

# **Texts in Theoretical Computer Science. An EATCS Series**

## **Series editors**

Monika Henzinger, Faculty of Science, Universität Wien, Vienna, Austria

Juraj Hromkovič, Department of Computer Science, ETH Zentrum, Zürich, Switzerland

Mogens Nielsen, Department of Computer Science, Aarhus Universitet, Aarhus, Denmark

Grzegorz Rozenberg, Leiden Center of Advanced Computer Science, Leiden, The Netherlands

Arto Salomaa, Turku Centre of Computer Science, Turku, Finland

More information about this series at <http://www.springer.com/series/3214>

Roberto Bruni · Ugo Montanari

# Models of Computation

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

© Springer International Publishing Switzerland 2017

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

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*<sup>1</sup>

---

<sup>1</sup> 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  $\lambda$ -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,  $\pi$ :** 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.

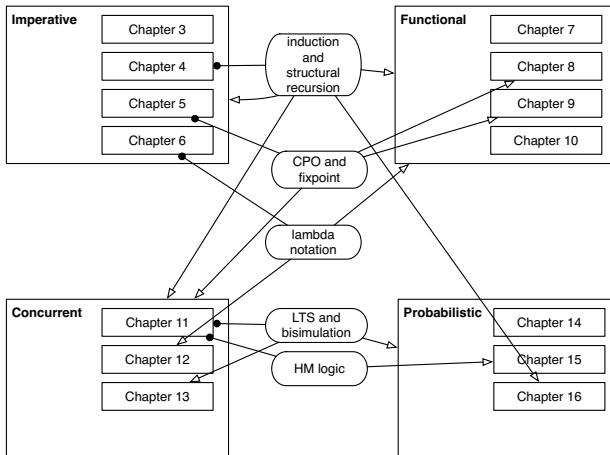
Each of the above models can be studied separately from the others, but previous parts introduce a body of notions and techniques that are also applied and extended in later parts.

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  $\pi$ -calculus, are just overviewed. The material in Part III can be used in conjunction with other textbooks, e.g., on model checking or the  $\pi$ -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:	$\lambda$ -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:	Hennessey-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 L<sup>A</sup>T<sub>E</sub>X preparation of preliminary versions of this book, in the form of lecture notes.

# Contents

## Part I Preliminaries

<b>1</b>	<b>Introduction</b>	3
1.1	Structure and Meaning	3
1.1.1	Syntax, Types and Pragmatics	4
1.1.2	Semantics	4
1.1.3	Mathematical Models of Computation	6
1.2	A Taste of Semantic Methods: Numerical Expressions	9
1.3	Applications of Semantics	17
1.4	Key Topics and Techniques	20
1.4.1	Induction and Recursion	20
1.4.2	Semantic Domains	22
1.4.3	Bisimulation	24
1.4.4	Temporal and Modal Logics	25
1.4.5	Probabilistic Systems	25
1.5	Chapter Contents and Reading Guide	26
1.6	Further Reading	28
	References	30
<b>2</b>	<b>Preliminaries</b>	33
2.1	Notation	33
2.1.1	Basic Notation	33
2.1.2	Signatures and Terms	34
2.1.3	Substitutions	35
2.1.4	Unification Problem	35
2.2	Inference Rules and Logical Systems	37
2.3	Logic Programming	45
	Problems	47

## Part II IMP: a Simple Imperative Language

<b>3</b>	<b>Operational Semantics of IMP</b>	53
3.1	Syntax of IMP	53
3.1.1	Arithmetic Expressions	54
3.1.2	Boolean Expressions	54
3.1.3	Commands	55
3.1.4	Abstract Syntax	55
3.2	Operational Semantics of IMP	56
3.2.1	Memory State	56
3.2.2	Inference Rules	57
3.2.3	Examples	61
3.3	Abstract Semantics: Equivalence of Expressions and Commands	66
3.3.1	Examples: Simple Equivalence Proofs	67
3.3.2	Examples: Parametric Equivalence Proofs	68
3.3.3	Examples: Inequality Proofs	70
3.3.4	Examples: Diverging Computations	72
	Problems	75
<b>4</b>	<b>Induction and Recursion</b>	77
4.1	Noether's Principle of Well-Founded Induction	77
4.1.1	Well-Founded Relations	77
4.1.2	Noetherian Induction	83
4.1.3	Weak Mathematical Induction	84
4.1.4	Strong Mathematical Induction	85
4.1.5	Structural Induction	85
4.1.6	Induction on Derivations	88
4.1.7	Rule Induction	89
4.2	Well-Founded Recursion	93
	Problems	98
<b>5</b>	<b>Partial Orders and Fixpoints</b>	103
5.1	Orders and Continuous Functions	103
5.1.1	Orders	104
5.1.2	Hasse Diagrams	106
5.1.3	Chains	110
5.1.4	Complete Partial Orders	111
5.2	Continuity and Fixpoints	114
5.2.1	Monotone and Continuous Functions	114
5.2.2	Fixpoints	116
5.3	Immediate Consequence Operator	119
5.3.1	The Operator $\hat{R}$	120
5.3.2	Fixpoint of $\hat{R}$	121
	Problems	124

<b>6</b>	<b>Denotational Semantics of IMP</b>	127
6.1	$\lambda$ -Notation	127
6.1.1	$\lambda$ -Notation: Main Ideas	128
6.1.2	Alpha-Conversion, Beta-Rule and Capture-Avoiding Substitution	131
6.2	Denotational Semantics of IMP	134
6.2.1	Denotational Semantics of Arithmetic Expressions: The Function $\mathcal{A}$	134
6.2.2	Denotational Semantics of Boolean Expressions: The Function $\mathcal{B}$	135
6.2.3	Denotational Semantics of Commands: The Function $\mathcal{C}$	136
6.3	Equivalence Between Operational and Denotational Semantics	141
6.3.1	Equivalence Proofs for Expressions	141
6.3.2	Equivalence Proof for Commands	142
6.4	Computational Induction	149
	Problems	152

### Part III HOFL: a Higher-Order Functional Language

<b>7</b>	<b>Operational Semantics of HOFL</b>	157
7.1	Syntax of HOFL	157
7.1.1	Typed Terms	158
7.1.2	Typability and Typechecking	160
7.2	Operational Semantics of HOFL	164
	Problems	171
<b>8</b>	<b>Domain Theory</b>	175
8.1	The Flat Domain of Integer Numbers $\mathbb{Z}_\perp$	175
8.2	Cartesian Product of Two Domains	176
8.3	Functional Domains	178
8.4	Lifting	181
8.5	Continuity Theorems	183
8.6	Apply, Curry and Fix	186
	Problems	189
<b>9</b>	<b>Denotational Semantics of HOFL</b>	191
9.1	HOFL Semantic Domains	191
9.2	HOFL Interpretation Function	192
9.2.1	Constants	192
9.2.2	Variables	193
9.2.3	Arithmetic Operators	193
9.2.4	Conditional	193
9.2.5	Pairing	194
9.2.6	Projections	194
9.2.7	Lambda Abstraction	195
9.2.8	Function Application	195

9.2.9	Recursion	196
9.2.10	Eager Semantics	196
9.2.11	Examples	197
9.3	Continuity of Meta-language's Functions	198
9.4	Substitution Lemma and Other Properties	200
	Problems	201
<b>10</b>	<b>Equivalence Between HOFL Denotational and Operational Semantics</b>	<b>205</b>
10.1	HOFL: Operational Semantics vs Denotational Semantics	205
10.2	Correctness	206
10.3	Agreement on Convergence	209
10.4	Operational and Denotational Equivalences of Terms	212
10.5	A Simpler Denotational Semantics	213
	Problems	214
 <b>Part IV Concurrent Systems</b>		
<b>11</b>	<b>CCS, the Calculus of Communicating Systems</b>	<b>221</b>
11.1	From Sequential to Concurrent Systems	221
11.2	Syntax of CCS	227
11.3	Operational Semantics of CCS	228
11.3.1	Inactive Process	228
11.3.2	Action Prefix	228
11.3.3	Restriction	229
11.3.4	Relabelling	229
11.3.5	Choice	229
11.3.6	Parallel Composition	230
11.3.7	Recursion	231
11.3.8	CCS with Value Passing	235
11.3.9	Recursive Declarations and the Recursion Operator	235
11.4	Abstract Semantics of CCS	237
11.4.1	Graph Isomorphism	237
11.4.2	Trace Equivalence	239
11.4.3	Strong Bisimilarity	241
11.5	Compositionality	252
11.5.1	Strong Bisimilarity Is a Congruence	253
11.6	A Logical View of Bisimilarity: Hennessy-Milner Logic	255
11.7	Axioms for Strong Bisimilarity	259
11.8	Weak Semantics of CCS	261
11.8.1	Weak Bisimilarity	262
11.8.2	Weak Observational Congruence	264
11.8.3	Dynamic Bisimilarity	265
	Problems	267

<b>12</b>	<b>Temporal Logic and the <math>\mu</math>-Calculus</b>	271
12.1	Specification and Verification	271
12.2	Temporal Logic	272
12.2.1	Linear Temporal Logic	273
12.2.2	Computation Tree Logic	275
12.3	$\mu$ -Calculus	278
12.4	Model Checking	282
	Problems	284
<b>13</b>	<b><math>\pi</math>-Calculus</b>	287
13.1	Name Mobility	287
13.2	Syntax of the $\pi$ -Calculus	291
13.3	Operational Semantics of the $\pi$ -Calculus	292
13.3.1	Inactive Process	293
13.3.2	Action Prefix	293
13.3.3	Name Matching	294
13.3.4	Choice	294
13.3.5	Parallel Composition	295
13.3.6	Restriction	295
13.3.7	Scope Extrusion	296
13.3.8	Replication	296
13.3.9	A Sample Derivation	297
13.4	Structural Equivalence in the $\pi$ -Calculus	297
13.4.1	Reduction Semantics	298
13.5	Abstract Semantics of the $\pi$ -Calculus	299
13.5.1	Strong Early Ground Bisimulations	300
13.5.2	Strong Late Ground Bisimulations	301
13.5.3	Compositionality and Strong Full Bisimilarities	302
13.5.4	Weak Early and Late Ground Bisimulations	303
	Problems	304

## Part V Probabilistic Systems

<b>14</b>	<b>Measure Theory and Markov Chains</b>	309
14.1	Probabilistic and Stochastic Systems	309
14.2	Probability Space	310
14.2.1	Constructing a $\sigma$ -Field	311
14.3	Continuous Random Variables	313
14.3.1	Stochastic Processes	318
14.4	Markov Chains	319
14.4.1	Discrete and Continuous Time Markov Chains	320
14.4.2	DTMCs as LTSs	320
14.4.3	DTMC Steady State Distribution	323
14.4.4	CTMCs as LTSs	324
14.4.5	Embedded DTMC of a CTMC	325

14.4.6 CTMC Bisimilarity .....	326
14.4.7 DTMC Bisimilarity .....	328
Problems .....	329
<b>15 Discrete Time Markov Chains with Actions and Nondeterminism . . .</b>	<b>333</b>
15.1 Reactive and Generative Models .....	333
15.2 Reactive DTMC .....	334
15.2.1 Larsen-Skou Logic .....	336
15.3 DTMC with Nondeterminism .....	337
15.3.1 Segala Automata .....	337
15.3.2 Simple Segala Automata .....	338
15.3.3 Nondeterminism, Probability and Actions .....	339
Problems .....	340
<b>16 PEPA - Performance Evaluation Process Algebra .....</b>	<b>343</b>
16.1 From Qualitative to Quantitative Analysis .....	343
16.2 CSP .....	344
16.2.1 Syntax of CSP .....	344
16.2.2 Operational Semantics of CSP .....	345
16.3 PEPA .....	347
16.3.1 Syntax of PEPA .....	347
16.3.2 Operational Semantics of PEPA .....	349
Problems .....	354
<b>Solutions .....</b>	<b>357</b>



# 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)
$\sqsupseteq$	CCS strong bisimilarity (see Definition 11.5)
$\approx$	CCS weak bisimilarity (see Definition 11.16)
$\approx\!\!\approx$	CCS weak observational congruence (see Section 11.8.2)
$\approx\!\!\approx\!\!\approx$	CCS dynamic bisimilarity (see Definition 11.18)
$\sim_E$	$\pi$ -calculus early bisimilarity (see Definition 13.4)
$\sim_L$	$\pi$ -calculus late bisimilarity (see Definition 13.4)
$\simeq_E$	$\pi$ -calculus strong early full bisimilarity (see Section 13.5.3)
$\simeq_L$	$\pi$ -calculus strong late full bisimilarity (see Section 13.5.3)
$\approx_E$	$\pi$ -calculus weak early bisimilarity (see Section 13.5.4)
$\approx_L$	$\pi$ -calculus weak late bisimilarity (see Section 13.5.4)
$\mathcal{A}$	interpretation function for the denotational semantics of IMP arithmetic expressions (see Section 6.2.1)
<i>ack</i>	Ackermann function (see Example 4.18)
<i>Aexp</i>	set of IMP arithmetic expressions (see Chapter 3)
$\mathcal{B}$	interpretation function for the denotational semantics of IMP boolean expressions (see Section 6.2.2)
<i>Bexp</i>	set of IMP boolean expressions (see Chapter 3)
$\mathbb{B}$	set of booleans
$\mathcal{C}$	interpretation function for the denotational semantics of IMP commands (see Section 6.2.3)
CCS	Calculus of Communicating Systems (see Chapter 11)
<i>Com</i>	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)

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