SEARCH ENGINE FOR GIVEN SET OF DOCUMENTS

ANIMESH ANAND

IIT(BHU), Varanasi

26/04/2017

Table Of Contents

What is Search Engine?

- In simplest terms, a search engine is any software program that searches for sites based on the words users have keyed in, that have to do with the subject of their interest.Search Engines essentially act as filters for the wealth of information available on the Internet.
- They allow users to quickly and easily find information that is of genuine interest or value to them, without the need to wade through numerous irrelevant web pages.

What is need of search engine.

- In our world, the information overload is challenging. Today we have huge amount of data on World Wide Web. To sort out and retrieve information for recent and future use is not an easy task.To get information from the billions of documents in efficient time and efficient manner is not so easy. Therefore, the need of a Search Engine is obvious.
- The goal of the Search Engines is to provide users with search results that lead to relevant information on high-quality websites. The operative word here is "relevant".

How to search in faster way

- Since if we will go through each documents to search for the query then obviously time to traverse through billions of documents available will be very high.
- So we build index of documents we have.

Building inverted index of documents

- To gain the speed benefits of indexing at retrieval time, we have to build the index in advance. The major steps in this are:
- 1. Collect all the documents in which we will look for the queries asked by the user.
- 2. Tokenize the text, turning each document into a list of tokens
- 3. Do linguistic preprocessing, producing a list of normalized tokens, which are the indexing terms
- 4. Index the documents that each term occurs in by creating an inverted index, consisting of a dictionary and postings
- We will make the dictionary of documents name as a key and a unique ID corresponding to it,i.e., we serialise the documents. We list the documents in a list also and the the document name will be present in list at the index of its respective serial no.
- The pseudo code for the steps 1-3 is shown below.

Building inverted index

```
import os
cur_dir=os.getcwd()
search_dir=cur_dir+"\search_files"
#search_dir::the directory in which the all documents are collected
Documents=[]
#list of wll have all the documents
docid={}
#Dictionary will have all the documents with a dcID
for files in os.walk(search_dir):
    Documents=files[2]
"""Now all the documents are listed"""
x=0
for i in l:
    docid[i]=x
    x+=1
"""
The dictioanary now contains all the documents with a ID
Notice the document is present in list at the same index as it's docID
"""
```

Figure: Pseudo Code

Building inverted index of documents

- Consider two Documents:
- **Doc 1 and Doc 2** as shown below and their corresponding index.

Building inverted index of documents

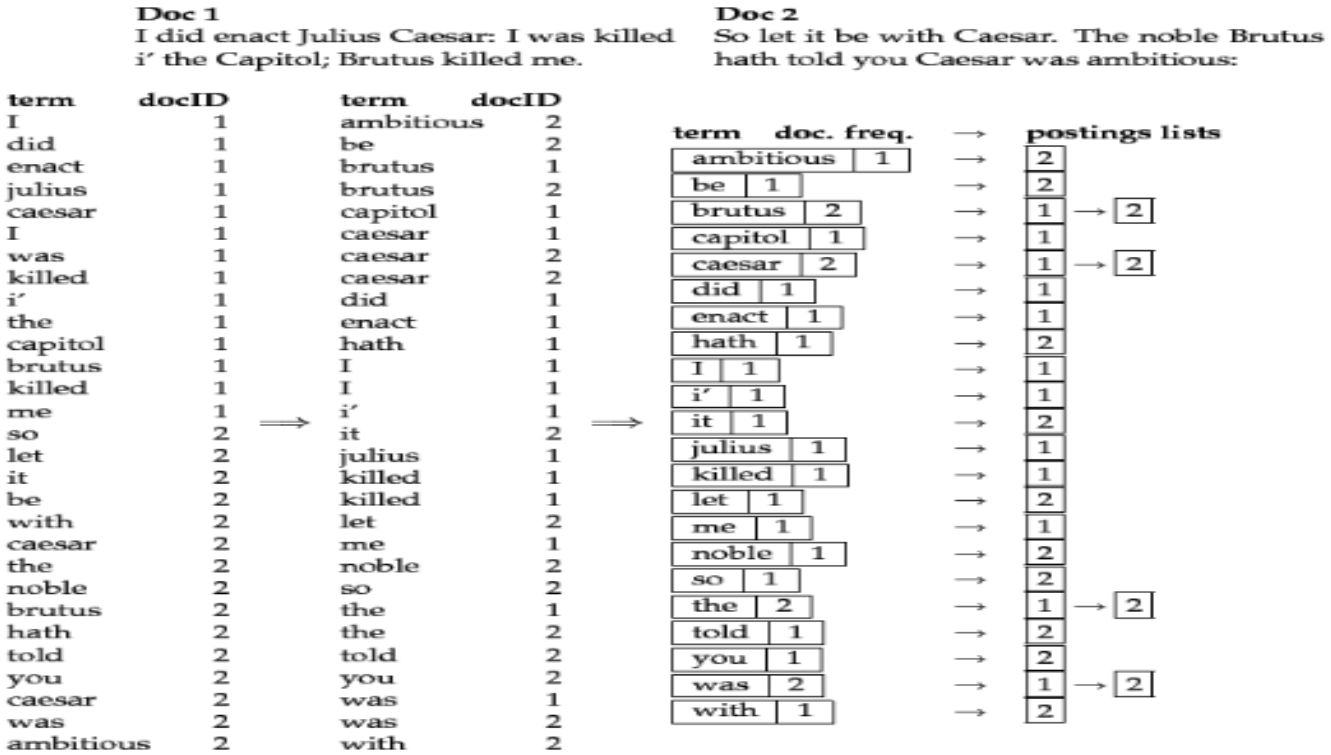

Figure: Original Image

SEARCHING FOR THE RESULTS AND RANKING IT

- When user searches for a query then we split it into words.We look into index for each word. Since each word is hashed as key in the *index* and contains list in which *docIDs* it is present.The pseudo code below shows how it search and its ranking is implemented.
- So the maximum numbers of words of the query the document will have, it will be ranked higher.

SEARCHING FOR THE RESULTS AND RANKING IT

```
from sets import Set
ignore=['is','am','are','in','on','to'] # words to be ignored
display=[] #the list will contain address of documents in ranked way
query=raw_input("Enter the query to be searched\n") #the query by user
query=list(Set(query.split()))
doclist={}
""" dictionary will contain name of document as key and how many
number of words it contains from the query """
for word in query:
    if word in ignore:
        continue
    if word in di:
        for i in di[word]:
            if i in doclist:
                doclist[i]+=1
            else:
                doclist[i]=1
for w in sorted(doclist, key=doclist.get, reverse=True):
  display.append(cur_dir+'\\search_files\\'+w)
"""now display contains doc names in sorted order of number of words
containing from the query.If no match is there,list will be empty.
"""
if display==[]:
    print "No results found"
```

Figure: Pseudo Code

Thank You

References I