Virginia Tech. **CS 4604 – Introduction to DBMS**
Computer Science **Spring 2014, Prakash**

Homework 5: XML and MapReduce

(due April 1, 2014, 3:30pm, hard-copy in-class and code on Scholar please)

Reminders:

a. Out of 100 points. Contains 4 pages.
b. Rough time estimates 3-5 hours.
c. Please type your answers. Illegible handwriting may get no points, at the discretion of the grader. Only drawings may be hand-drawn, as long as they are neat and legible.
d. There could be more than one correct answer. We shall accept them all.
e. Whenever you are making an assumption, please state it clearly.
f. Lead TA for his homework: Qianzhou Du.

Q1: XML [15 points]

Recall the soccer DB scenario in HW2 where you designed the database schema for a professional soccer league. The league consists of teams (clubs) identified by a name (e.g., Real Madrid), and each team has a location (e.g., Madrid) and many players. For each player, he or she has a unique name, and weekly wage (e.g., $200,000) after tax. Consider the following relation TEAMS as a SQL DDL statement.

```
CREATE TABLE TEAMS (
        TeamName CHARACTER VARYING,
        Location CHARACTER VARYING,
        PlayerName CHARACTER VARYING,
        WeeklyWage CHARACTER VARYING
);
```

Q1.1. (10 points) Rewrite the above relation as a XML DTD.

Q1.2. (5 points) Write an XPath expression based on the DTD to return the player-names whose weekly wages are more than 200,000 Dollars.

Q2: AWS MapReduce [85 points + 10 points *bonus*]

In this problem, we will learn how to use Hadoop to generate degree distributions of a large graph. The idea is to convince you that using Hadoop on AWS has now really become a low-cost/effort proposition (compared to setting up your own cluster).

Familiarize yourself with AWS (Amazon Web Services). **Read the set-up guidelines posted on the website to set up your AWS account and redeem your free credit ($100)---do this early!**
Link: http://courses.cs.vt.edu/~cs4604/Spring14/homeworks/hw5/AWS-setup.pdf

The pricing for various services provided by AWS can be found at http://aws.amazon.com/pricing/. The services we would be primarily using for this assignment are the Amazon S3 storage, the Amazon Elastic Cloud Computing (EC2) virtual servers in the cloud and the Amazon Elastic MapReduce (EMR) managed Hadoop framework. Play around with AWS and try to create MapReduce job flows (not required, or graded) or try the sample job flows on AWS.

The questions in this assignment will ideally use up only a **very small fraction of your $100 credit**. AWS allows you to use up to 20 instances total (that means 1 master instance and up to 19 core instances) without filling out a "limit request form". For this assignment, **you should not exceed this quota of 20 instances**. You can learn about these instance types by going through the extensive AWS documentations. Of course, after you are done with the HW, feel free to use your remaining credits for any other fun computations/applications you may have in mind! These credits are applicable more generally for AWS as a whole, not just MapReduce.

We will use data from LiveJournal (http://www.livejournal.com/): it is a free on-line community with almost 10 million members; a significant fraction of these members are highly active. (For example, roughly 300,000 update their content in any given 24-hour period.) LiveJournal allows members to maintain journals, individual and group blogs, and for people to declare which other members are their friends. So we can get a directed social network (as friendships need not be reciprocated) among the LiveJournal users e.g. if A has declared B as a friend, then there is a directed edge from user A to user B in the social network.

We have uploaded the LiveJournal (LJ) friendship network (a directed graph) to the following Amazon S3 bucket (directory):

s3n://cs4604-2014-vt-cs-data/livejournal/

Refer to the guidelines to see how to set this data as input to your MapReduce job (Section 6 in the guidelines). We have provided a screenshot to configure the EMR cluster, which demonstrates how to access the input data from the given bucket.

We have stored the graph as an edge-list. As it is a directed graph, each unordered pair of nodes is saved once. The format of the data is:

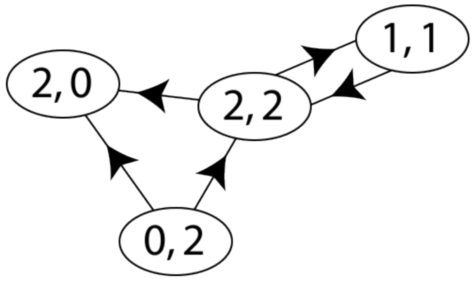
FromNodeId *ToNodeId* (tab separated)

This means that there is an edge from Node #*FromNodeId* to Node #*ToNodeId.* Hence a data snippet will look like:

```
0	1
0	2
0	3
0	7
1	0
1	7
......
```

The data has # nodes N = 4847571.

Recall the definitions of in and out-degrees of a node. For example, in the toy directed graph below, we have labeled each node with its (in-degree, out-degree).



Toy Directed graph. Nodes labeled with (in-degree, out-degree)

In simple terms, in-degree is the number of times a node appears in the *ToNodeId* of the given edge-list and out-degree is the number of times a node appears in the *FromNodeId* of the given edge-list.

We want to get the in-degree and out-degree distributions of the LJ network i.e.:

In-degree Distribution

$P_{in}(k) =$ fraction of nodes in the graph with in-degree $k > 0$

Out-degree Distribution

$P_{out}(k) =$ fraction of nodes in the graph with out-degree $k > 0$

Thus if there are N nodes in the network and n_k^i of them have in-degree k, we have $P_{in}(k) = n_k^i / \mathrm{N}$ (for each k). Similarly, if n_k^o of them have out-degree k, we have $P_{out}(k) = n_k^o / \mathrm{N}$ (for each k). The in-degree distribution for the example network above is: [(1, 1/4), (2, 2/4)] and the out-degree distribution is: [(1, 1/4), (2, 2/4)] (note that the in- and out- degree distributions are the same here).

While this is an easy task in general, we want to use Hadoop/MapReduce because of the large size of the graph. You can run your code using the AWS console by following the instructions in AWS-Setup (the easier way). You are also welcome to use the elastic-mapreduce command-line interface based on Ruby (read the responding part in AWS-Setup document).

Q2.1. (10 points) How many edges are present in the LiveJournal graph (just write the number)? Note: you can write a simple MapReduce program to calculate this on AWS, similar to checking for the number of nodes.

Q2.2. (10 points) Before computing the actual empirical distributions, write down your expectations of them i.e. what probability distributions do you expect to find for the in-degree and out-degree of this LJ social network (e.g. Gaussian? Or Exponential? Or Poisson? etc.)? Explain in 1-2 lines.

Q2.3. (30 points) Write the mapper and reducer in Python to calculate the in-degree distribution $P_{in}(k)$ for the LJ graph. Run it on the AWS and copy the output to your local machine and then plot $P_{in}(k)$ *(y-axis)* vs k *(x-axis)* in log-log scales. Is the distribution skewed?

Q2.4. (30 points) Write the mapper and reducer in Python to calculate the out-degree distribution $P_{out}(k)$ for the LJ graph. Run it on the AWS and copy the output to your local machine and then plot $P_{out}(k)$ *(y-axis)* vs k *(x-axis)* in log-log scales. Is the distribution skewed?

Q2.5. (5 points) Do your answers in Q2.2 match with what you actually got in Q2.3 and Q2.4? If not, did any of your prior assumptions fail? (Explain in 1-2 lines).

Q2.6. (BONUS 10 points) Is there any analytical function y = f(x) that will 'fit' the empirical distributions you found in Q2.3 and Q2.4? Write down the functional form f(x), and show the fit (i.e. plot the function over your empirical distributions). You may have to manually guess and set the 'correct' parameters in the function to match the empirical distribution approximately e.g. the general functional form of a line is y = mx + c, so if you want to fit a line to some given data, you need to guess some suitable m and c. Feel free to search the web for this question.

Hint 1: You will need two consecutive MapReduce jobs for each of Q2.3 and Q2.4. It might help to first write down the two SQL queries you will need on the edge-list table to get say the in-degree distribution and then translate them to two consecutive MR jobs (and similarly for the out-degree distribution).

Hint 2: Feel free to use the fact that you are given N = 4847571. Hence first just get the *number of* nodes with in-degree k (for each k) using AWS, and then simply divide everything by N while plotting on your local machine. Similarly for the out-degree distribution.

Note: The deliverables for Q2.3 include in-mapper-1.py, in-mapper-2.py, in-reducer-1.py, in-reducer-2.py, in-degree-dist.out (the raw final output from AWS) and in-degree-dist.jpg (similarly for out-degree in Q2.4). Zip all of these as YOUR-LASTNAME.zip. Then to submit the file:

1. Login to the Scholar Website.
2. Go to Assignment list for this class.
3. Click on HW5.
4. Click 'Add Attachment' and then submit this zip file.

Email Qianzhou (qiand12@vt.edu) if you face any issues.

Also include these in the hard-copy (everything except the raw .out files).