Texts in Theoretical Computer Science. An EATCS Series

Roberto Bruni · Ugo Montanari

Models of Computation

Springer

Roberto Bruni
Dipartimento di Informatica
Università di Pisa
Pisa
Italy

Ugo Montanari
Dipartimento di Informatica
Università di Pisa
Pisa
Italy

ISSN 1862-4499
Texts in Theoretical Computer Science. An EATCS Series
ISBN 978-3-319-42898-7 ISBN 978-3-319-42900-7 (eBook)
DOI 10.1007/978-3-319-42900-7

Library of Congress Control Number: 2017937525

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Mathematical reasoning may be regarded rather schematically as the exercise of a combination of two facilities, which we may call intuition and ingenuity.

Alan Turing[1]

[1] The purpose of ordinal logics (from Systems of Logic Based on Ordinals), Proceedings of the London Mathematical Society, series 2, vol. 45, no. 1, 161–228, 1939.

Foreword

Based on more than fifteen years of teaching, this book provides an exceptionally useful addition to the introductory literature on Models of Computation. It covers a wealth of material on imperative, functional, concurrent and probabilistic computation, their models and reasoning techniques. It does so in a carefully pedagogic way, honed over the years with the help of the students at Pisa. Supplemented with educative examples, explanations and problems (accompanied by solutions) and including a coverage of the techniques of probabilistic and quantitative computation, it provides the student with a rich background and solid foundation for future study and research.

If quality of teaching is to be judged from the quality of students, then teaching in Pisa scores very highly indeed. Over several generations brilliant students from Pisa have made their mark in Computer Science in Europe and the US. All those I know have come under the influence of Ugo Montanari and more recently that of his colleague Roberto Bruni. The broader access this book gives to their expertise and teaching is very welcome.

On a personal note it is a pleasure to acknowledge the friendly collaboration that Ugo Montanari has helped foster over many years, from the first time I visited Pisa for the Pisa Summer School in 1978 — my PhD thesis was born on the grassy area by the Cathedral and Leaning Tower — through to Ugo leading an EU project in which I participated. On the possible start of the UK's withdrawal from the EU I look back with nostalgia to those days of openness and cooperation, now fallen victim to the selfish interference of politicians. This book in its unifying treatment of a criss-cross of work carried out in Europe and the US is a testimony to the power of collaboration in research and education.

Cambridge
July 2016

Glynn Winskel

Preface

The origins of this book have their roots in more than 15 years of teaching a course on formal semantics to Computer Science graduate students in Pisa, originally called *Fondamenti dell'Informatica: Semantica* (*Foundations of Computer Science: Semantics*) and covering models for imperative, functional and concurrent programming. It later evolved into *Tecniche di Specifica e Dimostrazione* (*Techniques for Specifications and Proofs*) and finally into the currently running *Models of Computation*, where additional material on probabilistic models is included.

The objective of this book, as well as of the above courses, is to present different *models of computation* and their basic *programming paradigms*, together with their mathematical descriptions, both *concrete* and *abstract*. Each model is accompanied by some relevant formal techniques for reasoning on it and for proving some properties.

To this aim, we follow a rigorous approach to the definition of the *syntax*, the *typing* discipline and the *semantics* of the paradigms we present, i.e., the way in which well-formed programs are written, ill-typed programs are discarded and the meaning of well-typed programs is unambiguously defined, respectively. In doing so, we focus on basic proof techniques and do not address more advanced topics in detail, for which classical references to the literature are given instead.

After the introductory material (Part I), where we fix some notation and present some basic concepts such as term signatures, proof systems with axioms and inference rules, Horn clauses, unification and goal-driven derivations, the book is divided into four main parts (Parts II-V), according to the different styles of the models we consider:

IMP: imperative models, where we apply various incarnations of well-founded induction and introduce λ-notation and concepts like structural recursion, program equivalence, compositionality, completeness and correctness, and also complete partial orders, continuous functions, and fixpoint theory;

HOFL: higher-order functional models, where we study the role of type systems, the main concepts from domain theory and the distinction between lazy and eager evaluation;

CCS, π: concurrent, nondeterministic and interactive models, where, starting from operational semantics based on labelled transition systems, we introduce the notions of bisimulation equivalences and observational congruences, and overview some approaches to name mobility, and temporal and modal logic system specifications;

PEPA: probabilistic/stochastic models, where we exploit the theory of Markov chains and of probabilistic reactive and generative systems to address quantitative analysis of, possibly concurrent, systems.

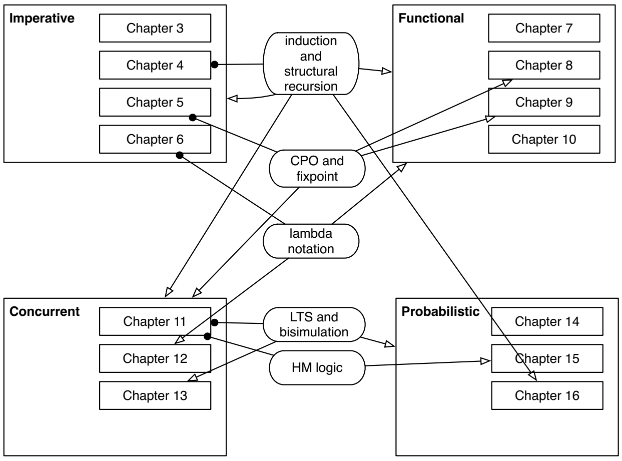
Each of the above models can be studied separately from the others, but previous parts introduce a body of notions and techniques that are also applied and extended in later parts.

Parts I and II cover the essential, classic topics of a course on formal semantics.

Part III introduces some basic material on process algebraic models and temporal and modal logic for the specification and verification of concurrent and mobile systems. CCS is presented in good detail, while the theory of temporal and modal logic, as well as the π-calculus, are just overviewed. The material in Part III can be used in conjunction with other textbooks, e.g., on model checking or the π-calculus, in the context of a more advanced course on the formal modelling of distributed systems.

Part IV outlines the modelling of probabilistic and stochastic systems and their quantitative analysis with tools like PEPA. It provides the basis for a more advanced course on quantitative analysis of sequential and interleaving systems.

The diagram that highlights the main dependencies is represented below:



The diagram contains a rectangular box for each chapter/part and a round-cornered box for each subject: a line with a filled-circle end joins a subject to the chapter where it is introduced, while a line with an arrow end links a subject to a chapter or part where it is used. In short:

Induction and recursion:	various principles of induction and the concept of structural recursion are introduced in Chapter 4 and used extensively in all subsequent chapters.
CPO and fixpoint:	the notions of complete partial order and fixpoint computation are first presented in Chapter 5. They provide the basis for defining the denotational semantics of IMP and HOFL. In the case of HOFL, a general theory of product and functional domains is also introduced (Chapter 8). The notion of fixpoint is also used to define a particular form of equivalence for concurrent and probabilistic systems, called bisimilarity, and to define the semantics of modal logic formulas.
Lambda notation:	λ-notation is a useful syntax for managing anonymous functions. It is introduced in Chapter 6 and used extensively in Part III.
LTS and bisimulation:	Labelled transition systems are introduced in Chapter 11 to define the operational semantics of CCS in terms of the interactions performed. They are then extended to deal with name mobility in Chapter 13 and with probabilities in Part V. A bisimulation is a relation over the states of an LTS that is closed under the execution of transitions. The above mentioned bisimilarity is the coarsest bisimulation relation. Various forms of bisimulation are studied in Parts IV and V.
HM-logic:	Hennessy-Milner logic is the logic counterpart of bisimilarity: two states are bisimilar if and only if they satisfy the same set of HM-logic formulas. In the context of probabilistic systems, the approach is extended to Larsen-Skou logic in Chapter 15.

Each chapter of the book concludes with a list of exercises that span over the main techniques introduced in that chapter. Solutions to selected exercises are collected at the end of the book.

Pisa,
February 2016

Roberto Bruni
Ugo Montanari

Acknowledgements

We want to thank our friend and colleague Pierpaolo Degano for encouraging us to prepare this book and submit it to the EATCS Texts series. We thank the anonymous reviewers of the book for helping us to improve the presentation and remove as many errors and typos as possible: those left are our full responsibility. We thank Ronan Nugent and all the people at Springer for their patience and support: they have done great editorial work. We acknowledge all the students of the course on *Models of Computation (MOD)* in Pisa for helping us to refine the presentation of the material in the book and to eliminate many typos and shortcomings from preliminary versions of this text. Last but not least, we thank Lorenzo Galeotti, Andrea Cimino, Lorenzo Muti, Gianmarco Saba, and Marco Stronati, former students of the course on *Models of Computation*, who helped us with the LaTeX preparation of preliminary versions of this book, in the form of lecture notes.

Contents

Acronyms

$\sim$	operational equivalence in IMP (see Definition 3.3)
$\equiv_{den}$	denotational equivalence in HOFL (see Definition 10.4)
$\equiv_{op}$	operational equivalence in HOFL (see Definition 10.3)
$\simeq$	CCS strong bisimilarity (see Definition 11.5)
$\approx$	CCS weak bisimilarity (see Definition 11.16)
$\approxeq$	CCS weak observational congruence (see Section 11.8.2)
$\cong$	CCS dynamic bisimilarity (see Definition 11.18)
$\sim_E$	π-calculus early bisimilarity (see Definition 13.4)
$\sim_L$	π-calculus late bisimilarity (see Definition 13.4)
$\simeq_E$	π-calculus strong early full bisimilarity (see Section 13.5.3)
$\simeq_L$	π-calculus strong late full bisimilarity (see Section 13.5.3)
$\approx_E$	π-calculus weak early bisimilarity (see Section 13.5.4)
$\approx_L$	π-calculus weak late bisimilarity (see Section 13.5.4)
$\mathscr{A}$	interpretation function for the denotational semantics of IMP arithmetic expressions (see Section 6.2.1)
ack	Ackermann function (see Example 4.18)
Aexp	set of IMP arithmetic expressions (see Chapter 3)
$\mathscr{B}$	interpretation function for the denotational semantics of IMP boolean expressions (see Section 6.2.2)
Bexp	set of IMP boolean expressions (see Chapter 3)
$\mathbb{B}$	set of booleans
$\mathscr{C}$	interpretation function for the denotational semantics of IMP commands (see Section 6.2.3)
CCS	Calculus of Communicating Systems (see Chapter 11)
Com	set of IMP commands (see Chapter 3)
CPO	Complete Partial Order (see Definition 5.11)
$\text{CPO}_\perp$	Complete Partial Order with bottom (see Definition 5.12)
CSP	Communicating Sequential Processes (see Section 16.2)
CTL	Computation Tree Logic (see Section 12.2.2)
CTMC	Continuous Time Markov Chain (see Definition 14.15)
DTMC	Discrete Time Markov Chain (see Definition 14.14)

Env	set of HOFL environments (see Chapter 9)
fix	(least) fixpoint (see Section 5.2.2)
FIX	(greatest) fixpoint
gcd	greatest common divisor
HML	Hennessy-Milner modal Logic (see Section 11.6)
HM-logic	Hennessy-Milner modal Logic (see Section 11.6)
HOFL	a Higher-Order Functional Language (see Chapter 7)
IMP	a simple IMPerative language (see Chapter 3)
int	integer type in HOFL (see Definition 7.2)
Loc	set of locations (see Chapter 3)
LTL	Linear Temporal Logic (see Section 12.2.1)
LTS	Labelled Transition System (see Definition 11.2)
lub	least upper bound (see Definition 5.7)
$\mathbb{N}$	set of natural numbers
$\mathscr{P}$	set of closed CCS processes (see Definition 11.1)
PEPA	Performance Evaluation Process Algebra (see Chapter 16)
Pf	set of partial functions on natural numbers (see Example 5.13)
PI	set of partial injective functions on natural numbers (see Problem 5.12)
PO	Partial Order (see Definition 5.1)
PTS	Probabilistic Transition System (see Section 14.4.2)
$\mathbb{R}$	set of real numbers
$\mathscr{T}$	set of HOFL types (see Definition 7.2)
Tf	set of total functions from $\mathbb{N}$ to $\mathbb{N}_\perp$ (see Example 5.14)
Var	set of HOFL variables (see Chapter 7)
$\mathbb{Z}$	set of integers