von beliebig verschachtelten regulären Ausdrücken für die Pfadmengen, wie sie von Han et al. als Repräsentation von Gegenbeispielen vorgeschlagen wurden [HKD09]. Dadurch kann eine weitere Reduktion der Größe der Gegenbeispiele gegenüber unserem implementierten Ansatz mit azyklischen Pfaden, die mit einfachen Schleifen annotiert sind, erreicht werden.

Bisher haben wir die konkreten Übergangswahrscheinlichkeiten ignoriert und angenommen, dass kürzere Pfade in der Regel für den Benutzer zur Fehlersuche nützlicher sind als längere. Man kann jedoch auch die Wahrscheinlichkeit der Pfade als Optimierungskriterium verwenden, indem man anstelle eines rein propositionalen Erfüllbarkeitsproblems ein sogenanntes SMT-Problem erzeugt, bei dem neben booleschen Atomen lineare Ungleichungen über reellen Variablen in der Formel vorkommen. Damit lässt sich eine Formel konstruieren, die genau für diejenigen Pfade der Länge k erfüllt ist, die zu einem kritischen Zustand führen und eine vorgegebene Mindestwahrscheinlichkeit p haben. Von Systemen, die Pfade zu einem sicherheitskritischen Zustand mit sehr unterschiedlichen Wahrscheinlichkeiten enthalten, erhoffen wir uns durch diese Technik bessere Gegenbeispiele und eine Reduktion der Laufzeit, da insgesamt weniger Pfade benötigt werden.

Weiterhin lässt sich die Erzeugung von Gegenbeispielen mit der Bisimulationsminimierung kombinieren. Dabei wendet man zunächst die symbolische Minimierung mittels starker Bisimulation auf das System an. Dadurch reduziert sich in den meisten Fällen die Anzahl der Zustände deutlich. Für das minimierte System berechnet man dann mittels BMC ein Gegenbeispiel. Ein Pfad im minimierten System entspricht einer Folge von Äquivalenzklassen im Originalsystem. Man kann effizient das Gegenbeispiel für das minimierte System zurückübersetzen in ein Gegenbeispiel für das Originalsystem. In vielen Fällen wird dies für die Fehlerkorrektur nicht nötig sein, sondern es dürfte genügen, von allen schrittweise äquivalenten Pfaden einen Repräsentanten zur Verfügung zu stellen. Dadurch lässt sich das Gegenbeispiel weiter verkleinern.

Mit demselben Ansatz, mit dem man Pfade mit höherer Wahrscheinlichkeit bevorzugen kann, ist man auch in der Lage, Markow-Reward-Modelle zu behandeln. Bei diesen sind Zustände und/oder Transitionen einer DTMC um Kosten bzw. Belohnungen erweitert. Es werden Eigenschaften der Form „Die Wahrscheinlichkeit, dass das Erreichen eines Zustands

Kosten größer als c verursacht, ist höchstens λ“ betrachtet. Gegenüber den DTMCs müssen die gefundenen Pfade dahingehend eingeschränkt werden, dass sie Kosten $> c$ verursachen müssen. Dies lässt sich direkt in das erzeugte SMT-Problem integrieren.

Insgesamt ist das BMC-Verfahren ein mächtiges und flexibles Werkzeug, um effizient Gegenbeispiele für Markow-Ketten mit sehr großem Zustandsraum zu erzeugen. Durch die vorgeschlagenen Verbesserungen und Erweiterungen wird sich seine Laufzeit noch weiter reduzieren und die unterstützte Modellklasse erweitern lassen.

4 Zusammenfassung und Ausblick

In der Dissertation haben wir zwei Probleme bei der Verifikation stochastischer Systeme gelöst: Im ersten Teil haben wir ein symbolisches *Minimierungsverfahren* vorgestellt, das zu einem Markow-Modell das kleinste berechnet, das in den interessierenden Eigenschaften mit dem ursprünglichen übereinstimmt. Im zweiten Teil haben wir gezeigt, wie man mit Hilfe von Bounded Model Checking effizient *Gegenbeispiele* für DTMCs berechnen kann. Dadurch, dass beide Verfahren symbolische Datenstrukturen verwenden und dafür optimiert sind, sind sie insbesondere auch auf sehr große Systeme anwendbar.

Das Tool SIGREF ist dabei, sich zu einem Standardwerkzeug für die symbolische Minimierung zu entwickeln und fand bereits Eingang in mehrere Anwendungen in der probabilistischen Verifikation. Das Werkzeug zur Generierung von Gegenbeispielen wird aktiv weiterentwickelt. Im Moment werden die in Kapitel 11 beschriebenen Optimierungen und Erweiterungen integriert, um das Tool SBMC noch mächtiger zu machen. Es ist geplant, SBMC beispielsweise für die gegenbeispielgesteuerte Abstraktionsverfeinerung (CEGAR) einzusetzen.

Literatur

[AL10] Husain Aljazzar und Stefan Leue. Directed Explicit State-Space Search in the Generation of Counterexamples for Stochastic Model Checking. *IEEE Trans. on Software Engineering*, 36(1):37–60, 2010.

[BCC+03] Armin Biere, Alessandro Cimatti, Edmund M. Clarke, Ofer Strichman und Yunshan Zhu. Bounded Model Checking. *Advances in Computers*, 58:118–149, 2003.

[BCM+92] Jerry R. Burch, Edmund M. Clarke, Kenneth L. McMillan, David L. Dill und L. J. Hwang. Symbolic Model Checking: 10^{20} States and Beyond. *Information and Computation*, 98(2):142–170, 1992.

[Bd92] Amar Bouali und Robert de Simone. Symbolic Bisimulation Minimisation. In *Proc. of CAV*, Band 663 von LNCS, Seiten 96–108. Springer, 1992.

[BHH+09] Eckard Böde, Marc Herbstritt, Holger Hermanns, Sven Johr, Thomas Peikenkamp, Reza Pulungan, Jan Rakow, Ralf Wimmer und Bernd Becker. Compositional Dependability Evaluation for STATEMATE. *IEEE Trans. on Software Engineering*, 35(2):274–292, 2009.

[BHHK03] Christel Baier, Boudewijn Haverkort, Holger Hermanns und Joost-Pieter Katoen. Model-Checking Algorithms for Continuous-Time Markov Chains. *IEEE*

Trans. on Software Engineering, 29(7):1–18, 2003.

[BHKH05] Christel Baier, Holger Hermanns, Joost-Pieter Katoen und Boudewijn R. Haverkort. Efficient Computation of Time-Bounded Reachability Probabilities in Uniform Continuous-Time Markov Decision Processes. *Theoretical Computer Science*, 345(1):2–26, 2005.

[BO05a] Stefan Blom und Simona Orzan. A Distributed Algorithm for Strong Bisimulation Reduction of State Spaces. *Software Tools for Technology Transfer*, 7(1):74–86, 2005.

[BO05b] Stefan Blom und Simona Orzan. Distributed State Space Minimization. *Software Tools for Technology Transfer*, 7(3):280–291, 2005.

[Der07] Salem Derisavi. A Symbolic Algorithm for Optimal Markov Chain Lumping. In *Proc. of TACAS*, Band 4424 von LNCS, Seiten 139–154. Springer, 2007.

[HJ94] Hans Hansson und Bengt Jonsson. A Logic for Reasoning about Time and Reliability. *Formal Aspects of Computing*, 6(5):512–535, 1994.

[HKD09] Tingting Han, Joost-Pieter Katoen und Berteun Damman. Counterexample Generation in Probabilistic Model Checking. *IEEE Trans. on Software Engineering*, 35(2):241–257, 2009.

[HKNP06] Andrew Hinton, Marta Kwiatkowska, Gethin Norman und David Parker. PRISM: A Tool for Automatic Verification of Probabilistic Systems. In *Proc. of TACAS*, Band 3920 von LNCS, Seiten 441–444. Springer, 2006.

[WB10] Ralf Wimmer und Bernd Becker. Correctness Issues of Symbolic Bisimulation Computation for Markov Chains. In *Int'l GI/ITG Conf. on Measurement, Modelling and Evaluation of Computing Systems (MMB)*, Band 5987 von LNCS, Seiten 287–301. Springer, 2010.

[WBB09] Ralf Wimmer, Bettina Braitling und Bernd Becker. Counterexample Generation for Discrete-time Markov Chains using Bounded Model Checking. In *Proc. of VMCAI*, Band 5403 von LNCS, Seiten 366–380. Springer, 2009.

[WDH10] Ralf Wimmer, Salem Derisavi und Holger Hermanns. Symbolic Partition Refinement with Automatic Balancing of Time and Space. *Performance Evaluation*, 67(9):815–835, 2010.

[WHH+06] Ralf Wimmer, Marc Herbstritt, Holger Hermanns, Kelley Strampp und Bernd Becker. Sigref – A Symbolic Bisimulation Tool Box. In *Proc. of ATVA*, Band 4218 von LNCS, Seiten 477–492. Springer, 2006.

[ZN10] Lijun Zhang und Martin R. Neuhäußer. Model Checking Interactive Markov Chains. In *Proc. of TACAS*, Band 6015 von LNCS, Seiten 53–68. Springer, 2010.

Ralf Wimmer studierte an der Albert-Ludwigs-Universität Freiburg Informatik und Mikrosystemtechnik und erhielt 2004 das Diplom mit Auszeichnung in Informatik. Für seine Diplomarbeit wurde er mit dem VDI-Nachwuchsförderpreis ausgezeichnet. 2011 promovierte er mit Auszeichnung an der Albert-Ludwigs-Universität bei Prof. Dr. Bernd Becker im Transregio-Sonderforschungsbereich SFB/TR 14 AVACS zu symbolischen Methoden für die Verifikation probabilistischer Systeme. Derzeit ist Ralf Wimmer Akademischer Rat an der Universität Freiburg. Er ist Autor von über 20 Publikationen.