Project 7: Harp PageRank
Cloud Computing
Spring 2017

Professor Judy Qiu

Goal

For this project you will implement PageRank on Harp[1] framework.

Deliverables

Zip your source code and output as username_harp-pagerank.zip. Please submit this file to the Canvas Assignments page.

Evaluation

The point total for this project is 6, where the distribution is as follows:

- Completeness of your code (5 points)
- Correct output (1 point)

Prerequisites

From now on, you don't need the old VM. To avoid any conflict and inconvenience, we have prepared a new VM (ubuntu 16.04) for you. The link will be posted on canvas. All necessary tools/libaries such Maven, JDK, Github, Hadoop 2.6.0, Harp are configured and ready for use. Intellij is installed as well. You can also use your own VM. But you will need to setup those tools/libraries by yourself. Some tutorials are available at the harp website[1]. Here this instruction is based on the configurations in this new VM.

HarpPageRank Implementation

Most of the code is completed for you and your task will be to perform the **Compute PR** step in the above diagram. The code for this can be found in **simplepagerank/PageRankMapper.java**

```
1  public void computePartialPR(Map<Long, ArrayList<Long>> partialGraph, Long2DoubleKVTable
       localPRTable, Long2DoubleKVTable globalPRTable){
2
3          for (Entry<Long, ArrayList<Long>> entry : partialGraph.entrySet()) {
4              Long sourceUrl = entry.getKey();
5              ArrayList<Long> targetUrls = entry.getValue();
6              System.out.println("sourceURL: "+sourceUrl);
7              if(targetUrls == null ){
8                  // simply assume that the IDs of pages are: 0,1,2,...,(numUrls-1)
9                  System.out.println("targetUrls is null");
10                 double pr = localPRTable.getVal(sourceUrl) / numUrls;
11                 // TODO - Students write Code here
```

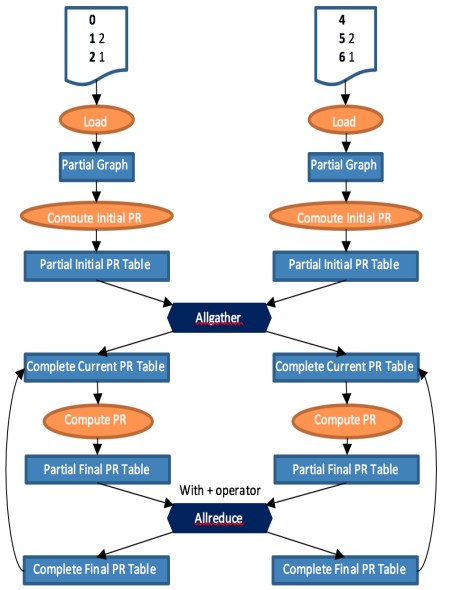

Figure 1: Harp PageRank Architecture

```
                // Add pr to the page rank of all other URLs in globalPRTable
                // Note. The addKeyVal(key, val) method in globalPRTable will
                // automatically sum values if the key exists. If the key does
                // NOT exist then a new entry will be made.


            }else{
                int numOfOutLinks = targetUrls.size();
                double pr = localPRTable.getVal(sourceUrl) / numOfOutLinks;
                // TODO - Students write Code here
                // Add pr to the page rank of all target URLs.



            }
        }
    }
```

Compilation and Running

- To make the modification to the code, you can use Intellij IDE or linux terminal. The source code is located at

```
/home/cc/Documents/harp/harp-tutorial-app/src/main/java/edu/iu/simplepagerank
```

- To compile the code, type the following commands in terminal

```
$ cd $HARP_ROOT_DIR
$ mvn clean package
```

- If hadoop is not started, start hadoop by:

```
$ $HADOOP_HOME/sbin/start-dfs.sh
$ $HADOOP_HOME/sbin/start-yarn.sh
```

  Then you can view the web UI at localhost:50070 and localhost:8088

- We prepared the input dataset (input5K-2partitions) for you. Use the following command to put it to hdfs. Please note there is a "dot" at the end of the second command.

```
$ cd $HARP_ROOT_DIR/data/tutorial/simplepagerank
$ hdfs dfs -put input5K-2partitions .
```

- Run the program:

```
$ cd $HARP_ROOT_DIR
$ hadoop jar harp-tutorial-app/target/harp-tutorial-app-1.0-SNAPSHOT.jar edu.iu.
    simplepagerank.HarpPageRank input5K-2partitions output5k 5000 10
```

  This will run PageRank against input2K-2partitions dataset. It has 5000 URLs in total. The program will run 2 parallel map tasks for 10 iterations. If you want to launch N map tasks, you need to divide the dataset into N partitions.

- To get the output, perform the following commands to get the output to the Desktop. Then you can submit it to canvas.

```
$ hdfs dfs -get output5k /home/cc/Desktop
```

References

[1] Indiana University. `https://dsc-spidal.github.io/harp`.