Cluster and Cloud

Computing Assignment 1

**Samir Patel(968621)-** [**samirashwini.patel@student.unimelb.edu.au**](mailto:%20samirashwini.patel@student.unimelb.edu.au)

**Amar Babuta(1031452)-** [**amar.babuta@student.unimelb.edu.au**](mailto:amar.babuta@student.unimelb.edu.au)

**HPC Twitter GeoProcessing**

This report describes different strategies, actions and approaches to solve a problem of searching for a broad Twitter dataset in order to identify Melbourne - wide tweet spots using high - performance computer facilities in University of Melbourne(Spartan).

**Brief introduction**

This project is designed to solve the problem of finding a large dataset of high performance (HPC) geocode files called Spartan from any of 16 possible geography or from outside. This geocode is a high performance file. We need to create a program to search each box, each row and column for a number of tweets. This is an extremely computer-intensive task because we have about 10 GB of tweet files and we use HPC to solve this. As Python provides a very potent programming experience, python programs are also comparatively compact and easy to understand. For implementing the Message Passing Interface (MPI), I used mpi4py external package. For the parallel software application, MPI is a strong and wide utilized system. Sometimes it is the de facto industry standard, and as an open source implementation it is widely available.

We have a huge data set which our personal laptop CPU can't process (it will take a few days). Therefore, we use spartan for that. Spartan is a multi-core hybrid computer with more than 400 virtual machines on a cloud partition, with more than 3,000 cores. There's no queuing system there directly at the shared computer facility. That allocates resources to work as a priority and as a necessity. SLURM is a job programmer for the system queuing. Simple Linux utility for resource management. We must write and submit a.slurm/.sh file with this system's sbatch command.

**Parallel Routing with MPI**

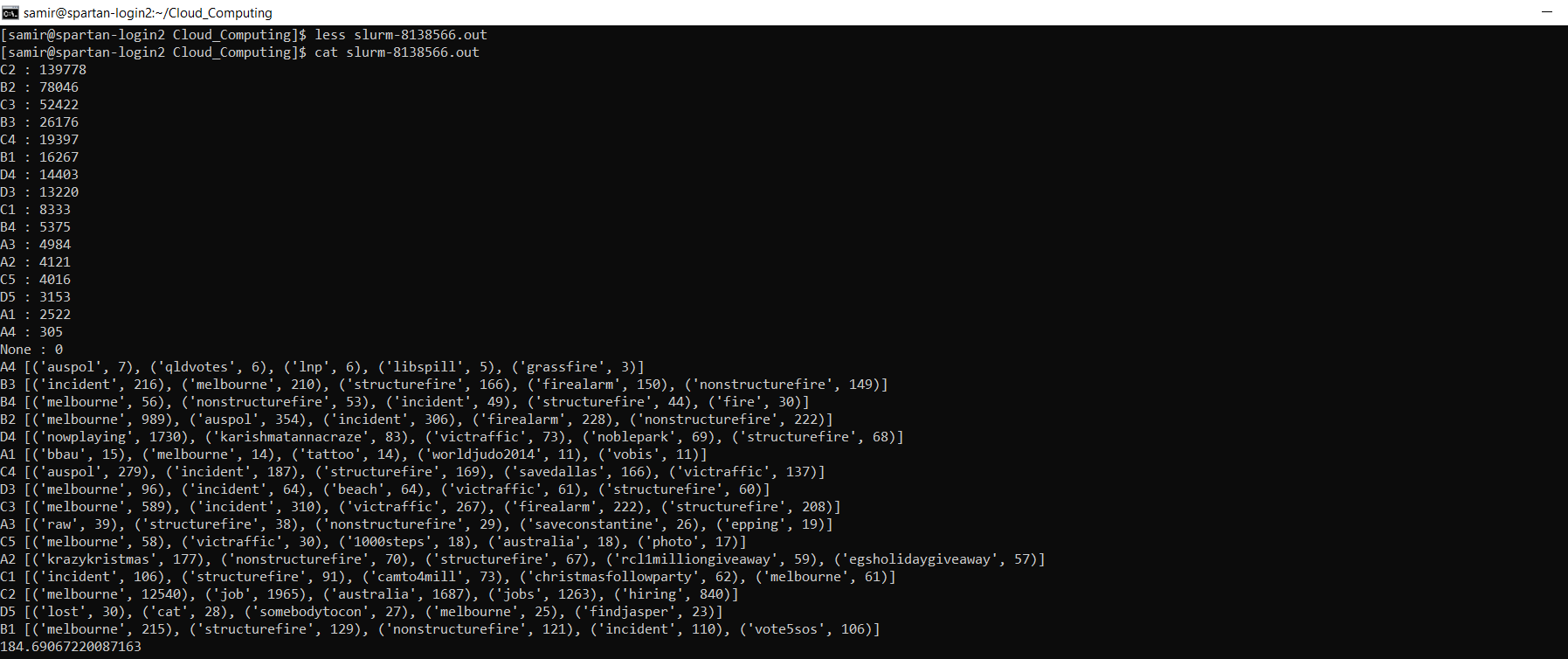
I used steps below to find the final solution after trying various strategies:

1. First step is to link data files to folders symbolically.
2. So I need to install a package for running and testing the MPI, using Python as programming language.
3. Write.sh script to run this program on Spartan.
4. The data to be processed are JSON format, the entire JSON can not be retrieved and RAMed, so I read the large json file from doc, coordinates, coordinates to handle this approach and make JSON out of the line and the location for each parser.
5. After the JSON is parsed, it will be appended to a list of location lists.
6. This enormous list is then divided into different parts depending on several processes to obtain quicker results.
7. Each process is included in the list and finds from it all relevant geolocation.
8. The collection method adds output processed by each process and heapq finds top 5 hashtags. In order to extract various outcomes like number of tweets and five most frequent hashtags made in each box, we have converted the result into a dictionary and processed the dictionary.

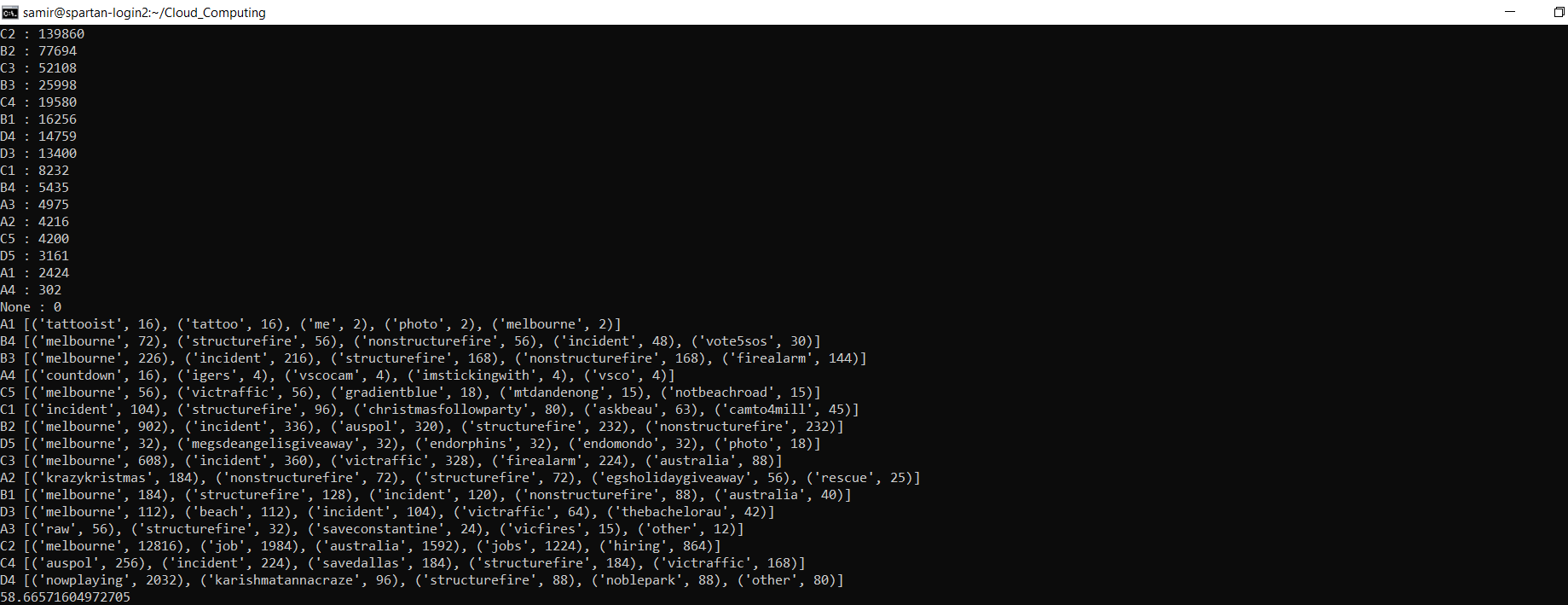
**Result Analysis:**

**Performance on different number of nodes and cores**

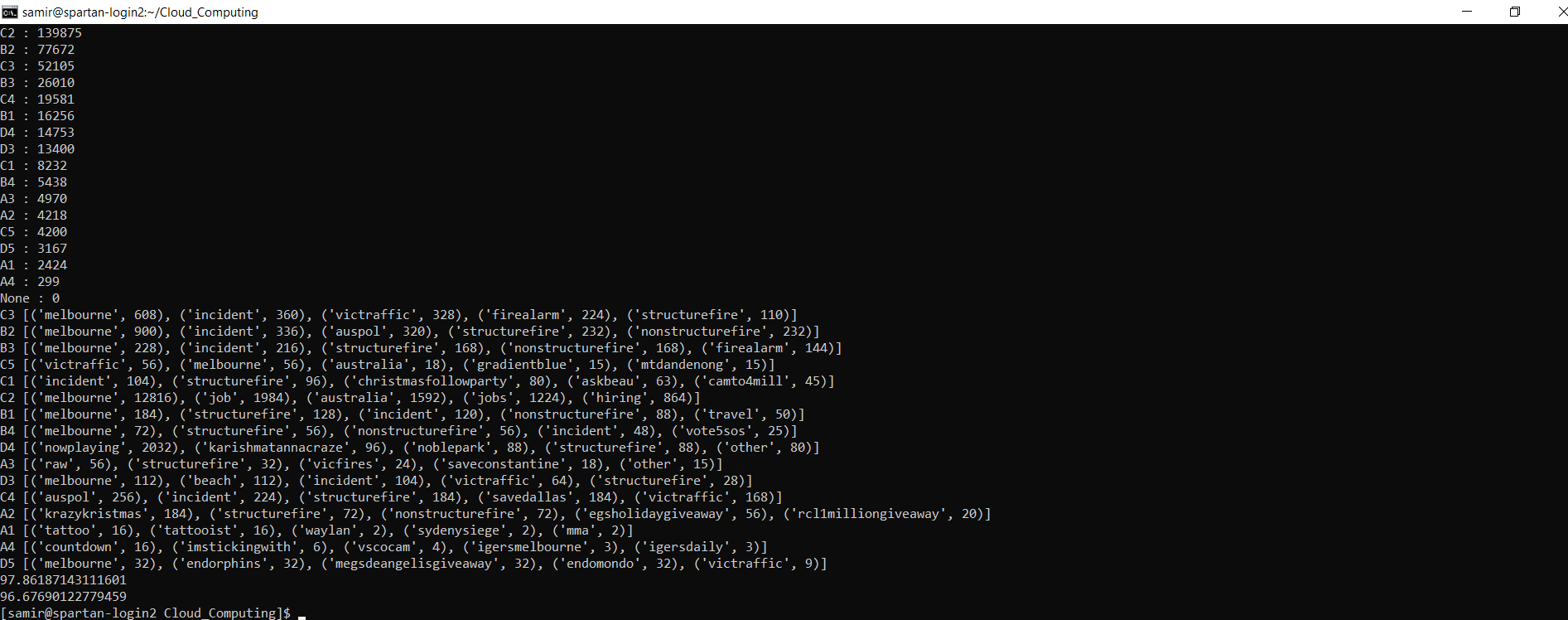
-\*\*\*\*Result for 1 node 1 core (Time Taken to run bigTwitter 184.69sec)\*\*\*\*--



--\*\*\*\*Result for 1 node 8 core (Time Taken to run bigTwitter 58.66sec)\*\*\*\*--



--\*\*\*\*Result for 2 node 8 core (Time Taken to run bigTwitter on 96.68 sec)\*\*\*\*--





C2

139778

B4

5375

C1

8333

D3

13220

D4

14403

C3

52422

C4

19397

C5

4016

D5

3153

B3

26176

B2

78046

B1

16267

A2

4121

A3

4984

A4

305

A1

2522