Texts in Theoretical Computer Science
An EATCS Series

For further volumes:
http://www.springer.com/series/3214

Fedor V. Fomin · Dieter Kratsch

Exact Exponential Algorithms

Springer

Prof. Dr. Fedor V. Fomin
University of Bergen
Inst. Informatics
PO Box 7800
5020 Bergen
Norway
fomin@ii.uib.no

Prof. Dieter Kratsch
Université Paul Verlaine - Metz
LITA, UFR MIM
Dépt. Informatique
Ile du Saulcy
57045 Cedex 1
France
kratsch@univ-metz.fr

Series Editors

ISSN 1862-4499
ISBN 978-3-642-16532-0 e-ISBN 978-3-642-16533-7
DOI 10.1007/978-3-642-16533-7
Springer Heidelberg Dordrecht London New York

ACM Codes: F.2, G.1, G.2

Cover design: KuenkelLopka GmbH, Heidelberg

Printed on acid-free paper

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

Preface

For a long time computer scientists have distinguished between fast and slow algorithms. Fast (or good) algorithms are the algorithms that run in polynomial time, which means that the number of steps required for the algorithm to solve a problem is bounded by some polynomial in the length of the input. All other algorithms are slow (or bad). The running time of slow algorithms is usually exponential. *This book is about bad algorithms.*

There are several reasons why we are interested in exponential time algorithms. Most of us believe that there are many natural problems which cannot be solved by polynomial time algorithms. The most famous and oldest family of hard problems is the family of NP-complete problems. Most likely there are no polynomial time algorithms solving these hard problems and in the worst-case scenario the exponential running time is unavoidable.

Every combinatorial problem is solvable in finite time by enumerating all possible solutions, i.e. by brute-force search. But is brute-force search always unavoidable? Definitely not. Already in the nineteen sixties and seventies it was known that some NP-complete problems can be solved significantly faster than by brute-force search. Three classic examples are the following algorithms for the TRAVELLING SALESMAN problem, MAXIMUM INDEPENDENT SET, and COLORING. The algorithm of Bellman [17] and Held and Karp [111] from 1962 solves the TRAVELLING SALESMAN problem with n cities in time $\mathcal{O}(2^n n^2)$, which is much faster than the trivial $\mathcal{O}(n!n)$ brute-force search. In 1977, Tarjan and Trojanowski [213] gave an $\mathcal{O}(2^{n/3})$ time algorithm computing a maximum independent set in a graph on n vertices, improving on the brute-force search algorithm which takes $\mathcal{O}(n2^n)$. In 1976, Lawler [150] constructed an $\mathcal{O}(n(1+\sqrt[3]{3})^n)$ time algorithm solving the COLORING problem for which brute-force solution runs in time $\mathcal{O}(n^{n+1})$. On the other hand, for some NP-complete problems, like SATISFIABILITY, regardless of all developments in algorithmic techniques in the last 50 years, we know of no better algorithm than the trivial brute-force search that tries all possible solutions. It is a great intellectual challenge to find whether enumeration of solutions is the only approach to solve NP problems in general.[1]

With the development of the area of exact algorithms, it has become possible to improve significantly the running time of the classical algorithms for MAXIMUM INDEPENDENT SET and GRAPH COLORING. Moreover, very often the techniques for solving NP problems can be used to solve problems that are presumably harder than NP-complete problems, like #*P* and PSPACE-complete problems. On the other hand, for some problems, like the TRAVELLING SALESMAN problem, no progress has been made in the last 50 years. To find an explanation of the wide variation in the worst-case complexities of known exact algorithms is another challenge.

Intellectual curiosity is not the only reason for the study of exponential algorithms. There are certain applications that require exact solutions of NP-hard problems, although this might only be possible for moderate input sizes. And in some situations the preference of polynomial time over exponential is debatable. Richard Lipton in his blog "*Gödel's Lost Letter and P≠ NP*"[2], attributes the following saying to Alan Perlis, the first Turing Award winner, "*for every polynomial-time algorithm you have, there is an exponential algorithm that I would rather run*". The point is simple: $n^3 > 1.0941^n \cdot n$ for $n \leq 100$ and on instances of moderate sizes, the exponential time algorithm can be preferable to polynomial time algorithms. And a reduction of the base of the exponential running time, say from $\mathcal{O}(1.8^n)$ to $\mathcal{O}(1.7^n)$, increases the size of the instances solvable within a given amount of time by a constant *multiplicative* factor; running a given exponential algorithm on a faster computer can enlarge the size only by a (small) *additive* factor. *Thus "bad" exponential algorithms are not that bad in many situations!*

And the final reason of our interest in exact algorithms is of course that the design and analysis of exact algorithms leads to a better understanding of NP-hard problems and initiates interesting new combinatorial and algorithmic challenges.

Our interest in exact algorithms was attracted by an amazing survey by Gerhard Woeginger [220]. This influential survey fascinated many researchers and, we think, was one of the main reasons why the area of exact algorithms changed drastically in the last decade. Still being in a nascent stage, the study of exact algorithms is now a dynamic and vibrant area. While there are many open problems, and new techniques to solve these problems are still appearing, we believe it is the right moment to summarize the work on exact algorithms in a book. The main intention of this book is to provide an introduction to the area and explain the most common algorithmic

---

[1] The origin of this question can be traced to the famous letter of Gödel to von Neumann from 1956:

> "Es wäre interessant zu wissen, ... wie stark im allgemeinen bei finiten kombinatorischen Problemen die Anzahl der Schritte gegenüber dem blossen Probieren verringert werden kann."

English translation: It would be interesting to know, ... how strongly in general the number of steps in finite combinatorial problems can be reduced with respect to simple exhaustive search. [205]

[2] Weblog post *Fast Exponential Algorithms* from February 13, 2009, available at `http://rjlipton.wordpress.com/`

techniques. We have tried to make the results accessible not only to the specialists working in the area but to a more general audience of students and researchers in Computer Science, Operations Research, Optimization, Combinatorics, and other fields related to algorithmic solutions of hard optimization problems. Therefore, our preferences in presenting the material were for giving the basic ideas, avoiding improvements which require significant technical effort.

Bergen, Metz, *Fedor V. Fomin*
August 2010 *Dieter Kratsch*

Acknowledgements

We are deeply indebted to Gerhard Woeginger for attracting our attention to Exact Algorithms in 2002. His survey on exact algorithms for NP-hard problems was a source of inspiration and triggered our interest in this area.

This work would be impossible without the help and support of many people. We are grateful to all our coauthors in the area of exact algorithms: Omid Amini, Hans Bodlaender, Hajo Broersma, Jianer Chen, Frederic Dorn, Mike Fellows, Henning Fernau, Pierre Fraigniaud, Serge Gaspers, Petr Golovach, Fabrizio Grandoni, Gregory Gutin, Frédéric Havet, Pinar Heggernes, Pim van 't Hof, Kjartan Høie, Iyad A. Kanj, Joachim Kneis, Arie M. C. A. Koster, Jan Kratochvíl, Alexander Langer, Mathieu Liedloff, Daniel Lokshtanov, Frédéric Mazoit, Nicolas Nisse, Christophe Paul, Daniel Paulusma, Eelko Penninkx, Artem V. Pyatkin, Daniel Raible, Venkatesh Raman, Igor Razgon, Frances A. Rosamond, Peter Rossmanith, Saket Saurabh, Alexey A. Stepanov, Jan Arne Telle, Dimitrios Thilikos, Ioan Todinca, Yngve Villanger, and Gerhard J. Woeginger. It was a great pleasure to work with all of you and we profited greatly from all this cooperation. Special thanks go to Fabrizio Grandoni for introducing us to the world of Measure & Conquer.

Many of our colleagues helped with valuable comments, corrections and suggestions. We are grateful for feedback from Evgeny Dantsin, Serge Gaspers, Fabrizio Grandoni, Petteri Kaski, Mikko Koivisto, Alexander Kulikov, Mathieu Liedloff, Daniel Lokshtanov, Jesper Nederlof, Igor Razgon, Saket Saurabh, Uwe Schöning, Gregory Sorkin, Ioan Todinca, K. Venkata, Yngve Villanger, Ryan Williams, and Peter Rossmanith.

It is a special pleasure to thank our wives Nora and Tanya. Without their encouragement this work would have taken much longer.

Contents