Cornell University CS5300 - SP14

Fast Convergence PageRank in Hadoop Report

Yun Hao (yh539), Yang Yang (yy565), Chen Zhang (cz294)

0. Overall Structures

1. Preprocess the data: before MapReduce, we need to use our netID to create our own input edge files for each different implementation.
2. PageRank: this is the main part of the project. It starts a MapReduce job for each pass. These MapReduce jobs will then call the Map function and Reduce function.
3. Map: Map function will parse the input data and transfer the output to Reduce.
4. Reduce: Reduce function will calculate the new PageRank value and other information (i.e. highest rank node in each block) and produce output to Map for next pass.
5. PRCounter: This is a Hadoop counter. It will transfer the results from Reduce to our main part PageRank. In block implementation, we also create blockCounter in each block so that we can get the highest rank node in each block.

1. Input Data Preprocess

1.1 Filter Parameter:

- **Reject edges are based on netID:** cz294
- **rejectMin:** 0.99*0.492 (0.48708)
- **rejectLimit:** 0.99*0.492+0.01 (0.49708)

1.2 Data Format:

SimpleNode:

src_node /t *pagerank* /t *LIST(des_node)*

BlockNode:

src_node+src_block /t *pagerank* /t *LIST(des_node+des_block)*

* Note that the pagerank we put here is 1.0, is amplified by N(total # of nodes) so in the reducer, we calculate the new pagerank value by (1-d)+d*incoming_PR_sum, also amplified by N(total # of nodes).

2. Simple Computation of PageRank

2.1 MapReduce Task

Mapper get information from master node, separate sub-divided problems and pass them to each branch node.
Reducer collect the information from each branch node and updates the PR value for the node based on the PR values of its neighbor nodes, and finally pass the updated information back to the master node.

2.2 Data Format

Mapper Input/ Reducer Output Format:

<*srcNodeID* /t *PR* /t *dstNodeIDLIST*>

Mapper Output/ Reducer Input Format:

<*srcNodeID*: *list*: *dstNodeIDLIST*>

<*srcNodeID*: *pr*: *PR*>

if outdegree > 0:

<*dstNodeID*: *PR(srcNode)/degree(srcNode)*>

Use Hadoop counter to calculate the residual error of each pass.

2.3 Map

Mapper basically does the pass tasks. Load information from the last MapReduce pass. (*The first pass will load from the preprocessed file)

<*srcNodeID* /t *PR* /t *dstNodeIDLIST*>

There are three kinds of information we need to pass:

1. <*srcNodeID*: *list*:*dstNodeIDLIST*>
2. <*srcNodeID*: *pr*:*PR*>

*Use those two remember the old pagerank value and list of dstNodes.

3. <*dstNodeID*: *PR(srcNode)/degree(srcNode)*> (if not a sink)

*Use this infomation to caculate the new pagerank value for each node in the reducer.

2.4 Reduce

Collect information from the branch node and reduce then do the computation based on information mapper passed.
There are three kinds of information we received:

1. <*srcNodeID*: *pr*:*PR*> --store to oldPageRank
2. <*srcNodeID*: *list*:*dstNodeIDLIST*> --store to dstNodeIDList
3. <*dstNodeID*: *PR(srcNode)/degree(srcNode)*> (if not a sink)

For each coming edge from the srcNode:

pageRankSum += prevPageRank / srcDegree

newPageRank = d * pageRankSum + (1 - d)

Here d is the damping factor (d=0.85).

(*Not divided by N because the pagerank value of preprocess file is not divided by N, so the value here is actually amplified)

2.5 Calculate the Residual

At the same time, reducer will calculate the residual error of this node and add it to counter (Hadoop counter).

ResidualError = (newPageRank-oldPageRank)/newPageRank

*We need to amplify this by 10^6 to convert to long before we add it to counter.
PageRank Value gets updated every time a collection of key-value pair passes the reducer, and a new residual error will be calculated for this node. After all pairs are done, we can calculate the average residual error of this Pass:

Avg Residual Error= Sum of Residual Error/ total # of Nodes.

After computation, output the updated information to master node using the newPageValue it calculated and the dstNodeIDList mapper passes:

<srcNodeID /t PR /t dstNodeIDLIST>

2.6 Result

Average Residual Value: 2.3388981179020183
Average Residual Value: 0.3229210087269968
Average Residual Value: 0.19205631760138933
Average Residual Value: 0.094025042292369
Average Residual Value: 0.06280159072136364

It demonstrates the slow convergence. We found it actually converging but the residual error is still too far from 0.001 after 5 MapReduce Passes.

3. Blocked Computation of PageRank

3.1 Data Format

Mapper Input/ Reducer Output Format:

< u_ndoeID+u_blockID /t u_PR /t LIST(v_nodeID+v_blockID) >

(*u is the srcNode and v is the dstNode: u->v)

Mapper Output/ Reducer Input Format:

<u_blockID: list:u_nodeID /t LIST(v_nodeID+v_blockID)>

<u_blockID: pr:u_nodeID /t u_PR>

If degree(u) > 0:

<v_blockID: v_NodeID /t u_PR /t u_nodeID+u_blockID /t u_degree>

Else u is a sink:

<u_blockID: u_NodeID /t u_PR /t u_nodeID+u_blockID /t 0>

(*Denote a sink by setting its degree to 0)

3.2 Map

Load information from the last MapReduce pass:
(*The first pass will load from the preprocessed file)

< u_ndoeID+u_blockID /t u_PR /t LIST(v_nodeID+v_blockID) >

(*u is the srcNode and v is the dstNode: u->v)
There are three kinds of information we need to pass:

1. *<u_blockID: list:u_nodeID /t LIST(v_nodeID+v_blockID)>*
2. *<u_blockID: pr:u_nodeID /t u_PR>*

*Use those two remember the old pagerank value and information of list of dstNodes

3. If degree(u) > 0:

<v_blockID: v_NodeID /t u_PR /t u_nodeID+u_blockID /t u_degree>

Else u is a sink:

<u_blockID: u_NodeID /t u_PR /t u_nodeID+u_blockID /t 0>

(*Denote a sink by setting its degree to 0)
We use this info to calculate the new pagerank value for each node in reducer.

3.3 Reduce

In reduce section, we firstly collect information from the brand node, reduce and then do the computation based on the information mapper passed. We also need to use Hadoop counters to store the sum of the residual errors, sum of the number of iterations and the pr value of the highest numbered node in each block.

Since each block can have many nodes, we need created several maps as below to keep track the related information we need:

HashMap<String, Double> oldPR : <u_nodeID, u_pr>

for nodes u in block B (store old pagerank value).

HashMap<String, String> v_list_map :<u_nodeID, LIST(v_nodeID+v_blockID)>

for nodes u in block B used for reducer output.

HashMap<String, ArrayList<String> > BE:<v_nodeID, arraylist(nodeID_u) >

for nodes u and v in block B and u->v, BE is for the edges from nodes in block B.

HashMap<String, Integer> degreeMap:<u_nodeID, u_degree>

for all nodes u in block B.

HashMap<String, ArrayList<Double>> BC:<v1_nodeID, arraylist(u1_PR/u1_degree)>

for u1->v1 and v1 in block B and u1 not belong to block B. BC is for boundary condition.

We do the preparations as following:
There are three kinds of information we received

1.<u_blockID: list:u_nodeID /t LIST(v_nodeID+v_blockID)>

Find the input that has info of list_v_node_block and get u_nodeID and list of nodes v from the input, store into Map v_list_map.

2. <u_blockID: pr:u_nodeID /t u_PR>

Find the input that has info of old PR of node u and put node u and the old PR value of node u into Map oldPR.

3.<v_blockID: v_NodeID /t u_PR /t u_nodeID+u_blockID /t u_degree>

or

<u_blockID: u_NodeID /t u_PR /t u_nodeID+u_blockID /t 0>

If u, v are from the same block: BE.put(v_nodeID,update_u_nodeID_arraylist) & degreeMap.put(u_nodeID, u_degree). If they are from different block: calculate PR(u)/degree(u), BC.put(v_nodeID, update_calc_list)--boundary value, fixed.

```
Algorithm:
  for ( v in B ) { NPR [v] = 0; }
  for ( v in B ) {
     for ( u where <u,v> in BE ) {
        if(deg[u]>0)
           NPR[v] +=  PR[u] / deg[u];
     }
     for ( u, R where <u,v,R> in BC ) {
        NPR[v] +=  R;
     }
     NPR[v] = d * NPR[v] + (1 – d) ;
 }
 for ( v in B ) {
    localResidualError += |NPR[v]-PR[v]| / NPR[v]
    PR[v] = NPR[v] ;
 }
```

Repeat the iteration until the localResidualError < 0.001 * PR.size(), that means the avg residual error in this block is smaller than 0.001. Use an int type numIter to record the total number of iteration of this block.

3.4 Result

MapReduce Pass 0:
Average number of iterations per Block: 18
Global Average Residual Error: 2.815
MapReduce Pass 1:
Average number of iterations per Block: 7
Global Average Residual Error: 0.03800
MapReduce Pass 2:

Average number of iterations per Block: 6
Global Average Residual Error: 0.02391
MapReduce Pass 3:
Average number of iterations per Block: 4
Global Average Residual Error: 0.009879
MapReduce Pass 4:
Average number of iterations per Block: 3
Global Average Residual Error: 0.003823
MapReduce Pass 5:
Average number of iterations per Block: 1
Global Average Residual Error: 0.0009708

Brief Summary:
Total Number of Passes: 6
Average Residual Error for Pass 0: 2.815
Average Residual Error for Pass 1: 0.03800
Average Residual Error for Pass 2: 0.02391
Average Residual Error for Pass 3: 0.009879
Average Residual Error for Pass 4: 0.003823
Average Residual Error for Pass 5: 0.0009708

The pagerank value of the highest numbered Node in each block:
The page rank of highest numbered Node in Block 0 is 9.780e-07
The page rank of highest numbered Node in Block 1 is 5.181e-07
The page rank of highest numbered Node in Block 2 is 3.070e-07
The page rank of highest numbered Node in Block 3 is 2.809e-07
The page rank of highest numbered Node in Block 4 is 3.003e-07
The page rank of highest numbered Node in Block 5 is 2.189e-07
The page rank of highest numbered Node in Block 6 is 3.001e-07
The page rank of highest numbered Node in Block 7 is 2.189e-07
The page rank of highest numbered Node in Block 8 is 0.0008665
The page rank of highest numbered Node in Block 9 is 2.900e-07
The page rank of highest numbered Node in Block 10 is 1.771e-06
The page rank of highest numbered Node in Block 11 is 4.853e-07
The page rank of highest numbered Node in Block 12 is 2.358e-07
The page rank of highest numbered Node in Block 13 is 6.287e-07
The page rank of highest numbered Node in Block 14 is 2.189e-07
The page rank of highest numbered Node in Block 15 is 2.189e-07
The page rank of highest numbered Node in Block 16 is 5.667e-07
The page rank of highest numbered Node in Block 17 is 3.516e-07
The page rank of highest numbered Node in Block 18 is 5.027e-06

```
The page rank of highest numbered Node in Block 19 is 4.104e-06
The page rank of highest numbered Node in Block 20 is 2.617e-07
The page rank of highest numbered Node in Block 21 is 0.001198
The page rank of highest numbered Node in Block 22 is 0.0001145
The page rank of highest numbered Node in Block 23 is 5.266e-07
The page rank of highest numbered Node in Block 24 is 2.189e-07
The page rank of highest numbered Node in Block 25 is 2.994e-07
The page rank of highest numbered Node in Block 26 is 2.189e-07
The page rank of highest numbered Node in Block 27 is 3.370e-07
The page rank of highest numbered Node in Block 28 is 2.617e-07
The page rank of highest numbered Node in Block 29 is 2.564e-06
The page rank of highest numbered Node in Block 30 is 8.780e-07
The page rank of highest numbered Node in Block 31 is 2.189e-07
The page rank of highest numbered Node in Block 32 is 5.174e-07
The page rank of highest numbered Node in Block 33 is 2.520e-07
The page rank of highest numbered Node in Block 34 is 3.877e-07
The page rank of highest numbered Node in Block 35 is 3.552e-07
The page rank of highest numbered Node in Block 36 is 3.347e-06
The page rank of highest numbered Node in Block 37 is 4.239e-07
The page rank of highest numbered Node in Block 38 is 7.235e-07
The page rank of highest numbered Node in Block 39 is 2.469e-07
The page rank of highest numbered Node in Block 40 is 3.118e-07
The page rank of highest numbered Node in Block 41 is 1.611e-06
The page rank of highest numbered Node in Block 42 is 2.856e-06
The page rank of highest numbered Node in Block 43 is 3.851e-06
The page rank of highest numbered Node in Block 44 is 6.043e-07
The page rank of highest numbered Node in Block 45 is 1.114e-05
The page rank of highest numbered Node in Block 46 is 9.741e-07
The page rank of highest numbered Node in Block 47 is 5.826e-07
The page rank of highest numbered Node in Block 48 is 2.189e-07
The page rank of highest numbered Node in Block 49 is 6.646e-06
The page rank of highest numbered Node in Block 50 is 6.166e-07
The page rank of highest numbered Node in Block 51 is 0.0009804
The page rank of highest numbered Node in Block 52 is 2.502e-05
The page rank of highest numbered Node in Block 53 is 0.002508
The page rank of highest numbered Node in Block 54 is 0.001176
The page rank of highest numbered Node in Block 55 is 1.599e-06
The page rank of highest numbered Node in Block 56 is 1.101e-06
The page rank of highest numbered Node in Block 57 is 9.686e-07
The page rank of highest numbered Node in Block 58 is 4.004e-06
The page rank of highest numbered Node in Block 59 is 4.268e-07
```

The page rank of highest numbered Node in Block 60 is 6.176e-07
The page rank of highest numbered Node in Block 61 is 1.511e-05
The page rank of highest numbered Node in Block 62 is 1.071e-06
The page rank of highest numbered Node in Block 63 is 3.119e-07
The page rank of highest numbered Node in Block 64 is 2.189e-07
The page rank of highest numbered Node in Block 65 is 9.674e-07
The page rank of highest numbered Node in Block 66 is 1.157e-06
The page rank of highest numbered Node in Block 67 is 3.566e-07

Jacobi PageRank Computation takes 6 passes (38 iterations per block) to converge.

4. Extra Credit

4.1 Gauss-Seidel Iteration

The Gauss-Seidel Iteration method uses the most recent pagerank values wherever possible to improve convergence rate. According to the following equation:

$$x_i^{(k+1)} = (1-\alpha) + \alpha \sum_{j<i} a_{ij} x_j^{(k+1)} + \alpha \sum_{j>i} a_{ij} x_j^{(k)} \quad \forall i$$

The only difference between Gauss Iteration and Blocked Iteration is the following implementation:
double u_pr = (NPR.get(u_nodeID) > 0) ? NPR.get(u_nodeID) : PR.get(u_nodeID);
This code will get the most recent pagerank values wherever possible.

Input file: block_edge
Output:
MapReduce Pass 0:
Average number of iterations per Block: 12
Global Average Residual Error: 3.151
MapReduce Pass 1:
Average number of iterations per Block: 6
Global Average Residual Error: 0.03688
MapReduce Pass 2:
Average number of iterations per Block: 5
Global Average Residual Error: 0.02304
MapReduce Pass 3:
Average number of iterations per Block: 3
Global Average Residual Error: 0.008108

```
MapReduce Pass 4:
Average number of iterations per Block: 2
Global Average Residual Error: 0.003485
MapReduce Pass 5:
Average number of iterations per Block: 1
Global Average Residual Error: 0.001006
MapReduce Pass 6:
Average number of iterations per Block: 1
Global Average Residual Error: 0.0006046

Summary:
Total Number of Passes: 7
Average Residual Error for Pass 0: 3.151
Average Residual Error for Pass 1: 0.03688
Average Residual Error for Pass 2: 0.02304
Average Residual Error for Pass 3: 0.008108
Average Residual Error for Pass 4: 0.003485
Average Residual Error for Pass 5: 0.001006
Average Residual Error for Pass 6: 0.0006046

The page rank of highest numbered Node in Block 0 is 9.544e-07
The page rank of highest numbered Node in Block 1 is 5.144e-07
The page rank of highest numbered Node in Block 2 is 2.653e-07
The page rank of highest numbered Node in Block 3 is 2.809e-07
The page rank of highest numbered Node in Block 4 is 2.343e-07
The page rank of highest numbered Node in Block 5 is 2.189e-07
The page rank of highest numbered Node in Block 6 is 2.491e-07
The page rank of highest numbered Node in Block 7 is 2.189e-07
The page rank of highest numbered Node in Block 8 is 0.0005743
The page rank of highest numbered Node in Block 9 is 2.454e-07
The page rank of highest numbered Node in Block 10 is 2.242e-06
The page rank of highest numbered Node in Block 11 is 4.216e-07
The page rank of highest numbered Node in Block 12 is 2.224e-07
The page rank of highest numbered Node in Block 13 is 4.532e-07
The page rank of highest numbered Node in Block 14 is 2.189e-07
The page rank of highest numbered Node in Block 15 is 2.189e-07
The page rank of highest numbered Node in Block 16 is 6.255e-07
The page rank of highest numbered Node in Block 17 is 3.446e-07
The page rank of highest numbered Node in Block 18 is 4.090e-06
The page rank of highest numbered Node in Block 19 is 4.417e-06
The page rank of highest numbered Node in Block 20 is 2.608e-07
```

The page rank of highest numbered Node in Block 21 is 0.0007830
The page rank of highest numbered Node in Block 22 is 2.243e-05
The page rank of highest numbered Node in Block 23 is 5.031e-07
The page rank of highest numbered Node in Block 24 is 2.189e-07
The page rank of highest numbered Node in Block 25 is 2.369e-07
The page rank of highest numbered Node in Block 26 is 2.189e-07
The page rank of highest numbered Node in Block 27 is 3.275e-07
The page rank of highest numbered Node in Block 28 is 2.608e-07
The page rank of highest numbered Node in Block 29 is 2.514e-06
The page rank of highest numbered Node in Block 30 is 7.709e-07
The page rank of highest numbered Node in Block 31 is 2.189e-07
The page rank of highest numbered Node in Block 32 is 4.724e-07
The page rank of highest numbered Node in Block 33 is 2.462e-07
The page rank of highest numbered Node in Block 34 is 3.670e-07
The page rank of highest numbered Node in Block 35 is 3.381e-07
The page rank of highest numbered Node in Block 36 is 3.674e-06
The page rank of highest numbered Node in Block 37 is 4.779e-07
The page rank of highest numbered Node in Block 38 is 9.675e-07
The page rank of highest numbered Node in Block 39 is 2.535e-07
The page rank of highest numbered Node in Block 40 is 2.844e-07
The page rank of highest numbered Node in Block 41 is 1.300e-06
The page rank of highest numbered Node in Block 42 is 3.020e-06
The page rank of highest numbered Node in Block 43 is 5.876e-06
The page rank of highest numbered Node in Block 44 is 3.524e-07
The page rank of highest numbered Node in Block 45 is 1.561e-05
The page rank of highest numbered Node in Block 46 is 8.791e-07
The page rank of highest numbered Node in Block 47 is 4.872e-07
The page rank of highest numbered Node in Block 48 is 2.189e-07
The page rank of highest numbered Node in Block 49 is 5.594e-06
The page rank of highest numbered Node in Block 50 is 4.686e-07
The page rank of highest numbered Node in Block 51 is 0.0004877
The page rank of highest numbered Node in Block 52 is 1.933e-05
The page rank of highest numbered Node in Block 53 is 0.0009960
The page rank of highest numbered Node in Block 54 is 0.0006277
The page rank of highest numbered Node in Block 55 is 1.467e-06
The page rank of highest numbered Node in Block 56 is 7.824e-07
The page rank of highest numbered Node in Block 57 is 5.833e-07
The page rank of highest numbered Node in Block 58 is 3.333e-06
The page rank of highest numbered Node in Block 59 is 3.730e-07
The page rank of highest numbered Node in Block 60 is 5.082e-07
The page rank of highest numbered Node in Block 61 is 6.385e-06

The page rank of highest numbered Node in Block 62 is 1.368e-06
The page rank of highest numbered Node in Block 63 is 3.119e-07
The page rank of highest numbered Node in Block 64 is 2.189e-07
The page rank of highest numbered Node in Block 65 is 1.041e-06
The page rank of highest numbered Node in Block 66 is 1.403e-06
The page rank of highest numbered Node in Block 67 is 3.003e-07

Comparison: Jacobi vs. Gauss-Seidel:

MapReduce Pass	Average number of iterations per Block		Global Average Residual	
	Jacobi	Gauss-Seidel	Jacobi	Gauss-Seidel
0	17	12	2.815	3.151
1	7	6	0.03800	0.03688
2	6	5	0.02391	0.02304
3	4	3	0.009879	0.008108
4	3	2	0.003823	0.003485
5	1	1	0.0009708	0.001006
6		1		0.0006046

The two versions take the similar mapreduce passes to converge to the same rate. But since Gauss-Seidel uses the new pagerank value as soon as possible, it needs less average number of iterations (30) per block than Jacobi, thus helping to converge faster than Jacobi.

4.2 Random Block Partition

Random Hash Function:

$$\text{Hash}(nodeID) = (nodeID+random)\%68$$

The following results shows that it takes 21 mapreduce passes(44 iterations per block) to converge.

Results:
MapReduce Pass 0:
Average number of iterations per Block: 3
Global Average Residual Error: 2.341
MapReduce Pass 1:
Average number of iterations per Block: 3
Global Average Residual Error: 0.3220
MapReduce Pass 2:

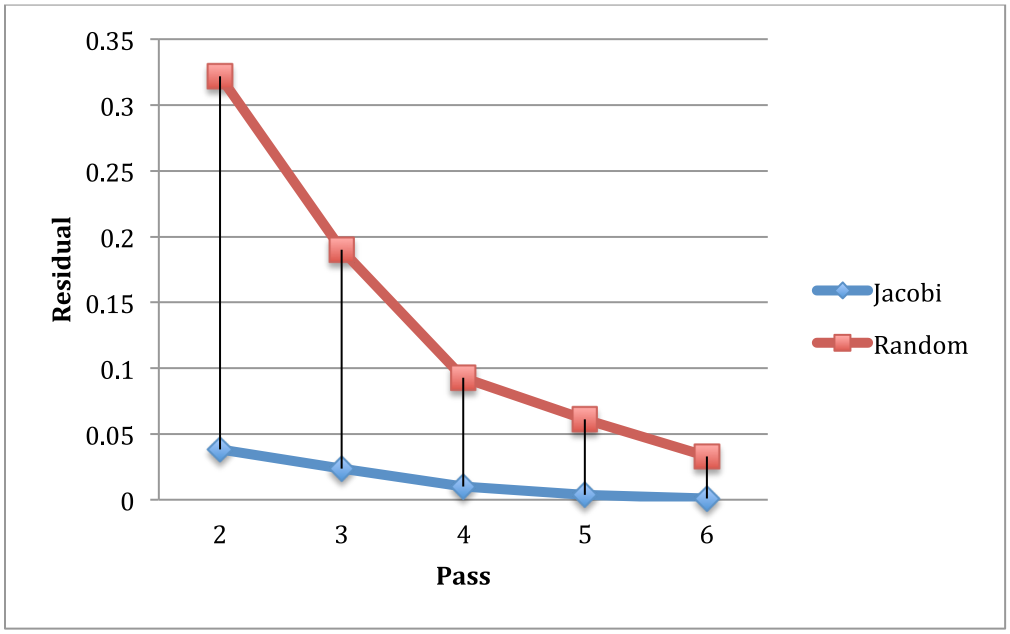
```
Average number of iterations per Block: 3
Global Average Residual Error: 0.1900
MapReduce Pass 3:
Average number of iterations per Block: 2
Global Average Residual Error: 0.09272
MapReduce Pass 4:
Average number of iterations per Block: 2
Global Average Residual Error: 0.06116
MapReduce Pass 5:
Average number of iterations per Block: 2
Global Average Residual Error: 0.03293
MapReduce Pass 6:
Average number of iterations per Block: 2
Global Average Residual Error: 0.02603
MapReduce Pass 7:
Average number of iterations per Block: 2
Global Average Residual Error: 0.01585
MapReduce Pass 8:
Average number of iterations per Block: 2
Global Average Residual Error: 0.01351
MapReduce Pass 9:
Average number of iterations per Block: 2
Global Average Residual Error: 0.009200
MapReduce Pass 10:
Average number of iterations per Block: 2
Global Average Residual Error: 0.007837
MapReduce Pass 11:
Average number of iterations per Block: 2
Global Average Residual Error: 0.005694
MapReduce Pass 12:
Average number of iterations per Block: 2
Global Average Residual Error: 0.004905
MapReduce Pass 13:
Average number of iterations per Block: 2
Global Average Residual Error: 0.003676
MapReduce Pass 14:
Average number of iterations per Block: 2
Global Average Residual Error: 0.003160
MapReduce Pass 15:
Average number of iterations per Block: 2
Global Average Residual Error: 0.002446
```

```
MapReduce Pass 16:
Average number of iterations per Block: 2
Global Average Residual Error: 0.002083
MapReduce Pass 17:
Average number of iterations per Block: 2
Global Average Residual Error: 0.001645
MapReduce Pass 18:
Average number of iterations per Block: 2
Global Average Residual Error: 0.001398
MapReduce Pass 19:
Average number of iterations per Block: 2
Global Average Residual Error: 0.001118
MapReduce Pass 20:
Average number of iterations per Block: 1
Global Average Residual Error: 0.0009415

Total Number of Passes: 21
Average Residual Error for Pass 0: 2.341
Average Residual Error for Pass 1: 0.3220
Average Residual Error for Pass 2: 0.1900
Average Residual Error for Pass 3: 0.09272
Average Residual Error for Pass 4: 0.06116
Average Residual Error for Pass 5: 0.03293
Average Residual Error for Pass 6: 0.02603
Average Residual Error for Pass 7: 0.01585
Average Residual Error for Pass 8: 0.01351
Average Residual Error for Pass 9: 0.009200
Average Residual Error for Pass 10: 0.007837
Average Residual Error for Pass 11: 0.005694
Average Residual Error for Pass 12: 0.004905
Average Residual Error for Pass 13: 0.003676
Average Residual Error for Pass 14: 0.003160
Average Residual Error for Pass 15: 0.002446
Average Residual Error for Pass 16: 0.002083
Average Residual Error for Pass 17: 0.001645
Average Residual Error for Pass 18: 0.001398
Average Residual Error for Pass 19: 0.001118
Average Residual Error for Pass 20: 0.0009415
The page rank of highest numbered Node in Block 0 is 3.100e-07
The page rank of highest numbered Node in Block 1 is 3.144e-07
The page rank of highest numbered Node in Block 2 is 2.925e-07
The page rank of highest numbered Node in Block 3 is 4.480e-07
The page rank of highest numbered Node in Block 4 is 3.121e-07
The page rank of highest numbered Node in Block 5 is 6.776e-07
The page rank of highest numbered Node in Block 6 is 2.908e-07
The page rank of highest numbered Node in Block 7 is 2.937e-07
The page rank of highest numbered Node in Block 8 is 6.213e-07
The page rank of highest numbered Node in Block 9 is 3.124e-07
The page rank of highest numbered Node in Block 10 is 6.267e-07
The page rank of highest numbered Node in Block 11 is 2.903e-07
The page rank of highest numbered Node in Block 12 is 2.723e-07
The page rank of highest numbered Node in Block 13 is 2.920e-07
The page rank of highest numbered Node in Block 14 is 3.234e-07
The page rank of highest numbered Node in Block 15 is 3.305e-07
The page rank of highest numbered Node in Block 16 is 3.945e-07
The page rank of highest numbered Node in Block 17 is 4.489e-07
The page rank of highest numbered Node in Block 18 is 2.734e-07
The page rank of highest numbered Node in Block 19 is 2.892e-07
The page rank of highest numbered Node in Block 20 is 5.967e-07
The page rank of highest numbered Node in Block 21 is 2.883e-07
The page rank of highest numbered Node in Block 22 is 2.900e-07
The page rank of highest numbered Node in Block 23 is 3.126e-07
The page rank of highest numbered Node in Block 24 is 5.520e-07
The page rank of highest numbered Node in Block 25 is 4.265e-07
The page rank of highest numbered Node in Block 26 is 2.911e-07
The page rank of highest numbered Node in Block 27 is 3.051e-07
The page rank of highest numbered Node in Block 28 is 2.942e-07
The page rank of highest numbered Node in Block 29 is 3.154e-07
The page rank of highest numbered Node in Block 30 is 3.286e-07
The page rank of highest numbered Node in Block 31 is 3.104e-07
The page rank of highest numbered Node in Block 32 is 3.129e-07
The page rank of highest numbered Node in Block 33 is 2.898e-07
The page rank of highest numbered Node in Block 34 is 6.051e-07
The page rank of highest numbered Node in Block 35 is 3.135e-07
The page rank of highest numbered Node in Block 36 is 5.017e-07
The page rank of highest numbered Node in Block 37 is 3.131e-07
The page rank of highest numbered Node in Block 38 is 4.208e-07
The page rank of highest numbered Node in Block 39 is 6.082e-07
The page rank of highest numbered Node in Block 40 is 5.987e-07
The page rank of highest numbered Node in Block 41 is 1.217e-06
The page rank of highest numbered Node in Block 42 is 9.778e-07
The page rank of highest numbered Node in Block 43 is 2.906e-07
```

The page rank of highest numbered Node in Block 44 is 3.132e-07
The page rank of highest numbered Node in Block 45 is 3.270e-07
The page rank of highest numbered Node in Block 46 is 8.376e-07
The page rank of highest numbered Node in Block 47 is 5.478e-07
The page rank of highest numbered Node in Block 48 is 1.015e-06
The page rank of highest numbered Node in Block 49 is 3.384e-07
The page rank of highest numbered Node in Block 50 is 3.118e-07
The page rank of highest numbered Node in Block 51 is 3.259e-07
The page rank of highest numbered Node in Block 52 is 2.900e-07
The page rank of highest numbered Node in Block 53 is 2.557e-07
The page rank of highest numbered Node in Block 54 is 5.485e-07
The page rank of highest numbered Node in Block 55 is 3.201e-07
The page rank of highest numbered Node in Block 56 is 2.967e-07
The page rank of highest numbered Node in Block 57 is 2.919e-07
The page rank of highest numbered Node in Block 58 is 6.826e-07
The page rank of highest numbered Node in Block 59 is 5.521e-07
The page rank of highest numbered Node in Block 60 is 3.252e-07
The page rank of highest numbered Node in Block 61 is 3.077e-07
The page rank of highest numbered Node in Block 62 is 4.148e-07
The page rank of highest numbered Node in Block 63 is 2.918e-07
The page rank of highest numbered Node in Block 64 is 6.064e-07
The page rank of highest numbered Node in Block 65 is 3.271e-07
The page rank of highest numbered Node in Block 66 is 3.131e-07
The page rank of highest numbered Node in Block 67 is 2.881e-07

Comparison:



5. Running our Code

We have made our own pre-filtered input files, customized for the netid cz294.

Step 0: Upload our pre-filtered input files to s3 bucket. You might want to create a new folder such as "data", and store all the input files in that folder.

Step 1: Upload custom JAR files to s3 bucket.

Step 2: Create a cluster called "pageRank", store the log files to a folder such as "s3n://bucket-name/log/".

Step 3: You can start a master instance and 10 core instances.

Step 4: Add step "Custom JAR". For example:

```
Name: SimplePageRank
JAR S3 location: s3n://bucket-name/SimplePageRank.jar
Arguments: s3n://bucket-name/folder-name/singlenode_edge
           s3n://bucket-name/folder-name
```

In the "Arguments" field above, the first line is the path of the input file, the second line is the output path. Note that we have appended "/pass0", "/pass1"... to the output path in our code. For the example above, the real output path will be

s3n://bucket-name/folder-name/pass0, s3n://bucket-name/folder-name/pass1,

...

As a result, for the output argument, please end the argument with the folder-name only, without any other special characters like '/'. Otherwise, it will fail to create the output folder.

Step 5: Create the cluster and run the program. You should see appropriate information in stdout and log files.