APPLICATIONS OF PROCESS ALGEBRA

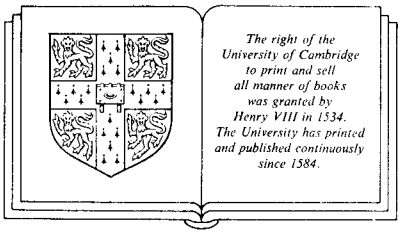
Cambridge Tracts in Theoretical Computer Science

Titles in the series

1. G. Chaitin *Algorithmic Information Theory*
2. L.C. Paulson *Logic and Computation*
3. M. Spivey *Understanding Z*
4. G. Revesz *Lambda Calculus, Combinators and Logic Programming*
5. S. Vickers *Topology via Logic*
6. A. Ramsay *Formal Methods in Artificial Intelligence*
7. J-Y. Girard, Y. Lafont & P. Taylor *Proofs and Types*
8. J. Clifford *Formal Semantics & Pragmatics for Natural Language Processing*
9. M. Winslett *Updating Logical Databases*
10. K. McEvoy & J.V. Tucker (eds) *Theoretical Foundations of VLSI Design*
12. G. Brewka *Nonmonotonic Reasoning*
13. G. Smolka *Logic Programming over Polymorphically Order-Sorted Types*

APPLICATIONS OF PROCESS ALGEBRA

Edited by
J. C. M. BAETEN
CWI, Amsterdam



CAMBRIDGE UNIVERSITY PRESS

Cambridge

New York *Port Chester* *Melbourne* *Sydney*

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

First published 1990
First paperback edition 2004

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

ISBN 0 521 40028 7 hardback
ISBN 0 521 60750 7 paperback

Preface

In this book, we give applications of the theory of process algebra, known by the acronym ACP (Algebra of Communicating Processes), as it has been developed since 1982 at the Centre for Mathematics and Computer Science, Amsterdam (see [7]), since 1985 in cooperation with the University of Amsterdam and the University of Utrecht. An important stimulus for this book was given by the ESPRIT contract no. 432, An Integrated Formal Approach to Industrial Software Development (Meteor). The theory itself is treated in [3], which will be revised, translated and published in this series. The theory is briefly reviewed in the first article in this book, *An introduction to process algebra,* by J.A. Bergstra and J.W. Klop.

This book gives applications of the theory of process algebra. By the term process algebra we mean the study of concurrent or communicating processes in an algebraic framework. We endeavour to treat communicating processes in an axiomatic way, just as for instance the study of mathematical objects as groups or fields starts with an axiomatization of the intended objects. The axiomatic method which will concern us, is algebraic in the sense that we consider structures which are models of some set of (mostly) equational axioms; these structures are equipped with several operators. Thus we use the term 'algebra' in the sense of model theory.

There is ample motivation for such an axiomatic-algebraic approach to the theory of communicating processes. An important reason is that there is not one definite notion of process. There is a staggering amount of properties which one may or may not attribute to processes, there are dozens of views ('semantics') which one may have on processes, and there are infinitely many models of processes. So an attempt to organize this field of process theories leads very naturally to an axiomatic methodology.

The aspect of the axiomatic-algebraic approach that is most prominent in

this book, is the obvious computational aspect. Much more than in mathematics or mathematical logic, in computer science it is 'algebra' that counts - the well-known etymology of the word 'algebra' should be convincing enough. In system verification, the use of transition diagrams is very illuminating, but especially for larger systems it is desirable to have a formalized mathematical language at our disposal in which specifications, computations, proofs can be given in what is in principle a linear notation. Only then can we expect something of attempts to mechanize our dealings with the objects of interest. In our case the mathematical language is algebraic, with basic constants, operators to construct larger processes and equations defining the nature of the processes under consideration. (The format of pure equations will not be enough, though. We also will use conditional equations and some infinitary proof-rules.)

Of course, the present axiomatizations for communicating processes do not cover the entire spectrum of interest. Several aspects of processes are as yet not well treated in the algebraic framework. The most notable examples are real-time behaviour of processes, and what sometimes is called 'true concurrency' (non-interleaving semantics).

The first system verifications that were done using this theory were [9] and [8]. The verification of the Alternating Bit Protocol of the first article is (in a revised version) part of the second article of this book, *Two simple protocols,* by F.W. Vaandrager. The second article, dealing with FIFO queues, contains problems, that are at this moment not yet well understood. This is the reason that it is not included in this book. From these articles, it became clear that more structuring mechanisms, more operators, were needed to deal with the verification of larger systems. The article [4] was written in order to provide more structuring power. Most of the operators introduced were then used in [28]. Parts of this article can be found in this book: The verification of the Positive Acknowledgement with Retransmission protocol is the second protocol discussed in *Two simple protocols,* and the modularization tool of redundancy in context is discussed (in an extensively revised form) in *Some observations on redundancy in a context,* the tenth article in this book. Other parts of [28], most notably the verification of a One Bit Sliding Window protocol, will appear elsewhere. In the meantime, another approach to modularization was pursued in [18]. The last article in this book is a revision of this paper. Later, more complicated protocol verifications were attempted. [12] gives a verification of a Sliding Window protocol, that takes over 100 pages. Further investigation is necessary, in order to incorporate Hoare-style proof-rules into the theory.

This book presents examples of algorithms other than communication protocols. The third article, a revision of [24], considers Peterson's protocol for ensuring mutual exclusion. Some properties are shown to hold for the process algebra specification. Interesting is the idea of considering global variables as processes communicating with all components in a system. Assigning a value to a variable is done by sending it the value, testing for an equality by reading

its value. In the following article, a revision of [20], an example is given of an automated plant (a CIM-architecture), and it is verified that the design goals were met. Next, [6] introduces the concept of process creation, that is used in the following articles. An example is given in the specification of the Sieve of Eratosthenes, an algorithm to generate prime numbers. Recently, in [29], we find a verification of this specification.

An application of the process creation operator is given in the following article, a revision of [16]. Here, a verification is given of two so-called systolic algorithms, an algorithm for recognizing palindromes, and an algorithm for sorting sequences of numbers. Systolic systems consist of regular configurations of simple components, which make them very suitable for chip design. The next article, a revision of [22], also considers a systolic algorithm. Two other articles about such algorithms in the present theory are [17] and [30].

The following article is the recent [21], that considers the distributed operating system Amoeba. It is found that there is a mistake in the description of this system in [23], and it is indicated how this mistake can be corrected. The corrected version is verified. Finally, there is an article in this book about the object-oriented programming language POOL, which is a revision of [27]. A translation is given to the language of ACP, which gives a number of different semantics for POOL. A direction that is not represented in this book is the investigation of circuit architecture, see [5].

Of course, we do not mean to imply that the verification techniques presented in this book constitute the only approach to these problems. In fact, there are many other approaches. From some we have benefited, others have inspired us. To mention just a few verifications in other theories, there are [19] and [25] in the context of CCS, [14] by Hennessy, [26] and [15] in the context of trace theory, [10] in the context of LOTOS, [11] in the context of ASL, [13] using knowledge-based reasoning, and [1,2] using denotational and operational semantics. However, I think this is the first time a collection of articles in this area is collected together, where a single approach is followed in all articles.

Finally, I would like to thank all authors, and all other participants of the PAM seminar, for their cooperation. Also, my thanks goes to the typing staff of the Centre for Mathematics and Computer Science and the desk editor W.A.M. Aspers.

J.C.M. BAETEN

REFERENCES

1. P. AMERICA, J.W. DE BAKKER, J.N. KOK, J.J.M.M. RUTTEN (1986). Operational semantics of a parallel object-oriented language. *Conference Record of the 13th ACM Symposium of Principles of Programming Languages,* St. Petersburg, Florida, 194-208.
2. P. AMERICA, J.W. DE BAKKER, J.N. KOK, J.J.M.M. RUTTEN (1986). *A Denotational Semantics of a Parallel Object-oriented Language,* CWI Report

CS-R8626, Centre for Mathematics and Computer Science, Amsterdam. To appear in *Information and Computation.*

3. J.C.M. Baeten (1986). *Procesalgebra,* Kluwer Programmatuurkunde, Deventer (in Dutch).
4. J.C.M. Baeten, J.A. Bergstra (1988). Global renaming operators in concrete process algebra. *Information and Computation 78(3),* 205-245.
5. J.C.M. Baeten, F.W. Vaandrager (1988). *Specification and Verification of a Circuit in ACP,* Report P8803, Programming Research Group, University of Amsterdam.
6. J.A. Bergstra (1985). *A Process Creation Mechanism in Process Algebra,* Logic Group Preprint Series Nr. 2, CIF, State University of Utrecht.
7. J.A. Bergstra, J.W. Klop (1982). *Fixed Point Semantics in Process Algebras,* MC Report IW 206, Centre for Mathematics and Computer Science, Amsterdam.
8. J.A. Bergstra, J.W. Klop (1984). *Fair FIFO Queues Satisfy an Algebraic Criterion for Protocol Correctness,* CWI Report CS-R8405, Centre for Mathematics and Computer Science, Amsterdam.
9. J.A. Bergstra, J.W. Klop (1986). Verification of an Alternating Bit Protocol by means of process algebra. W. Bibel, K.P. Jantke (eds.). *Math. Methods of Spec. and Synthesis of Software Systems '85, Math. Research 31,* Akademie-Verlag, Berlin, 9-23. Also appeared as CWI Report CS-R8404, Centre for Mathematics and Computer Science, Amsterdam, 1984.
10. F. Biemans, P. Blonk (1986). On the formal specification and verification of CIM architectures using LOTOS. *Computers in Industry 7(6),* 491-504.
11. M. Broy (1987). *Views of Queues,* Report MIP-8704, Fakultät für Mathematik und Informatik, Universität Passau.
12. R.A. Groenveld (1987). *Verification of a Sliding Window Protocol by Means of Process Algebra,* Report P8701, Programming Research Group, University of Amsterdam.
13. J.Y. Halpern, L.D. Zuck (1987). *A Little Knowledge Goes a Long Way: Simple Knowledge-based Derivations and Correctness Proofs for a Family of Protocols* (extended abstract), IBM Almaden Research Center.
14. M. Hennessy (1986). Proving systolic systems correct. *TOPLAS 8(3),* 344-387.
15. A. Kaldewaij (1987). The translation of processes into circuits. J.W. de Bakker, A.J. Nijman, P.C. Treleaven (eds.). *Proceedings PARLE Conference, Eindhoven, Volume I (Parallel Architectures),* LNCS 258, Springer-Verlag, 195-212.
16. L. Kossen, W.P. Weijland (1987). *Correctness Proofs for Systolic Algorithms: Palindromes and Sorting,* Report FVI 87-04, Computer Science Department, University of Amsterdam.
17. L. Kossen, W.P. Weijland (1987). *Verification of a Systolic Algorithm for String Comparison,* CWI Report CS-R8734, Centre for Mathematics and Computer Science, Amsterdam.

18. C.P.J. KOYMANS, J.C. MULDER (1989). *A Modular Approach to Protocol Verification using Process Algebra.* This volume.
19. K.G. LARSEN, R. MILNER (1987). A complete protocol verification using relativized bisimulation. TH. OTTMANN (ed.). *Proceedings 14th ICALP, Karlsruhe,* LNCS 267, Springer-Verlag, 126-135.
20. S. MAUW (1987). *Process Algebra as a Tool for the Specification and Verification of CIM-Architectures,* Report P8708, Programming Research Group, University of Amsterdam.
21. J.C. MULDER (1988). *On the Amoeba Protocol,* CWI Report CS-R8827, Centre for Mathematics and Computer Science, Amsterdam.
22. J.C. MULDER, W.P. WEIJLAND (1987). *Verification of an Algorithm for Log-time Sorting by Square Comparison,* CWI Report CS-R8729, Centre for Mathematics and Computer Science, Amsterdam.
23. S.J. MULLENDER (1985). *Principles of Distributed Operating System Design,* Ph.D. Thesis, Free University, Amsterdam.
24. E.R. NIEUWLAND (1987). *Proving Mutual Exclusion with Process Algebra,* Report FVI 87-10, Department of Computer Science, University of Amsterdam.
25. J. PARROW (1985). *Fairness Properties in Process Algebra - with Applications in Communication Protocol Verification,* DoCS 85/03, Ph.D. Thesis, Department of Computer Systems, Uppsala University.
26. M. REM (1987). Trace theory and systolic computations. J.W. DE BAKKER, A.J. NIJMAN, P.C. TRELEAVEN (eds.). *Proceedings PARLE Conference, Eindhoven, Volume I (Parallel Architectures),* LNCS 258, Springer-Verlag, 14-33.
27. F.W. VAANDRAGER (1986). *Process Algebra Semantics of POOL,* CWI Report CS-R8629, Centre for Mathematics and Computer Science, Amsterdam.
28. F.W. VAANDRAGER (1986). *Verification of Two Communication Protocols by means of Process Algebra,* CWI Report CS-R8608, Centre for Mathematics and Computer Science, Amsterdam.
29. J.L.M. VRANCKEN (1988). *The Implementation of Process Algebra Specifications in POOL-T,* Report P8807, Programming Research Group, University of Amsterdam.
30. W.P. WEIJLAND (1987). A systolic algorithm for matrix-vector multiplication. *Proceedings SION Conference CSN 87,* Centre for Mathematics and Computer Science, Amsterdam, 143-160.

Contents