Reactive Systems

A reactive system comprises networks of computing components, achieving their goals through interaction among themselves and their environment. Thus even relatively small systems may exhibit unexpectedly complex behaviours. As, moreover, reactive systems are often used in safety critical systems, the need for mathematically based formal methodology is increasingly important. There are many books that look at particular methodologies for such systems. This book offers a more balanced introduction for graduate students and describes the various approaches, their strengths and weaknesses and when they are best used. Milner's CCS and its operational semantics are introduced, together with the notions of behavioural equivalences based on bisimulation techniques and with recursive extensions of Hennessy-Milner logic. In the second part of the book the presented theories are extended to take timing issues into account. The book has arisen from various courses taught in Denmark and Iceland and is designed to give students a broad introduction to the area, with exercises throughout.

LUCA ACETO is Professor of Computer Science at Reykjavík University, Iceland, and Aalborg University, Denmark.

ANNA INGÓLFSDÓTTIR is Professor of Computer Science at Reykjavík University, Iceland, and Aalborg University, Denmark.

KIM G. LARSEN is Professor of Computer Science at Aalborg University, Denmark, and Twente University, The Netherlands.

JIŘÍ SRBA is Associate Professor in Computer Science at Aalborg University, Denmark.

'Many modern-day computing systems are reactive in nature; they persist indefinitely, responding to the interactions of users, and updating their internal structures accordingly. Over the last two decades, an elegant theory of these reactive systems has emerged, and is being increasingly applied in industrial settings.

And at last we have an accessible textbook for this area, written by a team who have played a central role in the development of the underlying theory, and the software tools which are essential to its successful application. It treats both timed and untimed systems and, although the underlying theory is carefully and methodically explained, the main trust of the book is to engage students with the material via a wealth of thought-provoking examples.

The clarity of the exposition is exceptional; it presents the essential ideas clearly, avoiding unnecessary detail, but at the same time has well-chosen pointers to more advanced concepts. The book is destined to become the standard textbook for reactive systems.'

Matthew Hennessy, Sussex University

'A must for anybody interested in formal analysis techniques for computing systems.'

Wan Fokkink, Vrije Universiteit Amsterdam

'This book is a gentle introduction to the basics of theories of interactive systems that starts with an introduction to CCS and its semantic theory and then moves to introducing modal logics and timed models of concurrency. By means of a number of small but intriguing examples and by using software tools based on sound theoretical principles, it leads the reader to appreciating and mastering a number of process algebra-based techniques that are also having a great impact outside academic circles.

The authors have managed to concentrate their expertise, enthusiasm and pedagogical ability in less than 300 pages. The presentation is very clear and conveys sufficient intuition to make the book appropriate also for students with limited mathematical background. An excellent advanced undergraduate text.'

Rocco De Nicola, Universitá di Firenze

'This book offers an introduction to model-based verification of reactive systems, a technology that is essential to all IT-developers of the future, given the global trend in information technology towards ubiquitous computing.

The book is unique in its pedagogical style, introducing the required theory (of models and specification formalisms for reactive systems) motivated carefully with its applications (in the development and use of automated verification tools in practice), and written as a textbook that can be used readily at many different levels of IT-related curricula.'

Mogens Nielsen, Aarhus University

Reactive Systems

Modelling, Specification and Verification

Luca Aceto[1 2] Anna Ingólfsdóttir[1 2]
Kim G. Larsen[1] Jiří Srba[1]

1 Department of Computer Science, Aalborg University, 9220 Aalborg Ø, Denmark
2 Department of Computer Science, School of Science and Engineering, Reykjavík University, Iceland

CAMBRIDGE
UNIVERSITY PRESS

CAMBRIDGE UNIVERSITY PRESS
Cambridge, New York, Melbourne, Madrid, Cape Town, Singapore, São Paulo

Cambridge University Press
The Edinburgh Building, Cambridge CB2 8RU, UK

Published in the United States of America by Cambridge University Press, New York

www.cambridge.org
Information on this title: www.cambridge.org/9780521875462

First published 2007

Printed in the United Kingdom at the University Press, Cambridge

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

ISBN 978-0-521-87546-2 hardback

Contents

Figures and tables

Figures

Tables

Preface

This book is based on courses that have been held at Aalborg University and at Reykjavík University over the last six years or so. The aim of these semester-long courses has been to introduce computer science students, at an early stage of their M.Sc. degrees or late in their B.Sc. degree studies, to the theory of concurrency and to its applications in the modelling and analysis of reactive systems. This is an area of formal-methods study that is finding increasing application outside academic circles and allows students to appreciate how techniques and software tools based on sound theoretical principles are very useful in the design and analysis of non-trivial reactive computing systems.

In order to carry this message across to students in the most effective way, the courses on which the material in this book is based have presented:

- some prime models used in the theory of concurrency (with special emphasis on state-transition models of computation such as labelled transition systems and timed automata);
- languages for describing actual systems and their specifications (with a focus on classic algebraic process calculi such as Milner's calculus of communicating systems and logics such modal and temporal logics); and
- the embodiment of these models and languages in tools for the automatic verification of computing systems.

The use of the theory and the associated software tools in the modelling and analysis of computing systems is a very important component in our courses, since it gives the students hands-on experience in the application of what they have learned and reinforces their belief that the theory they are studying is indeed useful and worth mastering. Once we succeed in awakening an interest in the theory of concurrency and its applications amongst students, it will be more likely that at least some will decide to pursue an in-depth study of the more advanced, and

mathematically sophisticated, aspects of our field – perhaps, during M.Sc. thesis work or at a doctoral level.

It has been very satisfying for us to witness a change in attitude in the students taking our courses over the years. Indeed, we have gone from a state in which most students saw very little point in taking the course on which this material is based to one in which the relevance of the material we cover is uncontroversial to many of them! At the time when an early version of our course was elective at Aalborg University, and taken only by a few mathematically inclined individuals, one student remarked in his course evaluation form 'This course ought to be mandatory for computer science students.' Now the course is indeed mandatory; it is attended by all M.Sc. students in computer science at Aalborg University, and most of them happily play with the theory and tools we introduce in the course.

How did this change in attitude come about? And why do we believe that this is an important change? In order to answer these questions, it might be best to describe first the general area of computer science to which this textbook aims at contributing.

The correctness problem and its importance Computer scientists build artifacts (implemented in hardware, or software or, as is the case in the fast growing area of embedded and interactive systems, a combination of both) that are supposed to offer some well-defined services to their users. Since these computing systems are deployed in very large numbers, and often control crucial and even safety-critical industrial processes, it is vital that they implement the specification of their intended behaviour correctly. The problem of ascertaining whether a computing system does indeed offer the behaviour described by its specification is called the *correctness problem* and is one of the most fundamental problems in computer science. The field of computer science that studies languages for the description of (models of) computer systems and their specifications and (possibly automated) methods for establishing the correctness of systems with respect to their specifications is called *algorithmic verification*.

Despite its fundamental scientific and practical importance, however, twentieth-century computer and communication technology did not pay sufficient attention to the correctness and dependability of systems in its drive toward faster and cheaper products. (See the editorial Patterson (2005) by a former president of the ACM for forceful arguments to this effect.) As a result, system crashes are commonplace, sometimes leading to very costly and even disastrous system failures, such as Intel's Pentium-II bug in the floating-point division unit Pratt (1995) and the crash of the Ariane-5 rocket due to the conversion of a 64-bit real number to a 16-bit integer (Lions, 1996).

Classic engineering disciplines have a time-honoured and effective approach to building artifacts that meet their intended specifications: before actually constructing the artifacts, engineers develop models of the design to be built and subject them to a thorough analysis. Surprisingly, such an approach has only recently been used extensively in the development of computing systems.

This textbook, and the courses we have given over the years based on the material it presents, stem from our deep conviction that every well-educated twenty-first-century computer scientist should be well versed in the technology of algorithmic model-based verification. Indeed, as recent advances in algorithmic verification and applications of model checking (Clarke, Gruemberg and Peled, 1999) have shown, the tools and ideas developed within these fields can be used to analyse designs of considerable complexity that, until a few years ago, were thought to be intractable using formal analysis and modelling tools. (Companies such as AT&T, Cadence, Fujitsu, HP, IBM, Intel, Motorola, NEC, Siemens and Sun – to mention but a few – are using these tools increasingly in their own designs to reduce the time to market and to ensure product quality.)

We believe that the availability of automatic software tools for the model-based analysis of systems is one of the two main factors behind the increasing interest amongst students and practitioners alike in model-based verification technology. Another is the realization that even small reactive systems – for instance, relatively short concurrent algorithms – exhibit very complex behaviours, owing to their interactive nature. Unlike in the setting of sequential software, it is therefore not hard for students to realize that systematic and formal analysis techniques *are useful*, even when not altogether necessary, in obtaining some level of confidence in the correctness of designs. The tool support that is now available to explore the behaviour of models of systems expressed as collections of interacting state machines of some sort should make the theory presented in this textbook very appealing for many students at several levels of their studies.

It is our firmly held belief that only by teaching the beautiful theory of concurrent systems, together with its applications and associated verification tools, to our students shall we be able to transfer the available technology to industry and improve the reliability of embedded software and other reactive systems. We hope that this textbook will offer a small contribution to this pedagogical endeavour.

Why this book? This book is by no means the first devoted to aspects of the theory of reactive systems. Some books that have been published in this area over the last 20 years or so are Baeten and Weijland (1990), Fokkink (2000), Hennessy (1988), Hoare (1985), Magee and Kramer (1999), Milner (1989), Roscoe (1999), Schneider (1999) and Stirling (2001), to mention but a few. However, unlike all

the aforementioned books except Fokkink (2000), Magee and Kramer (1999) and Schneider (1999), the present book was written explicitly to serve as a *textbook*, and it offers a distinctive pedagogical approach to its subject matter that derives from our extensive use in the classroom of the material presented here in book form. In writing this textbook we have striven to transfer to paper the spirit of the lectures on which this text is based. Our readers will find that the style is often colloquial and attempts to mimic the Socratic dialogue with which we try to entice our student audience to take an active part in the lectures and associated exercise sessions. Explanations of the material presented in this textbook are interspersed with questions to our readers and exercises that invite the readers to check straight away whether they understand the material, as it is being presented. We believe that this makes the book suitable for self-study, as well as for use as the main reference text in courses ranging from advanced B.Sc. courses to M.Sc. courses in computer science and related subjects.

Of course, it is not up to us to say whether we have succeeded in conveying the spirit of the lectures in the book you now hold in your hands, but we sincerely hope that our readers will experience some of the excitement that we still have in teaching this material and seeing our students appreciate it and we hope that readers will enjoy working with concurrency theory and the tools it offers for the analysis of reactive systems.

For the instructor We have used much of the material presented in this textbook in several one-semester courses at Aalborg University and at Reykjavík University, amongst others. These courses usually consist of about 30 hours of lectures and a similar number of hours of exercise sessions, where the students solve exercises and work on projects related to the material in the course. As stated above, we believe strongly that these practical sessions play a very important role in enabling students to appreciate the importance of the theory they are learning and to understand it in depth. Examples of recent courses based on this book may be found at the URL

www.cs.aau.dk/rsbook/.

There the instructor will find suggested schedules for his or her courses, exercises that can be used to supplement those in the textbook, links to other useful teaching resources available on the web, further suggestions for student projects and electronic slides that can be used for the lectures. (As an example, we usually supplement lectures covering the material in this textbook with from four to six 45-minute lectures on binary decision diagrams (Bryant, 1992), and their use in verification; these are based on Henrik Reif Andersen's excellent lecture

notes (Andersen, 1998), which are freely available on the web, and on Randel Bryant's survey paper (Bryant, 1992).)

We recommend that the teaching of the material covered in this book be accompanied by the use of software tools for verification and validation. In our courses, we usually employ the Edinburgh Concurrency Workbench (Cleaveland, Parrow and Steffen, 1993) for the part of the course devoted to classic reactive systems and, not surprisingly, UPPAAL (Behrmann, David and Larsen, 2004) for lectures on real-time systems. Both these tools are freely available and their use makes the theoretical material covered during the lectures come alive for the students. Using these tools the students will be able to analyse systems of considerable complexity, and we suggest that courses based upon this book be accompanied by a couple of practical projects involving the use of these, or similar, tools for verification and validation.

We shall maintain a page with all the supporting material and other useful resources for students and instructors alike at the URL

www.cs.aau.dk/rsbook/.

In writing this book, we have tried to be at once pedagogical, careful and precise. However, despite our efforts, we are sure that there is still room for improvement in this text and for the correction of any mistake that may have escaped our attention. We shall use the above web page to inform the reader about additions and modifications to this book.

We welcome corrections (typographical or otherwise), comments and suggestions from our readers. You can contact us by sending an email to the address

rsbook@cs.aau.dk

with subject line 'RS Book'.

Historical remarks and acknowledgments As already stated we have used this material in its present form for courses given at several institutions during the last few years. However, the story of its development is much older and goes back at least to 1986. During that year, the third author (Kim G. Larsen, then a freshly minted Ph.D. graduate from Edinburgh University) took up an academic position at Aalborg University. He immediately began designing a course on the theory of concurrency – the branch of theoretical computer science that he had worked on during his doctoral studies, under the supervision of Robin Milner. His aim was to use the course, and the accompanying set of notes and slides, to attract students to his research area by conveying his enthusiasm for it as well as his belief that

the theory of concurrency is important in applications. That material has stood the 'lecture-room test' well and forms the basis for the first part of the book.

The development of those early courses was strongly influenced by the teaching and supervision of Robin Milner that Kim G. Larsen enjoyed during his doctoral studies in Edinburgh and would not have been possible without them. Even though the other three authors were not Milner's students themselves, the strong intellectual influence of his work and writings on their view of concurrency theory will probably be evident to the readers of this book. Indeed, the 'Edinburgh concurrency-theory school' features prominently in the academic genealogy of each of the authors. For example, Rocco De Nicola and Matthew Hennessy had a strong influence on the view of concurrency theory and/or the work of Luca Aceto and Anna Ingólfsdóttir, and Jiří Srba enjoyed the liberal supervision of Mogens Nielsen.

The material upon which the courses we have held at Aalborg University and elsewhere since the late 1980s were based has undergone gradual change before reaching the present form. Over the years, the part of the course devoted to Milner's calculus of communicating systems and its underlying theory has decreased, and so has the emphasis on some topics of mostly theoretical interest. At the same time, the course material has grown to include models and specification languages for real-time systems. The present material aims at offering a good balance between classic and real-time systems, and between the theory and its applications.

Overall, as already stated, the students' appreciation of the theoretical material covered here has been greatly increased by the availability of software tools based on it. We thank all the developers of the tools we use in our teaching; their work has made our subject matter come alive for our students and has been instrumental in achieving whatever level of success we might have had in our teaching based on this textbook.

This book was partly written while Luca Aceto was on leave from Aalborg University at Reykjavík University, Anna Ingólfsdóttir was working at deCODE Genetics and Jiří Srba was visiting the University of Stuttgart, sponsored by a grant from the Alexander von Humboldt Foundation. They thank these institutions for their hospitality and excellent working conditions. Luca Aceto and Anna Ingólfsdóttir were partly supported by the project 'The Equational Logic of Parallel Processes' (No. 060013021) of The Icelandic Research Fund. Jiří Srba received partial support from a grant of the Ministry of Education of the Czech Republic, project No. 1M0545.

We thank Silvio Capobianco, Pierre-Louis Curien, Gudmundur Hreidarson, Rocco De Nicola, Ralph Leibmann, MohammadReza Mousavi, Guy Vidal-Naquet and the students of the Concurrency Course (Concurrence) (number 2-3) 2004–5,

Master Parisien de Recherche en Informatique, for useful comments on and corrections of drafts of this text.

The authors used drafts of the book in courses taught in the spring terms of 2004, 2005 and 2006 and in the autumn term of 2006 at Aalborg University, Reykjavík University and the University of Iceland. The students who took those courses offered valuable feedback on the text and gave us detailed lists of errata. We thank Claus Brabrand for using a draft of the first part of this book in his course 'Semantics' (Q1, 2005 and 2006) at Aarhus University. The suggestions from Claus and his students helped us improve the text further. Moreover, Claus and Martin Mosegaard designed and implemented an excellent simulator for Milner's calculus of communicating systems and the 'bisimulation-game' game, which our students can use to experiment with the behaviour of processes written in this language and to play the bisimulation game described in the textbook.

Last, but not least, we thank David Tranah at Cambridge University Press for his enthusiasm for our project and also the three anonymous reviewers, who provided useful comments on a draft of this book.

Any remaining infelicity is solely our responsibility.

Luca Aceto and Anna Ingólfsdóttir dedicate this book to their son Róbert, to Anna's sons Logi and Kári and to Luca's mother, Imelde Diomede Aceto. Kim Larsen dedicates the book to his wife Merete and to his two daughters Mia and Trine. Finally, Jiří Srba dedicates the book to his parents, Jaroslava and Jiří and to his wife, Vanda.