Natural Computing Series

Founding Editor

Grzegorz Rozenberg

Series Editors

Thomas Bäck, Natural Computing Group–LIACS, Leiden University, Leiden, The Netherlands

Lila Kari, School of Computer Science, University of Waterloo, Waterloo, ON, Canada

Susan Stepney, Department of Computer Science, University of York, York, UK

Scope

The Natural Computing book series covers theory, experiment, and implementations at the intersection of computation and natural systems. This includes:

- **Computation inspired by Nature**: Paradigms, algorithms, and theories inspired by natural phenomena. Examples include cellular automata, simulated annealing, neural computation, evolutionary computation, swarm intelligence, and membrane computing.
- **Computing using Nature-inspired novel substrates**: Examples include biomolecular (DNA) computing, quantum computing, chemical computing, synthetic biology, soft robotics, and artificial life.
- **Computational analysis of Nature**: Understanding nature through a computational lens. Examples include systems biology, computational neuroscience, quantum information processing.

José Raúl Romero · Inmaculada Medina-Bulo · Francisco Chicano
Editors

Optimising the Software Development Process with Artificial Intelligence

Springer

Editors
José Raúl Romero
Department of Computer Science
and Numerical Analysis
University of Córdoba
Córdoba, Córdoba, Spain

Inmaculada Medina-Bulo
Department of Computer Science
and Engineering
University of Cádiz
Puerto Real, Cádiz, Spain

Francisco Chicano
ITIS Software
Department of Languages and Computing
Sciences
University of Malaga
Málaga, Málaga, Spain

ISSN 1619-7127
Natural Computing Series
ISBN 978-981-19-9947-5 ISBN 978-981-19-9948-2 (eBook)
https://doi.org/10.1007/978-981-19-9948-2

© The Editor(s) (if applicable) and The Author(s), under exclusive license to Springer Nature Singapore Pte Ltd. 2023
This work is subject to copyright. All rights are solely and exclusively licensed by the Publisher, whether the whole or part of the material is concerned, specifically the rights of reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
The publisher, the authors, and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Singapore Pte Ltd.
The registered company address is: 152 Beach Road, #21-01/04 Gateway East, Singapore 189721, Singapore

Foreword

Optimisation has really taken off as an area of research, in general, not just in the software engineering community, since the 1940s. However, we can trace the origins of software optimisation right back to the very birth of software itself, with Ada Lovelace's prescient comments on the nature of software and the need for its optimisation. That is, 180 years ago, she wrote:

> In almost every computation a great variety of arrangements for the succession of the processes is possible, and various considerations must influence the selection amongst them for the purposes of a Calculating Engine. One essential object is to choose that arrangement which shall tend to reduce to a minimum the time necessary for completing the calculation.

This is an extract from "Note D" of "Sketch of the Analytical Engine Invented by Charles Babbage" by Ada Augusta Lovelace, 1842. It is surely the first instance of any author writing about the need to improve software execution performance by optimising the expression of the computation itself: optimising the program.

The application of meta heuristic search to software optimisation dates back to the 1970s, but has really taken off in the last two decades with the advent of the field that has come to be known as "Search-Based Software Engineering" (SBSE). The interest in optimisation techniques and evolutionary computation as a vehicle for softer optimisation rendered the software engineering community highly receptive to artificial intelligence techniques more generally. Recent breakthroughs in machine learning have further stimulated software engineers' interest in artificial intelligence in all of its forms and many applications.

Artificial intelligence techniques have been used to optimise almost every kind of engineering artefact and the processes by which they are made, right across the spectrum of engineering disciplines, including mechanical, biological, chemical, and even social engineering disciplines. Nevertheless, it is in software engineering that these techniques find a uniquely well-fitted potential in their application: Not only is the engineering artefact optimised using AI techniques but also the AI techniques are, themselves, implemented in the same engineering material—software. This opens up possibilities for self-application, and, with that, the potential for continuous, dynamic adaptive optimisation in deployed software environments.

I firmly believe that a deeper understanding of the nature of optimisation and its applications to software engineering will yield many more insights into the nature of optimisation itself and also, in so doing, will shed further light on the specific properties of software as an exciting (arguably the most exciting, and certainly the most peculiar) engineering material with which our species has yet worked.

The foundational nature of optimisation has led not only to an explosion of research on Search-Based Software Engineering but also to widespread uptake of computational search as a practical and flexible tool for software engineering optimisation. Most notably, techniques associated with software testing and repair have found their way into widespread industrial practice in many companies, both large and small. This impact has been felt by almost every user of the Internet, through applications at companies such as Meta, Google, and Microsoft. Currently, approximately 2.9 billion people are running software (consisting of tens of millions of lines of code) on smartphones. They are running software that has been automatically fixed by computational search techniques that automatically found fixes for bugs that were automatically detected by test cases that were, themselves, automatically designed by computational search.

This is but one recent example of the application of search-based software engineering to testing and bug fixing, and we can be sure that this is just the beginning. We are witnessing the early development of a fundamentally new approach to software engineering in which artificial intelligence and human intelligence combine to optimise software systems and the processes by which they are produced. Our grandchildren will surely consider the idea that the word "programmer" refers exclusively to a human as being just as quaint and anachronistic as we, today, would consider the view of our nineteenth-century engineering forebearers, who thought that the word "computer" would naturally and solely only ever refer to a human.

Much of the overall body of work on software optimisation has focused on software products, such as optimisation of the code itself, and also documentation, design diagrams, and test suites. However, optimisation has also been applied to software engineering processes. It is this set of related application areas that forms the concern of this excellent compendium of work on optimising software development processes with artificial intelligence.

In this single volume, the reader will find many aspects of the overall software development life cycle tackled using optimisation, from the elicitation of requirements and systems modelling through to development and on to deployment (e.g. cloud-based deployment) and on-going maintenance. These are the processes of most interest and importance to practising software engineers, and form the foundations for multiple research disciplines within the overall body of work on software engineering. The reader will also find work on optimising the overall management of processes and on developer productivity assistance (such as code completion) as well as work on software testing.

The chapters in this book have been written by leading researchers in the field of software process optimisation using AI techniques. The collection into a single volume provides an overview and introduction to an exciting field of engineering optimisation, with many practical applications and challenges for further research.

To further support practitioners and researchers from the software engineering community, seeking to deploy these techniques, the final chapters in this book provide foundational tutorial work on meta heuristics and machine learning. This ensures that the book is useful for researchers and practitioners alike. The interested reader, who wants to follow up further having read the book, will also find many other surveys and tutorials available online.

London, UK Mark Harman
July 2022

Preface

A software development project is complex, and poses constant challenges to the professionals who supervise, plan, design, analyse, develop, and maintain it. When we think of a software project, we must think of all its phases and activities: both in those phases referring to the planning and control of the engineering project, and in the phases of the development process itself, from the specification of the requirements and the architectural design to the testing, refactoring, and maintenance of the software.

In recent years, there has been a clear interest in automating and providing support to the professionals, with the goal of reducing the effort, time, cost, and risk of the different phases of a software project. To this end, initiatives related to the use of artificial intelligence (AI) for optimising these tasks have been tested for decades, from expert systems to search and optimisation techniques to machine learning. The editors of this volume have been working for more than 20 years in software engineering (SE), including the application of AI techniques for the optimisation of the different phases of the software life cycle. And we must recognise that during this time great advances have been made, but the most important comes from the industry itself, which acknowledges the need to apply techniques that improve its software engineering process and demands solutions for it. AI-enhanced software engineering or AI4SE (artificial intelligence for software engineering) was born.

An important factor is the popularisation of general-purpose tools among the developer community based on these techniques. Examples of these applications are the GitHub Copilot intelligent assistant, which had a great impact on the community by making visible and tangible the pros and cons of using AI for software development assistance. Other development assistant tools such as DeepMind AlphaCode or OpenAI Codex have also attracted the attention of professionals in this field—a field that had already been generating assistant tools in academia for some time, such as EvoSuite for the automatic generation of test suites.

The discussions generated about these tools, their internal foundations, their scope, and practicability, as well as the near future of AI4SE research and development have motivated the need for a volume like this. With this book, we aim to

provide Information Technology (IT) professionals (practitioners, developers, software engineers, or managers) as well as advanced graduate and Ph.D. students with a practical introduction to the use of AI techniques to improve and/or optimise the different phases of the software development process, from the initial project planning to the latest deployment. Notice that AI is a broad term, and the book is intended to cover it from different perspectives: optimisation and search algorithms applied to software engineering (also known as Search-Based Software Engineering, SBSE), machine learning, and pattern mining in software analytics, mining software repositories (MSR), natural language processing (NPL), etc. The AI solutions for SE used throughout the book are explained in a didactic way to provide the reader with a sufficient basis for a complete understanding of its content.

For producing this contributed volume, we have relied on renowned authors, highly experienced in each of the areas of the book. We first divided the project according to the classic phases of software project development and studied how AI had been used for the optimisation and improvement of each of them. The authors were invited not only to write descriptively about the state of AI4SE in their specific domain but also to describe real use cases and to develop practical and reproducible examples to enable the reader to understand, execute, and/or modify practical case studies. Most of the chapters come with a reproducibility package composed of source code and datasets for the reader to experience the techniques described in the chapters. A snapshot of the reproducibility packages has been uploaded to Zenodo as complementary material for this volume. But the chapters also provide links to a live version of the packages in GitHub, where the interested reader could find new updates of the software tools presented in this book. Finally, all chapters were reviewed by experts from industry and academia and were double-checked by the editors. The result is a homogeneous book that, despite dealing with complex problems in each topic, allows the reader to go deeper into the areas of interest from the ground up. The two final chapters of the volume cover the necessary background on artificial intelligence techniques for understanding the rest of the text.

Córdoba, Spain — José Raúl Romero
Cádiz, Spain — Inmaculada Medina-Bulo
Málaga, Spain — Francisco Chicano
October 2022

Acknowledgment

The editors would like to thank the qualified experts in AI4SE who have contributed to this volume. Their view opens a unique window to the readers through which they can observe future trends in the application of AI to the different problems that arise during the software development process.

Special mention should be made to Dr. Mark Harman, Professor at University College London and Research Scientist in a renowned IT company. Dr. Harman is considered one of the fathers of search-based software engineering, and we thank him for his willingness to participate in this project by writing the foreword to this volume.

We would also like to thank the reviewers for their selfless work, who have volunteered their time to provide an even better version of each of the chapters:

Dr. Shaukat Ali
Chief Research Scientist, Simula Research Laboratory, Norway

Dr. Jaume Bacardit
Reader, Newcastle University, United Kingdom

Dr. Juan J. Durillo
Scientific Staff, Leibniz Supercomputing Centre of the Bavarian Academy of Sciences and Humanities, Germany

Mr. Aitor Facio Valero
Project Manager, Navantia, Spain

Dr. Arnaud Liefooghe
Associate Professor, University of Lille, France

Mrs. María Mora
CIO at Municipal Information Technology Center, Málaga, Spain

Dr. Gregorio Robles
Full Professor, Universidad Rey Juan Carlos, Spain

Mr. Rubén Salado
Head of Backend, Genially, Spain

Dr. Christopher L. Simons
Lecturer, University of the West of England, United Kingdom.

The development of this volume has been financially supported by Grant PID2020-115832GB-I00 funded by MICIN/AEI/10.13039/501100011033; and the Spanish Research Network SEBASENet[1] funded by the Spanish Ministry of Science and Innovation (SMSI), and co-funded by the European Regional Development Fund (ERDF), project RED2018-102472-T. This work has also been supported by SMSI and ERDF through the project AwESOMe (PID2021-122215NB-C33), and by the University of Malaga (project B4-2019-05).

[1] SEBASENet Research Network: https://www.uco.es/SEBASENet/

Contents

Contributors

José Antonio Parejo Smart Computer Systems Research and Engineering Lab (SCORE), Research Institute of Computer Engineering (I3US), Universidad de Sevilla, Seville, Spain

Antoine Barbez Polytechnique Montreal, Montreal, Canada;
Dalhousie University, Halifax, Canada

Dominik Bork TU Wien, Business Informatics Group, Vienna, Austria

Francisco Chicano University of Málaga, Málaga, Spain

Francisco Gomes de Oliveira Neto Chalmers and the University of Gothenburg, Gothenburg, Sweden

Pedro Delgado-Pérez Department of Computer Science and Engineering, University of Cádiz, Cádiz, Spain

Martin Eisenberg JKU Linz, CDL-MINT, Institute of Business Informatics - Software Engineering, Linz, Austria

Eduard Paul Enoiu Mälardalen University, Vësterås, Sweden

Michael Felderer University of Innsbruck, Innsbruck, Austria;
Blekinge Institute of Technology, Karlskrona, Sweden

Robert Feldt Chalmers and the University of Gothenburg, Gothenburg, Sweden

Thiago Ferreira University of Michigan-Flint, College of Innovation & Technology, Michigan, USA

Javier Ferrer ITIS Software, University of Malaga, Malaga, Spain

Afonso Fontes Chalmers and the University of Gothenburg, Gothenburg, Sweden

Antonio Garmendia JKU Linz, Institute of Business Informatics - Software Engineering, Linz, Austria

Gregory Gay Chalmers and the University of Gothenburg, Gothenburg, Sweden

Marouane Kessentini Oakland University, School of Engineering and Computer Science, Michigan, USA

Foutse Khomh Polytechnique Montreal, Montreal, Canada

Fitsum Kifetew Fondazione Bruno Kessler, Trento, Povo, Italy

Mohammad Masudur Rahman Dalhousie University, Halifax, Canada

Inmaculada Medina-Bulo University of Cádiz, Cádiz, Spain

Leandro L. Minku School of Computer Science, University of Birmingham, Birmingham, UK

Breno Miranda Informatics Center, Federal University of Pernambuco, Recife, Brazil

Tien N. Nguyen University of Texas at Dallas, Dallas, USA

Anna Perini Fondazione Bruno Kessler, Trento, Povo, Italy

Aurora Ramírez Dept. Computer Science and Numerical Analysis, University of Córdoba, Córdoba, Spain

José Raúl Romero University of Córdoba, Córdoba, Spain

Ana Belén Sánchez Smart Computer Systems Research and Engineering Lab (SCORE), Research Institute of Computer Engineering (I3US), Universidad de Sevilla, Seville, Spain

Liyan Song Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China

Angelo Susi Fondazione Bruno Kessler, Trento, Povo, Italy

Sahar Tahvili Mälardalen University, Vësterås, Sweden;
Ericsson AB, Stockholm, Sweden

Manuel Wimmer JKU Linz, CDL-MINT, Institute of Business Informatics - Software Engineering, Linz, Austria