Authors: Xihao Jiang

Algorithm Design

```
1: class MAPPER
2:     method MAP1(string s)
3:         word list = s.split()
4:         for all i = 2 → word list.length do // word list[1] is the document id of each line
5:         EMIT(pair (word list[1], word list[i]), integer 1)
6:     method MAP2(pair(docId, token), integer tf)
7:         EMIT(string token,integer 1)
8:         method MAP3(stringtoken, integer sum)
9:         EMIT(string token,float log10( T sum ))
10:    method MAP4(string token, pair (pair (string docId, float tf),float idf))
11:        EMIT(string docId, pair (string token, integer tf ∗ idf))
12:    method MAP5(string docId, pair (string token, integer tf ∗ idf))
13:        a ← BUILD ARRAY (token)
14:        EMIT(string docId, array a)
15:    method MAP6(string docId, words [w1, w2, ...])
16:        count ← words.length
17:        EMIT(string docId, integer count)
18:    method MAP7(string docId, pair (pair (string token, integer tc),integer count))
19:        tf ← tc count
20:        EMIT(string token, pair (string docId, float tf))
21: class REDUCER
22:    method REDUCE1(pairs (docId, token), integers [r1, r2, ..])
23:        sum ← 0
```

24: initialize vocabulary count T

25: for all integer r ∈ integers [r1, r2, ...] do

26: sum ← sum + r

27: tf ← sum/T

28: EMIT(pair(docId, token), integer tf)

29: method REDUCE2(token,integers [r1, r2, ..])

30: sum ← 0

31: for all integer r ∈ integers [r1, r2, ...] do

32: sum ← sum + r

33: EMIT(string token, integer sum)

34: class JOINER

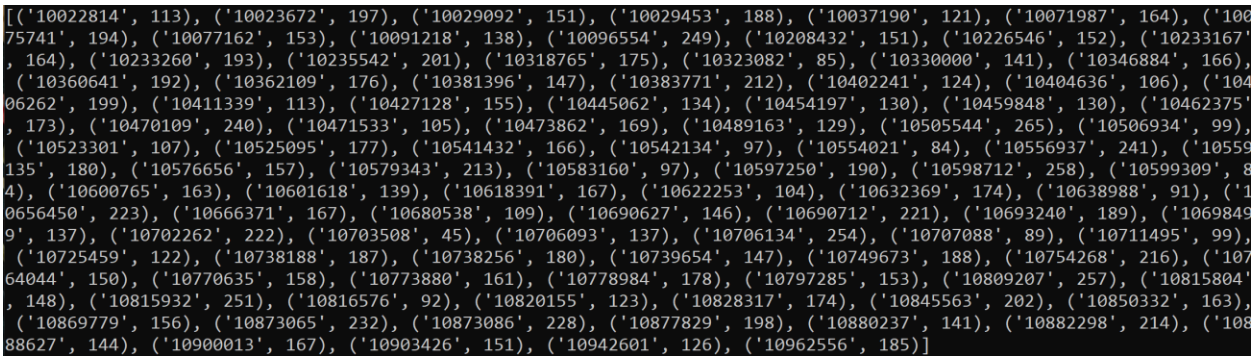
35: method JOIN1(pair(string token, pair (string docId, float tf)), pair(string token, float idf))

36: EMIT(string token, pair (pair (string docId, float tf), float idf))

37: method JOIN2(pair(string docId, pair (string token, integer tc)), pair(string docId, integer count))

38: EMIT(string docId, pair (pair (string token, integer tc), integer count))

Results



```
[0.00764713 0.00199598 0.00169936 ... 0.         0.         0.        ]
[0.00198793 0.00291865 0.00795174 ... 0.         0.         0.        ]
[0.00250858 0.00147322 0.00501717 ... 0.         0.         0.        ]
...
[0.00381741 0.00089674 0.00305393 ... 0.         0.         0.        ]
[0.00108619 0.00446524 0.00434476 ... 0.         0.         0.        ]
[0.00060552 0.         0.0060552  ... 0.         0.         0.        ]]
```

```
[('expression', 'cell'), ('expression', 'nuclear'), ('expression', 'much'), ('expression', 'antisense-cdna'), ('expressi
on', 'tpa'), ('expression', 'resection'), ('expression', '18'), ('expression', 'gradual'), ('expression', 'patient'), ('
expression', 'concept'), ('expression', 'dis_adenocarcinoma_dis'), ('expression', 'peptide'), ('expression', 'catalyz'),
 ('expression', 'electron-microscopic'), ('expression', 'egf-pdgf'), ('expression', 'stromelysins'), ('expression', '8%'
), ('expression', 'thereby'), ('expression', 'wide'), ('expression', 'bladder'), ('expression', 'correspond'), ('express
ion', 'gamma'), ('expression', 'subclone'), ('expression', 'doxorubicin'), ('expression', 'bt474'), ('expression', 'inte
rferon-gamma'), ('expression', 'phosphopeptide'), ('expression', 'defective'), ('expression', 'dis_pkc_dis'), ('expressi
on', 'apoptotical'), ('expression', 'impairment'), ('expression', 'asthmatic'), ('expression', 'n=10'), ('expression', '
dis_carcinogenesis_tumor_dis'), ('expression', '8-fold'), ('expression', 'subcutaneou'), ('expression', '111in-dtpa-hegf
'), ('expression', 'atp'), ('expression', 'x'), ('expression', 'dis_astrocytoma_dis'), ('expression', 'tgf-alpha)-induce
d'), ('expression', 'jnk2'), ('expression', 'stability'), ('expression', 'keratin'), ('expression', 'require'), ('expres
sion', 'il-1beta'), ('expression', 'growth-stimulating'), ('expression', 'dimerization'), ('expression', 'monolayers'),
('expression', 'megabase-sized'), ('expression', 'dis_submucosal_layer_dis'), ('expression', 'start'), ('expression', 'p
redicted'), ('expression', 'alpha-cyano-beta-hydroxy-beta-methyl-n-[4-(trifluoromethoxy)phe'), ('expression', 'energy'),
 ('expression', '30'), ('expression', 'chelation'), ('expression', 'deduce'), ('expression', 'insulin-like'), ('expressi
on', 'tk'), ('expression', 'heterologou'), ('expression', 'protein-coupled'), ('expression', 'reversible'), ('expression
', 'jak3'), ('expression', 'orthotopic'), ('expression', 'c-dependent'), ('expression', 'connect'), ('expression', '16')
, ('expression', 'meningitidi'), ('expression', 'gene_egfr+_gene'), ('expression', 'vitro'), ('expression', 'staining'),
 ('expression', 'loss'), ('expression', 'mcf-tr5-egfr-cd533'), ('expression', 'fine-granulated'), ('expression', 'altoge
ther'), ('expression', 'gene_egfrviii_gene'), ('expression', 'enhance'), ('expression', 'il-6'), ('expression', 'whey'),
 ('expression', '1478'), ('expression', 'imp'), ('expression', 'thyroid'), ('expression', 'diploid'), ('expression', 'mt
1-mmp'), ('expression', 'alpha'), ('expression', 'four'), ('expression', 'comparable'), ('expression', 'see'), ('express
ion', 'ductal'), ('expression', 'plasmid'), ('expression', 'cis-diammedichloroplatinum'), ('expression', 'chemoresistanc
e'), ('expression', 'agonist'), ('expression', 'add'), ('expression', 'grow'), ('expression', 'extract'), ('expression',
 'resident'), ('expression', '3-28'), ('expression', 'physiology'), ('expression', 'endometrium'), ('expression', 'gbms'
), ('expression', 'remodel'), ('expression', '40%'), ('expression', '6-fold'), ('expression', 'compound'), ('expression'
, 'implant'), ('expression', '111in-dtpa-mab'), ('expression', 'nanomolar'), ('expression', 'upon'), ('expression', 'dis
_restenosis_dis'), ('expression', 'unrelated'), ('expression', 'deltaegfr-containing'), ('expression', 'non-clear'), ('e
xpression', 'tyr845'), ('expression', 'k18'), ('expression', 'brain'), ('expression', 'gastric'), ('expression', 'exert'
), ('expression', 'outline'), ('expression', 'keratinocyte'), ('expression', 'dis_pkd_dis'), ('expression', 'fractionate
```

Potential Improvements

1. Try to reduce operations like gropByKey(), reducebyKey(), join().
    a. The function groupBykey must hold all the key-value pair in memory and if a key has too many values, it can cause an out of memory error.
2. Reduce shuffling
    a. Spark uses shuffling to redistribute data.
    b. Shuffling is an expensive operation.
3. Caching
    a. Spark will store the dataset in memory which allows for faster access and retrieval.
4. Dynamic allocation
    a. Scaling up or down based number of executors based on workload.
5. Data Skewing
    a. There might be uneven distribution of data which reduces utilization.

6. Optimize the amount of Spark partitions
    a. Too much or too little spark partitions could mean some executors are idle or scheduling overhead.
7. Use mapPartitions() over map()
    a. Using mapPartitions provides initialization for many RDD elements rather once per RDD element.
8. Check for memory leaks
    a. Unchecked memory leaks can cause a host of memory issues and slow data processing.
9. Check for bottlenecks
    a. Bottlenecks can occur in any stage of our algorithm which can often slow data processing.
10. Improve queries
    a. Instead of returning every row or column we should only return the ones we are looking for.