DFG Projekt Gepavas II

Gerichtete und parallele Validation abstrakter Spezifikationen

Projektnummer LE 2377/1-2

Abschlussbericht

Jens Bendisposto, Ivaylo Dobrikov, Dominik Hansen,
Sebastian Krings, Michael Leuschel

März 2015

Inhaltsverzeichnis

1 Allgemeine Angaben

DFG Sachbeihilfe
DFG Geschäftszeichen LE 2377/1-2

1.1 Antragsteller

Michael Leuschel, Prof. Dr.
Universitätsprofessor (C4)
geb. 28.10.1967, deutsche Staatsbürgerschaft
Lehrstuhl für Softwaretechnik und Programmiersprachen
Institut für Informatik, Universität Düsseldorf
Universitätsstr. 1, D-40225 Düsseldorf
Tel: 0211 81 10710 Fax: 0211 81 10712
Email: `leuschel@cs.uni-duesseldorf.de`

1.2 Thema des Projekts

Gerichtete und parallele Validierung von abstrakten Spezifikationen (GEPAVAS)
Stichwort: formale Methoden, logische Programmierung

Fachgebiet und Arbeitsrichtung Informatik, Softwaretechnik, formale Methoden, B-Methode, Model Checking, logische Programmierung

1.3 Berichtszeitraum, Förderungszeitraum

- Datum der Bewilligung von Gepavas I: 8.11.2007 (GZ: LE 2377/1-1, AOBJ: 548218)
- Datum der Bewilligung von Gepavas II: 23.11.2010 (GZ: LE 2377/1-2, AOBJ: 582278)
- Abschluss von Gepavas II nach kostenneutraler Laufzeitverlängerung: 31.1.2015

Berichtszeitraum: 23.11.2010 bis 31.1.2015 (mit Zusatzinformationen über erste Phase des Projekts)

1.4 Publikationen

Artikel von Gepavas II Folgende Artikel dokumentieren den wissenschaftlichen Beitrag von Gepavas II:

- Dominik Hansen, Michael Leuschel Translating B to TLA+ for Validation with TLC. In Proceedings ABZ'14, LNCS 8477, 2014.

– Ivaylo Dobrikov, Michael Leuschel Optimising the ProB Model Checker for B using Partial Order Reduction. In SEFM 2014, LNCS 8702, 2014.
– Sebastian Krings, Michael Leuschel Inferring Physical Units in B Models. In Proceedings SEFM'2013, LNCS 8137, Springer, 2013.
– Sebastian Krings, Michael Leuschel Inferring Physical Units in Formal Models. Journal Software and Systems Modeling. Akzeptiert zur Veröffentlichung, März 2015. Erhältlich unter: `http://www.stups.uni-duesseldorf.de/w/Special:Publication/KringsLeuschelUnitsJournal`
– Jens Bendisposto, Philipp Körner, Michael Leuschel Parallel Model Checking of B Specifications. In Proceedings of the 4th Rodin User and Developer Workshop, TUCS Lecture Notes, TUCS, 2013.
– Jens Bendisposto, Michael Leuschel Automatic Flow Analysis for Event-B. In Proceedings of Fundamental Approaches to Software Engineering (FASE) 2011, volume 6603 of Lecture Notes in Computer Science, Springer, 2011.

Die Arbeiten zu dem letzten Artikel wurden zum Teil schon in Gepavas I erstellt.

Die folgenden Artikel wurden Anfang 2015 bei wissenschaftlichen Zeitschriften auf Einladung eingereicht:

– Ivaylo Dobrikov, Michael Leuschel Optimising the ProB Model Checker for B using Partial Order Reduction. Eingereicht an Formal Aspects of Computing. Erhältlich unter: `http://www.stups.uni-duesseldorf.de/w/Special:Publication/DobrikovLeuschelPORtechreport`
– Dominik Hansen, Michael Leuschel Translating B to TLA + for Validation with TLC. Eingereicht an Science of Computer Programming. Erhältlich unter: `http://www.stups.hhu.de/w/Special:Publication/HansenLeuschel_TLC4B_Journal`

Die Doktorarbeit von Jens Bendisposto ist im Rahmen von Gepavas I und II durchgeführt worden:

– Jens Bendisposto. Directed and Distributed Model Checking of B-Specifications. PhD Thesis, University of Düsseldorf. January 2015. Volltext als Preprint verfügbar unter: `http://www.stups.hhu.de/w/Jens_Bendisposto`

Der zweite Teil der Doktorarbeit wird als Artikel eingereicht. Es ist geplant aus dem ersten Teil der Arbeit einen Zeitschriftenartikel zu erstellen.

Artikel von Gepavas I Um einen Überblick über das gesamte Gepavas Projekt zu ermöglichen, zählen wir hier noch die Artikel auf die in der ersten Phase von Gepavas entstanden sind:

– Michael Leuschel, Jens Bendisposto Directed Model Checking for B: An Evaluation and New Techniques. In SBMF'2010, volume 6527 of Lecture Notes in Computer Science, Springer, 2010.
– Michael Leuschel, The High Road to Formal Validation, Proceedings ABZ 2008, LNCS 5238, p. 4–23.

– Jens Bendisposto, Michael Leuschel: Proof Assisted Model Checking for B, Proceedings ICFEM 2009, LNCS 5885 , p. 504–520.

– Jens Bendisposto and Michael Leuschel. Parallel Model Checking of Event-B Specifications with ProB. Preliminary Proceedings PDMC 2009, Eindhoven, November 2009.

– Mireille Samia, Harald Wiegard, Jens Bendisposto, Michael Leuschel : High-Level versus Low-Level Specifications: Comparing B with Promela and ProB with Spin, Proceedings TFM-B 2009, Nantes, June, 2009, APCB. ISBN:2951246102.

2 Arbeits- und Ergebnisbericht

2.1 Einführung

Formale Methoden haben das Ziel fehlerfreie Software zu produzieren. Die B-Methode ist eine formale Methode die erfolgreich in mehreren Industrie-Projekten angewandt wurde. Die B-Methode garantiert, dass die entwickelte Software der Anfangsspezifikation entspricht. Mit steigender Komplexität der Anforderungen an Softwaresysteme wird aber auch das Erstellen korrekter Spezifikationen schwieriger; daher stellt die Korrektheit der Anfangspezifikation einen kritischen Punkt dar. Das Hauptziel dieses Forschungsprojektes ist es, Methoden und Werkzeuge zu entwickeln, mit denen aus Industrieprojekten stammende, komplexe, formale B-Spezifikationen validiert werden können. Eine besondere Herausforderung dabei ist die hohe Ausdruckskraft der B Spezifikationssprache. Wir glauben, dass gerade diese hohe Abstraktionsebene ein großes Potential für intelligente Validierungstechniken mit sich bringt.

In diesem Forschungsprojekt sollten folgende zwei Techniken entwickelt werden: **gerichtete** Modellprüfung (engl. model checking) von B, um Fehlerzustände in großen Zustandsräumen anhand von Heuristiken zu finden, sowie **parallele** Modellprüfung, zur Verteilung des Rechenaufwandes. Beide Techniken sollten durch verschiedene **statische Analysen** und verstärkt durch **Beweisinformationen** unterstützt werden. Zusätzlich wollen wir auch das Potential der **Partial-Order Reduktion** für Event-B evaluieren.

Wir werden in diesem Abschlussbericht zeigen, dass die Ziele des Projekts erreicht worden sind. Es sind mehrere neue Techniken entwickelt, bewiesen und implementiert worden. Mit diesen Techniken wurde die Laufzeit der Modellprüfung um bis zu mehrere Grössenordnungen verringert. Die Techniken konnten erfolgreich auf mehrere industrielle Fallstudien angewendet werden.

2.2 Überblick

Aufbau des Projekts Dieses Projekt ist eine Weiterführung des Projekts Gepavas I. Eine Übersicht des Arbeitsprogrammes von Gepavas I und II befindet sich in Abbildung 1. In Gepavas I wurden verschiedene Grundlagen für das aktuelle Projekt Gepavas II geschaffen:

- die statische Analyse von B Modellen basierend auf abstrakter Interpretation wurde entwickelt und prototypisch in unserem Werkzeug ProB implementiert,
- ein erster Prototyp für Shared-Memory parallele Modellprüfung mit ProB wurde entwickelt,
- gerichtete Modellprüfung basierend auf Heuristiken wurde entwickelt, implementiert und evaluiert.

Die wichtigsten Arbeitspakete von Gepavas II sind:

– Optimierung für Industrielle Anwendungen (T4),
  Hier wurden auch komprimierte Darstellung der einzelnen Zustände untersucht, mit möglicher Auslagerung des Zustandsraums auf die Festplatte (nur die Hashtabelle würde dann im Hauptspeicher verwaltet werden). Diese Arbeiten sind in T7 eingeflossen.
– Beweis- und Fluss-gerichtete Modellprüfung (T5),
  Mit Hilfe eines Beweisers oder abstrakter Interpretation können wir herausfinden welche Operationen welche anderen Operationen frei- beziehungsweise abschalten können (wir nennen dies den Flow eines B Modells). Daraus kann man einen Plan erstellen um eine gewisse Zielkonfiguration zu erreichen.
– Partial Order Reduction für Event-B (T6),
  Bei der Partial Order Reduction versucht man die Grösse des Zustandsraums zu verringern indem man die Kommutativität unabhängiger Transitionen ausnutzt.
– Verbesserte Techniken zur Parallelisierung (T7),
  Das Ziel ist es, den Aufwand der Modellprüfung auf verschiedene Prozessoren oder Rechner zu verteilen.
– Implementierung (I7) und empirische Evaluation (E3).

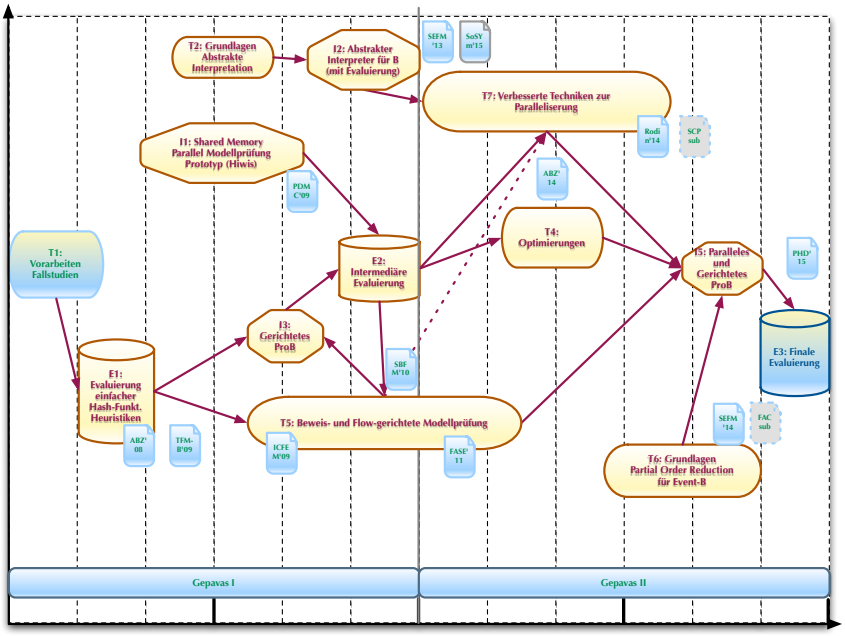

Abbildung 1. Projektübersicht mit Publikationen, Doktorarbeiten und Einreichungen

Übersicht über die wichtigsten Forschungsergebnisse Im Nachhinein hat sich dieses Projekt als sehr ambitioniert und umfangreich erwiesen. Dank der genehmigten Verlängerung des Projekts, und unter Mithilfe von Mitarbeitern des Lehrstuhls "STUPS" sind alle wissenschaftlichen und technischen Ziele erreicht worden. Insgesamt, sind

- 10 Artikel publiziert worden,
- 1 Zeitschriftenartikel Artikel ist akzeptiert worden,
- 2 weitere Zeitschriftenartikel sind eingereicht worden,
- und eine Promotion ist abgeschlossen worden.

Alle Gepavas Techniken sind praktisch in unserem Werkzeug ProB umgesetzt worden und stehen der Öffentlichkeit zur Verfügung.

In Sektion 2.3 beschreiben wir die Forschungsergebnisse von Gepavas II im Detail. Die Zuordnungen der Publikationen zu den einzelnen Arbeitspaketen haben wir im Diagramm 1 skizziert. Die grössten Fortschritte des Projekts sind wie folgt:

- Konsistenzprüfung physikalischer Einheiten in industriellen formalen Modellen und Systemen mit Hilfe abstrakter Interpretation (siehe Sektion 2.3 und Anhang A.1)
- optimiertes, paralleles und beweisgerichtete Modellprüfung für sehr grosse Zustandsräume und industrielle Anwendungen mit TLC (siehe Sektion 2.4 und Anhang A.2)
- die Entwicklung eines theoretischen Frameworks, das zur Extraktion von Programmfluss-Informationen aus Event-B Modellen dient und in den anderen Teilen von Gepavas und für modell-basiertes Testen Einsatz gefunden hat (siehe Sektion 2.4 und Anhang A.4)
- eine Entwicklung von Partial Order Reduction für Event-B und B Modelle mit guter Reduktion des Validerungsaufwands für Modelle mit einem hohen Anteil an Nebenläufigkeit (siehe Sektion 2.4 und Anhang A.3).
- Verbesserung der Laufzeit von ProB um zwei Größenordnungen mit verteilter Modellprüfung. Es können nun Modelle mit Milliarden von Zuständen validiert werden (siehe Sektion 2.4 und Anhang A.4).

Implementierung Die Implementierung ist unter folgender Adresse öffentlich zugänglich:

`http://www.stups.hhu.de/ProB`

Dort ist der Quelltext von ProB und allen in Gepavas entwickelten Techniken zugänglich. Die Grundlagen der Evaluierung sind ebenfalls öffentlich zugänglich und in den einzelnen Gepavas Artikeln sowie unten in diesem Abschlussbericht verlinkt. Die Dokumentation zur praktischen Verwendung der Gepavas-Techniken sind auch öffentlich verfügbar und unten aufgezählt.

2.3 Beschreibungen der wichtigsten Forschungsergebnisse

Abstrakte Interpretation Viele zustandsbasierte formale Methoden wie B, Event-B oder Z unterstützen statische Typsicherheit. Damit wird zum Beispiel sicher gestellt, dass Zahlen nur mit anderen Zahlen verglichen werden. Statische Typüberprüfung fängt sehr früh Fehler auf und garantiert eine gewisse Konsistenz der formalen Spezifikationen. Oft werden mit formalen Modellen auch physikalische Prozesse und Steuerungen modelliert; es wäre deshalb von Vorteil nicht nur zu prüfen, dass nur Zahlen mit Zahlen verglichen werden sondern auch, dass nur Zahlen mit Zahlen einer kompatiblen physikalischen Einheit verglichen werden. Dies wird in den oben genannten formalen Methoden leider nicht unterstützt.

In diesem Teil des Projekts haben wir mit Hilfe der abstrakten Interpretation eine Lösung für dieses Problem entwickelt. Mit unserem Ansatz kann der Benutzer gewisse Variablen mit physikalischen Einheiten belegen; unsere Analyse propagiert diese Information und versucht für alle Skalarvariablen eine physikalische Einheit zu inferieren. Falls dies nicht möglich ist werden Fehlermeldungen im Modell generiert. Diese Technik basiert auf den Ergebnissen von Gepavas I, und verbindet abstrakte Interpretation, Constraintprogrammierung und Modellprüfung. Die Entwicklungen wurden in das ProB Werkzeug sowohl für klassisches B als auch in Rodin für Event-B integriert. Eine Anwendung auf die untypisierte formale Sprache TLA$^+$ ist auch möglich. Ausführliche empirische Evaluierung zeigt, dass unsere Technik für industrielle Modelle skaliert. Die meisten Modelle die bei der Evaluierung verwendet wurden sind öffentlich zugänglich[1]:

`http://www.stups.hhu.de/models/sosym_units/`

Eine Anwenderdokumentation gibt es auf folgenden Seiten:

`http://www.stups.hhu.de/ProB/index.php5/Tutorial_Unit_Plugin` und
`http://www.stups.hhu.de/ProB/index.php5/Tutorial_Unit_Plugin_With_Rodin`

2.4 Parallele Modellprüfung mit TLC

Die zustandsbasierten formalen Methoden B und TLA$^+$ beruhen beide auf einer gemeinsamen Basis aus Prädikatenlogik, Arithmetik, Mengen- und Relationentheorie. Es gibt aber auch eine Reihe an grundlegenden Unterschieden, wie z.B. die Notation von Zustandsübergängen, unterschiedliche Ansätze zur Typisierung (TLA$^+$ ist untypisiert), als auch unterschiedliche zur Verfügung stehende Validierungswerkzeuge. TLA$^+$ Spezifikation lassen u.a. durch den Modelchecker TLC validieren, der sowohl paralleles als auch verteilte Modellprüfung erlaubt. In diesem Teil des Projekts haben wir eine Übersetzung von B nach TLA$^+$ entwickelt, die es ermöglicht B Spezifikationen mit TLC zu validieren. Unsere Übersetzung beinhaltet verschiedene Optimierungen, um eine effiziente Validierung

[1] Einige Modelle von Industriepartner dürfen leider nicht offen zugänglich gemacht werden.

mit TLC zu ermöglichen. Auch werden Liveness- und Fairness-Eigenschaften von der Übersetzung unterstützt und können mit TLC validiert werden. Unser implementierter Übersetzer, Tlc4B, wandelt B Spezifikationen automatisch nach TLA$^+$ um, ruft den Modelchecker TLC auf, und übersetzt die Ergebnisse anschließend wieder zurück nach B. Gegenbeispiele, die von TLC gefunden werden, können somit nochmal mit dem ProB Animator nachgespielt und revalidiert werden. Ein Benutzer braucht somit keine Kenntnisse von TLA$^+$ um TLC als Validierungswerkzeug für B nutzen zu können. Es besteht auch die Möglichkeit Konstantenbelegungen, die für TLC teilweise schwer zu ermitteln sind, mit ProB zu berechnen und anschließend an TLC zu übermitteln. Dadurch kann der Benutzer die Stärken beider Werkzeuge kombinieren: Constraint-Solving mit ProB und effiziente Modellprüfung mit TLC. Zudem unterstüztzt Tlc4B beweisgerichtete Modellprüfung als Optimierungmethode, was auf Arbeiten in Gepavas I aufbaut. Dabei kodiert der Übersetzer abgeleitete Beweisinformationen direkt in dem übersetzten B Modell, und erlaubt es TLC somit bei der Modellprüfung die Evaluierung von bereits bewiesenen Invarianten in einigen Zuständen zu überspringen.

Wir validieren unseren Übersetzer anhand einer großen Anzahl an Beispielen und Fallstudien, und vergleichen auch die Modelchecker TLC und ProB. TLC erweist sich als sehr effizient bei der Validierung von B Modellen mit großen Zustandsräumen. Für „low-level"-Spezifikationen ist Tlc4B bis zu 50 mal schneller, bei „high-level"-Spezifikationen ist es genau umgekehrt.

Die B-Modelle, die als Grundlage der empirischen Auswertung dienen, sind unter folgender URL erhältlich:

`http://www.stups.hhu.de/models/tlc4b`

Eine Anwenderdokumentation zur Benutzung von TLC befindet sich auf folgender Webseite:

`http://www.stups.hhu.de/ProB/index.php5/TLC`

Flow Analyse Wir haben ein theoretisches Framework entwickelt, das zur Extraktion von Programmfluss-Informationen aus Event-B Modellen dient. Derartige Informationen sind in Event-B oft in der Verfeinerung versteckt. Dennoch sind diese Informationen über die algorithmische Struktur eines Modells für eine ganze Reihe von Anwendungen nützlich. Die Analyse wurde in drei Teile aufgeteilt:

- **(Un)Abhängigkeitsanalyse**: Berechnung, welche Events sich gegenseitig beeinflussen können. Die Analyse ist nicht nur rein syntaktisch, sondern verwendet Beweis- und Constraintsolving Techniken um nicht-triviale Unabhängigkeiten zu finden.
- **Enable Analyse**: Für abhängige Events liefert diese Analyse Aussagen über die Umstände, wann ein Ereignis ein anderes Ereignis ermöglicht oder verhindert.

– **Flow Analyse**: Schlussendlich werden die Ergebnisse der Enable Analyse verwendet um einen abstrakten Flussgraphen des Systems zu erzeugen.

Das Analyse Framework ist die theoretische Grundlage für eine ganze Reihe von Anwendungen, die im Rahmen des Gepavas II Projekts entwickelt wurden. Die Unabhängigkeitsanalyse wird zum Beispiel bei der Partial Order Reduction verwendet, die Enable Analyse ist die Grundlage der partiellen Guard-Evaluierung und die Flow Analyse kann benutzt werden um die Abwesenheit von Deadlocks und andere Aussagen über die Lebendigkeit eines Systems zu beweisen. Die Analyse wird mittlerweile auch für das modell-basierte Testen eingesetzt, um zielgerichtet Testfälle zu generieren. Die Effizienz wird dabei in erheblichem Maße gesteigert; ein Artikel über den praktischen Einsatz dieser neuen Technik (um Sicherheitslücken in Smartcards zu finden) ist in Vorbereitung.

Optimierung des ProB Modellprüfers mit Hilfe von Partial-Order Reduktion Der Modellprüfer von ProB ist ein praktisches Werkzeug, das für die Validierung von Modellen mit endlichen Zustandsräumen benutzt werden kann. Um zu überprüfen, ob das gegebene Modell korrekt ist, generiert der Modellprüfer alle möglichen Zustände des Modells und überprüft dabei jeden Zustand auf Fehler (wie zum Beispiel Deadlocks oder Invariantenverletzungen).

Die Laufzeit der Modellprüfung ist zumeist eng verknüpft mit der Größe des Zustandsraums. Daher stellt die Größe des Zustandsraumes auch eine der größten Einschränkungen für explizite Modellprüfung dar. Das Problem ist auch unter dem Namen Zustandsexplosion (engl. state space explosion) bekannt. Um diesem Problem entgegenzuwirken, haben wir die Partial Order Reduction (POR) für den ProB Modellprüfer implementiert und evaluiert. Desweiteren haben wir eine Optimierungstechnik, die partielle Guard-Auswertung (engl. partial guard evaluation, PGE) entwickelt, die den Aufwand der Berechnung jedes einzelnen States reduziert.

Partial Order Reduction (POR) POR wurde als Methode für die Überwindung des Zustandsexplosion-Problems mittels Reduktionen des Zustandsraumes vorgeschlagen. Die Methode nutzt die Unabhängigkeit von Transitionen, um redundante Pfade im Zustandsraum zu kürzen. Somit wird versucht nur ein Fragment des Zustandsraumes des Modells überprüft zu werden, das für die Überprüfung der jeweiligen Eigenschaft ausreicht. Die Intuition ist: je kleiner das Fragment des zu überprüfenden Zustandraums ist, desto effizienter ist die Überprüfung des Modells. Dabei müssen bei der Reduktion des Zustandsraumes die folgenden Kriterien beabsichtigt werden:

1. Der reduzierte Zustandsraum und der vollständige Zustandsraum müssen äquivalent in Bezug auf die zu überprüfende Eigenschaft sein (Korrektheit der Reduktion).
2. Der reduzierte Zustandsraum muss deutlich kleiner als der vollständige Zustandsraum sein.

3. Der Aufwand für die Generierung des reduzierten Zustandsraumes muss relativ klein sein.

Wir haben den Modellprüfer von ProB erweitert, um die Validierung von B und Event-B Modellen mit Hilfe von POR zu optimieren. Unsere Implementierung von POR benuzt die *Ample Set* Theorie, die als eine Methode für POR vorgeschlagen wurde. Der Reduktionsalgorithmus nutzt Relationen wie Unabhängigkeit von Transitionen um den Zustandsraum des zu überprüfenden Modells zu reduzieren. Solche Relationen werden mittels statischer Analysen wie die (Un)Abhängigkeitsanalyse und die Enablinganalyse (siehe Sektion 2.4) berechnet.

Die Evaluierung der Implementierung von POR zeigt, dass sie für Modelle, die einen hohen Anteil an Nebenläufigkeit und Unabhängigkeit aufweisen, eine sehr gute Reduktion erreichen kann und somit die Laufzeit für die Modellprüfung verringert wird. Wir haben ebenfalls beobachtet, daß die Reduktion auch von der zu überprüfenden Eigenschaft abhängt. Zum Beispiel ist bei reiner Deadlock Suche die Reduktion größer als bei kombinierter Suche nach Deadlocks und Invariantenverletzungen. Die Ergebnisse der Evaluierung haben ebenfalls gezeigt, dass der Overhead für die Generierung des reduzierten Zustandraumes klein ist.

Die Ergebnisse der Evaluierung der POR Optimierung sowie die Modelle, auf den die reduzierte Suche evaluiert wurde, können auf der folgenden Webseite heruntergeladen werden:

`http://nightly.cobra.cs.uni-duesseldorf.de/por/`

Partielle Guard-Evaluierung (PGE) Partielle Guard-Evaluierung (PGE) ist eine andere Möglichkeit die Modellprüfung eines B bzw. Event-B Modells zu optimieren. Die Validierung eines B Modells anhand des Modellprüfers von ProB erfolgt mittels Explorieren des Zustandsraumes des Modells. Jeder Zustand, der besucht wird und kein Fehlerzustand ist, wird exploriert, indem alle Operationen des entsprechenden B Modells auf die aktuelle Evaluierung der Variablen auf Enabledness getestet werden und im Falle, dass sie im Zustand eingeschaltet sind, ausgeführt (bei der Ausführung der im Zustand eingeschalteten Operationen werden die entsprechenden Nachfolgerzustände generiert). Die Enable Analyse (siehe Sektion 2.4) enthält wichtige Informationen darüber wie sich die Operationen eines B Modells beeinflussen können. Durch das Benutzen solcher Informationen kann man im Laufe der Modellprüfung vorhersagen, ob bestimmte Operationen im aktuellen Zustand aktiviert bzw. deaktiviert sind, d.h. es wird unnötig den Guard der Operation tatsächlich zu prüfen. Dadurch können überflüssigen Berechnungen gespart werden. Die Idee der PGE orientiert sich an der in Gepavas I entwickelten beweisgestützten Modelprüfung, bei der die Invariante durch statische Analyse reduziert wird.

Wir haben den Modellprüfer von ProB erweitert, so dass partielle Guard-Evaluierung unterstützt wird. Die Informationen über die Beziehungen der Operationen bzw. Events eines B bzw. Event-B Modells werden mittels statischer Analysen berechnet (siehe Sektion 2.4). Die Evaluierung der PGE Optimierung

zeigt, dass für manche B Modelle (insbesondere Modelle mit großem Zustandsraum) der Speedup Faktor[2] bis zu 1.5 groß sein kann. Zusätzlich haben wir in manchen Fällen beobachtet, dass die Effizienz der PGE Optimierung von dem gewählten Suchverfahren abhängt. Die Modellprüfung mit PGE kann manchmal viel effizienter sein wenn Breitensuche anstatt Tiefensuche als Suchstrategie verwendet wird.

Die Ergebnisse der Evaluierung der PGE Optimierung sowie die Modelle, auf denen die optimierte Modellprüfung evaluiert wurde, können auf der folgenden Webseite heruntergeladen werden:

`http://nightly.cobra.cs.uni-duesseldorf.de/pge/`

Eine Anwendungsdokumentation zu den Optimierungen POR und PGE befindet sich auf folgender Webseite:

`http://www.stups.hhu.de/ProB/index.php5/Tutorial_Various_Optimizations`

Parallele und verteilte Modellprüfung mit ProB Die Grundannahme, daß sich Spezifikationssprachen auf einem hohen Abstraktionsniveau ausgezeichnet für parallele Modellprüfung eignen ist durch unsere Implementierung bestätigt worden. Unser Prototyp wurde auf drei unterschiedlichen Systemen (standalone Multicore Rechner, Amazon Cloud Instanzen und HPC Cluster) evaluiert und skaliert auf allen drei Systemen fast perfekt. Wir konnten die Zeit für die Modellprüfung für das größte unserer Benchmark Modelle von geschätzten 17 Tagen auf 96 Minuten auf 140 Kernen reduzieren. Die Resultate unserer Experimente legen nahe, daß die Modellprüfung noch weiter skaliert, wir konnten aber keine Experimente mit mehr als 140 Kernen durchführen. Kleinere Modelle haben wir auf Instanzen der Amazon Cloud berechnet, auch hier zeigte sich, daß die Modellprüfung in etwa mit der Anzahl der echten Rechnerkerne skaliert.

Die Resultate des Gepavas Projekts haben eine Verbesserung der Laufzeit bei der Modellprüfung von zwei Grössenordnungen erreicht. Entsprechend können mit ProB potentiell Modelle mit Milliarden von Zuständen verifiziert werden. Bisher war die praktikable Grenze im Bereich von einigen 10 Millionen Zuständen, abhängig vom Model.

Die B-Modelle, die als Grundlage der empirischen Auswertung dienen, sind unter folgender URL erhältlich:

`www.stups.hhu.de/models/parb`

Eine Dokumentation der empirischen Auswertung befindet sich auf folgender Webseite:

`www.stups.hhu.de/ProB/index.php5/DMC`

2 $\frac{\text{(Modellprüfer Laufzeit)}}{\text{(Modellprüfer Laufzeit mit PGE)}}$

Eine Anwenderdokumentation zur Benutzung des parallelen Prototypen befindet sich auf folgender Webseite:

`www.stups.hhu.de/ProB/index.php5/ParB`

3 Zusammenfassung

Formale Methoden haben das Ziel, basierend auf mathematischen Analysen, fehlerfreie Software zu produzieren. Die B-Methode ist solch eine formale Methode, die erfolgreich in der Industriepraxis eingesetzt wird. Zum Beispiel benutzten in 2014 weltweit 25 % der fahrerlosen U-Bahnzüge Software die auf Basis der B-Methode erstellt wurde. Die B-Methode garantiert, dass die entwickelte Software einer formalen Spezifikation entspricht. Mit steigender Komplexität der Anforderungen an Softwaresysteme, wird aber auch das Erstellen korrekter Spezifikationen schwieriger; daher stellt die Korrektheit der Anfangsspezifikation einen kritischen Punkt dar. Das Hauptziel dieses Forschungsprojektes war es, Methoden und Werkzeuge zu entwickeln, mit denen aus Industrieprojekten stammende, komplexe, formale B-Spezifikationen validiert werden können. Eine besondere Herausforderung war dabei die hohe Ausdruckskraft der B Spezifikationssprache. Alle Gepavas Techniken sind praktisch in unserem Werkzeug ProB umgesetzt worden und stehen der Öffentlichkeit zur Verfügung (`http://www.stups.hhu.de/ProB`). Die größten Fortschritte des Projekts waren dabei:

- optimiertes, paralleles und beweisgerichtete Modellprüfung für industrielle Anwendungen mit sehr großen Zustandsräumen,
- eine Technik um B Modelle und Systeme, trotz eines hohen Anteils an Nebenläufigkeit, effizient zu Validieren,
- die Möglichkeit, dank gerichteter Modellprüfung, Fehler auch in unendlichen Zustandsräumen zu finden,
- eine Technik zur Konsistenzprüfung der Benutzung physikalischer Einheiten in industriellen formalen Modellen und Systemen.

Ein besonderer Erfolg des Projekts ist die neue Methode zur parallelen Modellprüfung. Der Rechenaufwand wird dabei automatisch auf verschiedene Rechner verteilt und skaliert selbst bei einer großen Anzahl an Rechnern. Die Laufzeit wurde dabei um bis zu zwei Größenordnungen verbessert und es können nun formale Modelle mit Milliarden von Zuständen auch verteilt in der "Cloud" validiert werden. Das DFG Projekt hat in wichtigem Maße zur Verbesserung des Standes der Technik und Forschung auf dem Gebiet der formalen B Methode beigetragen. Dank des Projekts wurde auch das Werkzeug ProB weiterentwickelt und die Neuentwicklungen der Wissenschaftsgemeinde zur Verfügung gestellt (mittlerweile gibt es mehr als 580 Zitationen für die zwei Hauptartikel über ProB, und etliche akademische und industrielle Werkzeuge wie DTVT, SafeCap, iUML-B, HRemo, B4MSecure, oder VTG bauen auf ProB auf). ProB wird zur Validierung von sicherheitskritischen Bahnsystemen verwendet, und wurde weltweit zur Prüfung mehrere fahrerloser U-Bahnlinien eingesetzt. ProB wird in diesem Rahmen bei Firmen wie Siemens, Alstom, und General Electric in der täglichen Praxis eingesetzt. Im Rahmen des Projekts Gepavas haben wir auch eine Brücke zu anderen Forschungsgebieten geschlagen, unter anderem zur Sprache TLA^+, welche auch durch den Turing Preis von Leslie Lamport in 2013 vermehrt im Ramplenlicht steht (siehe `http://amturing.acm.org/award_winners/lamport_1205376.cfm`).

A Anhang

Dieser Anhang dokumentiert im Detail die wissenschaftlichen Ergebnisse des Projekts. Um Redundanz zu vermeiden, haben wir hier drei Zeitschriftenartikel und eine Doktorarbeit ausgewählt.

A.1 Abstrakte Interpretation

– Sebastian Krings, Michael Leuschel: Inferring Physical Units in Formal Models. Journal Software and Systems Modeling. Akzeptiert zur Veröffentlichung, März 2015.

Erhältlich unter: `http://www.stups.uni-duesseldorf.de/w/Special:Publication/KringsLeuschelUnitsJournal`

A.2 Paralleles Modelchecking mit TLC

– Dominik Hansen, Michael Leuschel: Translating B to TLA + for Validation with TLC. Eingereicht an Science of Computer Programming.

Erhältlich unter: `http://www.stups.hhu.de/w/Special:Publication/HansenLeuschel_TLC4B_Journal`

A.3 Partial Order Reduction

– Ivaylo Dobrikov, Michael Leuschel: Optimising the ProB Model Checker for B using Partial Order Reduction Eingereicht an Formal Aspects of Computing.

Erhältlich unter: `http://www.stups.uni-duesseldorf.de/w/Special:Publication/DobrikovLeuschelPORtechreport`

A.4 Flow Analyse und Paralleles und verteiltes Modelchecking

– Jens Bendisposto: Directed and Distributed Model Checking of B-Specifications Doktorarbeit, Akzeptiert zur Veröffentlichung, Januar 2015.

Volltext als Preprint verfügbar unter: `http://www.stups.hhu.de/w/Jens_Bendisposto`