

Stochastic Models for Fault Tolerance

Katinka Wolter

Stochastic Models for Fault Tolerance

Restart, Rejuvenation and Checkpointing

Katinka Wolter
Institute of Computer Science
Freie Universität Berlin
Takustr. 9
D-14195 Berlin
Germany
katinka.wolter@fu-berlin.de

ISBN 978-3-642-11256-0 e-ISBN 978-3-642-11257-7
DOI 10.1007/978-3-642-11257-7
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2010921028

ACM Computing Classification (1998): C.4., G.3, D.4.8

© Springer-Verlag Berlin Heidelberg 2010

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilm or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

The use of general descriptive names, registered names, trademarks, 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.

Cover design: KuenkelLopka GmbH, Heidelberg

Printed on acid-free paper.

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

*To Tobias and our children, Louise,
Margarete, Elisabeth, Samuel and Jakob*

Foreword

The book you hold in your hands, entitled *Stochastic Models for Fault Tolerance - Restart, Rejuvenation and Checkpointing*, found its incarnation in the hidden wonders of the Internet. On the face of it, the Internet may look like just any other network, such as the telephony network of old, cable TV network, satellite communication networks or bank transfer networks. However, when considering the actual usage patterns of the Internet, an exciting world of mathematical interest and curiosity opens up, to which that of other networks pale in comparison. Caused by the sheer number of users and web services, as well as the elegant intricacy of the packet-based network technology, measurements of the Internet have revealed highly intriguing patterns. These patterns exhibit such phenomena as the small world effect, scale free networks and self-similarity, each of which one can find discussed extensively in popular and scientific literature alike.

The work reported in this book came about because of another interesting Internet phenomenon, namely that of the ‘heavy tail’. It says that high download times are relatively common, much more common than with the thin exponential tail that characterizes completion times in traditional communication networks. This has a fascinating consequence, which under certain assumptions can be proven mathematically: it is often faster to abort and retry a download attempt than to wait for it to complete. After all, one may have been caught in the heavy tail, from which one can only escape by clicking the reload button. This fascinating fact seduced us into conducting research into the optimal timing of these retries (or restarts as they will be called in this book), eventually leading to Katinka Wolter’s Habilitation and this book.

Our research resulted in a number of interesting theoretical results, accompanied by extensive experimental work (mostly carried out by Philipp Reinecke). In this book, Katinka Wolter embeds these results into an extensive survey of existing scientific literature in restarts, rejuvenation, checkpointing and preventive maintenance in general. These categories of techniques have in common the problem of timing: how frequent should one carry out the preventive or pre-emptive activity? Mathematically, this leads to a set of related problems and solutions, and this book provides the reader with a careful overview of the various mathematical results as well as their inter-relations.

Many authors have researched problems of timing with great success, and the book highlights the various contributions. From a historical perspective, computing related research on timing was preceded by that on preventive maintenance of systems in general, such as for engine parts, machinery, and so forth. The results for computing systems build on these generic insights, roughly in three phases. First, mostly in the eighties of the past century, the question of when to save intermediate results (a ‘checkpoint’) became important. If a long-running computation fails, a checkpoint assures one does not need to start from scratch, but can roll back to the last checkpoint. The mathematical problem then is to compute the checkpoint interval that minimizes the completion time of the computation, under various fault and load assumptions.

In the nineties, researchers became interested in rejuvenation, especially when it was proposed as a generic fault tolerance approach by Bell Labs researchers. In rejuvenation, a computer will be switched off and on deliberately to avoid imminent more serious uncontrolled crashes. The question is when to ‘rejuvenate’ the computer. This can be based solely on a timer, but as the survey in this book shows, rejuvenation models tend to become more complex since the key for this problem is to identify the state in which to rejuvenate the system. Finally, retry problems, such as in Katinka Wolter’s main line of research, have been discussed throughout the history of computing. In some sense, retries are the dual of preventive maintenance: both deal with the life time of a process, but maintenance tries to prolong life (of a system), while retries try to shorten life (of a job) as fast as possible. This fundamental difference introduces subtle differences in the mathematical models and makes that problems of interest in preventive maintenance do not always translate to retries. In the respective chapters, this book will illustrate these differences.

Readers of this book will find an up-to-date overview of the key theoretical results for timing restarts, rejuvenation and checkpointing, thus serving as a hand book for engineers facing this issue. Computer researchers may find inspiration in the elegance of many of the results, and contribute their own extensions or improvements. Non-specialist readers may hopefully catch a glimpse of the wonders hidden inside the Internet, which emerge from the combined behaviour of all of us browsing the web, downloading music and emailing friends. For all readers, the next time you are browsing the web and a page takes long to download, you may think of this book and use it to time your clicking of the reload button.

It has been a sincere pleasure to have been associated with the research that eventually accumulated into this book. It was great to work with Katinka as well as Philipp solving some of the research problems. I congratulate Katinka Wolter with completing the challenging but extremely valuable task of writing this book and trust the result will be solid information and occasional enjoyment for many readers.

Newcastle, UK,
October 2009

Aad van Moorsel

Preface

As modern society relies on the fault-free operation of complex computing systems, system fault-tolerance has become a matter of course. Common agreement exists that large software systems always contain faults and precautions must be taken to avoid system failure. Failure of hardware components often is caused by external factors that can be neither predicted, avoided, nor corrected. Therefore, mechanisms are needed that guarantee correct service in the presence of failure of system components, be it software or hardware elements. Commonly used are redundancy patterns. These can be either redundancy in space or redundancy in time. Redundancy in space means the inclusion of additional hardware or software modules in the system that can replace a failed component. Different types of redundancy in space exist, such as cold, warm and hot stand-by. Redundancy in time includes methods such as restart, rejuvenation and checkpointing, where execution of tasks is repeated, or the task environment is restarted, either following a system failure or, preventively, during normal operation.

This book is concerned with methods of redundancy in time that need to be issued at the right moment. In particular we address the question of choosing the right time for the different fault-tolerance mechanisms. This includes a brief introduction to the regarded methods, i.e., restart, rejuvenation and checkpointing and aspects of their practical implementation in real-world systems. But the focus of the book is on selecting the right time, or timeout, for restart, rejuvenation and checkpointing. In general, this is the timeout selection problem.

Selecting the right timeout is a problem that is subject to a number of uncertainties. It is, in general, unknown when the system will fail. Furthermore, upcoming busy and idle periods of the system are not known, as is future user interaction with the system. Due to the many uncertainties, the timeout selection problem lends itself for a stochastic treatment. Consequently, many stochastic models addressing the timeout selection problem in restart, rejuvenation and checkpointing have been proposed. This book gives an overview of existing stochastic models of restart, rejuvenation and checkpointing.

The second part of this book treats a stochastic model of restart in various facets. Restart operates on program, or application level. If a task does not complete within a given time it is suspected to have failed and is, consequently, aborted and restarted. The timeout after which to abort and restart the task must be carefully chosen

because if it is too short the task might be aborted just before completion while if it is chosen too long one must wait unnecessarily.

The second method is software rejuvenation. Software rejuvenation restarts the operating environment of a task in order to prevent failures. Rejuvenation is a proactive fault-tolerance treatment. Hence it is issued before the system fails. This is particularly challenging, as it implies assumptions as to when the system would fail if no measures were taken. Ideally, the rejuvenation interval would always end just before the system fails. A conservative choice of the rejuvenation interval will select short intervals. But rejuvenation comes with a cost of saving the operating environment and all processes, restarting the system and reinitialising the operating environment and all processes. If rejuvenation is performed too often the rejuvenation cost accumulates unnecessarily, while if the system is rejuvenated at too long intervals it will often fail, rendering rejuvenation ineffective.

The third method, checkpointing, is the most complex mechanism of the three as it has a preventive component, saving a checkpoint, and a reactive component, roll-back recovery. Checkpointing systems save the system state in regular or irregular time intervals. Upon failure the system recovers by rolling back to the most recent checkpoint. The work performed since the most recent checkpoint is lost with a failure. If checkpoints are taken too frequently the interrupt and save operation incurs too high a cost, while if checkpoint intervals are too long much work is lost upon system failure.

For all three methods similar trade-offs exist. The fault-tolerance mechanisms come at a cost that must be traded against the cost of a potential system failure. If the timeouts are well chosen the fault-tolerance mechanism will avoid a failure and be worth-while. The trade-offs can be evaluated and optimised using stochastic models. The focus of this book is to collect, summarise and compare those stochastic models. This can be seen as a first step towards understanding and solving the generic timeout selection problem.

This book is based on the author's habilitation thesis at Humboldt-University in 2008. The habilitation thesis, and hence this book, would not be as it is without the careful and thorough reading from the first to the last page of Mirek Malek. I would like to thank him for his efforts. His many valuable comments helped to improve this text tremendously. I am thankful to Boudewijn Haverkort and Miklos Telek as they agreed on reviewing and commenting on the thesis. Miklos Telek even came to Berlin for the habilitation lecture, even though it was on yet another topic.

I thank the members of the habilitation committee for their invested time and consideration. It was a pleasure to work with the gifted students at Humboldt-University over the five years I spent there until completion of my habilitation thesis. I particularly thank Willi Engel for being a member of the habilitation committee and Philipp Reinecke for the many little shell scripts he quickly set up and for his dedicated work in the restart project. I thank Steffen Tschirpke for technical support and Christine Henze for help in all administrative matters. I am grateful to my former colleagues at Humboldt-University for many fervid discussions that broadened my view.

Aad van Moorsel has been a colleague on the restart project for many years. He is the perfect person to write a foreword and I am happy he agreed to do so.

I want to take the opportunity to thank Jochen Schiller and his group for giving me such a warm welcome and making me feel at home immediately at Freie Universität. This made me anticipate future work related to restart and also broadened my horizon.

I would like to thank Springer Verlag and Ralf Gerstner in particular for agreeing to publish this book.

Finally, the most crucial support throughout the past years came from my family. I thank Tobias Zepter and our children Louise, Margarete, Elisabeth, Samuel and Jakob just for being there and being my delight every day.

Berlin, Germany,
October 2009

Katinka Wolter

Contents

Part I Introduction

1 Basic Concepts and Problems	3
1.1 The Timeout Problem	3
1.2 System and Fault Models	6
1.3 Preventive Maintenance	9
1.4 Note on Terminology	10
1.5 Outline	11
2 Task Completion Time	13
2.1 Bounded Downtime	18
2.1.1 System Lifetime	18
2.1.2 Cumulative Uptime	20
2.1.3 Probability of Task Completion	21
2.2 Bounded Accumulated Downtime	22
2.2.1 System Lifetime	23
2.2.2 Cumulative Uptime	24
2.2.3 Probability of Task Completion	25
2.3 Bounded Number of Failures	28
2.3.1 System Lifetime	28
2.3.2 Cumulative Uptime	29
2.3.3 Probability of Task Completion	29

Part II Restart

3 Applicability Analysis of Restart	35
3.1 Applications of Restart	35
3.1.1 Randomised Algorithms	35
3.1.2 Optimal Restart Time for a Randomised Algorithm	37
3.1.3 Failure Detectors	39
3.1.4 Congestion Control in TCP	40

3.2	Criteria for Successful Restarts	42
3.2.1	When Does Restart Improve the Expected Completion Time?	43
3.2.2	When Does Restart Improve the Probability of Meeting a Deadline?	48
3.3	Conclusions	50
4	Moments of Completion Time Under Restart	51
4.1	The Information Captured by the Moments of a Distribution	52
4.2	Models for Moments of Completion Time	54
4.2.1	Unbounded Number of Restarts	54
4.2.2	Finite Number of Restarts	60
4.3	Optimal Restart Times for the Moments of Completion Time	62
4.3.1	Expected Completion Time	62
4.3.2	Optimal Restart Times for Higher Moments	69
4.4	Case Study: Optimising Expected Completion Time in Web Services Reliable Messaging	79
4.4.1	Metrics for the Fairness-Timeliness tradeoff	82
4.4.2	Oracles for Restart	83
4.4.3	Results	87
4.5	HTTP Transport	89
4.5.1	60 s Disruption	89
4.5.2	Packet Loss	91
4.5.3	Mail Transport	92
5	Meeting Deadlines Through Restart	95
5.1	A Model for the Probability of Meeting a Deadline Under Restart	95
5.2	Algorithms for Optimal Restart Times	97
5.3	An Engineering Rule to Approximate the Optimal Restart Time	100
5.4	Towards Online Restart for Self-Management of Systems	106
5.4.1	Estimating the Hazard Rate	107
5.4.2	Experiments	109
 Part III Software Rejuvenation		
Introduction		119
6	Practical Aspects of Preventive Maintenance and Software Rejuvenation	123
6.1	Preventive Maintenance	123
6.2	Software Rejuvenation	127

7	Stochastic Models for Preventive Maintenance and Software Rejuvenation	133
7.1	A Markovian Software Rejuvenation Model	133
7.2	Aging in the Modelling of Software Rejuvenation	137
7.2.1	Behaviour in State A under Policy I	141
7.2.2	Behaviour in State A under Policy II	142
7.3	A Petri Net Model	146
7.4	A Non-Markovian Preventive Maintenance Model	151
7.5	Stochastic Processes for Shock and Inspection-Based Modelling	153
7.5.1	The Inspection Model with Alert Threshold Policy	154
7.5.2	The Shock Model with a Risk Policy	159
7.6	Inspection-Based Modelling using the Möbius Modelling Tool	162
7.7	Comparative Summary of the Stochastic Models	164
7.8	Further Reading	166
 Part IV Checkpointing		
Introduction		169
8	Checkpointing Systems	171
8.1	Checkpointing Single-Unit Systems	171
8.2	Checkpointing in Distributed Systems	174
9	Stochastic Models for Checkpointing	177
9.1	Checkpointing at Program Level	178
9.1.1	Equidistant Checkpointing	179
9.1.2	Checkpointing Real-Time Tasks	189
9.1.3	Random Checkpointing Intervals	194
9.1.4	Algorithms for Optimum Checkpoint Selection	197
9.2	Checkpointing at System Level	207
9.2.1	Analytic Models for Checkpointing Transaction-Based Systems	208
9.2.2	Checkpointing Policies for Transaction-Based Systems	214
9.2.3	A Queueing Model for Checkpointing Transaction-Based Systems	224
9.3	A Trade-Off Metric for Optimal Checkpoint Selection	232
9.4	Summary	235
10	Summary, Conclusion and Outlook	237
A	Properties in Discrete Systems	241
A.1	Cumulative First Moment	241
A.2	The Gamma Function	242

B	Important Probability Distributions	243
B.1	Discrete Probability Distributions	243
B.1.1	The Binomial Distribution	243
B.1.2	The Multinomial Distribution	243
B.1.3	The Geometric Distribution	244
B.1.4	The Poisson Distribution	244
B.2	Continuous Probability Distributions	244
B.2.1	The Exponential Distribution	244
B.2.2	The Erlang Distribution and the Hypo-exponential Distribution	246
B.2.3	The Hyperexponential Distribution	247
B.2.4	The Mixed Hyper/Hypo-exponential Distribution	247
B.2.5	The Weibull Distribution	248
B.2.6	The Lognormal Distribution	249
C	Estimating the Hazard Rate	251
C.1	Cumulative Hazard Rate	251
C.2	Epanechnikov Kernel	251
C.3	Bandwidth Estimation	252
D	The Laplace and the Laplace-Stieltjes Transform	253
	References	255
	Index	265
	Glossary	267