Rolf Drechsler · Mathias Soeken *Editors*

Advanced Boolean Techniques

Selected Papers from the 13th International Workshop on Boolean Problems

Springer

Advanced Boolean Techniques

Rolf Drechsler • Mathias Soeken
Editors

Advanced Boolean Techniques

Selected Papers from the 13th International Workshop on Boolean Problems

Springer

Editors
Rolf Drechsler
Arbeitsgruppe Rechnerarchitektur
Universität Bremen
Bremen, Germany

Mathias Soeken
École Polytechnique Fédérale de Lausanne
Lausanne, Switzerland

ISBN 978-3-030-20322-1 ISBN 978-3-030-20323-8 (eBook)
https://doi.org/10.1007/978-3-030-20323-8

This Springer imprint is published by the registered company Springer Nature Switzerland AG.
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

Boolean functions are at the core of today's computer science and find application in circuit and system specification, synthesis, design understanding, and cryptography. The International Workshop on Boolean Problems[1] is a bi-annually held and well-established forum to discuss the recent advances on problems related to Boolean logic and Boolean algebra. In 2018, the 13th edition of the workshop was held in Bremen, Germany, from September 19th to September 21st. The workshop provided a forum for researchers and engineers from different disciplines to exchange ideas as well as to discuss problems and solutions. The workshop is devoted to both theoretical discoveries and practical applications.

This edited book contains a selection of best papers presented at the workshop. The papers in this volume demonstrate new accomplishments in the theory of Boolean functions. Furthermore, several papers illustrate how these results find their way into important practical applications such as cryptography and design understanding.

The first two chapters in the book are contributions that resulted from the invited keynotes at the conference. In Chap. 1, Görschwin Fey and Rolf Drechsler describe *Self-Explaining Digital Systems: Technical View, Implementation Aspects, and Completeness*. In Chap. 2, Tobias Oder, Tobias Schneider, and Tim Güneysu write about a *Secure Implementation of Lattice-Based Encryption Schemes*. The following nine chapters are extended manuscripts based on the workshop handouts. In Chap. 3, Bernd Steinbach and Christian Posthoff write about *Derivative Operations for Classes C_N of Boolean Functions*. In Chap. 4, Radomir S. Stanković, Milena Stanković, Jaakko T. Astola, and Claudio Moraga investigate bent functions in *Towards the Structure of a Class of Permutation Matrices With Bent Functions*. Oliver Keszocze, Kenneth Schmitz, Jens Schloeter, and Rolf Drechsler show how to improve the performance of SAT solvers in *Improving SAT Solving Using Monte Carlo Tree Search-based Clause Learning* in Chap. 5. The following three chapters are about logic synthesis applications. In Chap. 6, Evandro C. Ferraz,

[1] See www.informatik.uni-bremen.de/iwsbp.

Jeferson de Lima Muniz, Alexandre C. R. da Silva, and Gerhard W. Dueck explore majority-based logic synthesis in *Synthesis of Majority Expressions through Primitive Function Manipulation*. Anna Bernasconi, Fabrizio Luccio, Linda Pagli, and Davide Rucci target synthesis for switching lattices in Chap. 7 *Literal Selection in Switching Lattice Design*. In Chap. 8, Heinz Riener, Rüdiger Ehlers, Bruno Schmitt, and Giovanni De Micheli propose an exact synthesis approach in *Exact Synthesis of ESOP Forms*. D. Michael Miller and Mathias Soeken introduce *An Algorithm for Linear, Affine and Spectral Classification of Boolean Functions* in Chap. 9. Chapter 10 targets reversible functions with *New Results on Reversible Boolean Functions Having Component Functions with Specified Properties* by Paweł Kerntopf, Krzysztof Podlaski, Claudio Moraga, and Radomir S. Stanković. The book is concluded in Chap. 11 by Danila Gorodecky and Tiziano Villa on *Efficient Hardware Operations for the Residue Number System by Boolean Minimization*.

We would like to express our thanks to the program committee of the 13th International Workshop on Boolean Problems as well as to the organizational team, in particular Lisa Jungmann and Kristiane Schmitt. Furthermore, we thank all the authors of contributed chapters who did a great job in submitting their manuscripts of very high quality. A special thank goes to the keynote speakers of the workshop, Prof. Görschwin Fey (Hamburg University of Technology) and Prof. Tim Güneysu (Ruhr-Universität Bochum). Finally, we would like to thank Brinda Megasyamalan, Brian Halm, and Charles Glaser from Springer. All this would not have been possible without their steady support.

Bremen, Germany — Rolf Drechsler
Lausanne, Switzerland — Mathias Soeken
March 2019

Contents

Chapter 1
Self-explaining Digital Systems: Technical View, Implementation Aspects, and Completeness

Görschwin Fey and Rolf Drechsler

1.1 Introduction

Digital systems continuously increase in their complexity due to integration of various new features. Systems handle failures and have complex decision mechanisms for adaptability and autonomy. Understanding why a system performs certain actions becomes more and more difficult for users. Also designers have to cope with the complexity while developing the system or parts of it. The main difficulties are the inaccessibility of the inner logic of the digital system or a lack in understanding all the details. An explanation for actions executed by a digital system unveils the reasons for these actions and, by this, can serve various purposes.

From the outside a user may be puzzled why a technical device performs a certain action, e.g., "why does the traffic light turn red?" In simple cases the user will know the reason, e.g., "a pedestrian pushed the button, so pedestrians get green light, cars get red light." In more complex cases, explanations for actions may not as easily be accessible. When the digital system that controls the larger technical device provides an explanation, the user can understand why something happens. This raises the user's confidence in the correct behavior. The explanation for actions required in this case must refer to external input to the system, e.g., through sensors, and to an abstraction of the internal state that is understandable for a user.

G. Fey (✉)
Hamburg University of Technology, Hamburg, Germany
e-mail: goerschwin.fey@tuhh.de

R. Drechsler
University of Bremen, Bremen, Germany

DFKI Bremen, Bremen, Germany

R. Drechsler, M. Soeken (eds.), *Advanced Boolean Techniques*,
https://doi.org/10.1007/978-3-030-20323-8_1

Also designers of digital systems can benefit from explanations. A typical design task is debugging where a designer has to find the reason for certain actions executed by a digital system. Depending on the current design task a designer may use the same explanations that help users. Additionally, more detailed explanations, e.g., justifying data exchange between functional units may be useful. Thus, debugging and development are supported by explanations giving simple access points for a designer justifying the system's execution paths. At design time a designer can use explanations to understand the relation between the specification and the implementation.

Correctness of the system is validated through explanations if these explanations provide an alternative view that justifies the actual output. For in-field operation explanations may even be exploited for monitoring as a side-check that validates the actual execution of the system to detect failures and unexpected usage. In particular, problems are detected earlier when explanations cannot be generated, are not well-formed, or are not consistent with respect to the actual behavior.

The notion of explanation used here are cause–effect chains as often used in the philosophical domain. Moreover, granularity, addressee, and purpose are defined at design time of a system that then explains all its observable actions at run time. We consider such as system to be self-explaining.

Given a digital system the question is how to provide an explanation for observable actions online. While on first sight this mainly concerns functional aspects also non-functional aspects like actual power consumption or response time of the system deserve explanations.

During online operation either the system itself or some dedicated additional entity must provide the explanations. This incurs a cost, e.g., for storing historical data that explains and, by this, also justifies current and future actions. This overhead must be kept as low as possible.

A non-trivial challenge is to provide concise explanations in a cost-efficient way. While some actions of a system may have very simple explanations, e.g., "the power-on button has been pressed," other actions may require a deep understanding of the system, e.g., "when the distance to an energy source is large and the battery level is low, we save energy by reducing speed as well as light and move towards the energy source." Such an explanation may in turn require knowledge about what an energy source is, what thresholds are used, and how the system detects where the next energy source may be found.

Our contributions are the following:

- We formalize explanations and define what a self-explaining system is. We explain how to verify whether a system is self-explaining.
- We propose a conceptual framework that yields layered explanations providing details where necessary, but keeping explanations understandable at the same time.
- We provide a technical solution for explanations on the functional level and discuss how to automatically infer them.