PUBLICATIONS OF THE NEWTON INSTITUTE

Semantics and Logics of Computation

Publications of the Newton Institute

Edited by H. P. F. Swinerton-Dyer
Executive Director, Isaac Newton Institute for Mathematical Sciences

The Isaac Newton Institute of Mathematical Sciences of the University of Cambridge exists to stimulate research in all branches of the mathematical sciences, including pure mathematics, statistics, applied mathematics, theoretical physics, theoretical computer science, mathematical biology and economics. The four six-month long research programmes it runs each year bring together leading mathematical scientists from all over the world to bring together leading mathematical scientists from all over the world to exchange ideas through seminars, teaching and informal interaction.

Associated with the programmes are two types of publication. The first contains lecture courses, aimed at making the latest developments accessible to a wider audience and providing an entry to the area. The second contains proceedings of workshops and conferences focusing on the most topical aspects of the subjects.

Semantics and Logics of Computation

Edited by

Andrew M. Pitts
Computer Laboratory, Cambridge University

Peter Dybjer
Institutionen för Datavetenskap, Chalmers Tekniska Högskola

CAMBRIDGE
UNIVERSITY PRESS

CAMBRIDGE UNIVERSITY PRESS
Cambridge, New York, Melbourne, Madrid, Cape Town, Singapore, São Paulo

Cambridge University Press
The Edinburgh Building, Cambridge CB2 8RU, UK

Published in the United States of America by Cambridge University Press, New York

www.cambridge.org
Information on this title: www.cambridge.org/9780521580571

First published 1997

A catalogue record for this publication is available from the British Library

ISBN 978-0-521-58057-1 hardback

Transferred to digital printing 2007

Contents

Contributors

Samson Abramsky Department of Computer Science, University of Edinburgh, The King's Buildings, Mayfield Road, Edinburgh EH9 3JZ, UK

Thierry Coquand Institutionen för Datavetenskap, Chalmers Tekniska Högskola, S-412 96 Göteborg, Sweden

Peter Dybjer Institutionen för Datavetenskap, Chalmers Tekniska Högskola, S-412 96 Göteborg, Sweden

Martin Hofmann Fachbereich Mathematik, Technische Hochschule Darmstadt, Schlossgartenstrasse 7, D-64289 Darmstadt, Germany

Martin Hyland Department of Pure Mathematics and Mathematical Statistics, Cambridge University, 16 Mill Lane, Cambridge CB2 1SB, UK

Eugenio Moggi Dipartimento di Informatica, Università di Genova, via Dodecaneso 35, 16146 Genova, Italy

Mogens Nielsen Department of Computer Science, Aarhus Universitet, Ny Munkegade, Bldg. 540, DK-8000 Aarhus C., Denmark

Andrew Pitts Computer Laboratory, Cambridge University, Pembroke Street, Cambridge CB2 3QG, UK

Glynn Winskel Department of Computer Science, Aarhus Universitet, Ny Munkegade, Bldg. 540, DK-8000 Aarhus C., Denmark

Preface

This book is based on material presented at a summer school on Semantics and Logics of Computation that took place at the Isaac Newton Institute for Mathematical Sciences, Cambridge UK, in September 1995. The school was sponsored by the EU ESPRIT Basic Research Action on *Categorical Logic in Computer Science* (CLiCS I & II) and aimed to present some modern developments in semantics and logics of computation in a series of graduate-level lectures. Most of the material presented here has not previously been accessible in such a coherent and digestible form. This Preface gives a thematic overview of the contents of the book. It also briefly sketches the history of the two CLiCS projects which came to an end with the summer school.

Games, proofs and programs One of the most exciting recent developments in programming semantics has been the use of games and strategies to provide a more fine grained semantics than that provided by domain-theoretic models. This 'intensional semantics' aims to combine the good mathematical and structural properties of traditional denotational semantics with the ability to capture dynamical and interactive aspects of computation, and to embody the computational intuitions of operational semantics. More pragmatically, game semantics have been used to solve long-standing 'full abstraction' problems for PCF (a simple language for Programming Computable arithmetic Functions of higher type) and Idealised Algol. In other words games have provided syntax-independent models of these languages which precisely capture the operationally defined notion of 'contextual equivalence' of program expressions.

Abramsky's chapter on *Semantics of Interaction: an Introduction to Game Semantics* and Hyland's chapter on *Game Semantics* together provide an excellent introduction to these developments. They both present basic notions of games and strategies, although from somewhat different perspectives. The connection between strategies and partial computation and between winning strategies and terminating computation is discussed. Games and strategies are organised into a category with sufficient structure to model Linear Logic. Abramsky describes the use of games for modelling relationally parametric polymorphism. Hyland describes an exponential comonad and the associated cartesian closed co-Kleisli category. He goes on to introduce the 'dialogue games' which he and Ong used for constructing a fully abstract model of PCF. (Another fully abstract game semantics of PCF due to Abramsky, Jagadeesan, and Malacaria instead uses games and history-free strategies.)

Games are also one of the themes in Coquand's chapter on the *Computational Content of Classical Logic*. This area of research has developed rapidly during the last few years partly because of its connection to classical Linear Logic and to theories of 'continuation-passing' semantics. Coquand discusses three ways of extracting computational content from proofs in classical logic: the negative translation of

classical proofs into intuitionistic proofs; the interpretation of formulas and proofs in classical logic as games and strategies; and the use of formal topology. He makes a special point of connecting the old tradition of proof theory of classical logic with recent developments in computer science. He also follows Gentzen in analysing a number of actual mathematical examples.

Categories, types and computation Type theories of one kind or another form an integral part of many computer-oriented formalisations of mathematics. For example, several mechanised proof assistants for intuitionistic type theory have been developed (notably as part of CLiCS's sister project, the ESPRIT Basic Research Action on *Types for Proofs and Programs*). Within the context of semantics and logics of computation, dependent types have been incorporated into metalanguages for denotational semantics.

Hofmann's chapter on the *Syntax and Semantics of Dependent Types* describes the category-theoretic interpretation of intuitionistic type theory. He shows how categorical notions can be used to clarify what it means to be a model of a theory with dependent types and thus to facilitate metamathematical studies. He begins by giving an introduction to dependent types and to some of the most important type theories such as Martin-Löf's Type Theory and The Calculus of Constructions. He then discusses several kinds of models, both syntactic and semantic. The chapter ends by using the categorical machinery to prove conservativity of Martin-Löf's Logical Framework over ordinary type theory.

Moggi's chapter on *Metalanguages and Applications* develops an incremental approach to the denotational semantics of programming languages making use of the categorical notion of 'strong monad'. The basic idea is to use monads as a structuring device. Monads have been shown to capture in abstract form essential common properties of various types of computation (such as partial, imperative, non-deterministic, parallel, and exception-raising computation); and for this reason they have also recently become popular as a structuring device in functional programming. In his chapter, Moggi also makes essential use of a metalanguage of dependent types (of the kind discussed in Hofmann's chapter) when presenting denotational semantics. The chapter ends with a treatment of recursive definitions in the context of metalanguages, incorporating some ideas from recent axiomatic and synthetic treatments of domain theory.

Operationally-Based Theories of Program Equivalence Pitts' chapter describes techniques for proving properties of programs which are based directly on the syntax and operational semantics of programming languages. A central result is the co-inductive characterisation of contextual equivalence between programs in a lazy functional programming language. As a consequence contextual equivalences between programs can be proved by co-induction. This technique has already been exploited extensively in concurrency theory, where it is used for proving bisimulation equivalence between processes. Pitts gives a number of

concrete example of how the technique is used for proving equivalences between lazy functional programs. Although this operationally-based approach bypasses some of the mathematical intricacies of denotational semantics, it also borrows denotational ideas. This is illustrated in the last part of the chapter where a syntactic analogue of Scott's induction principle for fixpoint recursion is developed.

Categories in Concurrency The chapter by Winskel and Nielsen presents a range of models for parallel and concurrent computation. They consider both interleaving models such as transition system, synchronisation trees and languages, and models where concurrency is represented more explicitly ('true concurrency') by a form of causal independence, such as Petri Nets, event structures, pomsets and Mazurkiewicz traces. They show how these models can be unified by putting them in a category-theoretic framework. As a consequence they are able to give language independent characterisations of various constructs such as parallel composition. They also discuss bisimulation equivalence, the most important notion of equivalence for transition systems, and show how the categorical framework helps to find an appropriate corresponding notion for other models such as Petri nets.

The CLiCS projects The summer school which gave rise to this book marked the end of six years of funding by the EU of ESPRIT Basic Research Actions on *Categorical Logic in Computer Science*. The first CLiCS project started in October 1989 and was succeeded by CLiCS-II which continued until December 1995. The following universities and research institutes participated: Aarhus in Denmark; Cambridge, Imperial College, Manchester, and Sussex in England; ENS, Paris and INRIA (Rocquencourt) in France; Darmstadt and GMD (Bonn) in Germany; Genova and Parma in Italy; and Chalmers in Sweden. One purpose of this book is to publish in accessible form some of the highlights of the research which evolved during these projects.

The initial aim of CLiCS was to bring together a group of mathematicians and theoretical computer scientists to work on foundational problems in computer science. This was motivated by the belief that significant advances in language design, programming methodology, and even language implementation and computer architecture, are based on the insights and analytical techniques made available by theoretical work. As the name of the project suggests a unifying theme was the application of concepts and techniques from Category Theory and Categorical Logic. The project encompassed four main areas: domain theory; semantics and logics of programming languages; concurrency theory and linear logic; and type theory and constructive mathematics. It was organised into nine research topics: sequentiality and stability; duality in domain theory; logic of inductive and recursive datatypes; subtypes and polymorphism in higher order languages; higher order modal program logics; monadic programming language semantics; applied linear logic; computational content of classical logic; and symbolic computation. Material on many, but by no means all, of these topics may be found in the chapters that follow.

Acknowledgements We wish to thank the European Union and the Isaac Newton Institute for Mathematical Sciences for the funding which made the summer school possible; to thank the lecturers and participants for making it such an enjoyable, if exhausting week; and to thank the staff of the Newton Institute for their quiet efficiency. Finally, we would like to thank David Tranah at CUP for his encouragement and assistance in turning the lectures into this book.

Andrew Pitts
Peter Dybjer September 1996