Texts in Theoretical Computer Science
An EATCS Series

Springer-Verlag Berlin Heidelberg GmbH

Wan Fokkink

Introduction to Process Algebra

With 11 Figures and 11 Tables

Springer

Author

Dr. Wan Fokkink

Centrum voor Wiskunde en Informatica (CWI)
P.O. Box 94079
1090 GB Amsterdam, The Netherlands

wan@cwi.nl
http://www.cwi.nl/~wan

Library of Congress Cataloging-in-Publication Data applied for

Die Deutsche Bibliothek - CIP-Einheitsaufnahme

Fokkink, Wan:
Introduction to process algebra/Wan Fokkink. - Berlin; Heidelberg;
New York; Barcelona; Hong Kong; London; Milan; Paris;
Singapore; Tokyo: Springer 2000
(Texts in theoretical computer science)

ACM Computing Classification (1998): D.2.4, D.3.1–3, F.3.2, F.1.2, F.4.2

ISBN 978-3-642-08584-0 ISBN 978-3-662-04293-9 (eBook)
DOI 10.1007/978-3-662-04293-9

Cover Design: Künkel + Lopka, Werbeagentur, Heidelberg
Typesetting: Camera ready by the author
SPIN 10747036 45/3142/SR - 5 4 3 2 1 0 - Printed on acid-free paper

Preface

Computer software and network protocols are increasingly important in daily life. At the same time the complexity of software has rocketed, so that its correctness is at stake. New methodologies and disciplines are being developed to bring structure to the ever growing jungle of computer technology. (Semi-)automated manipulation has become an important means in discovering flaws in software and hardware systems. Process algebra is a mathematical framework in which system behaviour is expressed in the form of algebraic terms, enhancing the available techniques for manipulation.

Concurrency is omnipresent in system behaviour, and in a large part responsible for its complexity: even simple behaviours become wildly complicated when they are executed in parallel. In order to study such systems in detail, it is imperative that they are dissected into their concurrent components. Fundamental to process algebra is a parallel operator, to break down systems into their concurrent components. A set of equations is imposed to derive whether two terms are behaviourally equivalent. In this framework, non-trivial properties of systems can be established in an elegant fashion. For example, it may be possible to equate an implementation to the specification of its required input/output relation. In recent years a variety of automated tools have been developed to facilitate the derivation of such properties.

Applications of process algebra exist in diverse fields such as safety critical systems, network protocols, and biology. In the educational vein, process algebra has been recognised to teach skills to deal with complex concurrent systems, by representing and reasoning about such systems in a mathematically clear and precise manner.

This text developed from an undergraduate course on process algebra at the computer science department of the University of Wales Swansea during the autumn of 1997 and of 1998. Chapters 2-7 contain sufficient material for more than twenty hours of lecturing; a set of slides and further material to support such a course are available from my homepage (currently at http://www.cwi.nl/∼wan). It is recommended to use a tool set based on process algebra, such as the μCRL tool set or the Concurrency Workbench Edinburgh, to enliven the course. Appendices A and B provide useful background information; they are not intended to be included in the course.

I am grateful to John Tucker for his encouragement to further develop a raw set of lecture notes, and to Judi Romijn for her support. Over the years I have benefited from discussions with Jan Bergstra, Rob van Glabbeek, Jan Friso Groote, Frits Vaandrager, Alban Ponse, Chris Verhoef, Jaco van de Pol, Jos Baeten, Luca Aceto, Jos van Wamel, Steven Klusener, Bas Luttik, Dennis Dams, and many others.

Amsterdam, November 1999 *Wan Fokkink*

Contents

1. Introduction 1

2. Basic Process Algebra 7
2.1 Basic Process Terms 7
2.2 Transition Rules for BPA 8
2.3 Bisimulation Equivalence 10
2.4 Axioms for BPA 12

3. Algebra of Communicating Processes 17
3.1 Parallelism and Communication 17
3.2 Left Merge and Communication Merge 19
3.3 Axioms for PAP 20
3.4 Deadlock and Encapsulation 25

4. Recursion 31
4.1 Guarded Recursive Specifications 31
4.2 Transition Rules for Guarded Recursion 33
4.3 Recursive Definition and Specification Principles 36
4.4 Completeness for Regular Processes 39
4.5 Approximation Induction Principle 42

5. Abstraction 47
5.1 Rooted Branching Bisimulation Equivalence 47
5.2 Guarded Linear Recursion Revisited 51
5.3 Axioms for the Silent Step 53
5.4 Abstraction Operators 57
5.5 An Example with Queues 60
5.6 Cluster Fair Abstraction Rule 63

6. Protocol Verifications 69
6.1 Alternating Bit Protocol 69
6.2 Bounded Retransmission Protocol 78
6.3 Specification and Verification Techniques 88
6.4 Tools 92

7. Extensions 97
7.1 Renaming 97
7.2 State Operator 99
7.3 Priorities 105

A. Equational Logic 109
A.1 Signatures 109
A.2 Axiomatisations 110
A.3 Initial Models 111
A.4 Term Rewriting 113

B. Structural Operational Semantics 119
B.1 Transition System Specifications 119
B.2 The Meaning of Negative Premises 121
B.3 Bisimulation as a Congruence 125
B.4 Branching Bisimulation as a Congruence 128
B.5 Conservative Extension 131
B.6 Modal Logics 133

Solutions to Selected Exercises 137

References 151

Index 161