Cluster and Cloud Computing Assignment 1 – Social Media Analytics

Problem Description

Your task in this programming assignment is to implement a parallelized application leveraging the University of Melbourne HPC facility SPARTAN. Your application will process a large Twitter dataset. The objective is to identify the:

- happiest hour ever in the data,
- happiest day ever in the data,
- most active hour ever, i.e. the hour that had the most posts in the data,
- most active day ever, i.e. which day had the most posts in the data.

You should be able to log in to SPARTAN through running the following command:

ssh your-unimelb-username@spartan.hpc.unimelb.edu.au

with the password you set for yourself on *karaage* (https://dashboard.hpc.unimelb.edu.au/karaage). Thus, I would log in as:

ssh rsinnott@spartan.hpc.unimelb.edu.au
password = my karaage password (not my UniMelb password)

If you are a Windows user then you may need to install an application like Putty.exe to run *ssh*. (If you are coming from elsewhere with different firewall rules, then you may need to use a VPN).

The files to be used in this assignment are accessible at:

- /data/gpfs/projects/COMP90024/twitter-100gb.json
  - this is the main file to use for your final analysis and report write up, i.e., **do not use this file for software development and testing**. It is 120Gb of JSON and will take a long time to process.
- /data/gpfs/projects/COMP90024/twitter-50mb.json
  - is a smaller (59Mb) JSON file that should be used for testing.
- /data/gpfs/projects/COMP90024/twitter-1mb.json
  - is a small (1.2Mb) JSON file that should be used for initial testing.

You may decide to use the smaller JSON files on your own PC/laptop to start with for development and testing.

You should make a symbolic link to these files on SPARTAN, i.e., you should run the following commands at the Unix prompt **from your own user directory on SPARTAN**:

```
ln –s /data/gpfs/projects/COMP90024/twitter-100gb.json
ln –s /data/gpfs/projects/COMP90024/twitter-50mb.json
ln –s /data/gpfs/projects/COMP90024/twitter-1mb.json
```

(Do not cut and paste these commands as it will include the carriage return. Type them in manually).

Once done you should see something like the following **in your home directory**:

```
lrwxrwxrwx 1 rsinnott unimelb  40 Mar 22 15:06 twitter-100gb.json-> /data/gpfs/projects/COMP90024/twitter-100gb.json
lrwxrwxrwx 1 rsinnott unimelb  40 Mar 22 15:06 twitter-50mb.json-> /data/gpfs/projects/COMP90024/twitter-50mb.json
lrwxrwxrwx 1 rsinnott unimelb  40 Mar 22 15:06 twitter-1mb.json-> /data/gpfs/projects/COMP90024/twitter-1mb.json
```

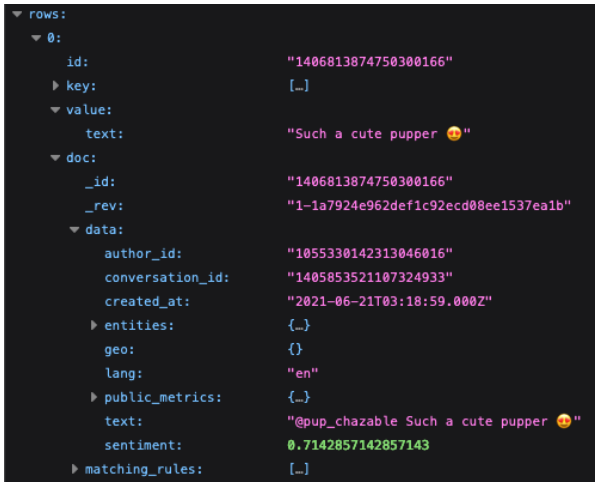
Your assignment is to (*eventually*!) search the large Twitter data file and identify:

- the happiest hour ever, e.g. 3-4pm on 23rd November with an overall sentiment score of +12,

- the happiest day ever, e.g. 25th February was the happiest day with an overall sentiment score of +23,
- the most active hour ever, e.g. 4-5pm on 3rd March had the most tweets (#1234),
- the most active day ever, e.g. 3rd October had the most tweets (#12345),

Noting that these dates and the sentiment scores are representative! There is no need for deeper statistical analysis of the data, e.g. averaging the number of posts per hour/day or the average sentiment etc.

You will note that the vast majority of tweets have a sentiment score already, e.g. the first tweet in the JSON array:



Your task is to use the *created_at* attribute (when the tweet was made) and the pre-calculated sentiment score to answer the bullet points above. Tweets that do not have the sentiment attribute can be ignored.

Your application should allow a given number of nodes and cores to be utilized. Specifically, **your application should be run once** to search the *largest JSON* file on each of the following resources:

- 1 node and 1 core;
- 1 node and 8 cores;
- 2 nodes and 8 cores (with 4 cores per node).

The resources should be set when submitting the search application with the appropriate *SLURM* options. Note that you should run a single *SLURM* job three separate times on each of the resources given here, i.e. you should not need to run the same job 3 times on 1 node 1 core for example to benchmark the application. (This is a shared facility and this many COMP90024 students will consume a lot of resources especially with the large file!).

You can implement your solution using any routines and libraries you wish however it is strongly recommended that you follow the guidelines provided on access and use of the SPARTAN cluster. Do not for example think that the job scheduler/SPARTAN automatically parallelizes your code – it doesn't! You may wish to use the pre-existing MPI libraries that have been installed for C, C++ or Python, e.g., mpi4py. You should feel free to make use of the Internet to identify which JSON processing libraries you might use. You may also use any regular expression libraries that you might need for string comparison.

Your application should return the final results and the time to run the job itself, i.e. the time for the first job starting on a given SPARTAN node to the time the last job completes. You may ignore the queuing time. The focus of this assignment is not to optimize the application to run faster, but to learn about HPC and how basic benchmarking of applications on a HPC facility can be achieved and the lessons learned in doing this on a shared resource.

Final packaging and delivery

You should write a brief report on the application – **no more than 4 pages**!, outlining how it can be invoked, i.e. it should include the scripts used for submitting the job to SPARTAN, the approach you took to parallelize your code, and describe variations in its performance on different numbers of nodes and cores. Your report should include the actual results tables as outlined above and a single graph (e.g., a bar chart) showing the time for execution of your solution on 1 node with 1 core, on 1 node with 8 cores and on 2 nodes with 8 cores.

Deadline

The assignment should be submitted to Canvas as a zip file. The zip file must be named with the students named in each team and their student Ids. That is, *ForenameSurname-StudentId:ForenameSurname-StudentId* might be *<SteveJobs-12345:BillGates-23456>.zip*. Only one report is required per student pair and only one student needs to upload this report. The deadline for submitting the assignment is: **Friday 12th April (by 12 noon!)**.

It is strongly recommended that you do not do this assignment at the last minute, as it may be the case that the Spartan HPC facility is under heavy load when you need it and hence it may not be available! You have been warned…!!!!

Marking

The marking process will be structured by evaluating whether the assignment (application + report) is compliant with the specification given. This implies the following:

- A working demonstration – **60% marks**
- Report and write up discussion – **40% marks**

Timeliness in submitting the assignment in the proper format is important. **A 10% deduction per day will be made for late submissions.**

You are free to develop your system where you are more comfortable with (at home, on your PC/laptop, in the labs, on SPARTAN itself - but not on the largest file until you are ready!). Your code should of course work on SPARTAN.