COMP90024 Cluster and Cloud Computing Assignment1

HPC Twitter GeoProcessing

Xinjie Lan, Liuyi Chai

# 1.Introduction

The project is to design a simple paralyzed application to identify Twitter usage around Melbourne and calculate the most frequently occurring hashtags that are being sent from those areas. The application allows a given number of nodes and cores to be utilized. This report would briefly describe the approach we took to parallelize our code and analyze the performances of executing the solution on different resources.

**1.1 Tools**

• The University of Melbourne HPC facility SPARTAN

• Python

**1.2 Dataset**

• bigTwitter.json

This file includes the coordinates of tweets.

• melbGrid.json

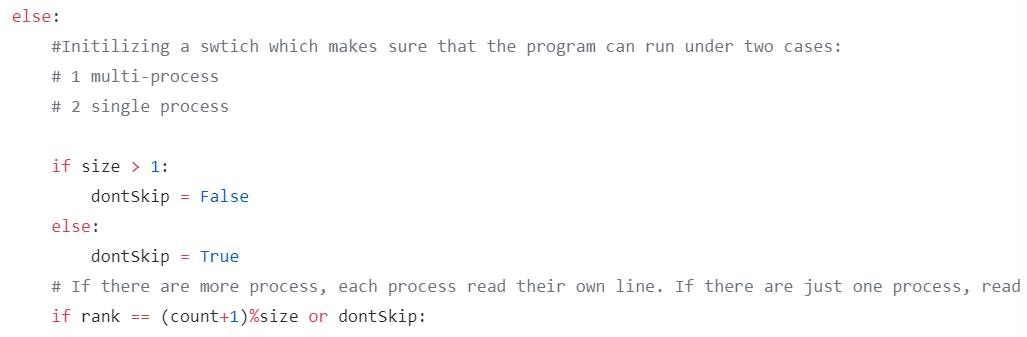
This file includes the coordinates of a range of gridded boxes.

# 2.Description of the python program

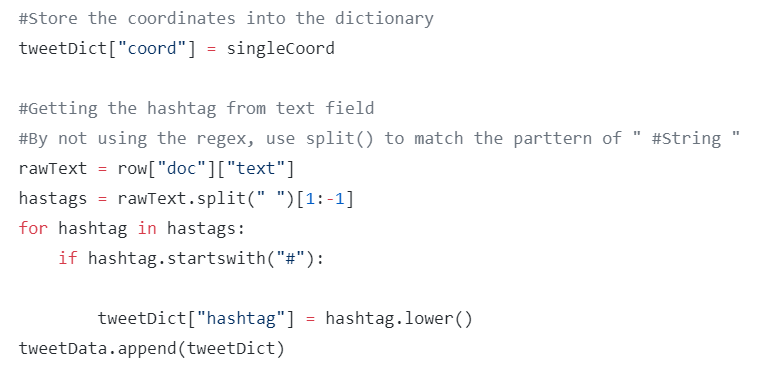
**2.1 Load and parse the grid file**

The python file has three parts. The first part aims to load the melbGrid file and parse the data. To implement this, we stored the grid data into a dictionary which contains each grid cell’s ID and their range of coordinates.

**2.2 Parallel parse the Twitter file**



Screenshot 1



Screenshot 2

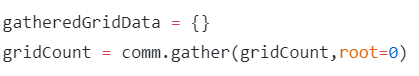
The main job of the second part is to parallel parse the bigTwitter file. When there is only one node, the program should read the file line by line, while in other cases when multiple nodes exist it should skip some lines. By calculating the node size and defining a “dontSkip” Boolean, we determined whether the current line belongs to the current rank. If it doesn’t belong to the current ran, we skipped, otherwise parsed the data and then stored the coordinates and the hashtags of each tweet into a dictionary (shown in screenshot 1).

In case of rewriting the arguments in the dictionary we appended the dictionaries to a list in every loop turn (shown in screenshot 2)

**2.3 Parallel calculate the number of posts and hashtags in each cell**

After parsing the data in bigTwitter, we classified each coordinate into the cell it belongs to, and then parallel calculated the number of the posts as well as their hashtags in this cell.

**2.4 Gather data from all processes**



Screenshot 3

In part3, the master node gathers the data from all the processes by comm.gather method (shown in screenshot 3).

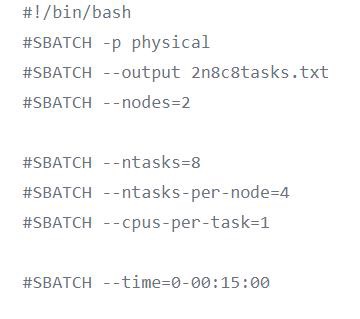
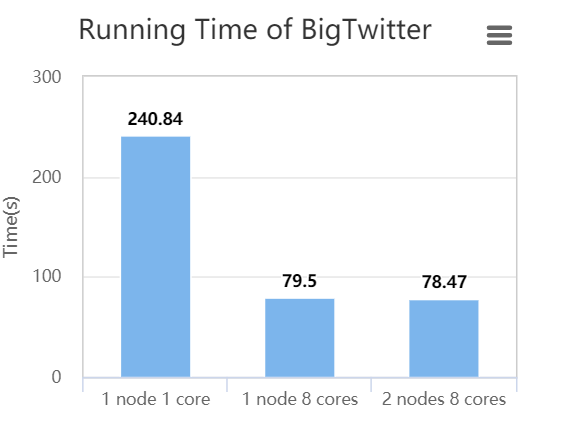
**2.5 Rank the list and Generate the result**



Screenshot 4

Since the gathered data is in a list, we firstly parsed the list datatype into dictionary type. From the dictionary, we counted the number of occurrences of the hashtags and stored the top5 into a post ranking list (screenshot 4). Finally, we generated the ranking result by sorting the list.

**3. Results**



Screenshot 5

Slurm Script Example 2 Nodes 8 Cores Chart 1

From the running time bar chart (Chart 1) we can see that the program consumed much more time on 1 node 1 core, and on 1 node 8 cores and 2 nodes 8 cores the running time decreased. While the difference of running time between 1 node 8 cores and 2 nodes 8 cores is slight.