

COMP90042 Web Search & Text Analysis

Workshop Week 3

Zenan Zhai

March 19, 2019

University of Melbourne

Indexing

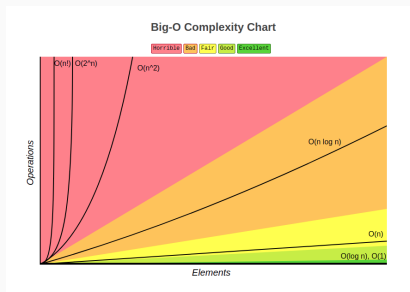
- Data Structure
 - Document-Term Matrix
 - Inverted Index
- Compression
 - Variable Byte Compression
 - OptPFor Delta Compression
- Index Construction
 - Invert Batch Indexing
 - Auxiliary Indexing
 - Logarithmic Indexing

Search

- Vector Space Models
 - TF-IDF
 - BM25
- Efficient Query Processing
 - Operation GEQ
 - WAND
- Query Completion
 - Prefix Trie
 - Range Maximum Query
- Query Expansion
 - Relevance Feedback
 - Semantic-Based Methods
- Phrase Search
 - Inverted Index + Positional Information
 - Suffix Array

Warm Up - Complexity

Time Complexity



Big O cheat sheet

Notation

- $T(n) = O(f(n)) \Leftrightarrow \exists c, n_0, \forall n > n_0, T(n) \leq c \cdot O(f(n))$
- $T(n) = \Omega(f(n)) \Leftrightarrow \exists c, n_0, \forall n > n_0, T(n) \geq c \cdot \Omega(f(n))$
- $T(n) = \Theta(f(n)) \Leftrightarrow \exists c_1, c_2, n_0, \forall n > n_0,$
 $c_1 \cdot \Theta(f(n)) \leq T(n) \leq c_2 \cdot \Theta(f(n))$

Warm Up - Complexity

Space Complexity

- Amount of auxiliary space the algorithm need in the function of input size.

Example - Merge Sort

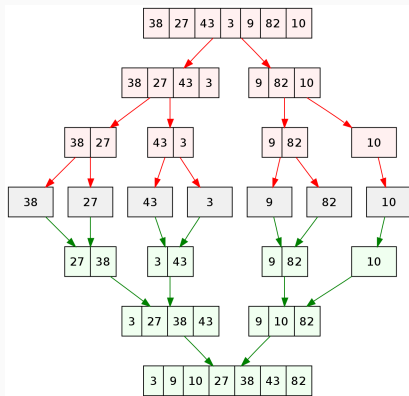


Fig from Wikipedia.

- Index Compression
 - Variable Byte Compression
 - OptPFor Delta Compression
- Index Construction
 - Invert Batch Indexing
 - Auxiliary Indexing
 - Logarithmic Indexing
- Efficient Query Processing
 - Operation GEQ
 - WAND

Variable Byte Compression

1Byte = 8bits

Biggest integer can be stored in 64/32 bits.

- Max. 64-bits unsigned integer
 $2^{64} - 1 = 18,446,744,073,709,551,615$
- Max. 32-bits unsigned integer
 $2^{32} - 1 = 4,294,967,295$

In practice, 95% of integers in posting lists are < 128
(Can be stored in 7 bits).

Variable Byte Compression

Compression of integer 315700

- Divide the integer by 128 (2^7).
- Add "indicator bit" to the residual.
1 indicates end of encoded number.

315700 \Rightarrow 0010011|0100010|0110100

Indicator	Remainder	Decimal
0	0110100	52
0	0100010	34
1	0010011	147

Decompression of integer 315700

- Multiply the remainder by $2^{7 \times (i-1)}$ for the i^{th} chunk.
- Take sum of all chunks.

$$52 \times 2^0 + 34 \times 2^7 + (147 - 128) \times 2^{14} = 315700$$

More flexible chunk size b than VByte compression

- Encode most numbers in a block by b bits
- Leave exception unchanged.
- Record number and positions of exceptions in header.

Encode [1 4 7 2 4 5 123 48] with $b = 3$

Header	Content	Exceptions
$b=3, \#e=2, \text{epos}=[6, 7]$	[1, 4, 7, 2, 4, 5]	[123, 48]

Parameter b need to be tuned to get optimal performance.

- Index Compression
 - Variable Byte Compression
 - OptPFor Delta Compression
- Index Construction
 - Static Construction
 - Auxiliary Indexing
 - Logarithmic Indexing
- Efficient Query Processing
 - Operation GEQ
 - WAND

Static Construction

Traverse once to get global vocabulary.

{1:Apple, 2:Pear, 3:Banana, 4:Grape}

Split document collection in to batches and compute inverted indexes in parallel, then merge terms with same ID.

Apple $\rightarrow d_1, d_2$

Pear $\rightarrow d_2$

Banana $\rightarrow d_1$

+

Apple $\rightarrow d_3$

Pear $\rightarrow d_3, d_4$

Grape $\rightarrow d_4$

\Downarrow

Apple $\rightarrow d_1, d_2, d_3$

Pear $\rightarrow d_2, d_3, d_4$

Banana $\rightarrow d_1$

Grape $\rightarrow d_4$

Auxiliary Indexing

Combination of static and incremental indexing

- 1 static index on disk, 1 incremental index on memory.
- Merge when incremental index is too big.
- Query both indexes and merge results

Note that merge requires number of IOs equals to the size of the posting lists

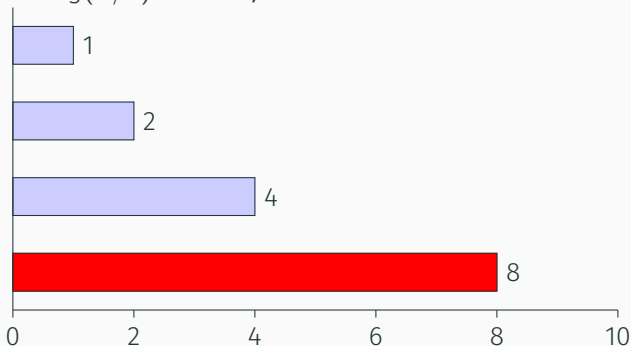
Suppose now we have N postings on disk and we merged the postings when it reaches size n .

$$T(N, n) = \sum_{i=1}^{\frac{N}{n}} i \times n = \frac{(1 + \frac{N}{n})}{2} n \times \frac{N}{n} = \frac{N}{2} + \frac{N^2}{2n} (n < N)$$

$$T(N, n) = O\left(\frac{N^2}{n}\right)$$

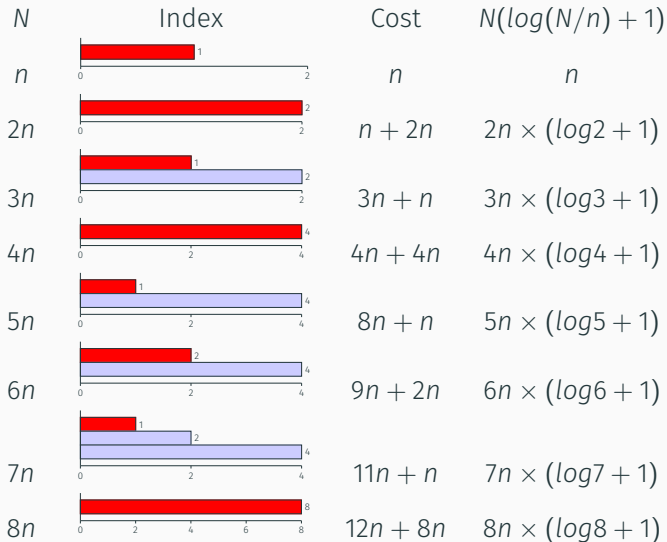
Logarithmic Indexing

Use $\log(N/n)$ indexes, each level i has size $2^i \times n$.

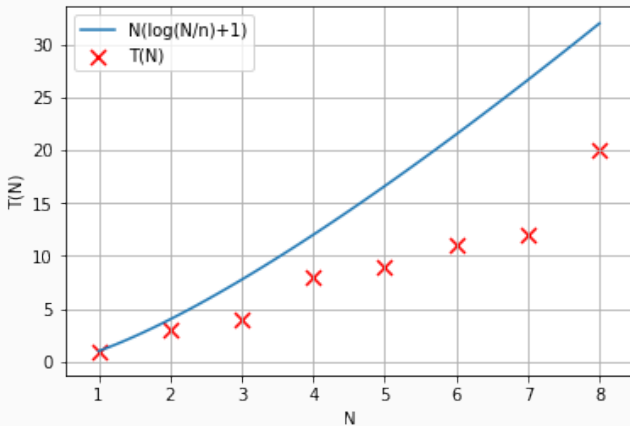


Consider total number of merges needed when $N = 8n$.

Logarithmic Indexing



Logarithmic Indexing



- Index Compression
 - Variable Byte Compression
 - OptPFor Delta Compression
- Index Construction
 - Static Construction
 - Auxiliary Indexing
 - Logarithmic Indexing
- Efficient Query Processing
 - Operation GEQ
 - WAND

GEQ - Greater or equal to x .

2, 4, 7, 9, 13, 20, 34, 54, 67, 87, 96, 103

Binary search in sorted list.

- Time complexity ?
- Space complexity ?

Compressed posting lists with sample value

[2, 4, 7, 9] [13, 20, 34, 54] [67, 87, 96, 103]

max=9

max=54

max=103

$$W_{d,t} = \frac{(k_1 + 1)tf_{d,t}}{k_1((1 - b) + b(\frac{L}{L_{avg}})) + tf_{d,t}} \times \log \frac{N - df_t + 0.5}{df_t + 0.5} \times \frac{(k_3 + 1)tf_{Q,t}}{k_3 + tf_{Q,t}}$$

$$Score(Q, d) = \sum_{q \in Q} W_{d,q}$$

Question: $T(Score(Q, d))$?

Concepts

- Top- k retrieval
- Maximum Contribution

Questions

- Why do we want to sort lists by their current pivot?
- Why do we still need to compute similarity score if Max. contribution of a doc is greater than Min. score in top- k list?
- How does operation GEQ helps WAND?

Benefits and Restrictions

Exercise - What does the WAND do?

Query Q: The quick brown fox

with $k = 2$

