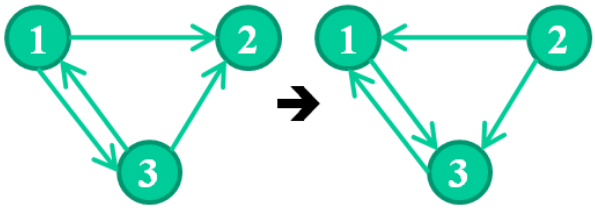
COMP9313 2016s2 Assignment 4

Processing Graph Data using MapReduce on EMR

Problem 1 (10 pts): Reverse graph edge direction

Given a directed graph, reverse the direction of all edges.



Input files:

In the input file, each line contains a pair of node ids: "FromNodeId\tToNodeId". In the above example, the input contains four lines: "1\t2", "1\t3", "3\t1", "3\t2".

The sample file "tiny-web-Stanford.txt" can be downloaded at:
https://webcms3.cse.unsw.edu.au/COMP9313/16s2/resources/5391

Download the entire graph "web-Stanford.txt" at:

https://webcms3.cse.unsw.edu.au/COMP9313/16s2/resources/5392

Output:

The output is the adjacency list of the reversed graph, and the nodes are sorted in ascending order in each list. Format each line as: "NodeId\t$Neighbor_1$, $Neighbor_2$, …, $Neighbor_m$", using only a comma to separate the node IDs in the list.

Given the above example, the output file contains three lines: "1\t3", "2\t1, 3", "3\t1".

Cluster configuration:

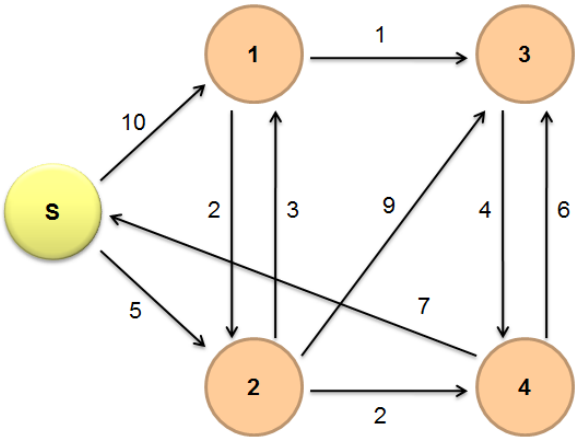
Create a cluster with 3 nodes of instance type m3.xlarge. Create an S3 bucket with name "comp9313.<YOUR_STUDENTID>". Create a folder "problem1" in this bucket, and upload the input file "web-Stanford.txt" into this folder. Save your output folder in "problem1" as well. Finally, please make the output folder public (right click the folder, and click "Make public").

Code format:

Name your java file as "ReverseGraph.java", and put it in the package "comp9313.ass4". Set the number of reducers to 3 when configuring the MapReduce job. The input and output are taken from the parameters of the main function.

Problem 2 (20 pts): Single-source shortest path

Given a graph and a node "s", find the distances of all nodes to "s".



Input files:

In the input file, each line is in format of:

"EdgeId FromNodeId ToNodeId Distance".

In the above example, the input contains:

```
0 0 1 10.0
1 0 2 5.0
2 1 2 2.0
3 1 3 1.0
4 2 1 3.0
5 2 3 9.0
6 2 4 2.0
7 3 4 4.0
8 4 0 7.0
9 4 3 6.0
```

This sample file "tiny-graph.txt" can be downloaded at:

https://webcms3.cse.unsw.edu.au/COMP9313/16s2/resources/5393

Download the two larger graphs “NA.cedge.txt” and “SF.cedge.txt” at:

https://webcms3.cse.unsw.edu.au/COMP9313/16s2/resources/5394 and

https://webcms3.cse.unsw.edu.au/COMP9313/16s2/resources/5395

Output:

The output file contains distances of all nodes to the given node. Each line is in format of “QueryNodeId TargetNodeID Distance”. The distances are of double precision. Given the example graph, the output file is like:

```
0 0 0.0
0 1 8.0
0 2 5.0
0 3 9.0
0 4 7.0
```

Cluster configuration:

Create a cluster with 3 nodes of instance type m3.xlarge. Create an S3 bucket with name “comp9313.<YOUR_STUDENTID>”. Create a folder “problem2” in this bucket, and upload the input files “NA.cedge.txt” and “SF.cedge.txt” into this folder. ~~Use node “0” as the query node for both graphs, and save your results in “problem2” as well. Finally, please make the output folder public (right click the folder, and click “Make public”).~~ Set the name of the final result folder as: “NA”(“SF”)+“QueryNodeId”, and store it in the HDFS output folder as specified in the main function parameters (This means that you need to perform another MapReduce job to extract the result).

Code format:

Name your java file as “SingleSourceSP.java”, and put it in the package “comp9313.ass4”. Your program should take three parameters: the input folder, the output folder, and the query node id.

One difficulty of this problem is how to do the iterative MapReduce jobs and how to check the termination criterion. You can download the code template at: https://webcms3.cse.unsw.edu.au/COMP9313/16s2/resources/5396, which may help you to solve this problem.

Notes

1. Create a project locally in Eclipse, test everything in your local computer, and finally do it in EMR.

~~2. In the second problem, generate all intermediate results in HDFS, and then extract the result as a single file, and save it in S3.~~

2. In the second problem, you can use more than one reducers when doing the shortest path computation. However, when extracting the final result, please just use one reducer to save the output into one file.

3. In problem2, we will randomly select 10 query nodes to check the correctness of your algorithm.

Documentation and code readability

Your source code will be inspected and marked based on readability and ease of understanding. The documentation (comments of the codes) in your source code is also important. Below is an indicative marking scheme:

Result correctness: 80%
Code structure, Readability, and Documentation: 20%

Submission:

Deadline: Monday 7th November 09:59:59 AM

Log in any CSE server (williams or wagner), and use the give command below to submit your solutions:

$ give cs9313 assignment4 ReverseGraph.java SingleSourceSP.java

Or you can submit through:
https://cgi.cse.unsw.edu.au/~give/Student/give.php

If you submit your assignment more than once, the last submission will replace the previous one. To prove successful submission, please take a screenshot as assignment submission instructions show and keep it by yourself.

Late submission penalty

10% reduction of your marks for the 1st day, 30% reduction/day for the following days.

Plagiarism:

The work you submit must be your own work. Submission of work partially or completely derived from any other person or jointly written with any other person is not permitted. The penalties for such an offence may include negative marks, automatic failure of the course and possibly other academic discipline. Assignment submissions will be examined manually.

Relevant scholarship authorities will be informed if students holding scholarships are involved in an incident of plagiarism or other misconduct.

Do not provide or show your assignment work to any other person - apart from the teaching staff of this subject. If you knowingly provide or show your assignment work to another person for any reason, and work derived from it is submitted you may be penalized, even if the work was submitted without your knowledge or consent.