Lecture Notes in Computer Science 1089
Edited by G. Goos, J. Hartmanis and J. van Leeuwen

Advisory Board: W. Brauer D. Gries J. Stoer

G. Ramalingam

Bounded Incremental Computation

Springer

Series Editors

Gerhard Goos, Karlsruhe University, Germany

Juris Hartmanis, Cornell University, NY, USA

Jan van Leeuwen, Utrecht University, The Netherlands

Author

G. Ramalingam
IBM T.J. Watson Research Center
P.O. Box 704, Yorktown Heights, NY 10598, USA

Cataloging-in-Publication data applied for

Die Deutsche Bibliothek - CIP-Einheitsaufnahme

Ramalingam, G.:
Bounded incremental computation / G. Ramalingam. - Berlin ; Heidelberg ; New York ; Barcelona ; Budapest ; Hong Kong ; London ; Milan ; Paris ; Santa Clara ; Singapore ; Tokyo : Springer, 1996
(Lecture notes in computer science ; 1089)
ISBN 3-540-61320-X
NE: GT

CR Subject Classification (1991): F.2, G.2, F.1, D.4.7, D.2.6, D.3.4

ISBN 3-540-61320-X Springer-Verlag Berlin Heidelberg New York

Typesetting: Camera-ready by author
SPIN 10513144 06/3142 - 5 4 3 2 1 0 Printed on acid-free paper

to

my mother, my father, and Raji

with love

என் தாய், என் தந்தை, ராஜி மூவருக்கும்

அன்புடன்

Preface

Algorithms and programs transform their input into their output. *Incremental computation* concerns the re-computation of output after a change in the input. An incremental algorithm, consequently, transforms a *change in input* into a *change in output.* Incremental algorithms, also called dynamic or on-line algorithms, are becoming increasingly important given the popularity of interactive systems, which must respond efficiently to a user's actions, which are usually "modifications" of an "input document".

In the context of incremental computation, small changes in the input are likely to cause correspondingly small changes in the output. It is natural to attempt to identify the part of the previous output that is no longer "correct" and "update" it. Where it is not possible to identify the affected part of the output *exactly*, an incremental algorithm may attempt to identify a *conservative approximation* (that is, an over-estimation) of the affected part of the output. Given some part of the output that needs to be recomputed, an incremental algorithm would benefit by processing only the portion of the input it needs to process in order to generate that part of the output. The effectiveness or efficiency of an incremental algorithm depends on how accurate an approximation to the affected region it can identify and on the overhead it incurs in doing this.

This book presents results — upper bound results, lower bound results, and experimental results — for several incremental computation problems. What is common to all these results is that we seek to determine the efficiency of an algorithm by analyzing how accurate an approximation to the affected region it identifies and on the overhead it incurs in doing this. In particular, we try to analyze the complexity of incremental algorithms and problems in terms of a parameter $\|\delta\|$ that measures the *size of the change in the input and output.* An incremental algorithm is said to be *bounded* if the time it takes to update the output depends only on the size of the *change* in the input and output (i.e., $\|\delta\|$), and not on the size of the *entire* current input. Otherwise, an incremental algorithm is said to be *unbounded.* A problem is said to be *bounded* (unbounded) if it has (does not have) a bounded incremental algorithm. The results established in this book illustrate a complexity hierarchy for incremental computation from this point of view. These results are summarized below.

We present efficient $O(\|\delta\| \log \|\delta\|)$ incremental algorithms for several shortest-path problems — the single-sink shortest-path problem, the all-pairs shortest-path problem, and a generalization of the single-sink shortest-path problem due to Knuth — establishing that these problems are all polynomially bounded. These results show that it is possible in these problems to identify, without much of an overhead, exactly the part of the output that needs to be updated.

We present an $O(2^{\|\delta\|})$ incremental algorithm for the circuit value annotation problem, which matches a previous $\Omega(2^{\|\delta\|})$ lower bound for this problem. Consequently, this establishes that the circuit value annotation problem is an exponentially bounded problem. We also present experimental results that show that our algorithm,

in spite of a worst-case complexity of $\Theta(2^{\|\delta\|})$, works well in practice, often identifying a very good approximation to the affected output with very little overhead.

We present lower bounds showing that a number of problems, including graph reachability, dataflow analysis, and algebraic path problems, are unbounded with respect to a model of computation called the *sparsely-aliasing pointer machine* model.

We present an $O(\|\delta\| \log n)$ incremental algorithm for the reachability problem in reducible flowgraphs, which identifies the affected output exactly, but with an $O(\log n)$ factor overhead.

We present an algorithm for maintaining the dominator tree of a reducible flowgraph, which identifies a reasonable approximation to the affected output and updates it efficiently.

Acknowledgements

This book is a slightly revised version of my Ph.D. thesis. I am greatly indebted to my thesis advisor Thomas Reps for his guidance, both personal and professional, both while I was a student and after my graduation, which, among other things, shaped the thesis and its form. I thank also

- Susan Horwitz and Marvin Solomon, for their patient reading of my thesis and their valuable suggestions which helped improve the thesis;
- Eric Bach and Richard Brualdi, the other members of my thesis committee;
- Prasoon Tiwari, for various useful discussions I had with him;
- Charles Fischer, for his excellent teaching;
- the members of the "PL" group, in particular, Paul Adams, Tom Ball, Sam Bates, Dave Binkley, Lorenz Huelsbergen, Phil Pfeiffer, Todd Proebsting, and Wuu Yang;
- IBM, for partially supporting me through a graduate fellowship while I was a student, and for its support subsequently;
- my friends, too numerous to list here, who made my stay at Madison, Wisconsin, a memorable one;

I especially thank my family, without whose support this would not have been possibe. I thank my parents, my wife, and my sister for their love and support over the years.

May 1996 G. Ramalingam

Table Of Contents