# OpenMP and Parallel Programming

Christopher Booth

Computer Engineering Department, University of Ottawa

Ottawa, ON Canada

Cboot095@uottawa.ca

Chenming Yu

Computer Engineering Department, University of Ottawa

Ottawa, ON Canada

byu083@uottawa.ca

Abstract— This paper describes our project using openMP parallel programming in order to parallel programming in multiple processes. In order to spend less time on searching for number of tweets without repeating, we created different number of parallel processes to fetch and save them in a text file. Also, openMP multithreading loops are developed to achieve the goal. The program runs based on the functions provided by digitalocean.

Keywords— openMP, parallel process, textblob, manager, dictionary, sentiment, multithread, digitalocean

I. INTRODUCTION

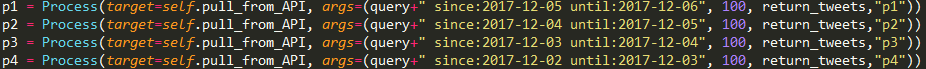
This paper is a final report for CEG4136 lab session as undergraduate students. In this lab, we uses cython codes to implement different programs to compare the time of searching different number of tweets in different methods. We run the simulations on digital ocean which allows for SSD-based cloud services for deploy and scale applications that run simultaneously on the computer. In order to retrieve the goal as reducing the time for searching the tweets, we used openMP multithreading method and parallel processes to decrease the searching time. What is more, we used different methods and implemented some algorithm to reduce the time for searching and illustrating tweets in a file. The comparison of time is also offered in the report.

II. DigitalOcean

[2] The program is run on the DigitalOcean platform which provides cloud services that help to deploy and scale applications that run simultaneously on multiple computers. It supports for restful API which allows to automatic and scale the works. The droplets deploy an SSD cloud server in seconds with intuitive control panel and flexible API.

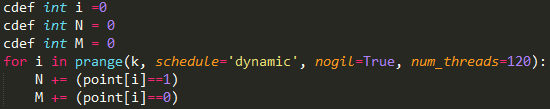
III. Parallel processes

The program is using parallel processes to get tweets from the Internet which does not use openMP. The program needs to get a great amount of tweets and parse them formally. It sets a pointer to count for the number of tweets and call twitter API to fetch tweets. In order to fetch the tweets faster, parallel processes are created to fetch tweets at the same time. The program creates the parallel processes in cython. Then the processes fetch the same amount of tweets and parse tweets one by one in the file. The processes are save to a shared dictionary to store required parameters of a tweet. [1] By parsing the tweets, it saves text of tweets which saves time for the searching. [3] On the other hand, the processes are able to save sentiments of tweets with the sentiment function in textblob. Then they append the parsed tweets to tweets list. During this process, the processors are able to figure out if the tweets has been searched and been appended in the text file. If the tweets are retweets, the processors can just drop the retweets since they are already on the list. Then the manager function can put tweets together and illustrates the non-repeating tweet list in the file.

We create 50 processes that each pull 100 tweets each process has an individual query passed by an argument. Ex.

IV. OpenMP Multithreading

In the main function of the program, open multi-processing is used for creating multithreading. [4] By using Cython, the program is able to run on both python and c language. Since the python environment does not support for multithreading because of global interpreter lock. Global interpreter lock is a mechanism to synchronize the execution of threads so that only one native thread is able to run at a time. After implementing openMP, the program can define the variables in c language through cython. It is now able to implement multithreading loop in c using c defined variables. By implementing multithreading loops, our program can run multiple threads to handle a lot of instructions. It would save a lot of time for huge amount of tweet searching if we were aggregating more tweets unfortunately twitter API can only collect 5000 tweets at a time and with such a small data set the overhead added by openMP in Cython outweighs the benefits of the multiple threads.. OpenMP multithreading loops are used to classify the type of tweets and print the tweets on the screen.



V. Python Functions

*A. Tweepy*

[5] Tweepy is an open-sourced application which enables python to communicate to twitter with twitter platform and use its API. In the code, the program create tweepy API objects to fetch tweets.

*B. Textblob (sentiment)*

[6] TextBlob is a python library for processing textual data. It provides a simple API for diving into common natural language processing tasks. In our code, we used textblob utility function for sentiment analysis. The program classifies sentiment of passed tweet using textblob’s sentiment method. It creates textblob objects of passed tweet text and set sentiment for each tweet.

*C.Manager(dictionary)*

Manager is imported from multiprocessing. A manager object returned by manager command controls a server process which holds python objects and allow other processes to manipulate them using proxies. In the program, shared memory dictionary put tweets which is searched by the parallel processors together using manager functions and return the tweets.

VI. CONCLUSION

For this lab, the primary objective is completed. Our program is able to find a great number of tweets containing the specified key word ‘engineering’. By using openMP multithreading and parallel processes, the program decreases the searching time for tweets. Serial computation is implemented in python and it is able to run on digitalocean platform. Unfortunately this program is not very scalable and we can never utilize the high instance CPU we use as a platform. We never manage to pass 1% CPU utilization, even though we experience significant speedup. So while the platform may have proved inappropriate and openMP have prohibitive overhead at this scale parallelism in python is definitely achievable.

ACKNOWLEDGEMENT

Thanks to our professor Mr. Miodrag Bolic and our lab TA Rajshekhar Mukherjee. Our TA helped us on the programming part and explain some cython functions when we are doing the lab. His comments in the source codes allowed us to understand the functions about searching tweets.

REFERENCE

[1] (2017) Using Parallelism. [Online]. Available:

<https://cython.readthedocs.io/en/latest/src/userguide/parallelism.html>

[2] DigitalOcean, cloud computing, simplified. [Online]. Available:

<https://www.digitalocean.com/products/>

[3] Parallel Processing and Multiprocessing n Python. [Online]. Available:

<https://wiki.python.org/moin/ParallelProcessing>

[4] Multiprocessing in Python with Fortran and OpenMP. [Online]. Available:

<http://www.admin-magazine.com/HPC/Articles/Multicore-Processing-in-Python>

[5] Introduction to tweepy, Twitter for Python. [Online]. Availavle:

<https://www.pythoncentral.io/introduction-to-tweepy-twitter-for-python/>

[6] Sentiment Analysis using TextBlob. [Online]. Available:

<https://wafawaheedas.gitbooks.io/twitter-sentiment-analysis-visualization-tutorial/sentiment-analysis-using-textblob.html>