THE UNIVERSITY OF MELBOURNE

Cluster and Cloud Computing Assignment 1

HPC Twitter GeoProcessing

Jigar Thakkar, 800301

04/04/17

This report explains various strategies, activities, and approaches performed to solve the problem of searching a large geocoded Twitter dataset to identify tweet hotspots around Melbourne using high-performance computing facility at the university of Melbourne (Spartan).

Jigar Thakkkar 800301

Brief introduction

This project is solving the problem of searching into a large dataset of twitter file geocode using high performance (HPC) facility named Spartan, this geocode is from one of any 16 possible geographic areas or from somewhere outside. We must write a program for searching for a number of tweets made in each box, each row, and each column. This is a very computationally intensive task as we have around 10 GB tweet file, and for solving this we are using HPC.

I have used python as a programming language because python provides very powerful programming experience, also python programs are comparatively compact and easy to understand. I have used mpi4py external package for implementing Message Passing Interface (MPI). MPI is very powerful and widely used system for the parallel software application. In a way, it is the de facto industry standard and it is widely available as open source implementation.

We have a huge dataset to process which can't be processed by our personal laptop's CPU (it'll take days to process). So that's why we are using spartan for that. Spartan is hybrid computing facilities with various cores at physical partition and over 400 virtual machines with over 3,000 cores at cloud partition. We can't run our program directly at shared computing facility there is a queuing system there. which assigns resources to the job according to their priority and need. SLURM (Simple Linux Utility for Resource Management) is job scheduler for queuing system. We have to write and submit .slurm/.sh file using sbatch command at this system.

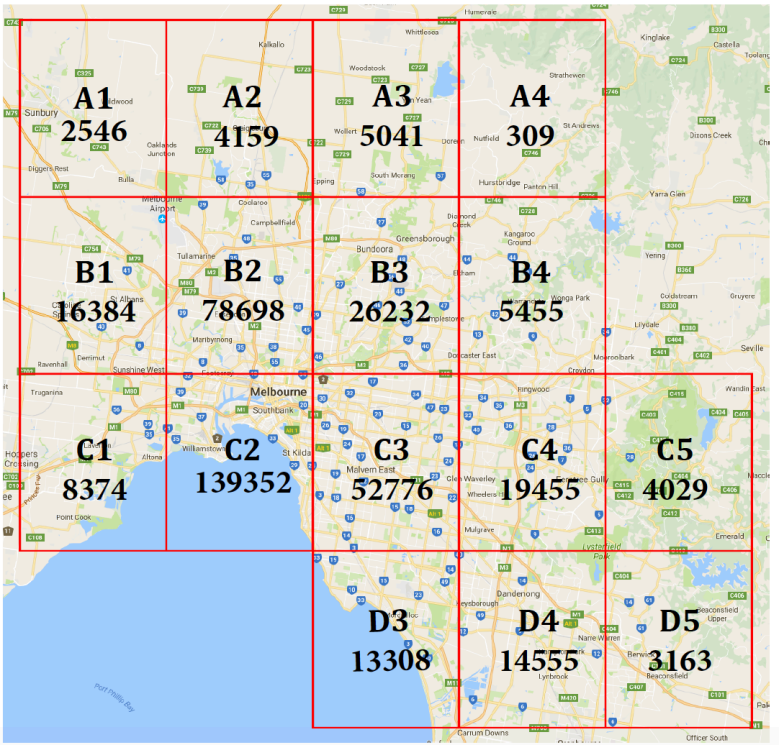
Parallel programming with MPI

After trying various strategies, I used below steps for final solution:

1) The first step is creating symbolic link of data file to over folder
2) I'm using python as Programming language so I must install mpi4py package to run and test MPI.
3) For running this program on Spartan we need to write .sh script (More Details of that are at later on report).
4) Data which needs to be processed is in JSON format, It is not possible to fetch the whole JSON and put into RAM, To handle this my approach is read the large file line by line and make JSON of each line and parse location from that line.
5) After fetching location list as line by line It will converted into a list of lists of location.
6) This huge list is then divided into various part depends on a number of processes.
7) Each process takes part of the list and finds all relevant geolocation from it.
8) After reaching barrier, gather method will aggregate result processed by each process. I have converted this result into dictionary, I have processed this dictionary to extract various result like number of tweets made in each box in descending order.

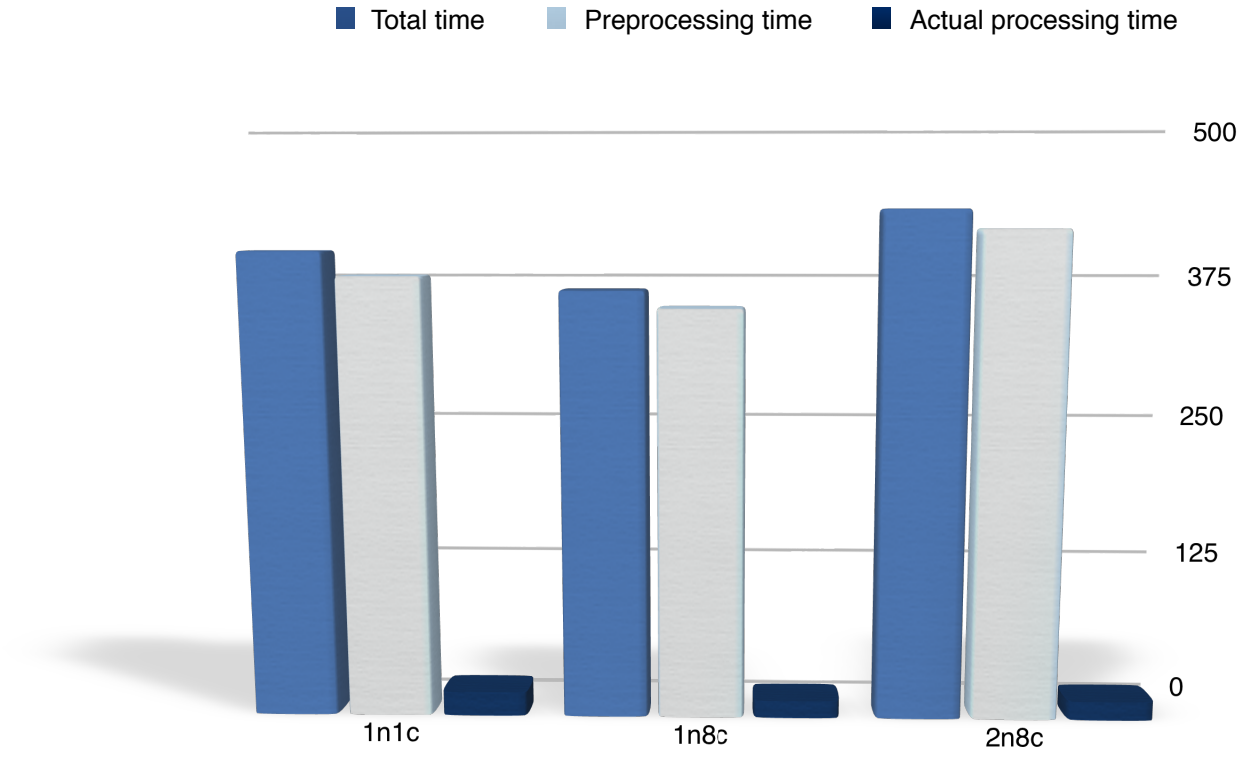
Result And Analysis

All result is generated into .out file once the program is completed. On the next page, the result is shown on the map. Analysis shown that near you go to CBD you more number of tweets are from there.



Row	Tweet
C-Row	223986
B-Row	126769
D-Row	31026
A-Row	12055

Column	Tweet
Column2	222209
Column3	97357
Column4	39774
Column1	27304
Column5	7192



In last graph:
1n1c means 1 node and 1 core;
1n8c 1 means 1 node and 8 cores;
2n8c means 2 nodes and 8 cores

Most of the time taken by the total process is for pre-processing data this includes reading from a file and catching location from each tweet and making a list of locations.Average actual processing time is around 8 seconds. In 8 seconds program finds from geo-location list that from which one of 16 box that location is from.

More CPU we allocate doesn't mean every time that program will run faster, It depends on various factors. I'm running program on a cloud partition for a faster-getting chance to the allocation of the resources. here preprocessing is a major bottleneck, Programme spending approximately 90% time on pre-processing which is sequential task here. But if we find a solution for that and make it parallel then program will be much faster. But for that, we have to create some index or convert dataset into some other format.

Invocation

To invoke 'geo.py' application on 'Spatan' requires a .slurm or .sh script.This script allocates resources to when it goes for processing. I have created 3 .sh files for this purpose, here is 1 node and 8 cores script file:

`#!/bin/bash`
This is fix for all shell script header.

`#SBATCH -p physical`
This line is to define on which partition you want to run your programme.

`#SBATCH --nodes=1`
Number of nodes you want to use for running your programme.

`#SBATCH --ntasks-per-node=8`
Number of tasks you want per node.

`#SBATCH --cpus-per-task=1`
CPU per task here 1 so total number of CPU use during this program is 8.

`#SBATCH --output=P1n8c`
Output name of the file.

`#SBATCH -t 0-1:00`
In case your job gets killed exceeding the specified walltime.

`module load Python/2.7.12-intel-2016.u3`
These will load the MPI and Python modules which are required for the execution of my application.

`time srun python geo.py`
Final run command for python file.