

Efficiency - Compression

(CS429)
Nazli Goharian
nazli@ir.iit.edu

Slides are *mostly* based on Information Retrieval Algorithms and Heuristics, Grossman, Frieder

© Goharian, Grossman, Frieder, 2002, 2009

1

Efficiency Techniques

- Indexing
- **Compression**
- Index Pruning (Top Doc)
- Efficient Query Processing
- Duplicate Document Detection

© Goharian, Grossman, Frieder, 2002, 2009

2

Facts

- Index (specially position index) may be similar size or larger than collection.
- Stop words removal eliminates about half the size of an inverted index. “the” occurs in 7 percent of English text.
- Half of terms occur only once (*hapax legomena*), so they only have one entry in their posting list
- Some terms have very long posting lists.
- Search engine must manage efficiently the memory hierarchy
- Approaches:
 - Stop word removal, stemming, case folding (*lossy*)
 - *Loss-less* Compression

© Goharian, Grossman, Frieder, 2002, 2009

3

Compression: Goal

- Reducing the storage requirements
- Reducing I/O
- Reducing disk seek time
- Storing more data in memory cache, and expedite query processing
- Compression ratio -- depending on CPU speed and memory throughput
- Efficiency of decompression algorithm is important!

© Goharian, Grossman, Frieder, 2002, 2009

4

Things to Compress

- Lexicon
 - In some applications fits in memory
 - In many it does not. Thus, may compress to fit in the memory to support query throughput
- Posting List
 - Term Frequency in each posting list entry
 - Document Identifier in each posting list entry

Lexicon Compression

- Storing terms in lexicon as a string with pointers indicating end of a term and start of the next term.
- Grouping strings of terms
- Hashing on terms

Delta Encoding

- Assumption: smaller numbers are more likely than large numbers! Thus: change the numbers to smaller numbers!
- Applied to posting lists
 - term: $(d_1, tf_1), (d_2, tf_2), \dots (d_n, tf_n)$
- Documents are ordered, so each d_i is replaced by the interval difference (*d-gaps*), namely, $d_i - d_{i-1}$
- Numbers are encoded using fewer bits for smaller, common numbers.
- Smaller *d-gaps* for more common terms.
- Index is reduced to 10-15% of database size.

© Goharian, Grossman, Frieder, 2002, 2009

7

Compression Techniques of Inverted Indices

- Fixed Length
 - Byte Aligned
- Variable Length
 - Elias Encoding (γ), a family of universal codes
 - Bernoulli
 - ...

© Goharian, Grossman, Frieder, 2002, 2009

8

Byte-Aligned Compression

- Done within byte boundaries to improve Run-time at slight cost to compression ratio.
- Each number is represented by fixed number of bytes, from which 2 bits are length indicators.
- Compression ratio of 15% of uncompressed inverted index, when stop words are used.

Byte-Aligned Compression

- Algorithm:
 - Take doc id differences (*d-gaps*)
 - Identify number of bytes needed for each *d-gap*.
 - Write length indicator for each *d-gap* in preceding 2 bits.
 - Write the binary representation of *d-gaps*.

Byte-Aligned Compression

0 - 63	00xxxxxxx
64 - (16K-1)	01xxxxxxx xxxxxxxx
16K - (4M-1)	10xxxxxxx xxxxxxxx xxxxxxxx
4M - (1G-1)	11xxxxxxx xxxxxxxx xxxxxxxx xxxxxxxx
0	00000000
1	00000001
...	...
63	00111111
64	01000000 01000000
65	01000000 01000001

- The hope here is that the document distance between posting list nodes will be small.

Gamma (Elias) Encoding (γ)

<u>X</u>	<u>γ</u>	<u>To represent value X:</u>
1	0	<ul style="list-style-type: none"> • $\lfloor \log_2 x \rfloor$ ones representing the highest power of 2 not exceeding X. • a 0 marker. • $\lfloor \log_2 x \rfloor$ bits representing the remainder $x - 2^{\lfloor \log_2 x \rfloor}$ in binary. • Uses $2\lfloor \log_2 x \rfloor + 1$ bits to represent value x. The smaller the integer, the fewer the bits used to represent the value. Most <i>tf</i>'s are relatively small.
2	10 0	
3	10 1	
4	110 00	
5	110 01	
6	110 10	
7	110 11	
8	1110 000	
63	111110 11111	

Gamma (Elias) Encoding (γ) Example

$$X = 22$$

$$\lfloor \log_2 22 \rfloor = 4 \quad 2^4 \leq x < 2^5$$

4 is highest power of 2 not exceeding 22 \Rightarrow 4 bits unary: 1111

$$x - 2^{\lfloor \log_2 x \rfloor} = 22 - 2^4 = 6$$

\Rightarrow 4 bits binary to represent the remaining number 6: 0110

1111 0 0110

4 bits unary for 16 0 marker 4 bits binary for 6

- Decompression is in one pass.

Reordering Documents Prior to Indexing

- Reduce doc id gap for better compression
- Similar documents contain similar terms
- Thus, find similar documents and process in that order d_3, d_{50}, d_{200} will be d_1, d_2, d_3
- Methods to measure document similarity:
 - Top-Down
 - Bottom-Up

Bottom-Up

- Each document in collection is grouped together progressively based on their similarity:
- *k-means* or variations used:
 - *Choose k documents as k clusters (centroids)*
 - *Assign remaining documents to k clusters*
 - *Recalculate centroids*
 - *Repeat iteration till centroids stabilize*

Top-Down - *Bisecting*

- Four phases:
 - **P1: Center selection:** select 2 documents
 - **P2: Redistribution:** assign remaining documents
 - **P3: Recursion:** repeat P1 & P2 over the 2 groups, till each become a singleton
 - **P4: Merging:** partitions formed from recursive call are merged bottom-up

Compression Summary

- Pro
 - Reducing the storage requirements of inverted index
 - Reducing I/O for querying the inverted index
 - Reducing disk seek time
 - Store more data in memory cache, and expedite processing
- Con
 - Takes longer to build the inverted index.
 - Software becomes *much* more complicated.
 - Uncompress required at query time – note that this time is usually offset by dramatic reduction in I/O.