Computer Science Department
CS632V – Introduction to Big Data Analytics
Spring 2017

Project #1

Let's build a **Basic Inverted Index** (aka Postings File or Inverted File).
An Inverted index is designed to allow very fast full-text searches.
An inverted index consists of a list of all the unique words that appear in any document, and for each word, a list of the documents in which it appears.
It is used in any website's document (html's) for text search and there are the main components of Apache Solr and EslasticSearch engines (both based on Apache Lucene library).

The main steps for building an Inverted Index are:

1) Collect the documents to be indexed.
2) Read the contents of each document.
   (Can you read each document's paragraphs in parallel? Extra point!)
3) Tokenize the text, turning each document into a list of tokens
   (Make things simpler, avoid Token Normalization – *'canonicalizing' tokens* – every word is unique)
   (Can you drop common terms – *stop words* – and be excluded? Extra point!)
4) Index the documents that each term occurs in by creating an inverted index, consisting of a dictionary and postings.

Write a **Python** script by utilizing the **MapReduce** paradigm to create such an Index.

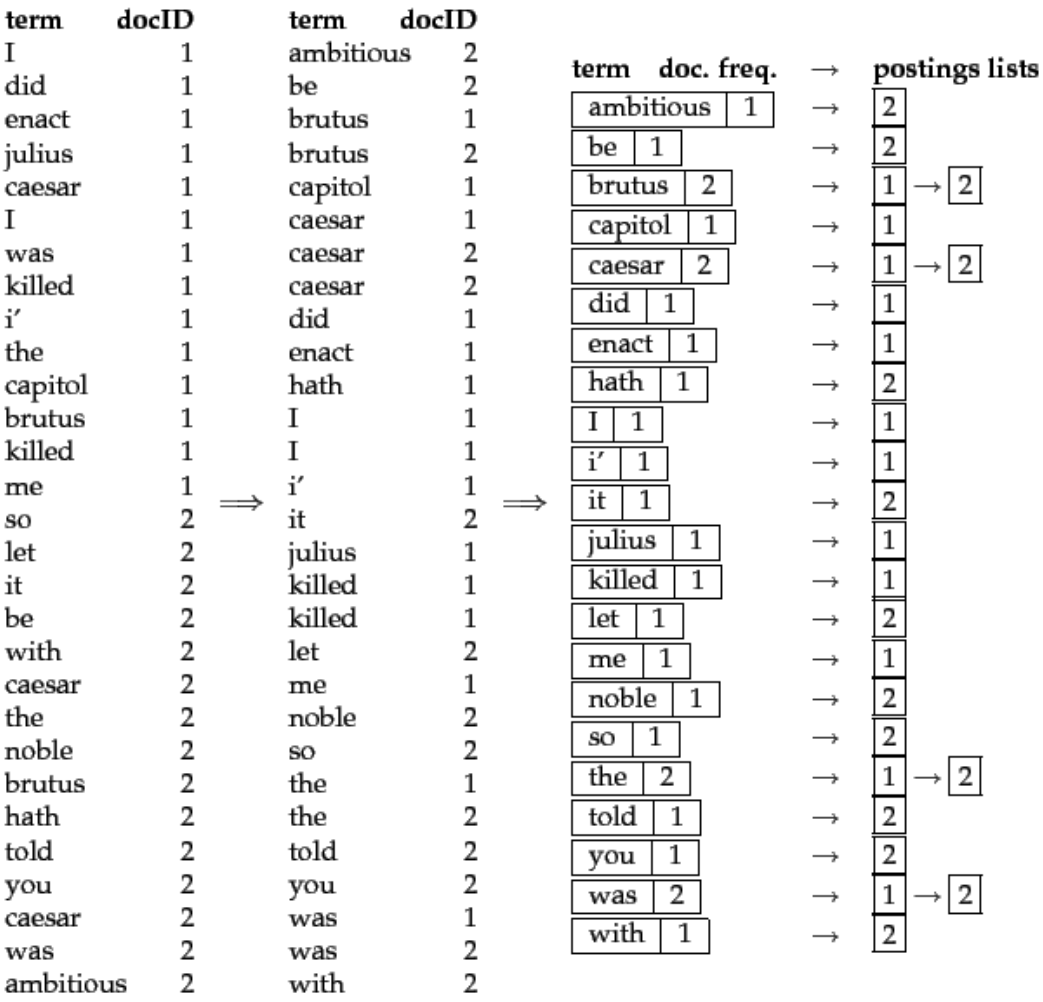
Example

Doc1
I did enact Julius CaesarL I was killed I' the Capitol; Brutus killed me.

Doc2
So let it be with Caesar. The noble Brutus hath told you Caesar was ambitious:

See page 2 details of the flow …

The index is being built by sorting and grouping.
The sequence of terms in each document, tagged by their documentID is sorted alphabetically.
Instances of the same term can then grouped by word and then by documentID.
The term and documentIDs are then separated out.
The dictionary stores the terms, and it has a pointer to the postings list for each term. It commonly also stores other summary information such as, the document frequency of each term.
This information is often used doe improving query time efficiency and for weighting in ranked retrieval models. Each postings list stores the list of documents in which a term occurs, and may store other information such as the term frequency (the frequency of each term in each document) or the positions of the term in each document.



For testing your code, draw the inverted index that would be built for the following document collection:

Doc1 new home sales top forecasts
Doc2 home sales rise in july
Doc3 increase in home sales in july
Doc4 july new home sales rise