Mathematics of Discrete Structures for Computer Science

Gordon J. Pace

Mathematics of Discrete Structures for Computer Science

Springer

Gordon J. Pace
Department of Computer Science,
Faculty of Information and
Communication Technology
University of Malta
Msida, Malta

ISBN 978-3-642-29839-4 ISBN 978-3-642-29840-0 (eBook)
DOI 10.1007/978-3-642-29840-0
Springer Heidelberg New York Dordrecht London

Library of Congress Control Number: 2012940954

ACM Computing Classification (1998): F.4, F.2, G.2

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Dedicated to Claudia
and to my parents

Foreword

Profound knowledge and skills in discrete mathematics are mandatory for a well educated computer scientist. Therefore a corresponding curriculum typically requires at least a basic course on this subject. This textbook, which is based on the lectures given by the author at the University of Malta, is a perfect companion for every student taking such a course.

When teaching a formal computer science topic one can either follow a so-called descriptive or algorithmic approach. Traditionally most textbooks on discrete mathematics follow the descriptive approach meaning that the properties of mathematical structures are listed first while the algorithms to compute corresponding results are of secondary interest. This textbook follows the algorithmic approach which seems to be better suited for computer science students: Mathematical structures are introduced and defined using algorithms which are then characterized by their properties in a second step.

The book covers the material that is essential knowledge for a computer science student. Starting with propositional and first order logic it discusses sets, relations and further discrete mathematical structures. It contains a chapter on type theory before introducing the natural numbers. In the final chapter it discusses how to reason about programs and gives the basics of computability theory.

Besides a thorough formal presentation the book always gives a convincing motivation for studying the corresponding structures and explains the main ideas using illustrations. A well chosen set of exercises rounds off each topic. After studying the presented material a computer science student will be well prepared for further, more specialized courses.

Lübeck, Germany — Martin Leucker

Preface

In computer science, as in other sciences, mathematics plays an important role. In fact, computer science has more in common with mathematics than with the traditional sciences. Consider physics, for example. Clearly, mathematics is right at the core of the subject, used to describe models which explain and predict the physical world. Without mathematics, most of modern physics would have been impossible to develop, since a purely qualitative analysis does not allow one to reason and deduce what logically follows from an observation or a hypothesised model. No manner of qualitative reasoning would have enabled Newton to formulate his law of gravity based on Kepler's claims about planetary motion. Computer science also exploits mathematical reasoning in a similar manner. Given a computer system, one can apply mathematical reasoning to prove that it will always calculate your paycheck correctly.

However, the link is even tighter than this. Computer science is a direct descendant of mathematics, and computers are nothing but the physical embodiment of mathematical systems developed long before the first modern computer was even conceived, let alone built.

In this book, we will be exploring the foundational mathematics which is necessary for the understanding of more advanced courses in computer science. Whether you are designing a digital circuit, a computer program or a new programming language, mathematics is needed to reason about the design—its correctness, robustness and dependability.

There are two distinct approaches used to present mathematical concepts and operators. In the first approach concepts and operators are defined in terms of properties which they satisfy. Based on these definitions, ways of computing the result of applying these operators are then developed and proved to be correct. In contrast, in computer science frequently one takes the opposite approach—one starts by defining ways of calculating the result of an operator, and then proves that the operator one knows how to compute satisfies a number of properties. The two approaches are, in the end, equivalent. However, given that this book is aimed at computer science students, the latter approach is usually adopted.

Finally, most sections are accompanied by exercises which are usually necessary as part of the learning process. Learning mathematics is surprisingly like learning to ride a bicycle. You must try to write formal expressions and proofs before you really understand the concepts you read about. Some of the concepts covered in this book may initially appear to be difficult. However, as you progress through the book and familiarise yourself with the concepts, you will hopefully start requiring less effort to follow the definitions and proofs, and begin to enjoy the beauty of how various concepts in mathematics can be built and reasoned about, based on just a small number of basic concepts.

Buenos Aires, Argentina Gordon J. Pace

Acknowledgements

The original inspiration which led to this book dates back almost two decades, when I followed a course on mathematics for system specification lectured by Joe Stoy at the University of Oxford. His approach of building up the necessary mathematical machinery from the ground up, akin to building a software system, provided the spark which years later led to the design of the structure and content of a course in mathematics of discrete structures, and consequently to this book. Without Joe Stoy's spark, this book would never have been.

This book has been long in the making. Life kept getting in the way. Many students would have benefitted had the book been finished earlier. Instead, they were exposed to the intended content in my lectures—questions and blank faces exposing the parts of the book which I had to rethink. Thanks go out to those hundreds of students without whom the book would have been much poorer.

Thanks also go out to those people who used initial drafts of the book to deliver and support taught courses in mathematics of discrete structures: Christian Colombo, Adrian Francalanza, Ingram Bondin, Mark Bonnici, Andrew Calleja, Karlston Demanuele, Kevin Falzon, Andrew Gauci and Christine Vella. Their feedback was invaluable.

Finally, the shortest thanks go out to those whose support can never be fully acknowledged, no matter how long I write about it—Claudia, Dodie and Josie. Thank you.

Contents