Programming, Specification, and Interactive Theorem Proving

Towards a Unified Language based on Equational Logic, Rewriting Logic, and Type Theory

Dissertation
zur Erlangung des Doktorgrades
am Fachbereich Informatik
der Universität Hamburg

vorgelegt von

Mark-Oliver Stehr

geboren in Hamburg

Hamburg 2002

Genehmigt vom Fachbereich Informatik der Universität Hamburg auf Antrag von

Prof. Dr. Rüdiger Valk (Universität Hamburg)
Prof. Dr. José Meseguer (University of Illinois at Urbana-Champaign)
Prof. Dr. Dr. h.c. Wilfried Brauer (Technische Universität München)

Hamburg, den 23. September 2002
(Datum der Disputation)

Prof. Dr. Siegfried Stiehl
(Dekan des Fachbereichs Informatik)

Abstract

Inspired by a number of different applications of rewriting logic, equational logic, and type theory that we present and further advance in this thesis, we study a unified formalism based on the key aspects of these quite different lines of research. The resulting formalism, that we call the open calculus of constructions, is intended as a step towards our long-term goal of developing a unified language for programming, specification and interactive theorem proving.

We begin our work by exploring the application of rewriting logic as a semantic framework for concurrency. To this end, we give a unified treatment of different classes of Petri nets, a typical and important representative of a class of formalisms that are used for the modeling and specification of concurrent and distributed systems based on a multiset representation of a distributed state space. Specifically, we continue the line of research initiated by Meseguer and Montanari under the motto "Petri nets are monoids" by giving a rewriting semantics for different Petri nets classes. In particular, we have covered important high-level Petri net models, namely algebraic net specifications and colored Petri nets, and we have proved that the models of our representations are naturally isomorphic to the well-known Best-Devillers process semantics. Apart from their contribution to a conceptual unification in this field, the main practical advantage of our representations in rewriting logic is their executability, which allows us to use a rewriting engine such as Maude for the efficient symbolic execution of system models and for their analysis.

The next application addressed in this thesis is the use of type theory, more precisely the calculus of inductive constructions, as a logical framework and for metalogical reasoning. Specifically, we have used the COQ proof assistant in a formally rigorous development of a UNITY-style temporal logic, which generalizes the original UNITY approach in important aspects. Since all inference rules of the temporal logic are proved as theorems in the metalogic, the result of the development is a verified temporal logic library, which due to the use of labeled transition systems as a semantic basis, can be employed for a wide range of system models, Petri nets and rewriting logic specifications being particular examples. The development also includes a new application of the proposition-as-types interpretation in the context of compositional reasoning.

The use of membership equational logic or rewriting logic as a semantic and logical framework for higher-order languages, or more generally languages with binding constructs, obviously requires a first-order treatment of names and relevant operations such as substitutions. To systematically address such applications, we develop CINNI, a new calculus of names and substitutions, that takes names seriously in the sense that it does not abstract from names, and is generic in the sense that it can be instantiated to arbitrary object languages. Our calculus unifies the standard named notation and a notation based on de Bruijn indices by employing

a representation that was originally developed by Berkling for the λ-calculus. It furthermore nicely generalizes the calculus $\lambda\upsilon$ of explicit substitutions developed by Lescanne, and, as we show, most metatheoretic results can be generalized to the new calculus. We furthermore give a very general confluence result for the composition of CINNI with the equations or rules capturing the dynamics of the object language, and we in particular discuss how our approach can be applied to the representation of the untyped λ-calculus, Abadi and Cardelli's object calculus, also called the ς-calculus, and Milner's π-calculus for communicating and mobile systems. As a real-world application of CINNI we briefly discuss a specification of an active network programming language in the rewriting-logic-based language Maude.

We more specifically address the use of membership equational logic and rewriting logic as a first-order logical framework by representing an important class of pure type systems. Pure type systems generalize a variety of different type theories, including the calculus of constructions and its well-known subsystems, and can be seen as higher-order logics via the propositions-as-types interpretation. Following a methodology based on Meseguer's general logics in combination with rewriting logic as a concrete logical framework, we have studied representations of pure type systems at different levels of abstractions, ranging from an abstract textbook representation to a more concrete executable representation of an important subclass, which can directly serve as a type inference and type checking algorithm. The latter representation is based on a new notion of uniform pure type systems, which take names seriously thanks to the CINNI calculus and simultaneously offer a possible solution to the known problem with α-closure pointed out by Pollack. Using an example, in which we validate proofs developed with the LEGO proof assistant in an extension of the calculus of constructions with universes, we have demonstrated how our approach directly leads to an executable prototype in a rewriting logic language such as Maude.

As an application of type theory in the context of classical reasoning we study Howe's HOL/Nuprl connection, which addresses the problem of formal interoperability between proof assistents, from the viewpoint of Meseguer's general logics. We supplement Howe's semantic justification by a proof-theoretic correctness argument, a piece of work which has lead to proof-translation as new interesting application (explored in joint work with Naumov) that goes beyond Howe's original HOL/Nuprl connection. From a theoretical perspective we found that the core idea of the HOL/Nuprl connection, namely the beneficial coexistence of an intensional and an extensional logic in the same formal system, does not rely on any of the advanced concepts of Nuprl, but can equally well be used in Martin-Löf's type theory and can further be easily adopted to type theories in the line of calculus of constructions.

The final and main contribution of this thesis is the development of a formalism that we call the open calculus of constructions (OCC). It is based on the sur-

prisingly powerful interaction between its two key features, namely dependent types, in the spirit of Martin-Löf's type theory and the calculus of constructions, and the computational system of rewriting logic and its underlying membership equational logic, which is based on conditional rewriting modulo equations. The applications of membership equational logic, rewriting logic, and type theory, studied in this thesis have not only inspired the development of this unifying formalism, but they become applications of OCC itself and benefit from its use in an essential way. On the theoretical side, we introduce OCC by presenting a classical set-theoretic semantics and a formal system for which we prove soundness and consistency as a logic. The formal system is used to define derivable judgements together with their operational semantics, and is based on the ideas that we developed earlier in the context of uniform pure type systems. The model-theoretic semantics that we develop in this thesis is a very intuitive semantics with proof-irrelevance for impredicative universes, but unlike existing approaches it is more direct and can be given independently of the formal system. Using an experimental prototype of OCC, that we implemented in Maude following the approach to the specification of type theories mentioned before in combination with reflective techniques, we have developed a large collection of examples, many of which are closely related to the applications discussed earlier in this thesis. These examples do not only convey the pragmatics of OCC, but they simultaneously provide a proof-of-concept for our approach. Among the topics covered by our examples we find executable equational/behavioral specifications, programming with dependent types, symbolic execution of system models, formalization of algebraic and categorical concepts, inductive/coinductive theorem proving, and theorem proving modulo equational theories.

Zusammenfassung

Ausgehend von verschiedenen Anwendungen von Termersetzungslogik (rewriting logic), Gleichungslogik (equational logic), und Typtheorie, die wir in dieser Arbeit untersuchen und weiterentwickeln, studieren wir einen einheitlichen Formalismus, der auf den Hauptcharakteristika dieser recht unterschiedlichen Forschungsrichtungen basiert. Der resultierende Formalismus, den wir als offenes Kalkül der Konstruktionen (open calculus of constructions) bezeichnen, ist als erster Schritt in Richtung unseres langfristigen Ziels, der Entwicklung einer einheitlichen Sprache zur Programmierung, Spezifikation, und interaktivem Theorembeweisen, zu verstehen.

Unsere Arbeit beginnt mit der Untersuchung der Anwendbarkeit von Termersetzungslogik als semantisches Rahmenwerk (semantic framework) für Nebenläufigkeit. Hierzu geben wir eine einheitliche Behandlung verschiedener Petrinetz-Modelle, eines typischen und wichtigen Repräsentanten einer Klasse von Formalismen, die zur Modellierung und Spezifikation von nebenläufigen und verteilten Systemen benutzt werden und auf einer Multimengen-Repräsentation des verteilten Zustandsraumes basieren. Speziell führen wir die Forschungsrichtung fort, die von Meseguer und Montanari unter dem Motto "Petrinetze sind Monoide" initiiert wurde, indem wir eine Termersetzungssemantik für verschiedene Petrinetz-Klassen angeben. Insbesondere decken wir wichtige höhere Petrinetz-Modelle, genauer algebraische Netzspezifikationen und gefärbte Petrinetze ab, und wir beweisen, daß die Modelle unserer Repräsentationen eine natürliche Isomorphie zur bekannten Prozess-Semantik von Best und Devillers aufweisen. Zusätzlich zu ihrem Beitrag zu einer konzeptuellen Vereinheitlichung ist der wichtigste praktische Vorteil unserer Repräsentationen in Termersetzungslogik ihre Ausführbarkeit, die es uns erlaubt, Termersetzungsmaschinen wie z.B. Maude zur effizienten symbolischen Ausführung von Systemmodellen sowie zu deren Analyse zu nutzen.

Die nächste Anwendung, die in dieser Arbeit behandelt wird, betrifft die Verwendung der Typtheorie, genauer des induktiven Kalküls der Konstruktionen (inductive calculus of constructions) als logisches Rahmenwerk (logical framework) und für metalogische Beweise. Speziell haben wir den COQ-Beweisassistenten in einer rigoros formalen Entwicklung einer Temporallogik im UNITY-Stil eingesetzt, die den ursprünglichen Ansatz in wichtigen Punkten verallgemeinert. Da die Inferenzregeln der Temporallogik als Theoreme in der Metalogik bewiesen wurden, ist das Resultat dieser Entwicklung eine verifizierte Temporallogik-Bibliothek, die dank der Verwendung von allgemeinen, beschrifteten Transitionssystemen als semantische Basis für ein weites Spektrum von Systemmodellen (Petrinetze und Termersetzungslogik-Spezifikationen wären konkrete Beispiele) eingesetzt werden kann. Unsere Entwicklung enthält u.a. eine neuartige Anwendung der Interpretation von logischen Formeln als Typen (propositions-as-types interpretation) im Kontext der kompositionalen Verifikation.

Die Verwendung von Membership-Gleichungslogik (membership equational logic) oder Termersetzungslogik als semantisches und logisches Rahmenwerk für Sprachen höherer Ordnung, oder allgemeiner für Sprachen mit Konstrukten zur Namensbindung, benötigt offensichtlich eine Behandlung von Namen und relevanten Operationen wie Substitutionen mit Mitteln erster Ordnung. Um solche Anwendungen systematisch anzugehen, haben wir CINNI entwickelt, ein neues Kalkül der Namen und Substitutionen, das Namen in dem Sinne respektiert, daß es nicht von ihnen abstrahiert, und das ferner generisch ist in dem Sinne, daß es für beliebige Objekt-Sprachen instantiiert werden kann. Unser Kalkül vereinheitlicht die übliche Notation mit Namen und die namenlose Notation basierend auf de-Bruijn-Indizes, indem es eine Repräsentation benutzt, die ursprünglich von Berkling für das λ-Kalkül entwickelt wurde. Ferner verallgemeinert CINNI das Kalkül $\lambda\upsilon$ der expliziten Substitutionen von Lescanne und, wie wir zeigen, lassen sich die meisten metatheoretischen Resultate auf das neue Kalkül verallgemeinern. Schließlich zeigen wir ein sehr allgemeines Konfluenz-Resultat für die Komposition von CINNI mit Gleichungen oder Regeln, die die Dynamik der Objektsprache widerspiegeln, und wir diskutieren, wie unserer Ansatz zu Repräsentationen des ungetypten λ-Kalküls, des Objekt-Kalküls von Abadi und Cardelli, auch als ς-Kalkül bezeichnet, und Milners π-Kalküls für kommunizierende und mobile Systeme verwendet werden kann. Als eine Anwendung von CINNI aus der Praxis diskutieren wir kurz die Spezifikation einer Programmiersprache für aktive Netzwerke in der auf Termersetzungslogik basierten Sprache Maude.

Etwas spezifischer behandeln wir die Verwendung von Membership-Gleichungslogik und Termersetzungslogik als logisches Rahmenwerk erster Ordnung, indem wir eine wichtige Klasse der reinen Typsysteme (pure type systems) repräsentieren. Reine Typsysteme verallgemeinern eine reichhaltige Klasse von verschiedenen Typtheorien einschließlich des Kalküls der Konstruktionen und seiner bekannten Teilsysteme und können ferner als Logiken höherer Ordnung entspechend der Interpretation von logischen Formeln als Typen (propositions-as-types interpretation) angesehen werden. Unter Verwendung eines methodischen Ansatzes, der auf Meseguers allgemeinen Logiken (general logics) in Kombination mit Termersetzunglogik als konkretes logisches Rahmenwerk basiert, haben wir Repräsentationen von reinen Typsystemen auf verschiedenen Abstraktionsebenen studiert, angefangen von einer abstrakten lehrbuchartigen Repräsentation bis hin zu einer konkreteren und ausführbaren Repräsentation einer wichtigen Unterklasse, die in sehr direkter Weise als Typinferenz- und Typprüfungsalgorithmus genutzt werden kann. Die letztere Repräsentation basiert auf einem neuen Begriff der einheitlichen reinen Typsysteme (uniform pure type systems), die dank der Verwending des CINNI-Kalküls Namen respektieren, und gleichzeitig eine mögliche Lösung zum bekannten Problem der α-Abgeschlossenheit darstellen, auf das Pollack hinwies. Mit Hilfe eines Beispiels, der Validation von Bewei-

sen, die mit dem LEGO-Beweisassistent in einer Erweiterung des Kalküls der Konstruktionen um Universen ausgeführt wurden, zeigen wir, wie unser Ansatz unmittelbar zu einem ausführbaren Prototyp in einer auf Termersetzungs-Logik basierenden Sprache wie Maude führt.

Als Anwendung der Typtheorie in Kontext des klassischen Beweisens studieren wir Howes HOL/Nuprl-Verbindung, die sich mit dem Problem der formalen Interoperabilität zwischen Beweisassistenten aus der Sicht von Meseguers allgemeinen Logiken beschäftigt. Wir ergänzen Howes semantische Rechtfertigung durch ein beweistheoretisches Korrektheitsargument, eine Arbeit die Beweisübersetzung als neue interessante Anwendung hat (untersucht in Zusammenarbeit mit Naumov), und damit über Howes urspüngliche HOL/Nuprl-Verbindung hinausgeht. Aus theoretischer Sicht fanden wir, daß die Kernidee der HOL/Nuprl-Verbindung, nämlich die Koexistenz einer intensionalen und einer extensionalen Logik in einem einzigen formalen System, nicht auf die speziellen Eigenschaften von Nuprl angewiesen ist, sondern ebenso in Martin-Löfs Typtheorie verwendet werden kann, und ferner leicht an Typtheorien auf der Linie des Kalküls der Konstruktionen angepasst werden kann.

Der letzte und Hauptbeitrag dieser Arbeit ist die Entwicklung eines Formalismus, den wir als offenes Kalkül der Konstruktionen (open calculus of constructions, OCC) bezeichnen. Er basiert auf der überraschend mächtigen Interaktion zwischen seinen beiden Hauptcharakteristika, nämlich abhängigen Typen im Sinne von Martin-Löfs Typtheorie und des Kalküls der Konstruktionen, und dem Berechnungssystem der Termersetzungslogik und seiner unterliegenden Membership-Gleichungslogik, das auf bedingter Termersetzung modulo Gleichungen basiert. Die Anwendungen von Membership-Gleichungslogik, Termersetzungs-Logik und Typtheorie, die wir in dieser Arbeit studierten, waren nicht nur eine wichtige Inspiration für die Entwicklung dieses einheitlichen Formalismus, sondern werden selbst zu Anwendungen von OCC und profitieren wesentlich von seinem Einsatz. Auf der theoretischen Seite führen wir OCC durch Angabe einer klassischen, mengentheoretischen Semantik und eines formalen Systems ein, für das wir Korrektheit und logische Konzistenz beweisen. Das formale System wird benutzt, um die ableitbaren Urteile und ihre operationale Semantik zu definieren und basiert auf den Ideen, die wir zuvor im Kontext der einheitlichen, reinen Typsysteme entwickelt haben. Die modelltheoretische Semantik, die wir in dieser Arbeit entwickeln, ist eine sehr intuitive Semantik mit Beweis-Irrelevanz (proof-irrelevance semantics) für imprädikative Universen, aber anders als in existierenden Ansätzen ist sie direkter und kann unabhängig vom formalen System angegeben werden. Unter Verwendung eines experimentellen Prototyps von OCC, den wir mit Hilfe des obigen Ansatzes zur Spezifikation von Typtheorien in Kombination mit reflektiven Techniken in Maude entwickelt haben, wurde eine umfangreiche Sammlung von Beispielen erstellt, von denen viele eng mit den vorher diskutierten Anwendungen zusammenhängen. Die Bei-

spiele vermitteln nicht nur die Pragmatik von OCC sondern zeigen gleichzeitig die Realisierbarkeit und den Nutzen seiner Konzepte. Unter den Themen, die beispielsartig behandelt werden, finden sich ausführbare, gleichungsbasierte und verhaltensorientierte Spezifikationen, Programmierung mit abhängigen Typen, symbolische Ausführung von Systemmodellen, Formalisierung von algebraischen und kategorientheoretischen Begriffen, induktives/coinduktives Theorembeweisen und Beweisen modulo Gleichungstheorien.