CSE-E5430 **Autumn 2014**
Scalable Cloud Computing P
Home Assignment 2 - Deadline: 25th of November 2014 at 10:15(strict deadline!)

Return your answer via email to the course assistant with "Home Assignment 2, [student number]" as the subject. Attach to the email a zip or tar file, packaging all your answer files inside the directory "`assignment-2`". A template .tar file for the home assignment 2 can be found from:
`https://noppa.aalto.fi/noppa/kurssi/cse-e5430/harjoitustyot`
under Assignment 2. Download it, unpack the files, and modify them to contain your answers. When you are done, pack the files to a tar or zip file. On UNIX workstations (including the linux virtual machine) this can be done with the command:
"`tar cvf assignment-2-answer.tar assignment-2`", when you have an existing directory "`assignment-2`" with your answer files in it. Then send the package to the course email address with the subject mentioned above. Submissions that arrive late are not graded! Be sure to send your answer in time, and remember that it may take some time for email messages to arrive. The course assistant will send you a confirmation once he receives your submission.
The home exercises are personal, no group work allowed! There are two rounds of home exercises of 10 points each. To pass the home exercises ≥ 10 points are needed and ≥ 16 points gives a +1 to the exam grade (no effect to exam grades 0 or 5).
The home assignments require you to have a working Apache Hadoop installation, please see the following Noppa page on how to have Hadoop set up:
`https://noppa.aalto.fi/noppa/kurssi/cse-e5430/hadoop_setup`
Note the hints and guides for further information towards the end of this document.

For this assignment assume that you were hired by a company that runs a social networking platform. Users can sign up free-of-charge to the system and connect to friends by sending friend requests, which means that the system establishes links between them (i.e., they appear on each other's *buddy list*). Users can also send wall posts to each others profile page, which are then visible to all users. Note that users generally do not need to be friends in order to post to each other's walls.
Since the company makes its revenue by selling advertising space in its web platform, it is very interested in learning about the *social graph* that users establish by friend-links and also the interaction by communication between users in order to draw more traffic to advertisers. Therefore, it is your task to answer two questions in this context:

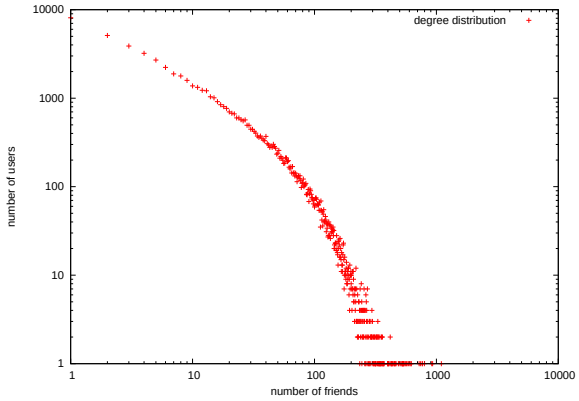
- What is the distribution of the *node degree* (that is, the number of friends) in the user population?
- What is the relationship between the number of friends a user has and the number of *wall posts* the user sends in a fixed time period (note: users do not need to be friends in order to send wall posts to each others and a user can write on his/her own wall)?

You are given two data sets, which you find in Noppa to download:

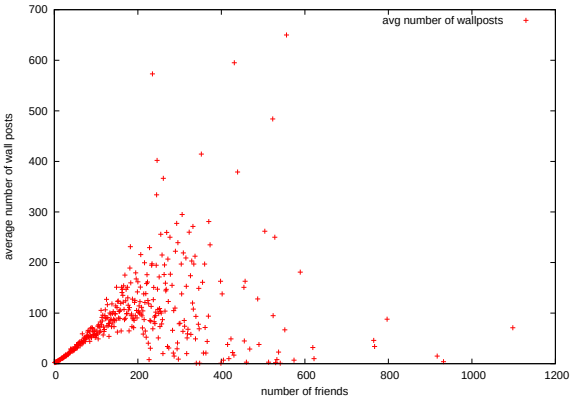
`user-links-small.txt.gz`	Each line of this first dataset contains two user-id's that establish an undirected link (friend-relationship) between the two.
`user-wall-small.txt.gz`	Each line of this second dataset contains three values, the first two being user-id's while the third one is a time-stamp. An entry `user1 user2 t` indicates that `user1` wrote on the wall of `user2` at time `t` (note: `user1` = `user2` is possible).

Both datasets share the same space of user-id's. However, there may be user-id's that are in the first and not in the second set and vice-versa. Both datasets are a subset of the data available at `http://socialnetworks.mpi-sws.org/data-wosn2009.html`, which is anonymized version of data collected from a subset of the users of a certain well-known social network.

1. *The advertisers are interested in finding out whether it would be worth contacting a few selected users to offer them discounts, hoping to interest a large fraction of the users in their product.*



(a) Question 1



(b) Question 2

Figure 1: Plots showing the end result for a larger dataset than the one that was distributed. For the source of this data see the text. **Note:** You do not need to re-produce these plots. Your task is to write Pig scripts that produce the data that this plots are based on.

Write a Pig script that computes the distribution (in order to draw a histogram to show to the advertisers) of the node degree in the social graph and writes this data into a text file. That is, your output should contain (**in increasing order of degree**) lines of the form
`degree count`
where `degree` is a value of node degree observed and `count` is the number of users in the dataset that have this node degree (i.e., this number of friends).

Add your script to the directory "`assignment-2/question-1`", together with the **output file you generated**. Also add the logging output generated by running the script to the file "`assignment-2/question-1/question-1-log.txt`". Note that when using the Grunt shell you can also cut-and-paste the Pig commands into this file.

2. *An important advertiser plans the launch of a new product. The advertiser asks the question whether informing (or sending incentives to) a few selected users would create a sufficient amount of "buzz" right in time before the planned date of the product launch.*

   Write a Pig script that computes for the time period that is covered by the data in the second data the following statistic **for user-id's that appear in both datasets**. Consider the set of users U_D with fixed node degree D. Among these users, compute the **average number of wall posts** that the users in U_D have made during the time time period in question. The scripts should store this value, for each observed node degree, into a file **in increasing order of degree**. That is, the output consists of lines of the form
   `degree value`
   where `degree` is a value of node degree observed and `value` is the average number of wall posts a user with degree `degree` has made (to their friends, to themselves, or to someone totally different). Note that **all** wall posts, including those made to users that are not in the first dataset **are to be taken into account**.

   Add your script to the directory `assignment-2/question-2`, together with the **output file you generated**. Also add the logging output generated by running the script to the file `assignment-2/question-2/question-2-log.txt`. Note that when using the Grunt shell you can also cut-and-paste the Pig commands into this file.

The data you computed can then be used to generate the plots shown in Figure 1 (you do not need to do this yourself!), which were generated from the complete dataset due to Viswanath et al. available at the webpage given above. Note that the distribution of the node degree shows the characteristic appearance of a so-called *scale-free* network, which has mostly nodes with small degree and a small number of hubs with much large degree than the average. The second plot also confirms the intuition that people with more friends also tend to send more wall posts.

Hints and further information

For more information on Pig, its installation into the Hadoop virtual machine and example scripts see Tutorial 5 and its set of demo solutions. Recall that you need to start Hadoop first (as in the WordCount example) before running Pig scripts or starting the Grunt shell. A small tutorial is part of any Pig distribution (see the `tutorial` folder inside the release). Here are further pointers.

`http://pig.apache.org/` Web page of Apache Pig
`http://pig.apache.org/docs/r0.11.1/start.html#tutorial` Pig tutorial (note: the tutorial files are inside the `pig-0.11.1.tar.gz` file but Pig's local mode, that is via the flag "-x local", may cause problems inside the older virtual machine image; the rest however should work just fine)
`https://noppa.aalto.fi/noppa/kurssi/cse-e5430/viikkoharjoitukset/CSE-E5430_tutorial_5__29.10-2014.pdf` Tutorial 5 and its demo solution, which are highly relevant

`http://pig.apache.org/docs/r0.12.0/basic.html`
`http://pig.apache.org/docs/r0.12.0/func.html`
Pig built-in functions and expressions

Note that Pig provides a large number of functions for processing sets of data tuples, for example aggregation functions such as computing the minimum, maximum or average of values over a given set.

Installing Pig inside the virtual machine

If you are using new virtual machine image, you may skip this steps since Pig is already installed. However, if you use older virtual machine image or you own setup, do the followings.

```
hadoop@ubuntu-vm:~$ cd /home/hadoop
hadoop@ubuntu-vm:~$ wget http://www.nic.funet.fi/pub/mirrors/apache.org/pig\
/pig-0.12.0/pig-0.12.0.tar.gz
hadoop@ubuntu-vm:~$ tar xvpzf pig-0.12.0.tar.gz
```

Similarly, you can download the files needed for completing the assignments straight from Noppa:

```
hadoop@ubuntu-vm:~$ wget https://noppa.aalto.fi/noppa/kurssi/cse-e5430/\
harjoitustyot/CSE-E5430_assignment-2-files.tar.gz
hadoop@ubuntu-vm:~$ tar xvpzf CSE-E5430_assignment-2-files.tar.gz
```

Note: the two datasets are in the `/data` subdirectory.
In order to run anything in the Pig shell or execute Pig scripts you need to have Hadoop up and running. Assuming you are currently in the directory `/home/hadoop`, you can start Hadoop as earlier via (note the ".")

```
hadoop@ubuntu-vm:~$ . WordCount/start_commands
```

Once Hadoop is up (there is one datanode alive), the Pig Grunt shell can then be started by

```
hadoop@ubuntu-vm:~$ pig-0.12.0/bin/pig
```

If you have created a file `script1.pig` that contains Pig command, you can execute all of these directly by

```
hadoop@ubuntu-vm:~$ pig-0.12.0/bin/pig script1.pig
```

For more information see the documentation of Pig under the link above.

Data import

Note that you can import both data sets into Pig even without uncompressing them:

```
grunt> A = LOAD 'user-links-small.txt.gz' as (user_a: int, user_b: int);
```

and similarly

```
grunt> B = LOAD 'user-wall-small.txt.gz' AS (user_a: int, user_b:int, time: int);
```

If you want to look at the data yourself, you can uncompress it and view it on the normal command line via

```
hadoop@ubuntu-vm:~$ gunzip user-wall-small.txt.gz
hadoop@ubuntu-vm:~$ less user-wall-small.txt
```

Debugging your scripts

For debugging purposes it may be convenient to work on a smaller subset of the two datasets. In this case Pig's LIMIT command comes in handy.

```
grunt> A = LOAD 'user-links-small.txt.gz' as (user_a: int, user_b: int);
grunt> B = LIMIT A 20;
```

In this case `B` gets assigned the first 20 elements of `A`. Similarly, SAMPLE can be used to obtain a random sample from a given set. DUMP can be used to dump its contents to the screen. See Pig's documentation for more details.

Data export

Assume that the Pig variable Z contains your result for the first problem. then by

```
grunt> STORE Z into 'question-1-output';
```

the output can be stored to HDFS as a text file, which can then be accessed as usual. Please add the contents of this file to the directory, which you package and send for submission.

Logging output

By redirecting the output you can directly produce logging output as required above. For example, assuming you are in the directory `assignment-1/question-1`, by executing inside the normal shell terminal as a single command

```
hadoop@ubuntu-vm:~$ /home/hadoop/pig-0.12.0/bin/pig \
   script1.pig > question-1-log.txt 2>&1
```

where `script1.pig` is your solution to the first question, you are re-directing the command line output of Pig to the file question-1-log.txt, as required. If you prefer using the Grunt shell, it is of course also fine to cut&paste the output from the terminal to the log file.