"MAP-REDUCE" related operations

Overview

Chapter 1,2,3.1-3.4

Step 3: An example of "MAP-REDUCE" OPERATIONS

```
Doc1  : docID=51:   algorithm data structure alex
Doc2  : docID=22:   algoritms data alex structure
Doc3  : docID=13:   structures alex data alex

Forward index is a triplet  (docID, wordID, wordoffset ) or a
                quadruplet  (docID, wordID, wordoffset, context )

            STAGE 0: Build forward index (in this simple example, a triplet).

            N : corpus size
            dj  : j-th document of corpus is dj
            ti  : i-th index term is ti
            ni  : number of distinct documents containing ti
            tfi : number of occurrences of ti in all documents (duplicates make tfi >= ni)
            tfij: number of occurrences of ti in dj only!
            idfi: lg(N/ni)
Forward Index                       MAP REDUCE SCALING: Millions of files per 'server'
(51,1,1)                                                Thousands of 'servers'
(51,2,2)
(51,3,3)                            M-R Question 1: How many docIDs?                ( N   )
(51,4,4)                              Answer: Scan top-to-bottom record changes of docID..
(22,1,1)
(22,2,2)                            M-R Question 2: How many words per server?      ( wj  )
(22,4,3)                              Answer: Count tuples (minus sentinel)
(22,3,4)
(13,3,1)                            M-R Question 3: How many words per docID        ( sum tfij over i)
(13,4,2)                              Answer: For given docID count tuples
(13,2,3)
(13,4,4)
( -,-,-)  'sentinel record'
```


Step 3: An example of "MAP-REDUCE" OPERATIONS

```
Doc1  : docID=51:   algorithm data structure alex
Doc2  : docID=22:   algoritms data alex structure
Doc3  : docID=13:   structures alex data alex

Forward index is a triplet  (docID, wordID, wordoffset ) or a
                qudruplet  (docID, wordID, wordoffset, context )

          STAGE 1: Invert Forward Index to obtain (inverted) index
                   Inversion means sort by wordID,docID,offset

Forward Index                  Inverted Index
(51,1,1)                      (51,1,1)      MAP REDUCE SCALING: Global sorting
(51,2,2)                      (22,1,1)                          across all servers
(51,3,3)                      (51,2,2)
(51,4,4)                      (22,2,2)     M-R Question 4: How many different words?   ( i )
(22,1,1)  - Sort by ---->     (13,2,3)       Answer: Scan top-to-bottom record changes
(22,2,2)     wordID           (51,3,3)               in wordID
(22,4,3)     docID            (22,3,4)
(22,3,4)     offset           (13,3,1)     M-R Question 5: How many docs per word?     (ni)
(13,3,1)                      (51,4,4)       Answer: For a given wordID count tuples
(13,4,2)                      (22,4,3)               with nonduplicate docIDs
(13,2,3)                      (13,4,2)
(13,4,4)                      (13,4,4)     M-R Question 6: How many instance of word?  ( sum tfij over j)
(-, -,-)                      (- ,-,-)       Answer: For a given wordID count tuples
```


Step 3: An example

```
             STAGE 2: The (inverted) index
                                    Inverted Index
                    (51,1,1)       1 :  (51,-,1) (22,-,1)
                    (22,1,1)       2 :  (51,-,2) (22,-,2) (13,-,3)
                    (51,2,2)       3 :  (51,-,3) (22,-,4) (13,-,1)
                    (22,2,2)       4 :  (51,-,4) (22,-,3) (13,-,2) (13,-,4)
                    (13,2,3) --->
                    (51,3,3)
                    (22,3,4)
                    (13,3,1)
                    (51,4,4)
                    (22,4,3)
                    (13,4,2)
                    (13,4,4)
 DocList : 51
********** 22
           13
 Vocabulary                                    Lexicon
 #word      #wordID  #hash? #other              #word    #wordID pointer-to-vocabulary
0 alex       4                                  data      2      2
1 algorithm  1                                  algorithm 1      1
2 data       2                                  structure 3      3
3 structure  3                                  alex      4      0
```


Step 3: Google's Barrels

```
Doc1  : docID=51:   algorithm data structure alex gerbessiotis computational geometry
Doc2  : docID=22:   algoritms data alex structure
Doc3  : docID=13:   structures alex data alex engineering design

        Google's Barrels : To avoid global 'sorting' that moves tuples around
                           when the forward index is generated it
               (a) does not group together all tuples of a given docID
               but
               (b) depending on wordID it

           Example:  This 'Barrel' stores only wordIDs 1,2,3,4  of any docID
                   i.e. tuples (?,1,?) or (?,2,?) or (?,3,?) or (?,4,?)

           Therefore sorting by wordID will be local sorting in that barrel only !

Forward Index
(51,1,1)
(51,2,2)
(51,3,3)
(51,4,4)
(22,1,1)
(22,2,2)
(22,4,3)
(22,3,4)
(13,3,1)
(13,4,2)
(13,2,3)
(13,4,4)
```

