Interactive Theorem Proving And Program Development Coqart The Calculus Of Inductive Constructions Texts In Theoretical Computer Science An Eatcs Series

This book constitutes the refereed proceedings of the 8th International Conference on Interactive Theorem Proving, ITP 2017, held in Brasilia, Brazil, in September 2017. The 28 full papers, 2 rough diamond papers, and 3 invited talk papers presented were carefully reviewed and selected from 65 submissions. The topics range from theoretical foundations to implementation aspects and applications in program verification, security and formalization of mathematical theories.

Commemorating the 50th anniversary of the first time a mathematical theorem was proven by a computer system, Freek Wiedijk initiated the present book in 2004 by inviting formalizations of a proof of the irrationality of the square root of two from scientists using various theorem proving systems. The 17 systems included in this volume are among the most relevant ones for the formalization of mathematics. The systems are showcased by presentation of the formalized proof and a description in the form of answers to a standard questionnaire. The 17 systems presented are HOL, Mizar, PVS, Coq, Otter/Ivy, Isabelle/Isar, Alfa/Agda, ACL2, PhoX, IMPS, Metamath, Theorema, Leog, Nuprl, Omega, B method, and Minlog.

This book constitutes the refereed proceedings of the International Symposium on Logical Foundations of Computer Science, LFCS 2013, held in San Diego, CA, USA in January 2013. The volume presents 29 revised refereed papers carefully selected by the program committee. The scope of the Symposium is broad and includes constructive mathematics and type theory; logic, automata and automatic structures; computability and randomness; logical foundations of programming; logical aspects of computational complexity; logic programming and constraints; automated deduction and interactive theorem proving; logical methods in protocol and program verification; logical methods in program specification and extraction; domain theory logic; logical foundations of database theory; equational logic and term rewriting; lambda and combinatory calculi; categorical logic and topological semantics; linear logic; epistemic and temporal logics; intelligent and multiple agent system logics; logics of proof and justification; nonmonotonic reasoning; logic in game theory and social software; logic of hybrid systems; distributed system logics; mathematical fuzzy logic; system design logics; and other logics in computer science.

This book constitutes the thoroughly refereed proceedings of the Third International Conference on Interactive Theorem Proving, ITP 2012, held in Princeton, NJ, USA, in August 2012. The 21 revised full papers presented together with 4 rough diamond papers, 3 invited talks, and one invited tutorial were carefully reviewed and selected from 40 submissions. Among the topics covered are formalization of mathematics; program abstraction and logics; data structures and synthesis; security; (non-)termination and automata; program verification; theorem prover development; reasoning about program execution; and prover infrastructure and modeling styles.

This text and software package introduces readers to automated theorem proving, while providing two approaches implemented as easy-to-use programs. These are semantic-tree theorem proving and resolution-refutation theorem proving. The early chapters introduce first-order predicate calculus, well-formed formulae, and their transformation to clauses. Then the author goes on to show how the two methods work and provides numerous examples for readers to try their hand at theorem-proving experiments. Each chapter comes with exercises designed to familiarise the readers with the ideas and with the software, and answers to many of the problems.

One-stop reference, self-contained, with theoretical topics presented in conjunction with implementations for which code is supplied.

Interactive Theorem Proving and Program DevelopmentCoq'Art: The Calculus of Inductive ConstructionsSpringer Science & Business Media

This book constitutes the refereed proceedings of the First International Conference on Interactive Theorem proving, ITP 2010, held in Edinburgh, UK, in July 2010. The 33 revised full papers presented were carefully reviewed and selected from 74 submissions. The papers are organized in topics such as counterexample generation, hybrid system verification, translations from one formalism to another, and cooperation between tools. Several verification case studies were presented, with applications to computational geometry, unification, real analysis, etc.

This book constitutes the refereed proceedings of the 4th International Conference on Interactive Theorem Proving, ITP 2013, held in Rennes, France, in July 2013. The 26 regular full papers presented together with 7 rough diamond papers, 3 invited talks, and 2 invited tutorials were carefully reviewed and selected from 66 submissions. The papers are organized in topical sections such as program verfication, security, formalization of mathematics and theorem prover development.

A textbook that teaches students to read and write proofs using Athena. Proof is the primary vehicle for knowledge generation in mathematics. In computer science, proof has found an additional use: verifying that a particular system (or component, or algorithm) has certain desirable properties. This book teaches students how to read and write proofs using Athena, a freely downloadable computer language. Athena proofs are machine-checkable and written in an intuitive natural-deduction style. The book contains more than 300 exercises, most with full solutions. By putting proofs into practice, it demonstrates the fundamental role of logic and proof in computer science as no other existing text does. Guided by examples and exercises, students are quickly immersed in the most useful high-level proof methods, including equational reasoning, several forms of induction, case analysis, proof by contradiction, and abstraction/specialization. The book includes auxiliary material on SAT and SMT solving, automated theorem proving, and logic programming. The book can be used by upper undergraduate or graduate computer science students with a basic level of programming and mathematical experience. Professional programmers, practitioners of formal methods, and researchers in logic-related branches of computer science will find it a valuable reference.

This book constitutes the proceedings of the 6th International Conference on Interactive Theorem Proving, ITP 2015, held in Nanjing, China, in August 2015. The 27 papers presented in this volume were carefully reviewed and selected from 54 submissions. The topics range from theoretical foundations to implementation aspects and applications in

program verification, security and formalization of mathematics.

Part I of this book is a practical introduction to working with the Isabelle proof assistant. It teaches you how to write functional programs and inductive definitions and how to prove properties about them in Isabelle's structured proof language. Part II is an introduction to the semantics of imperative languages with an emphasis on applications like compilers and program analysers. The distinguishing feature is that all the mathematics has been formalised in Isabelle and much of it is executable. Part I focusses on the details of proofs in Isabelle; Part II can be read even without familiarity with Isabelle's proof language, all proofs are described in detail but informally. The book teaches the reader the art of precise logical reasoning and the practical use of a proof assistant as a surgical tool for formal proofs about computer science artefacts. In this sense it represents a formal approach to computer science, not just semantics. The Isabelle formalisation, including the proofs and accompanying slides, are freely available online, and the book is suitable for graduate students, advanced undergraduate students, and researchers in theoretical computer science and logic.

The LNCS series reports state-of-the-art results in computer science research, development, and education, at a high level and in both printed and electronic form. Enjoying tight cooperation with the R&D community, with numerous individuals, as well as with prestigious organizations and societies, LNCS has grown into the most comprehensive computer science research forum available. The scope of LNCS, including its subseries LNAI and LNBI, spans the whole range of computer science and information technology including interdisciplinary topics in a variety of application fields. In parallel to the printed book, each new volume is published electronically in LNCS Online.

This book constitutes the refereed proceedings of the 7th International Conference on Interactive Theorem Proving, ITP 2016, held in Nancy, France, in August 2016. The 27 full papers and 5 short papers presented were carefully reviewed and selected from 55 submissions. The topics range from theoretical foundations to implementation aspects and applications in program verification, security and formalization of mathematical theories.

1. BASIC CONCEPTS OF INTERACTIVE THEOREM PROVING Interactive Theorem Proving ultimately aims at the construction of powerful reasoning tools that let us (computer scientists) prove things we cannot prove without the tools, and the tools cannot prove without us. Interaction typi cally is needed, for example, to direct and control the reasoning, to speculate or generalize strategic lemmas, and sometimes simply because the conjec ture to be proved does not hold. In software verification, for example, correct versions of specifications and programs typically are obtained only after a number of failed proof attempts and subsequent error corrections. Different interactive theorem provers may actually look quite different: They may support different logics (first-or higher-order, logics of programs, type theory etc.), may be generic or special-purpose tools, or may be tar geted to different applications. Nevertheless, they share common concepts and paradigms (e.g. architectural design, tactics, tactical reasoning etc.). The aim of this chapter is to describe the common concepts, design principles, and basic requirements of interactive theorem provers, and to explore the band width of variations. Having a 'person in the loop', strongly influences the design of the proof tool: proofs must remain comprehensible, - proof rules must be high-level and human-oriented, - persistent proof presentation and visualization becomes very important.

This book constitutes the refereed proceedings of the 13th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning, LPAR 2006, held in Phnom Penh, Cambodia in November 2006. The 38 revised full papers presented together with one invited talk were carefully reviewed and selected from 96 submissions.

This book constitutes the refereed proceedings of the 11th International Conference on Theorem Proving in Higher Order Logics, TPHOLs '98, held in Canberra, Australia, in September/October 1998. The 26 revised full papers presented were carefully reviewed and selected from a total of 52 submissions. Also included are two invited papers. The papers address all current aspects of theorem proving in higher order logics and formal verification and program analysis. Besides the HOL system, the theorem provers Coq, Isabelle, LAMBDA, LEGO, NuPrl, and PVS are discussed.

The book emphasizes the design of full-fledged, fully normalizing lambda calculus machinery, as opposed to the just weakly normalizing machines.

A practical introduction to the development of proofs and certified programs using Coq. An invaluable tool for researchers, students, and engineers interested in formal methods and the development of zero-fault software.

This book is concerned with techniques for formal theorem-proving, with particular reference to Cambridge LCF (Logic for Computable Functions). Cambridge LCF is a computer program for reasoning about computation. It combines the methods of mathematical logic with domain theory, the basis of the denotational approach to specifying the meaning of program statements. Cambridge LCF is based on an earlier theorem-proving system, Edinburgh LCF, which introduced a design that gives the user flexibility to use and extend the system. A goal of this book is to explain the design, which has been adopted in several other systems. The book consists of two parts. Part I outlines the mathematical preliminaries, elementary logic and domain theory, and explains them at an intuitive level, giving reference to more advanced reading; Part II provides sufficient detail to serve as a reference manual for Cambridge LCF. It will also be a useful guide for implementors of other programs based on the LCF approach.

Growing demands for the quality, safety, and security of software can only be satisfied by the rigorous application of formal methods during software design. This book methodically investigates the potential of first-order logic automated theorem provers for applications in software engineering. Illustrated by complete case studies on protocol verification, verification of security protocols, and logic-based software reuse, this book provides techniques for assessing the prover's capabilities and for selecting and developing an appropriate interface architecture.

This volume presents the proceedings of the Sixth International Joint Conference on the Theory and Practice of Software Engineering, TAPSOFT '95, held in Aarhus, Denmark in May 1995. TAPSOFT '95 celebrates the 10th anniversary of this conference series started in Berlin in 1985 to bring together theoretical computer scientists and software engineers (researchers and practitioners) with a view to discussing how formal methods can usefully be applied in software development. The volume contains seven invited papers, among them one by Vaugham Pratt on the recently revealed bug in the Pentium chip, and 44 revised full papers selected from a total of 147 submissions. In addition the TAPSOFT '95 proceedings contains 10 tool descriptions.

ACM Monograph Series: A Computational Logic focuses on the use of induction in proving theorems, including the use of lemmas and axioms, free variables, equalities, and generalization. The publication first elaborates on a sketch of the theory and two simple examples, a precise definition of the theory, and correctness of a tautology-checker. Topics include mechanical proofs, informal development, formal specification of the problem, well-founded relations, natural numbers, and literal atoms. The book then examines the use of type information to simplify formulas, use of axioms and lemmas as rewrite rules, and the use of definitions. Topics include nonrecursive functions, computing values, free variables in hypothesis, infinite backwards chaining, infinite looping, computing type sets, and type prescriptions. The manuscript takes a look at rewriting terms and simplifying clauses, eliminating destructors and irrelevance, using equalities, and generalization. Concerns include reasons for eliminating isolated hypotheses, precise statement of the generalization heuristic, restricting generalizations, precise use of equalities, and multiple destructors and infinite looping. The publication is a vital source of data for researchers interested in computational logic.

This book constitutes the refereed proceedings of the 9th International Conference on Interactive Theorem Proving, ITP 2018, held in Oxford, UK, in July 2018. The 32 full papers and 5 short papers presented were carefully reviewed and selected from 65 submissions. The papers feature research in the area of logical frameworks and interactive proof assistants. The topics include theoretical foundations and implementation aspects of the technology, as well as applications to verifying hardware and software systems to ensure their safety and security, and applications to the formal verication of mathematical results. Chapters 2, 10, 26, 29, 30 and 37 are available open access under a Creative Commons Attribution 4.0 International License via link.springer.com.

This book constitutes the proceedings of the 26th European Symposium on Programming, ESOP 2017, which took place in Uppsala, Sweden in April 2017, held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2017. The 36 papers presented in this volume were carefully reviewed and selected from 112 submissions. They cover traditional as well as emerging topics in programming languages. In detail they deal with semantic foundation and type system for probabilistic programming; techniqu3es for verifying concurrent or higher-order programs; programming languages for arrays or web data; program analysis and verification of non-standard program properties; foundation and application of interactive theorem proving; graph rewriting; separation logic; session type; type theory; and implicit computational complexity.

This book constitutes the refereed proceedings of the Second International Conference on Interactive Theorem proving, ITP 2011, held in Berg en Dal, The Netherlands, in August 2011. The 25 revised full papers presented were carefully reviewed and selected from 50 submissions. Among the topics covered are counterexample generation, verification, validation, term rewriting, theorem proving, computability theory, translations from one formalism to another, and cooperation between tools. Several verification case studies were presented, with applications to computational geometry, unification, real analysis, etc.

Handbook of Automated Reasoning.

A handbook to the Coq software for writing and checking mathematical proofs, with a practical engineering focus. The technology of mechanized program verification can play a supporting role in many kinds of research projects in computer science, and related tools for formal proof-checking are seeing increasing adoption in mathematics and engineering. This book provides an introduction to the Coq software for writing and checking mathematical proofs. It takes a practical engineering focus throughout, emphasizing techniques that will help users to build, understand, and maintain large Coq developments and minimize the cost of code change over time. Two topics, rarely discussed elsewhere, are covered in detail: effective dependently typed programming (making productive use of a feature at the heart of the Coq system) and construction of domain-specific proof tactics. Almost every subject covered is also relevant to interactive computer theorem proving in general, not just program verification, demonstrated through examples of verified programs applied in many different sorts of formalizations. The book develops a unique automated proof style and applies it throughout; even experienced Coq users may benefit from reading about basic Coq concepts from this novel perspective. The book also offers a library of tactics, or programs that find proofs, designed for use with examples in the book. Readers will acquire the necessary skills to reimplement these tactics in other settings by the end of the book. All of the code appearing in the book is freely available online.

This book constitutes the proceedings of the 5th International Conference on Interactive Theorem Proving, ITP 2014, Held as Part of the Vienna Summer of Logic, VSL 2014, in Vienna, Austria, in July 2014. The 35 papers presented in this volume were carefully reviewed and selected from 59 submissions. The topics range from theoretical foundations to implementation aspects and applications in program verification, security and formalization of mathematics.

Interactive theorem proving is the modern way of formalizing mathematics using a computer as a proof assistant, helping solve simple tasks and keeping an order on the proofs. As it is an overwhelming task to prove a program correct or prove that an implementation conforms to its UML-specification, this book draws a line to show up how far current cutting edge research has succeeded in tackling this problem. Using examples from algorithm development, Java bytecode verification and UML state machine analysis the author introduces current trends in interactive theorem proving technology using Coq, Isabelle, and model checking. -- from back cover.

This volume constitutes the proceedings of the 22nd International Conference on Theorem Proving in Higher Order Logics (TPHOLs 2009), which was held during August 17-20, 2009 in Munich, Germany. TPHOLs covers all aspects of theorem proving in higher order logics as well as related topics in theorem proving and veri?cation. There were 55 papers submitted to TPHOLs 2009 in the full research c- egory, each of which was refereed by at least three reviewers selected by the ProgramCommittee. Of these submissions, 26 researchpapers and 1 proofpearl were accepted for presentation at the conference and publication in this v- ume. In keeping with longstanding tradition, TPHOLs 2009 also o?ered a venue for the presentation of emerging trends, where researchers invited discussion by means of a brief introductory talk and then discussed their work at a poster session. A supplementary proceedings volume was published as a 2009 technical report of the Technische Universit¨ at Munc ¨ hen. The organizers are grateful to David Basin, John Harrison and Wolfram Schulte for agreeing to give invited talks. We also invited four tool devel- ers to give tutorials about their systems. The following speakers kindly accepted our invitation and we are grateful to them: John Harrison (HOL Light), Adam Naumowicz (Mizar), Ulf Norell (Agda) and Carsten Schur ¨ mann (Twelf).

This book presents reflections on the occasion of 20 years on the KeY project that focuses on deductive software verification. Since the inception of the KeY project two decades ago, the area of deductive verification has evolved considerably. Support for real world programming languages by deductive program verification tools has become prevalent. This required to overcome significant theoretical and technical challenges to support advanced software engineering and programming concepts. The community became more interconnected with a competitive, but friendly and supportive environment. We took the 20-year anniversary of KeY as an opportunity to invite researchers, inside and outside of

the project, to contribute to a book capturing some state-of-the-art developments in the field. We received thirteen contributions from recognized experts of the field addressing the latest challenges. The topics of the contributions range from tool development, effciency and usability considerations to novel specification and verification methods. This book should offer the reader an up-to-date impression of the current state of art in deductive verification, and we hope, inspire her to contribute to the field and to join forces. We are looking forward to meeting you at the next conference, to listen to your research talks and the resulting fruitful discussions and collaborations.

Computer-Aided Reasoning: ACL2 Case Studies illustrates how the computer-aided reasoning system ACL2 can be used in productive and innovative ways to design, build, and maintain hardware and software systems. Included here are technical papers written by twenty-one contributors that report on self-contained case studies, some of which are sanitized industrial projects. The papers deal with a wide variety of ideas, including floating-point arithmetic, microprocessor simulation, model checking, symbolic trajectory evaluation, compilation, proof checking, real analysis, and several others. Computer-Aided Reasoning: ACL2 Case Studies is meant for two audiences: those looking for innovative ways to design, build, and maintain hardware and software systems faster and more reliably, and those wishing to learn how to do this. The former audience includes project managers and students in survey-oriented courses. The latter audience includes students and professionals pursuing rigorous approaches to hardware and software engineering or formal methods. Computer-Aided Reasoning: ACL2 Case Studies can be used in graduate and upper-division undergraduate courses on Software Engineering, Formal Methods, Hardware Design, Theory of Computation, Artificial Intelligence, and Automated Reasoning. The book is divided into two parts. Part I begins with a discussion of the effort involved in using ACL2. It also contains a brief introduction to the ACL2 logic and its mechanization, which is intended to give the reader sufficient background to read the case studies. A more thorough, textbook introduction to ACL2 may be found in the companion book, Computer-Aided Reasoning: An Approach. The heart of the book is Part II, where the case studies are presented. The case studies contain exercises whose solutions are on the Web. In addition, the complete ACL2 scripts necessary to formalize the models and prove all the properties discussed are on the Web. For example, when we say that one of the case studies formalizes a floating-point multiplier and proves it correct, we mean that not only can you read an English description of the model and how it was proved correct, but you can obtain the entire formal content of the project and replay the proofs, if you wish, with your copy of ACL2. ACL2 may be obtained from its home page. The results reported in each case study, as ACL2 input scripts, as well as exercise solutions for both books, are available from this page.

The report presents an outline of the principle features of an on-line interactive theorem-proving program, and a brief account of the results of some experiments with it. The program has been used to obtain proofs of new mathematical results recently announced without proof in the Notices of the American Mathematical Society. (Author).