

# Contents

<b>1</b>	<b>Introduction</b>	1
<b>2</b>	<b>Basic Concepts</b>	5
2.1	Models of Computation	5
2.1.1	RAM Model	5
2.1.2	External-Memory Model	6
2.2	Notation	6
2.3	Classical Full-Text Indexes	7
2.3.1	Suffix Tree	7
2.3.2	Suffix Array	9
2.4	Compression	10
2.4.1	Empirical Entropy	11
2.4.2	Burrows-Wheeler Transform	12
<b>3</b>	<b>Optimally Partitioning a Text to Improve Its Compression</b>	15
3.1	The PPC-Paradigm and Motivations for Optimal Partitioning	16
3.2	The Problem and our Solution	19
3.2.1	A Pruning Strategy	20
3.2.2	Space and Time Efficient Algorithms for Generating $\mathcal{G}_\epsilon(T)$	21
3.3	On Zero-th Order Compressors	23
3.3.1	On Optimal Partition and Booster	26
3.4	On $k$ -th Order Compressors	27
3.5	On BWT-Based Compressors	28
<b>4</b>	<b>Bit-Complexity of Lempel-Ziv Compression</b>	33
4.1	Notation and Terminology	36
4.2	An Efficient and Bit-Optimal Greedy Parsing	37
4.3	On the Bit-Efficiency of the Greedy LZ77-Parsing	40
4.4	On Bit-Optimal Parsings and Shortest-Path Problems	42
4.4.1	An Useful and Small Subgraph of $\mathcal{G}(T)$	42
4.4.2	An Efficient Bit-Optimal Parser	45
4.4.3	On the Optimal Construction of $\mathcal{T}_B$	51
4.5	An Experimental Support to our Theoretical Findings	52

- 5 Fast Random Access on Compressed Data . . . . . 55**
  - 5.1 Our Storage Scheme for Strings . . . . . 56
  - 5.2 Huffman on Blocks of Symbols . . . . . 58
  - 5.3 BWT Compression and Access . . . . . 59
  
- 6 Experiments on Compressed Full-Text Indexing . . . . . 61**
  - 6.1 Basics and Background . . . . . 64
    - 6.1.1 Backward Search . . . . . 64
    - 6.1.2 Rank Query . . . . . 65
  - 6.2 Compressed Indexes . . . . . 66
    - 6.2.1 The FM-index Family . . . . . 66
    - 6.2.2 Implementing the FM-index . . . . . 68
    - 6.2.3 The Compressed Suffix Array . . . . . 69
    - 6.2.4 The Lempel-Ziv Index . . . . . 71
    - 6.2.5 Novel Implementations . . . . . 72
  - 6.3 The Pizza and Chili Site . . . . . 74
    - 6.3.1 Indexes . . . . . 75
    - 6.3.2 Texts . . . . . 76
  - 6.4 Experimental Results . . . . . 78
    - 6.4.1 Construction . . . . . 79
    - 6.4.2 Counting . . . . . 80
    - 6.4.3 Locate . . . . . 81
    - 6.4.4 Extract . . . . . 84
    - 6.4.5 Final Comparison . . . . . 87
  
- 7 Dictionary Indexes . . . . . 89**
  - 7.1 Background . . . . . 91
  - 7.2 Compressed Permuterm Index . . . . . 92
    - 7.2.1 A Simple, but Inefficient Solution . . . . . 93
    - 7.2.2 A Simple and Efficient Solution . . . . . 93
  - 7.3 Dynamic Compressed Permuterm Index . . . . . 98
    - 7.3.1 Deleting One Dictionary String . . . . . 99
    - 7.3.2 Inserting One Dictionary String . . . . . 100
  - 7.4 Experimental Results . . . . . 102
  - 7.5 Further Considerations . . . . . 106
  
- 8 Future Directions of Research . . . . . 107**
  
- Bibliography . . . . . 111**



<http://www.springer.com/978-94-6239-032-4>

Compressed Data Structures for Strings  
On Searching and Extracting Strings from Compressed  
Textual Data

Venturini, R.

2014, XIV, 118 p. 18 illus., Hardcover

ISBN: 978-94-6239-032-4

A product of Atlantis Press