Document/Page ranking using tf-idf Weighting Scheme

CSE 633(SPRING 2020)
Instructor: Dr. Russ Miller

Name: Mansi Shetty
Overview

- Terminologies and Definitions
- Problem Statement
- Applications
- Sequential Algorithm
- Parallel Implementation
- Results
- Observations
- References
Terminologies and Definitions

Document
A document is a collection of terms/words. Examples: Web page, tweet

Corpus
It is a collection of documents

Term Frequency (TF)
It is a document weighting scheme that takes into account the number of occurrences of the term t in document d.

Inverse Document Frequency (IDF)
It is defined as, idf(t) = \log(N/df)

where, N is the total number of documents in the corpus and df is the document frequency which indicates the number of documents in the corpus that contain the term t.
Problem Statement

Rank documents/search results based on Term Frequency – Inverse Document Frequency, a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus.

\[
\text{tf-idf}_{t,d} = \text{tf}_{t,d} \times \text{idf}_t
\]

\[
\text{Score}(q, d) = \sum_{t \in q} \text{tf-idf}_{t,d}
\]
Example

- Lets consider a corpus with two documents i.e. N=2

- For document 1 (d1),
  \[
  \begin{align*}
  t_{f, this, d1} & = \frac{1}{5} , \quad t_{f, is, d1} = \frac{1}{5} , \quad t_{f, a, d1} = \frac{2}{5} , \quad t_{f, sample, d1} = \frac{1}{5} \\
  i_{d, this} & = \log(2/2) = 0 \\
  i_{d, is} & = \log(2/2) = 0 \\
  i_{d, a} & = \log(2/1) = 0.301 \\
  i_{d, sample} & = \log(2/1) = 0.301
  \end{align*}
  \]

- For document 2 (d2),
  \[
  \begin{align*}
  t_{f, this, d2} & = \frac{1}{7} , \quad t_{f, is, d2} = \frac{1}{7} , \quad t_{f, another, d2} = \frac{2}{7} , \quad t_{f, example, d2} = \frac{3}{7} \\
  i_{d, this} & = \log(2/2) = 0 \\
  i_{d, is} & = \log(2/2) = 0 \\
  i_{d, another} & = \log(2/1) = 0.301 \\
  i_{d, example} & = \log(2/1) = 0.301
  \end{align*}
  \]

Therefore,

\[
\begin{align*}
  \text{tf-idf}_{\text{sample, d1}} &= \left(\frac{1}{5}\right) \times 0.301 = 0.0602 \\
  \text{tf-idf}_{\text{sample, d2}} &= 0 \times 0.301 = 0 \\
  \text{tf-idf}_{\text{example, d1}} &= 0 \times 0.301 = 0 \\
  \text{tf-idf}_{\text{example, d2}} &= \left(\frac{3}{7}\right) \times 0.301 = 0.129
  \end{align*}
\]
Applications

- Information retrieval
- Web search
- Keyword Extraction
- Stop words elimination

In applications like above, the time taken to return the results becomes the most important factor and given the amount of data that the internet has to offer today, computations takes a significantly longer time.
Sequential Algorithm

• Create an inverted index
  \{\text{term}: (\text{document frequency}, [\text{document list}])\}
• Read the query terms and create a list of resultant documents
• Calculate term frequency $tf_{t,d}$
• Calculate inverse document frequency $idf_t$
• Calculate tf-idf score for each document
• Sort the documents based on tf-idf score
Parallel Implementation

- Consider $N$ documents and $P$ processors.
- Assign $N/P$ documents to each processor.
- Each processor creates the inverted index for its $N/P$ documents independently.
- The file with the queries is read by each processor.
- Document frequencies of all the terms of the query are consolidated by recursive halving and broadcasting.
- Processors independently form the resultant set from their inverted index.
- Results in each of the processors are ranked according to tf-idf weighting scheme.
- Final result for each query is consolidated by recursive halving where the task is to merge two sorted lists.
RESULTS
Total number of articles: 6144

<table>
<thead>
<tr>
<th>Number of Processors</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>24.598</td>
</tr>
<tr>
<td>4</td>
<td>11.837</td>
</tr>
<tr>
<td>8</td>
<td>6.024</td>
</tr>
<tr>
<td>16</td>
<td>3.288</td>
</tr>
<tr>
<td>32</td>
<td>1.735</td>
</tr>
<tr>
<td>64</td>
<td>1.120</td>
</tr>
<tr>
<td>128</td>
<td>0.771</td>
</tr>
<tr>
<td>256</td>
<td>0.632</td>
</tr>
<tr>
<td>512</td>
<td>0.786</td>
</tr>
</tbody>
</table>
Total number of articles: 12288

<table>
<thead>
<tr>
<th>Number of Processors</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>56.093</td>
</tr>
<tr>
<td>4</td>
<td>27.075</td>
</tr>
<tr>
<td>8</td>
<td>13.719</td>
</tr>
<tr>
<td>16</td>
<td>6.757</td>
</tr>
<tr>
<td>32</td>
<td>3.330</td>
</tr>
<tr>
<td>64</td>
<td>1.852</td>
</tr>
<tr>
<td>128</td>
<td>1.126</td>
</tr>
<tr>
<td>256</td>
<td>0.739</td>
</tr>
</tbody>
</table>
Total number of articles: 24576

<table>
<thead>
<tr>
<th>Number of Processors</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>122.559</td>
</tr>
<tr>
<td>4</td>
<td>57.218</td>
</tr>
<tr>
<td>8</td>
<td>28.548</td>
</tr>
<tr>
<td>16</td>
<td>14.354</td>
</tr>
<tr>
<td>32</td>
<td>7.184</td>
</tr>
<tr>
<td>64</td>
<td>3.202</td>
</tr>
<tr>
<td>128</td>
<td>1.725</td>
</tr>
<tr>
<td>256</td>
<td>0.995</td>
</tr>
<tr>
<td>512</td>
<td>0.759</td>
</tr>
</tbody>
</table>
**Constant number of articles/processor (165 articles/processor)**

<table>
<thead>
<tr>
<th>Number of processors</th>
<th>Articles</th>
<th>Time(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>330</td>
<td>1.370</td>
</tr>
<tr>
<td>4</td>
<td>660</td>
<td>1.368</td>
</tr>
<tr>
<td>8</td>
<td>1320</td>
<td>1.459</td>
</tr>
<tr>
<td>16</td>
<td>2640</td>
<td>1.420</td>
</tr>
<tr>
<td>32</td>
<td>5280</td>
<td>1.454</td>
</tr>
<tr>
<td>64</td>
<td>10560</td>
<td>1.498</td>
</tr>
<tr>
<td>128</td>
<td>21120</td>
<td>1.562</td>
</tr>
<tr>
<td>256</td>
<td>42240</td>
<td>1.638</td>
</tr>
</tbody>
</table>

**Graph: Constant number of articles/processor**

![Graph showing the relationship between number of processors and time](image)
Observations

• For 6144 total articles, as we increase processors beyond 256, the total time increases as the communication overhead overpowers reduction in computation time.

• For 24576 total articles, there is not a significant decrease in total time when we increase the processors from 256 to 512.

• When we maintain a constant number of articles/processors, the computation time remains comparable with slight increase as we increase the number of processors.
References

- Algorithms Sequential & Parallel: A Unified Approach (Dr. Russ Miller, Dr.Laurence Boxer)


- https://en.wikipedia.org/wiki/Tf%E2%80%93idf


- https://www.kaggle.com/snapcrack/all-the-news
Thank You!