Efficient Index Maintenance for Text Databases

A thesis submitted for the degree of
Doctor of Philosophy

Nicholas Lester  B.E. (Hons.), B.Sc,
School of Computer Science and Information Technology,
Science, Engineering, and Technology Portfolio,
RMIT University,
Melbourne, Victoria, Australia.

1st August, 2006
Declaration

I certify that except where due acknowledgement has been made, the work is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program; and, any editorial work, paid or unpaid, carried out by a third party is acknowledged.

Nicholas Maxwell Lester
School of Computer Science and Information Technology
RMIT University
1st August, 2006
Acknowledgments

There are many people who deserve thanks for their help, leading to the publication of this thesis.

During the course of my candidature, I have been truly privileged to work with three excellent supervisors. Firstly, Hugh Williams, who introduced me to research and without whom I would not have undertaken this doctorate. Justin Zobel, my primary primary supervisor, whose friendly conversation and enthusiasm guided me through good times and bad. Finally, Alistair Moffat, who demonstrates a standard of research to aspire to and doesn’t mind sharing.

The Search Engine Group at RMIT also deserve my gratitude for the friendship and occasionally studious atmosphere they provided. I would like to particularly thank Simon Wilkinson, William Webber, Bodo von Billerbeck, Steven Garcia, Adam Cannane, Michael Cameron, and Yaniv Bernstein for our association. In addition, Bodo and Simon kindly volunteered to proof-read this thesis.

Naturally, none of this would have been possible without my family. My mother, Rosemary, is an amazingly capable woman and was my first intellectual role model. My father, Chris, who fostered my interest in computing and whose common-sense advice can be relied upon. I would also like to thank my sisters, Rebecca and Josie, for providing a sense of perspective whenever it was required, and my grandparents, Elaine and Ray, for offering the sanctuary of their home more often than I was able to escape there.

Finally, I would like to thank my girlfriend, Katherine Chong. Her patience and love have humbled me for four years, and I hope that they continue to do so.
Credits

Preliminary versions of material from this thesis have appeared in the following publications. Chapters 3, 4, 5 and 6 contain material previously published by Lester, Moffat, and Zobel [2005b]. Chapters 3, 4, and 5 also contain material previously published by Lester, Zobel, and Williams [2004], and by Lester, Zobel, and Williams [2005c]. Chapter 8 contains material previously published by Lester, Moffat, Webber, and Zobel [2005a]. William Webber, co-author of [Lester, Moffat, Webber, and Zobel, 2005a], is a research student at the University of Melbourne. Material produced in collaboration with William is individually attributed in Chapter 8.

This work has been supported by grants from the Australian Research Council and the RMIT Virtual Research and Innovation Institutes. We are grateful to MSN Search for the provision of a large (anonymised!) query log containing government-related queries. The thesis was written in the Vim editor on Debian Linux, and typeset using the \LaTeX document preparation system. All trademarks are the property of their respective owners.
# Contents

Abstract 1

1 Introduction 3

2 Background 9

  2.1 Query semantics ..................................................... 10
  2.2 Inverted indexes ..................................................... 14
  2.3 Index representation ................................................ 17
    2.3.1 Integer compression ............................................. 19
    2.3.2 Gap compression ............................................... 20
  2.4 Collections and queries ............................................ 22
  2.5 Index construction ................................................. 24
    2.5.1 Sort-based in-situ inversion ................................... 26
    2.5.2 Single-pass in-memory inversion ................................ 29
    2.5.3 Two-pass inversion ............................................. 33
  2.6 Index maintenance .................................................. 34
    2.6.1 Document modification .......................................... 34
    2.6.2 Document insertion ............................................. 35
    2.6.3 Document deletion .............................................. 37
  2.7 Efficient query evaluation ........................................ 40
    2.7.1 Accumulator pruning ............................................ 41
    2.7.2 Conjunctive query resolution ................................... 42
    2.7.3 Stopping ......................................................... 43
    2.7.4 Skipping ......................................................... 44
    2.7.5 Frequency and impact-ordered evaluation ....................... 45
CONTENTS

2.8 Discussion ................................................................. 47

3 Rebuild index maintenance ........................................ 49
  3.1 The REBUILD strategy ............................................. 49
  3.2 Analysis ................................................................. 51
  3.3 Experiments ............................................................ 52
  3.4 Discussion ............................................................... 54

4 Inplace index maintenance .......................................... 55
  4.1 The INPLACE strategy ................................................ 55
    4.1.1 Pulsing and bucketing ......................................... 58
    4.1.2 Predictive over-allocation .................................... 59
    4.1.3 Query piggy-backing ........................................... 62
    4.1.4 Partial flushing .................................................. 64
    4.1.5 Discontiguity ...................................................... 66
  4.2 Analysis ................................................................. 68
  4.3 Experiments ............................................................ 71
  4.4 Discussion ............................................................... 79

5 Remerge Index Maintenance ...................................... 81
  5.1 The REMERGE strategy ............................................ 81
    5.1.1 Hybrid Maintenance ............................................ 83
  5.2 Analysis ................................................................. 84
  5.3 Experiments ............................................................ 86
  5.4 Discussion ............................................................... 87

6 Geometric partitioning ............................................ 89
  6.1 The GEOM strategy .................................................. 90
    6.1.1 Hierarchical merging .......................................... 92
  6.2 Analysis ................................................................. 93
    6.2.1 Varying the radix .............................................. 96
  6.3 Experiments ............................................................ 99
  6.4 Discussion ............................................................... 103
# CONTENTS

## 7 Vocabulary management

7.1 The V-REMERGE strategy .................................................. 106
7.2 The V-INPLACE strategy .................................................... 106
7.3 The V-GEOM strategy ....................................................... 109
7.4 Experiments ................................................................. 112
7.5 Discussion ................................................................. 120

## 8 Query evaluation revisited

8.1 The QUIT and CONTINUE strategies ........................................ 124
8.2 Adaptive pruning .......................................................... 128
8.3 Experiments ................................................................. 133
8.4 Discussion ................................................................. 141

## 9 Conclusion

## A Experimental Specifications

Bibliography
## List of Figures

1.1 Timeline of queries for “Abu Ghraib” .......................... 5  
2.1 The operation of a text retrieval system .......................... 10  
2.2 The moby collection .................................................. 11  
2.3 Common ranking metrics ............................................. 13  
2.4 The moby collection, annotated with an inverted index and term identifiers . 15  
2.5 Cost estimates for querying with and without an index .................. 16  
2.6 The components of an index ........................................... 18  
2.7 The moby collection, annotated with an inverted index and gap information . 21  
2.8 The moby collection, annotated with postings information ............ 24  
2.9 Sort-based indexing of the moby collection ........................ 27  
2.10 Distributive inversion of the moby collection ........................ 29  
2.11 The hierarchy of index maintenance methods ........................ 36  
2.12 Document-at-a-time versus term-at-a-time query evaluation ............. 39  
2.13 A partial frequency-ordered index for the moby collection ............ 45  

3.1 The distinction between rebuild and alternative strategies ............... 50  
3.2 Construction time for REBUILD on the wt10g collection .............. 53  

4.1 An index being maintained using an inplace strategy .................. 56  
4.2 An index being maintained using predictive over-allocation .............. 60  
4.3 Distribution of in-memory inverted list sizes prior to merging events ..... 65  
4.4 An index being maintained using discontiguity ........................ 66  
4.5 Construction time for INPLACE on the wt10g collection ............... 70  
4.6 Fragmentation caused by INPLACE using pulsed inverted lists ........... 72  
4.7 Construction time for PROP INPLACE on the wt100g-50 collection ...... 74  

xi
4.8 Fragmentation caused by PROP INPLACE ............................................ 75
4.9 Postings relocated by INPLACE and PROP INPLACE during construction . . . 76
4.10 Time taken to execute logical seeks .................................................. 77
4.11 Number of logical seeks for INPLACE and PROP INPLACE ..................... 79

5.1 An index being maintained using a remerge strategy .............................. 82
5.2 Construction time for the REMERGE strategy on the \textit{wt100g-50} collection . 86

6.1 A geometrically partitioned index ....................................................... 90
6.2 Fixed-radix geometric partitioning merging pattern with $r = 3$ ................ 92
6.3 Fixed-partitions geometric partitioning merging pattern with $p = 2$ .......... 98
6.4 Construction times for GEOM on the \textit{wt100g} collection ...................... 100
6.5 Average time per query for varying numbers of queries .......................... 101
6.6 Query times during construction of indexes for the \textit{wt100g} collection ........ 102

7.1 Proportion of the vocabulary changed per merging event ........................ 108
7.2 Cost of index construction components using GEOM, $r = 3$, \textit{V-REMERGE} .... 111
7.3 Cost of index construction components using GEOM, $r = 3$, \textit{V-REMERGE} and
    a small buffer .................................................................................. 113
7.4 Cost of index construction components using GEOM, $r = 3$, \textit{V-GEOM}, $c = 1$ 115
7.5 Construction times for fixed-radix \textit{V-GEOM} on the \textit{gov2} collection ......... 116
7.6 Construction times for fixed-partition \textit{V-GEOM} on the \textit{gov2} collection ....... 117
7.7 Query times during construction of indexes for the \textit{gov2} collection ............ 118
7.8 Query times during construction of indexes for the \textit{gov2} collection (cont.) .... 119

8.1 Accumulator use and thresholds during adaptive pruning .......................... 136
8.2 Effectiveness trade-off for different pruning techniques .......................... 137
8.3 Efficiency trade-off for the ADAPTIVE PRUNING and CONTINUE strategies ... 138
8.4 Efficiency trade-off for the ADAPTIVE PRUNING and QUIT strategies .......... 139
List of Tables

2.1 Test collection characteristics .................................................. 22
2.2 The effect of stopping on query resolution time and effectiveness .......... 44
4.1 Cost estimates for REBUILD, INPLACE, and PROP INPLACE ................. 69
5.1 Cost estimates for the REMERGE algorithm .................................... 85
6.1 Cost estimates for GEOM with a fixed radix ................................. 94
6.2 Cost estimates for GEOM with a fixed number of partitions ................ 97
6.3 Cost estimates for GEOM ............................................................. 99
8.1 Comparison of pruning strategies ................................................. 134
# List of Algorithms

1. The **OFFLINE** index construction algorithm ........................................ 32  
2. The **REBUILD** online index construction algorithm .......................... 51  
3. The **INPLACE** online index construction algorithm .......................... 57  
4. The **PROP INPLACE** online index construction algorithm ................. 62  
5. The **REMERGE** online index construction algorithm ........................ 83  
6. The **QUIT-FULL** ranked query processing algorithm ....................... 124  
7. The **CONTINUE-FULL** ranked query processing algorithm ................. 125  
8. The **QUIT-PART** ranked query processing algorithm ....................... 126  
9. The **CONTINUE-PART** ranked query processing algorithm ................. 128  
10. The **ADAPTIVE PRUNING** ranked query processing algorithm .......... 130  
11. Threshold estimation while processing ranked queries using adaptive pruning 131
Abstract

All practical text search systems use inverted indexes to quickly resolve user queries. Offline index construction algorithms, where queries are not accepted during construction, have been the subject of much prior research. As a result, current techniques can invert virtually unlimited amounts of text in limited main memory, making efficient use of both time and disk space. However, these algorithms assume that the collection does not change during the use of the index.

This thesis examines the task of index maintenance, the problem of adapting an inverted index to reflect changes in the collection it describes. Existing approaches to index maintenance are discussed, including proposed optimisations. We present analysis and empirical evidence suggesting that existing maintenance algorithms either scale poorly to large collections, or significantly degrade query resolution speed. In addition, we propose a new strategy for index maintenance that trades a strictly controlled amount of querying efficiency for greatly increased maintenance speed and scalability. Analysis and empirical results are presented that show that this new algorithm is a useful trade-off between indexing and querying efficiency. In scenarios described in Chapter 7, the use of the new maintenance algorithm reduces the time required to construct an index to under one sixth of the time taken by algorithms that maintain contiguous inverted lists.

In addition to work on index maintenance, we present a new technique for accumulator pruning during ranked query evaluation, as well as providing evidence that existing approaches are unsatisfactory for collections of large size. Accumulator pruning is a key problem in both querying efficiency and overall text search system efficiency. Existing approaches either fail to bound the memory footprint required for query evaluation, or suffer loss of retrieval accuracy. In contrast, the new pruning algorithm can be used to limit the memory footprint of ranked query evaluation, and in our experiments gives retrieval accuracy not worse than previous alternatives.
The results presented in this thesis are validated with robust experiments, which utilise collections of significant size, containing real data, and tested using appropriate numbers of real queries. The techniques presented in this thesis allow information retrieval applications to efficiently index and search changing collections, a task that has been historically problematic.