README

Submitted by yl2458, dj327, wf85

1. File preprocessing

Text file ‘edges.txt’ is preprocessed by a java file, the hashmap is used to store each source node and corresponding neighbor node. After processed, each single line of the output file is in the format "source degree neighbor-1, neighbor-2, ... , neighbor-n". The preprocessed input file is stored as a S3 file ‘https://s3.amazonaws.com/edu-cornell-cs-cs5300s16-yl2458-pagerank/input/nodesForProject.txt’.
Netid: yl2458 rejectMin: 0.76878 rejectLimit: 0.77878 number of edges: 7524719

2. Overall Structure

1) simple page rank

The simple page rank program processes the filtered file.
For each node, the map task emits to each neighbor node with (its own page rank value) / (out degree of its node). In the reduce task, the graph is reconstructed and the page rank value from neighbor is added as new page rank value. Besides, residual error is computed in the reduce task. A Hadoop counter is used here to accumulate the overall residual errors of the whole graph. To keep the precision of the result, we multiply a number 100,000,000 for each node’s residual error before accumulating.

2) The blocked page rank

There are 5 classes in the blocked page rank package.’webNode.java’ includes all the key parameters (node ID, PageRank value, outgoing degrees and nodes which outgoing links point to) of a node in the web graph. ‘blockPageRank_map’ hard codes the content of file ‘blocks.txt’. The function ‘map’ parses received values into the parameters. It also emits the link relationships based on whether or not the nodes are in the same block. Plus, it provides a function to check the corresponding block ID of a node. ‘blockPageRank_reduce’ iteratively computes the residual error in the block until it is less than the threshold 0.001. ’accumulator’ is used as a counter to accumulate the residual error of different blocks over the whole graph. It also accumulate number of iteration times of different blocks so that we can compute the average number if iterations in each pass. ‘blockPageRank’ is responsible for coordinating map and reduce processes. It also does input and output work.

3) The Guass-Seidel page rank

All the difference between this and blocked page rank is the IterateBlockOnce() function. Guass-Seidel uses the updated PageRank values in this iteration instead of the ones in last iteration to compute current PageRank values.

4) The random page rank

The random page rank is different from blocked page rank by assigning block ID by (nodeID mod 68). The random partition took 22 iteration to converge.

3. How to deploy the program

Copy the JAR file (simplePageRank/blockPageRankPrint/randomPartitionPageRank) to amazon s3. In the EMR, upload the jar file

Run simple `simplePageRank/simplePageRank s3://edu-cornell-cs-cs5300s16-yl2458-pagerank/input/ s3://edu-cornell-cs-cs5300s16-yl2458-pagerank/outputSimple/`

Run block `blockPageRankPrint/blockPageRank s3://edu-cornell-cs-cs5300s16-yl2458-pagerank/input/ s3://edu-cornell-cs-cs5300s16-yl2458-pagerank/b3output/`

Run Guass `gaussPageRank.gaussPageRank s3://edu-cornell-cs-cs5300s16-yl2458-pagerank/input/ s3://edu-cornell-cs-cs5300s16-yl2458-pagerank/outputg4/`

Run random `randomBlockPartition/randomBlockPartition s3://edu-cornell-cs-cs5300s16-yl2458-pagerank/input/ s3://edu-cornell-cs-cs5300s16-yl2458-pagerank/output/`

4. Result Analysis

Simple PageRank: Results of the 5 MapReduce passes:

```
average residual error in iteration 0: 2.3385694
average residual error in iteration 1: 0.3229377
average residual error in iteration 2: 0.19198681
average residual error in iteration 3: 0.09410117
average residual error in iteration 4: 0.06279114
```

Blocked PageRank:

```
residual error in pass 0: 2.8112411
average number of iteration per block in pass 0: 17.485294
residual error in pass 1: 0.0379689
average number of iteration per block in pass 1: 7.1911764
residual error in pass 2: 0.023874294
average number of iteration per block in pass 2: 5.8235292
residual error in pass 3: 0.009805076
average number of iteration per block in pass 3: 3.9264705
residual error in pass 4: 0.0037792667
average number of iteration per block in pass 4: 2.5
residual error in pass 5: 9.516379E-4
average number of iteration per block in pass 5: 1.3382353
```

The PageRank value of the two lowest numbered is saved as an txt file named as ‘blockPageRankSampleResult.txt’, which is submitted in the zip file.

Guass-Seidel:

```
residual error in pass 0: 2.8120434
average number of iteration per block in pass 0: 9.661765
residual error in pass 1: 0.038855735
average number of iteration per block in pass 1: 5.25
residual error in pass 2: 0.025096823
average number of iteration per block in pass 2: 4.5735292
residual error in pass 3: 0.010956198
average number of iteration per block in pass 3: 3.3823528
residual error in pass 4: 0.004862435
average number of iteration per block in pass 4: 2.4117646
residual error in pass 5: 0.0016418326
average number of iteration per block in pass 5: 1.5588236
residual error in pass 6: 7.830803E-4
average number of iteration per block in pass 6: 1.3235294
before for loop
```

The PageRank value of the two lowest numbered is saved as an txt file named as ‘gaussPageRankSampleResult.txt’, which is submitted in the zip file.

Random partition:

```
residual error in pass 0: 2.3391917
average number of iteration per block in pass 0: 2.6323528
residual error in pass 1: 0.32238314
average number of iteration per block in pass 1: 2.1029413
residual error in pass 2: 0.19116461
average number of iteration per block in pass 2: 2.0
residual error in pass 3: 0.09354331
average number of iteration per block in pass 3: 2.0
residual error in pass 4: 0.062117696
average number of iteration per block in pass 4: 2.0
residual error in pass 5: 0.033542998
average number of iteration per block in pass 5: 2.0
residual error in pass 6: 0.026891865
average number of iteration per block in pass 6: 2.0
residual error in pass 7: 0.016401198
average number of iteration per block in pass 7: 2.0
residual error in pass 8: 0.01416704
average number of iteration per block in pass 8: 2.0
residual error in pass 9: 0.00965728
average number of iteration per block in pass 9: 2.0
residual error in pass 10: 0.00832954
average number of iteration per block in pass 10: 2.0
residual error in pass 11: 0.0060540307
average number of iteration per block in pass 11: 2.0
residual error in pass 12: 0.0052796355
average number of iteration per block in pass 12: 2.0
residual error in pass 13: 0.0039586793
average number of iteration per block in pass 13: 2.0
residual error in pass 14: 0.00343903
average number of iteration per block in pass 14: 2.0
residual error in pass 15: 0.0026664152
average number of iteration per block in pass 15: 2.0
residual error in pass 16: 0.002291269
average number of iteration per block in pass 16: 2.0
residual error in pass 17: 0.0018138172
average number of iteration per block in pass 17: 2.0
residual error in pass 18: 0.0015534619
average number of iteration per block in pass 18: 2.0
residual error in pass 19: 0.0012454765
average number of iteration per block in pass 19: 2.0
residual error in pass 20: 0.0010615748
average number of iteration per block in pass 20: 1.9705882
residual error in pass 21: 8.609366E-4
average number of iteration per block in pass 21: 1.0
```

The PageRank value of the two lowest numbered is saved as an txt file named as ‘randomBlockRankSampleResult.txt’, which is submitted in the zip file.

As we can see, although the number of iterations (pass) of the Guass-Seidel algorithm is more than the ones for the ordinary blocked PageRank, the average number of iterations for each block is much less, especially in the first several times passes. It is also obvious that for a bad partition scheme, the number of iteration time (22) is much more than a good partition scheme. In other words, a bad partition scheme can lead to a much more slower convergence.