SpringerBriefs in Computer Science

Series editors

Stan Zdonik, Brown University, Providence, Rhode Island, USA
Shashi Shekhar, University of Minnesota, Minneapolis, Minnesota, USA
Xindong Wu, University of Vermont, Burlington, Vermont, USA
Lakhmi C. Jain, University of South Australia, Adelaide, South Australia, Australia
David Padua, University of Illinois Urbana-Champaign, Urbana, Illinois, USA
Xuemin Sherman Shen, University of Waterloo, Waterloo, Ontario, Canada
Borko Furht, Florida Atlantic University, Boca Raton, Florida, USA
V. S. Subrahmanian, University of Maryland, College Park, Maryland, USA
Martial Hebert, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA
Katsushi Ikeuchi, University of Tokyo, Tokyo, Japan
Bruno Siciliano, Università di Napoli Federico II, Napoli, Italy
Sushil Jajodia, George Mason University, Fairfax, Virginia, USA
Newton Lee, Inst for Education Res and Scholarships, Los Angeles, California, USA

SpringerBriefs present concise summaries of cutting-edge research and practical applications across a wide spectrum of fields. Featuring compact volumes of 50 to 125 pages, the series covers a range of content from professional to academic.

Typical topics might include:

- A timely report of state-of-the art analytical techniques
- A bridge between new research results, as published in journal articles, and a contextual literature review
- A snapshot of a hot or emerging topic
- An in-depth case study or clinical example
- A presentation of core concepts that students must understand in order to make independent contributions

Briefs allow authors to present their ideas and readers to absorb them with minimal time investment. Briefs will be published as part of Springer's eBook collection, with millions of users worldwide. In addition, Briefs will be available for individual print and electronic purchase. Briefs are characterized by fast, global electronic dissemination, standard publishing contracts, easy-to-use manuscript preparation and formatting guidelines, and expedited production schedules. We aim for publication 8-12 weeks after acceptance. Both solicited and unsolicited manuscripts are considered for publication in this series.

More information about this series at http://www.springer.com/series/10028

Rui Peng • Yan-Fu Li • Yu Liu

Software Fault Detection and Correction: Modeling and Applications

Springer

Rui Peng
Donlinks School of Economics
& Management
University of Science and Technology
Beijing
Beijing, China

Yu Liu
Department of Systems Engineering
and Engineering Management
City University of Hong Kong
Hong Kong, China

Yan-Fu Li
Department of Industrial & Systems Engineering
Tsinghua University
Beijing, China

ISSN 2191-5768 ISSN 2191-5776 (electronic)
SpringerBriefs in Computer Science
ISBN 978-981-13-1161-1 ISBN 978-981-13-1162-8 (eBook)
https://doi.org/10.1007/978-981-13-1162-8

Library of Congress Control Number: 2018957277

© The Author(s), under exclusive license to Springer Nature Singapore Pte Ltd. 2018
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 translation, 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, express 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

Preface

Software plays an important role in both everyday life and industrial productions. In order to guarantee the reliability of software, a long testing process is usually needed before the software can be finally released to market. During the last few decades, numerous software reliability growth models (SRGMs) have been proposed to characterize the growth of software reliability during testing. These models are very important for management to make related decisions, such as determining the optimal software release time considering both software reliability and total cost.

The models typically can be classified as Markov models, nonhomogeneous Poisson process models, data-driven models, and simulation-based models. These models have been successfully applied into many software projects, especially the NHPP models. However, most traditional models assume that faults can be immediately removed upon detection and thus only construct the fault detection process model. In practice, a detected fault needs to be reported and verified before it can be finally corrected. Some researchers have tried to incorporate the debugging delay into the software reliability modeling process and tried to construct several paired models for both fault detection process and fault correction process. However, these models fail to consider some more practical factors, such as the fault introduction process, the testing resource allocation during testing, and the fault dependency.

In this book, five different works on the modeling of software fault detection and correction processes are introduced, with incorporation of different factors. For each work, the parameter estimation method is introduced, and the applications of the models proposed are illustrated by real dataset. In particular, the optimal release time problem is studied for each work. Chapters 1 and 2 are written by Dr. Yan-Fu LI and Dr. Rui PENG together. Chapters 3, 4, and 7 are written by Dr. Rui PENG. Chapters 5 and 6 are written by Dr. Yu LIU and Dr. Rui PENG together. The following people have also participated in the works included in this book: Prof. Xiangbin Yan and his student Hui Qiu from the University of Science and Technology Beijing, Dr. Qingqing Zhai from Shanghai University, and Chair Professor Min Xie from the City University of Hong Kong. This book is partially supported by the National Natural Science Foundation of China (NSFC) under grant

numbers 71531013 and 71671016. The valuable comments from anonymous reviewers and the assistant editor Jane Li are deeply indebted.

Through reading this book, readers should be able to command how the software reliability is modeled, in particular, when both fault detection process and fault correction process are of concern. To better understand the contents of this book, readers are supposed to command the fundamentals of probability theory and stochastic process. This book is especially used for researchers, practitioners, and graduate students in the field of software engineering.

Beijing, China Rui Peng
Beijing, China Yan-Fu Li
Hong Kong, China Yu Liu

Contents

1 Introduction 1
1.1 Reliability 1
1.2 Software Reliability 2
1.3 Structure of the Book 3
References 4

2 Classification of Models 5
2.1 Software Reliability Models 5
2.2 NHPP Models 5
2.3 Simulation Models 8
2.4 Data-Driven Models 9
2.5 Markov Models 10
2.6 Summary 11
References 12

3 TEF Dependent Software FDP and FCP Models 15
3.1 Introduction 15
3.2 The Modeling Process 16
3.2.1 FDP Model 17
3.2.2 FCP Model 17
3.2.3 Some Specific Models 18
3.2.4 A Summary of Various Testing-Effort Functions 22
3.3 Parameter Estimation 23
3.4 Illustrative Example 25
3.4.1 Dataset Description 25
3.4.2 Select the Most Suitable TEF for This Dataset 26
3.4.3 Performance Analysis 26
3.5 Software Release Policies 27
3.5.1 Software Release Policy Based on Reliability Criterion 27
3.5.2 Software Release Policy Based on Cost Criterion 28

3.5.3 Software Release Policy Based on Mixed Criterion 28
3.5.4 Numerical Examples for Software Release Policy 29
3.6 Summary 30
References 31

4 Software Reliability Models Considering Fault Dependency 33
4.1 Introduction 33
4.2 The Modeling Framework 35
4.2.1 Modeling FDP_L 35
4.2.2 Modeling FCP_L 36
4.2.3 Modeling FDP_D and FCP_D 37
4.2.4 Combined Models 37
4.2.5 Specific Models for Dependent FDP and FCP 38
4.3 Parameter Estimation 40
4.4 Numerical Example 41
4.4.1 Description of the Datasets 41
4.4.2 Performance Analysis 42
4.5 Software Release Policy 44
4.5.1 Software Release Policy Based on Reliability Criterion 45
4.5.2 Software Release Policy Based on Cost Criterion 45
4.5.3 Software Release Policy Based on Mixed Criterion 47
4.5.4 Numerical Examples 47
4.6 Summary 49
References 49

5 General Order Statistics-Based Model 53
5.1 Introduction 53
5.2 Modeling Framework 55
5.2.1 Modeling Fault Detection Process 55
5.2.2 Modeling Fault Correction Process 57
5.3 Parameter Estimation with Weighted Least-Square Estimation Method 59
5.4 Some Application Examples 61
5.4.1 Application to an Existing Dataset 62
5.4.2 Predictability Comparison 62
5.4.3 Optimal Software Release Time Analysis 64
5.4.4 Application to a New and Real Dataset with Unequal Time Interval 66
5.4.5 Performance Comparison 68
5.4.6 Optimal Software Release Time Analysis 70
5.5 Summary 71
References 71

6 Reliability of Multi-release Open-Source Software 75
6.1 Introduction 75
6.2 Modeling Framework in FDP and FCP with Semigrouped Test Data 76
6.2.1 Fault Correction Process Modeling 76
6.2.2 Fault Removing Matrix 76
6.2.3 Modeling FDP and FCP Based on the Fault Removing Matrix 77
6.2.4 Extended Modeling Framework with Multiple Releases . . . 80
6.3 Parameter Estimation and Comparison Criteria 81
6.3.1 Point Estimation 81
6.3.2 Interval Estimation 82
6.3.3 Comparison Criteria 83
6.4 Illustrative Example 84
6.4.1 Dataset Description 84
6.4.2 Model Application with Single Released Software 84
6.4.3 Model Application with Multiple Released Software 89
6.5 Summary 91
References 93

7 FDP and FCP with Four Types of Faults 95
7.1 Introduction 95
7.2 The Modeling Process 97
7.2.1 Fault Detection Process 97
7.2.2 Fault Correction Process 98
7.2.3 Some Special Cases 99
7.3 Parameter Estimation 100
7.4 Illustrative Examples 100
7.5 Software Release Policies 103
7.5.1 Software Release Policy Based on Reliability Criterion 103
7.5.2 Software Release Policy Based on Cost Criterion 104
7.5.3 Software Release Policy Based on Mixed Criterion 105
7.5.4 Numerical Examples for Software Release Policy 105
7.6 Summary 106
References 106

List of Abbreviations

SRGM	Software reliability growth model
"GO model" or "G-O model"	Goel-Okumoto model
MVF	Mean value function
FDP	Fault detection process
FCP	Fault correction process
NHPP	Nonhomogeneous Poisson process
TEF	Testing-effort function
LSE	Least squared error
OLS	Ordinary least square
WLS	Weighted least square
MSE	Mean squared error
RMSE	Root mean squared error
RMSPE	Root mean square prediction error
PDF	Probability density function
CDF	Cumulative distribution function
RADC	Rome Air Development Center
GOS	General order statistics
ETD	Expected time delay
MRE	Mean relative error
FRM	Fault removing matrix
EFRM	Expected FRM
DUCF	Detected but uncorrected faults

List of Notations

$m_d(t)$ or $m(t)$	Mean value function of the fault detection process
$\lambda_d(t)$ or $\lambda(t)$	Fault intensity function for the fault detection process, which equals to $\frac{dm_d(t)}{dt}$ *or* $\frac{dm(t)}{dt}$
$m_r(t)$ or $m_c(t)$	Mean value function of the fault correction process
$\lambda_r(t)$ or $\lambda_c(t)$	Fault intensity function for the fault correction process
$R(\Delta T \mid T)$	Software reliability, defined as the probability that no fault occurs during the following ΔT given that the software is released at T
$C(T)$	Total cost function if software is released at time T