

Cambridge University Press
0521543274 - The Pi-Calculus: A Theory of Mobile Processes
Davide Sangiorgi and David Walker
Frontmatter
[More information](#)

The π -calculus: a Theory of Mobile Processes

Davide Sangiorgi
INRIA Sophia Antipolis

David Walker
University of Oxford



Cambridge University Press
0521543274 - The Pi-Calculus: A Theory of Mobile Processes
Davide Sangiorgi and David Walker
Frontmatter
[More information](#)

PUBLISHED BY THE PRESS SYNDICATE OF THE UNIVERSITY OF CAMBRIDGE
The Pitt Building, Trumpington Street, Cambridge, United Kingdom

CAMBRIDGE UNIVERSITY PRESS
The Edinburgh Building, Cambridge CB2 2RU, UK
40 West 20th Street, New York NY 10011-4211, USA
477 Williamstown Road, Port Melbourne, VIC 3207, Australia
Ruiz de Alarcón 13, 28014 Madrid, Spain
Dock House, The Waterfront, Cape Town 8001, South Africa

<http://www.cambridge.org>

© Cambridge University Press 2001

This book is in copyright. Subject to statutory exception
and to the provisions of relevant collective licensing agreements,
no reproduction of any part may take place without
the written permission of Cambridge University Press.

First published 2001
First paperback edition 2003

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

ISBN 0 521 78177 9 hardback
ISBN 0521 54327 4 paperback

Cambridge University Press
0521543274 - The Pi-Calculus: A Theory of Mobile Processes
Davide Sangiorgi and David Walker
Frontmatter
[More information](#)

*Quelli che s'innamoran di pratica senza scientia
son come 'l nocchiere ch'entra in navilio senza timone o bussola,
che mai ha certezza dove si vada.*

– Leonardo da Vinci

Contents

<i>Foreword</i>	ix
<i>Preface</i>	xi
<i>General Introduction</i>	1
Part I: The π-calculus	5
<i>Introduction to Part I</i>	7
1 Processes	11
1.1 Syntax	11
1.2 Reduction	17
1.3 Action	36
1.4 Basic properties of the transition system	44
2 Behavioural Equivalence	54
2.1 Strong barbed congruence	54
2.2 Strong bisimilarity	64
2.3 Up-to techniques	80
2.4 Barbed congruence	92
<i>Notes and References for Part I</i>	118
Part II: Variations of the π-calculus	121
<i>Introduction to Part II</i>	123
3 Polyadicity and Recursion	127
3.1 Polyadicity	127
3.2 Recursion	132

Contents

v

3.3	Priority-queue data structures	138
3.4	Data as processes	145
4	Behavioural Equivalence, continued	154
4.1	Distinctions	154
4.2	Variants of bisimilarity	157
4.3	The late transition relations	158
4.4	Ground bisimilarity	162
4.5	Late bisimilarity	164
4.6	Open bisimilarity	166
4.7	The weak equivalences	172
4.8	Axiomatizations and proof systems	174
5	Subcalculi	189
5.1	The Asynchronous π -calculus	189
5.2	Syntax of $A\pi$	190
5.3	Behavioural equivalence in $A\pi$	194
5.4	Asynchronous equivalences	198
5.5	Expressiveness of asynchronous calculi	203
5.6	The Localized π -calculus	211
5.7	Internal mobility	215
5.8	Non-congruence results for ground bisimilarity	223
	<i>Notes and References for Part II</i>	227
	Part III: Typed π-calculi	231
	<i>Introduction to Part III</i>	233
6	Foundations	236
6.1	Terminology and notation for typed calculi	236
6.2	Base- π	238
6.3	Properties of typing	244
6.4	The simply-typed π -calculus	247
6.5	Products, unions, records, and variants	249
6.6	Pattern matching in input	255
6.7	Recursive types	257
7	Subtyping	260
7.1	i/o types	261
7.2	Properties of the type systems with i/o	265
7.3	Other subtyping	270

vi	<i>Contents</i>	
7.4	The priority queues, revisited.....	272
7.5	Encodings between union and product types.....	276
8	Advanced Type Systems	281
8.1	Linearity.....	281
8.2	Receptiveness.....	288
8.3	Polymorphism.....	296
	<i>Notes and References for Part III</i>	305
	Part IV: Reasoning about Processes using Types	309
	<i>Introduction to Part IV</i>	311
9	Groundwork	313
9.1	Using types to obtain encapsulation.....	313
9.2	Why types for reasoning?.....	316
9.3	A security property.....	317
9.4	Typed behavioural equivalences.....	319
9.5	Equivalences and preorders in simply-typed π -calculi.....	328
10	Behavioural Effects of i/o Types	329
10.1	Type coercion.....	329
10.2	Examples.....	330
10.3	Wires in the Asynchronous π -calculus.....	335
10.4	Delayed input.....	336
10.5	Sharpened Replication Theorems.....	338
10.6	Proof techniques.....	340
10.7	Context Lemma.....	342
10.8	Adding internal mobility.....	348
11	Techniques for Advanced Type Systems	351
11.1	Some properties of linearity.....	351
11.2	Behavioural properties of receptiveness.....	352
11.3	A proof technique for polymorphic types.....	359
	<i>Notes and References for Part IV</i>	365
	Part V: The Higher-Order Paradigm	367
	<i>Introduction to Part V</i>	369
12	Higher-Order π-calculus	373
12.1	Simply-typed HO π	373

12.2	Other $\text{HO}\pi$ languages	381
13	Comparing First-Order and Higher-Order Calculi	383
13.1	Compiling higher order into first order	383
13.2	Optimizations	397
13.3	Reversing the compilation	408
13.4	Full abstraction	412
	<i>Notes and References for Part V</i>	415
	Part VI: Functions as Processes	419
	<i>Introduction to Part VI</i>	421
14	The λ-calculus	424
14.1	The formal system	424
14.2	Contrasting λ and π	426
14.3	Reduction strategies: call-by-name, call-by-value, call-by-need	429
15	Interpreting λ-calculi	434
15.1	Continuation Passing Style	434
15.2	Notations and terminology for functions as processes	436
15.3	The interpretation of call-by-value	438
15.4	The interpretation of call-by-name	452
15.5	A uniform encoding	461
15.6	Optimizations of the call-by-name encoding	464
15.7	The interpretation of strong call-by-name	465
16	Interpreting Typed λ-calculi	469
16.1	Typed λ -calculus	469
16.2	The interpretation of typed call-by-value	470
16.3	The interpretation of typed call-by-name	474
17	Full Abstraction	477
17.1	The full-abstraction problem	477
17.2	Applicative bisimilarity	478
17.3	Soundness and non-completeness	479
17.4	Extending the λ -calculus	483
18	The Local Structure of the Interpretations	492
18.1	Sensible theories and lazy theories	492
18.2	Lévy–Longo Trees	493
18.3	The Local Structure Theorem for call-by-name	496

viii	<i>Contents</i>	
18.4	Böhm Trees	505
18.5	Local structure of the call-by-value interpretation	505
	<i>Notes and References for Part VI</i>	507
	Part VII: Objects and π-calculus	513
	<i>Introduction to Part VII</i>	515
19	Semantic Definition	517
19.1	A programming language	517
19.2	Modelling examples	522
19.3	Formal definition	528
20	Applications	533
20.1	Some properties of declarations and commands	533
20.2	Proxies	535
20.3	An implementation technique	539
20.4	A program transformation	541
	<i>Notes and References for Part VII</i>	546
	<i>List of Tables</i>	548
	<i>List of Notations</i>	550
	<i>Bibliography</i>	562
	<i>Index</i>	576

Foreword

Computer science aims to explain the way computational systems behave for us. The notion of calculational process, or algorithm, is a lot older than computing technology; so, oddly enough, a lot of computer science existed before modern computers. But the invention of real stored-program computers presented enormous challenges; these tools can do a lot for us if we describe properly what we want done. So computer science has made immense strides in ways of *presenting* data and algorithms, in ways of manipulating these presentations themselves as data, in matching algorithm description to task description, and so on. Technology has been the catalyst in the growth of modern computer science.

The first large phase of this growth was in free-standing computer systems. Such a system might have been a single computer program, or a multi-computer serving a community by executing several single programs successively or simultaneously. Computing theorists have built many mathematical models of these systems, in relation to their purposes. One very basic such model – the λ -calculus – is remarkably useful in this role, even if it was designed by Alonzo Church around 1940.

The second phase of the growth of computer science is in response to the advent of computer networks. No longer are systems freestanding; they interact, collaborate and interrupt each other. This has an enormous effect on the way we think about our systems. We can no longer get away with considering each system as sequential, goal-directed, deterministic or hierarchical; networks are none of these. So if we confine ourselves to such concepts then we remain dumb if asked to predict whether a network will behave in a proper or an improper – way; for example, whether someone logging in to his bank ~~bank~~ (as happened recently) find himself scanning someone else's account instead of his own.

The present book is a rigorous account of a basic calculus which aims to underpin our theories of interactive systems, in the same way that the λ -calculus did for freestanding computation. The authors are two of the original researchers

Cambridge University Press

0521543274 - The Pi-Calculus: A Theory of Mobile Processes

Davide Sangiorgi and David Walker

Frontmatter

[More information](#)

x

Foreword

on the π -calculus, which is now over ten years old and has served as a focus for much theoretical and practical experiment. It cannot claim to be definitive; in fact, since it was designed it has become common to express ideas about interaction and mobility in variants of the calculus. So it has become a kind of workshop of ideas.

That's the spirit in which the book is written. Half the book analyses the constructions of the calculus, searching out its meaning and exploring its expressivity by looking at weaker variants, or by looking at various type disciplines. Enthusiasts about types in programming will be struck to find that π -calculus types don't just classify *values*; they classify *patterns of behaviour*. This reflects the fact that what matters most in mobile interactive systems is not values, but connectivity and mobility of processes. With or without types, the unifying feature is *behaviour*, and what it means to say that two different processes behave the same.

The later part of the book deals with two generic applications. One of these is classical; how the π -calculus can actually do the old job which the λ -calculus does in underpinning conventional programming. The other is modern; how the calculus informs one of the most important models of interaction, the *object-oriented* model. These applications bring together much of the theory developed earlier; together, they show that a small set of constructs, provided that they emphasize *inter action* rather than calculation, can still bring some conceptual unity to the greatly extended scope of modern computing.

This book has been a labour of love for the authors over several years. Their scholarship is immense, and their organisation of ideas meticulous. As one privileged to have worked closely with them both, it's a great pleasure to be able to recommend the result as a storehouse of ideas and techniques which is unlikely to be equalled in the next decade or two.

Robin Milner
Cambridge
February 2001

Preface

Mobile systems, whose components communicate and change their structure, now pervade the informational world and the wider world of which it is a part. But the science of mobile systems is yet immature. This science must be developed if we are properly to understand mobile systems, and if we are to design systems so that they do what they are intended to do. This book presents the π -calculus, a theory of mobile systems, and shows how to use it to express systems precisely and reason about their behaviour rigorously.

The book is intended to serve both as a reference for the theory and as an extended demonstration of how to use the π -calculus to express systems and analyse their properties. The book therefore presents the theory in detail, with emphasis on proof techniques. How to use the techniques is shown both in proofs of results that form part of the theory and in example applications of it.

The book is in seven parts. Part I introduces the π -calculus and develops its basic theory. Part II presents variations of the basic theory and important subcalculi of the π -calculus. A distinctive feature of the calculus is its rich theory of types for mobile systems. Part III introduces this theory, and Part IV shows how it is useful for understanding and reasoning about systems. Part V examines the relationship between the π -calculus and higher-order process calculi. Part VI analyses the relationship between the π -calculus and the λ -calculus. Part VII shows how ideas from π -calculus can be useful in object-oriented design and programming.

The book is written at the graduate level and is intended for computer scientists interested in mobile systems. It assumes no prior acquaintance with the π -calculus: both the theory and the viewpoint that underlies it are explained from the beginning.

Although the book covers quite a lot of ground, several topics, notably logics for mobility, and denotational and non-interleaving semantics, are not treated at all. The book contains detailed accounts of a selection of topics, chosen for

Cambridge University Press

0521543274 - The Pi-Calculus: A Theory of Mobile Processes

Davide Sangiorgi and David Walker

Frontmatter

[More information](#)

xii

Preface

their interest and because they allow us to explore concepts and techniques that can also be used elsewhere. Each Part ends with some references to sources and additional notes on related topics. We have not attempted the arduous task of referring to all relevant published work. The references given provide starting points for a reader who wishes to go more deeply into particular topics. Sometimes, an element of arbitrariness in the choice of references was inevitable.

Many exercises are suggested to help appreciation of the material; the more difficult of them are marked with an asterisk. We intend to maintain a Web page for general information and auxiliary material about the book. At the time of writing, this page is located at

[http://www-sop.inria.fr/mimosa/personnel/Davide.Sangiorgi/
Book_pi.html](http://www-sop.inria.fr/mimosa/personnel/Davide.Sangiorgi/Book_pi.html)

Acknowledgements Our greatest debt is to Robin Milner. The field that is the subject of this book was shaped by his fundamental work on CCS and in creating and developing the π -calculus. Further, we have both been privileged to have worked with Milner, and his influence on our approach to the subject and how to write about it are profound.

We thank the many colleagues – too many to mention here – with whom we have worked on or discussed π -calculus and related topics, and whose insights and comments have contributed to our understanding.

We are grateful to the following people for reading parts of a draft of the book and offering comments that helped us improve it: Michael Baldamus, Silvia Crafa, Cédric Fournet, Daniel Hirshkoff, Kohei Honda, Naoki Kobayashi, Giovanni Lagorio, Cédric Lhoussaine, Huimin Lin, Barbara König, Robin Milner, Julian Rathke, Vasco Vasconcelos, Nobuko Yoshida, and especially Marco Pistore.

We record our appreciation of the work of David Tranah and his colleagues at Cambridge University Press in guiding the book into print.

Finally, we thank Laurence Sangiorgi and Katharine Grevling for their encouragement, assistance, and patience during the seemingly interminable process of writing.