Contents

I  What, Why and How  1

1  Introduction  2
   1.1 Publications derived from the Thesis  4
   1.2 Awards  6
   1.3 Research Stays and Internships  6

2  Basic Concepts  7
   2.1 Computation model  7
   2.2 Sequences, Documents and Collections  7
   2.3 Empirical Entropy  8
   2.4 Huffman Codes  8
   2.5 Integer Encoding Mechanisms  9
      2.5.1 Bit-aligned Codes  10
      2.5.2 Byte-aligned Codes  11
   2.6 Rank, Select and Access  13
      2.6.1 Binary Sequences  13
      2.6.2 General Sequences  15
   2.7 Wavelet Tree  16
      2.7.1 Data Structure  16
      2.7.2 Rank, Select and Access  16
      2.7.3 Wavelet Trees and Grids  18
      2.7.4 Two-Dimensional Range Queries  19
   2.8 Directly Addressable Codes  20
   2.9 $K^2$-tree  21
   2.10 Compact Trees  24
   2.11 Differentially Encoded Search Trees  25
   2.12 Range Maximum Queries  26
   2.13 Treaps and Priority Search Trees  27
   2.14 Text Indexes  27
      2.14.1 Generalized Suffix Tree and Suffix Array  28
      2.14.2 Self Indexes  30
      2.14.3 Inverted Indexes  33
   2.15 Muthukrishnan’s Algorithm  33
   2.16 Information Retrieval Concepts  35
      2.16.1 Scoring  35
      2.16.2 Top-k Query Processing and Two-stage Ranking Process  37
II Inverted Indexes

3 Query Processing and Compression of Posting Lists

3.1 Query Processing

3.1.1 Early Termination

3.1.2 Index Organization

3.1.3 Query Processing Strategies

3.2 Algorithms for Top-k Ranked Retrieval

3.2.1 Weak-And (WAND)

3.2.2 Block-Max WAND

3.2.3 Persin et al.’s Algorithm

3.2.4 Other Approaches

3.3 Dual-Sorted Inverted Lists

3.3.1 Basic operations

3.3.2 Complex Operations

3.4 Compression of Posting Lists

4 Inverted Treaps

4.1 Inverted Index Representation

4.1.1 Construction

4.1.2 Compact Treap Representation

4.1.3 Representing the Treap Topology

4.1.4 Practical Improvements

4.2 Query Processing

4.2.1 General Procedure

4.2.2 Intersections

4.2.3 Unions

4.2.4 Supporting Different Score Schemes

4.3 Incremental Treaps

4.3.1 Supporting Insertions

4.3.2 Gradual Growth

5 Experimental Evaluation

5.1 Datasets and Test Environment

5.1.1 Baselines and Setup

5.1.2 Dual-Sorted Implementation

5.1.3 Inverted Treaps Implementation

5.2 Results

5.2.1 Index Size

5.2.2 Construction Time

5.2.3 Ranked Union Query Processing

5.2.4 Ranked Intersection Query Processing

5.2.5 One-word Queries

5.2.6 Incremental Treaps

5.3 Discussion
III Practical Top-k on General Sequences

6 Top-k Document Retrieval on General Sequences

6.1 Linear Space Solutions
6.1.1 Hon et al. solution
6.1.2 Optimal solution

6.2 Practical Implementations
6.2.1 Hon et al.
6.2.2 GREEDY
6.2.3 NPV
6.2.4 SORT
6.2.5 Patil et al.
6.2.6 Other aproaches
6.2.7 Summary

7 Top-k on Grids

7.1 Wavelet Trees and RMQ
7.2 $K^2$-treap
7.2.1 Local Maximum Coordinates
7.2.2 Local Maximum Values
7.2.3 Tree Structure
7.2.4 Top-k Query Processing
7.2.5 Other Supported Queries

8 Practical Implementations

8.1 A Basic Compact Implementation
8.1.1 Searching for Patterns
8.1.2 Mapping to the Grid
8.1.3 Finding Isolated Documents
8.1.4 Representing the Grid
8.1.5 Representing Labels and Weights
8.1.6 Summing Up
8.1.7 Answering Queries

8.2 An Improved Index
8.2.1 Mapping the Suffix Tree to the Grid
8.2.2 Smaller Grid Representations
8.2.3 Efficient Construction for Large Collections
8.2.4 Summing Up

9 Experimental Evaluation

9.1 Datasets and Test Environment
9.2 Implementations
9.2.1 Baselines
9.2.2 Implementation of WRMQ
9.2.3 Implementation of $K^2$Treap $^H$
9.2.4 Implementation of $WIRMQ^H$
9.3 Results
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.3.1</td>
<td>Space</td>
<td>122</td>
</tr>
<tr>
<td>9.3.2</td>
<td>Retrieval Speed Varying $k$</td>
<td>124</td>
</tr>
<tr>
<td>9.3.3</td>
<td>Varying Pattern Length</td>
<td>129</td>
</tr>
<tr>
<td>9.3.4</td>
<td>Word Alphabet Collections</td>
<td>131</td>
</tr>
<tr>
<td>9.4</td>
<td>Discussion</td>
<td>133</td>
</tr>
<tr>
<td>IV</td>
<td>Extensions and Evangelization</td>
<td>139</td>
</tr>
<tr>
<td>10</td>
<td>Aggregated 2D Queries on Clustered Points</td>
<td>140</td>
</tr>
<tr>
<td>10.1</td>
<td>Motivation</td>
<td>140</td>
</tr>
<tr>
<td>10.2</td>
<td>Augmenting the $K^2$-tree</td>
<td>142</td>
</tr>
<tr>
<td>10.2.1</td>
<td>Supporting Counting Queries</td>
<td>143</td>
</tr>
<tr>
<td>10.3</td>
<td>Experiments and Results</td>
<td>145</td>
</tr>
<tr>
<td>10.3.1</td>
<td>Experiment Setup</td>
<td>146</td>
</tr>
<tr>
<td>10.3.2</td>
<td>Space Comparison</td>
<td>147</td>
</tr>
<tr>
<td>10.3.3</td>
<td>Query Processing</td>
<td>148</td>
</tr>
<tr>
<td>10.4</td>
<td>Discussion</td>
<td>150</td>
</tr>
<tr>
<td>11</td>
<td>Web Access Logs</td>
<td>152</td>
</tr>
<tr>
<td>11.1</td>
<td>Motivation</td>
<td>152</td>
</tr>
<tr>
<td>11.2</td>
<td>Our Index</td>
<td>154</td>
</tr>
<tr>
<td>11.2.1</td>
<td>Construction</td>
<td>154</td>
</tr>
<tr>
<td>11.2.2</td>
<td>Queries</td>
<td>155</td>
</tr>
<tr>
<td>11.3</td>
<td>Experiments and Results</td>
<td>157</td>
</tr>
<tr>
<td>11.3.1</td>
<td>Setup and Implementations</td>
<td>157</td>
</tr>
<tr>
<td>11.3.2</td>
<td>Experimental Data</td>
<td>157</td>
</tr>
<tr>
<td>11.3.3</td>
<td>Space Usage</td>
<td>158</td>
</tr>
<tr>
<td>11.3.4</td>
<td>Time Performance</td>
<td>158</td>
</tr>
<tr>
<td>11.4</td>
<td>Discussion</td>
<td>160</td>
</tr>
<tr>
<td>12</td>
<td>PcapWT: Indexing Network Packet Traces</td>
<td>161</td>
</tr>
<tr>
<td>12.1</td>
<td>Motivation</td>
<td>161</td>
</tr>
<tr>
<td>12.2</td>
<td>Related Work</td>
<td>162</td>
</tr>
<tr>
<td>12.3</td>
<td>The Design of PcapWT</td>
<td>163</td>
</tr>
<tr>
<td>12.3.1</td>
<td>Building Index Data</td>
<td>163</td>
</tr>
<tr>
<td>12.3.2</td>
<td>Packet Lookup and Extraction</td>
<td>163</td>
</tr>
<tr>
<td>12.4</td>
<td>Performance Evaluation</td>
<td>166</td>
</tr>
<tr>
<td>12.4.1</td>
<td>Packet Traces Datasets</td>
<td>166</td>
</tr>
<tr>
<td>12.4.2</td>
<td>Baselines</td>
<td>168</td>
</tr>
<tr>
<td>12.4.3</td>
<td>Space</td>
<td>168</td>
</tr>
<tr>
<td>12.4.4</td>
<td>Multi-threaded File I/O</td>
<td>169</td>
</tr>
<tr>
<td>12.4.5</td>
<td>Packet Extraction</td>
<td>169</td>
</tr>
<tr>
<td>12.5</td>
<td>Discussion</td>
<td>171</td>
</tr>
</tbody>
</table>
V And Finally... 174

13 Conclusions and Future Work 175
  13.1 Summary of Contributions 175
  13.2 Future Work 176

Bibliography 177
# List of Tables

2.1 Encoding positive integers using different bit-aligned encoding techniques. . . . 11
2.2 Different configurations for Simple9 depending on the selector value \( b \) . . . . 12
2.3 List of compact tree operations . . . . . . . . . . . . . . . . . . . . . . . . 25

5.1 Ranked union results grouped by percentiles. . . . . . . . . . . . . . . . . . . 80
5.2 Ranked intersection results grouped by percentiles. . . . . . . . . . . . . . . 85

6.1 Comparison of practical results. . . . . . . . . . . . . . . . . . . . . . . . . 103

9.1 Collection Statistics for top-\( k \) document retrieval experiments . . . . . . . 120

10.1 Real datasets used, and space required to represent them. . . . . . . . . . . . 147

11.1 Space usage, in bytes, of the data structures used in our index . . . . . . . . . 157
11.2 Comparison of the space consumption of the SSA, CSA, and CST. . . . . . . 158

12.1 Description of four sample pcap files. . . . . . . . . . . . . . . . . . . . . . . 167
12.2 Comparison of the index data size. . . . . . . . . . . . . . . . . . . . . . . . 169
12.3 Packet filtering queries and specifications. . . . . . . . . . . . . . . . . . . . 172
# List of Figures

1.1 Outline of the thesis ................................................. 5  
2.1 Concatenation $C$ of our 3-document example collection $D$. ........... 8  
2.2 Huffman tree example ............................................... 10  
2.3 Example of RANK and SELECT operations on a binary sequence $S[1,11]$. 13  
2.4 Example of a wavelet tree ......................................... 18  
2.5 Huffman shaped wavelet tree of an example sequence $S = 1511863875743288$. 18  
2.6 Example of grid representation using a wavelet tree .................... 20  
2.7 Example of a DAC representation of integers .......................... 22  
2.8 Example of a $K^2$-tree ............................................ 22  
2.9 Example of compact representation of a tree using BP, LOUDS and DFUDS. 25  
2.10 Example of a differentially encoded search tree ...................... 26  
2.11 Example of a Cartesian tree to support RMQ over $S$ .................. 27  
2.12 Example of a treap ................................................ 28  
2.13 Example of a generalized suffix tree ................................ 29  
2.14 Suffix array of collection $C$ ....................................... 29  
2.15 Matrix of all the cyclic rotations of “ATA#TAAA#TATA#$” ............... 31  
2.16 Example of positional inverted index ................................ 34  
2.17 Example of a Document array and $C$-Array .......................... 35  
3.1 Example of Dualsorted inverted list .................................. 49  
4.1 An example of posting list representation using a treap. .................. 56  
4.2 Compact treap using BP topology ................................... 58  
4.3 Compact treap using LOUDS ......................................... 59  
4.4 Compact treap using $\text{HEAP}$ .................................... 62  
4.5 Separating frequencies below $f_0 = 2$ in our example treap. ........... 63  
4.6 The left-to-right construction of an example treap ..................... 72  
5.1 Total sizes of the indexes depending on the scoring scheme ............ 76  
5.2 Total sizes of the indexes depending on the scoring scheme ............ 76  
5.3 Sizes of the topology components depending on the scoring scheme .... 77  
5.4 Construction time of the indexes depending on the scoring scheme, in minutes. 77  
5.5 Ranked union times for distinct $k$ values, in milliseconds ............ 79  
5.6 Ranked union times grouped by number of terms per query using $t\text{-idf}$ 81  
5.7 Ranked union times grouped by number of terms per query using BM25. 82  
5.8 Ranked union times for distinct $k$ values as a function of the query length. 83
10.7 Space-time trade-offs of the compared range counting variants .......................... 150

11.1 Layout of our index organization for representing Web Access Logs .................. 155
11.2 Average microseconds to perform ACCESS and USER_PATH .................................. 159
11.3 Average microseconds to perform COUNT (P) and the MOST_FREQUENT ............... 159
11.4 Average microseconds to list distinct users and retrieve top-10 most common ............... 160

12.1 Extracting index data from a source packet trace file ........................................ 164
12.2 Mapping between the extracted index data and positive integers .......................... 164
12.3 Building wavelet tree chunks ............................................................................. 165
12.4 Listing the final matched packets ....................................................................... 166
12.5 Retrieving the packets using multi-threaded file I/O .......................................... 167
12.6 Comparison of the index size ............................................................................. 170
12.7 Comparison of the building time of index data .................................................... 170
12.8 Comparison of the packet lookup time for different sampling sizes ....................... 171
12.9 Comparison of packet extraction with various number of threads for parallel ............ 172
    file I/O ............................................................................................................. 172
12.10 A performance comparison of packet extraction using RT-2/3 .............................. 173
List of Algorithms

<table>
<thead>
<tr>
<th></th>
<th>Basic wavelet tree algorithms</th>
<th>17</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Method RANGE_REPORT on a wavelet tree</td>
<td>21</td>
</tr>
<tr>
<td>3</td>
<td>Method K²_RANGE on a K²-tree.</td>
<td>23</td>
</tr>
<tr>
<td>4</td>
<td>Method BACKWARD_SEARCH on the SSA</td>
<td>32</td>
</tr>
<tr>
<td>5</td>
<td>Method LOCATE on the SSA</td>
<td>32</td>
</tr>
<tr>
<td>6</td>
<td>Method DOCUMENT_LISTING returns the distinct documents</td>
<td>35</td>
</tr>
<tr>
<td>7</td>
<td>Method SVS</td>
<td>42</td>
</tr>
<tr>
<td>8</td>
<td>Method WAND</td>
<td>44</td>
</tr>
<tr>
<td>9</td>
<td>Method EVALUATE</td>
<td>45</td>
</tr>
<tr>
<td>10</td>
<td>Method TRY_REPORT</td>
<td>45</td>
</tr>
<tr>
<td>11</td>
<td>Method BLOCK-MAX-WAND</td>
<td>47</td>
</tr>
<tr>
<td>12</td>
<td>Method RANGE_INTERSECT on a wavelet tree</td>
<td>51</td>
</tr>
<tr>
<td>13</td>
<td>Method TREAP_INTERSECT using treaps.</td>
<td>65</td>
</tr>
<tr>
<td>14</td>
<td>Method REPORT using treaps.</td>
<td>66</td>
</tr>
<tr>
<td>15</td>
<td>Method CHANGEV using treaps.</td>
<td>66</td>
</tr>
<tr>
<td>16</td>
<td>Method CHANGED using treaps.</td>
<td>66</td>
</tr>
<tr>
<td>17</td>
<td>Method TREAP_UNION using treaps.</td>
<td>68</td>
</tr>
<tr>
<td>18</td>
<td>Method CHANGED_UNION using treaps.</td>
<td>69</td>
</tr>
<tr>
<td>19</td>
<td>Method WT_GREEDY returns the top-k most frequent symbols</td>
<td>101</td>
</tr>
<tr>
<td>20</td>
<td>Top-k algorithm using WRMQ index</td>
<td>113</td>
</tr>
<tr>
<td>21</td>
<td>Top-k algorithm using H-mapping index</td>
<td>116</td>
</tr>
</tbody>
</table>