

AKDENİZ UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING



Parallel Computing – Final Project

Temmuz Burak Yavuzer

20160808026

Problem

1. Write a parallel MPI application that finds 2-grams (bigram) in the news dataset (35 MB compressed) in a shortest time.
2. As an output you should provide the ngram and its frequency
3. You can use any thread or MPI function.
4. Applications should display the total time spent to load the file, total time to compute the 2-grams

What is N-Gram

N-gram is probably the easiest concept to understand in the whole machine learning space, I guess. An N-gram means a sequence of N words. Well, in Natural Language Processing, or NLP for short, n-grams are used for a variety of things. Some examples include auto completion of sentences (such as the one we see in Gmail these days), auto spell check (yes, we can do that as well), and to a certain extent, we can check for grammar in a given sentence. For instance, let us take a look at the following examples.

1. San Francisco (is a 2-gram)
2. The Three Musketeers (is a 3-gram)
3. He stood up slowly (is a 4-gram)

System and Environment that Used in the Experiment

In my experiment I used my computer which have 4 core. Environment is Ubuntu x64 18.04.4 via VirtualBox 6.1 which is using 11 GB of Ram with number of 4 core CPU, Video Memory is 128 MB of 6GB of the GTX1060. For the storage I used my HDD 5400RPM Sata3 128 Cache disk .

According to these specifications the number of process can be maximum 4. Parallel computation is increased consecutive form 1 to 4.

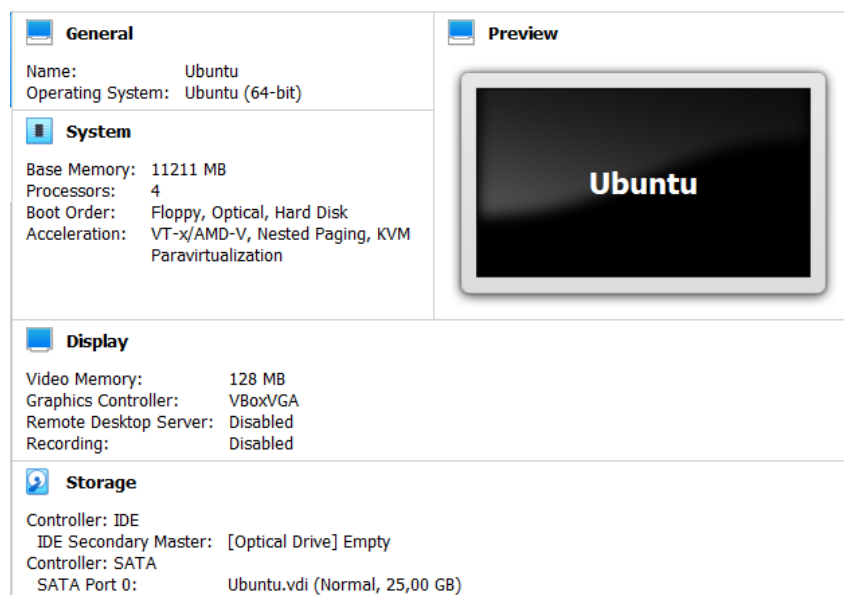


Figure 1.1 Ubuntu via VirtualBox

Installation/Envirionment and Usage of the Code

To run the code you need to have Python3

“sudo apt-get install python3”

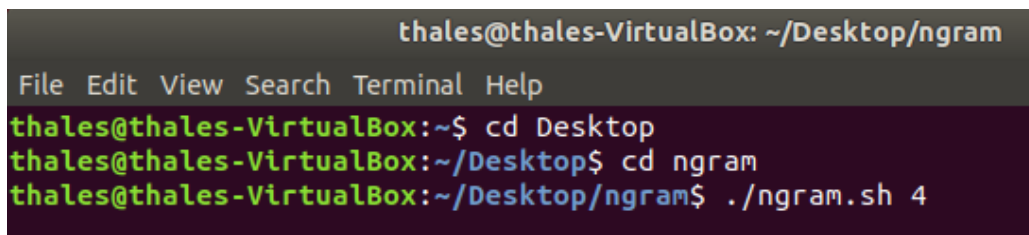
We need to have numpy library, to download that first we need to get pip

“ sudo apt get install python3-pip”

After all of that we are free to download the mpi4py library

“sudo apt-get update -y”

“sudo apt-get install -y python3-mpi4py”



```
thales@thales-VirtualBox: ~/Desktop/ngram
File Edit View Search Terminal Help
thales@thales-VirtualBox:~$ cd Desktop
thales@thales-VirtualBox:~/Desktop$ cd ngram
thales@thales-VirtualBox:~/Desktop/ngram$ ./nggram.sh 4
```

Figure 1.1 Ubuntu via VirtualBox

```
DATA="/home/thales/Desktop/ngram/Data"
OUTPUT="/home/thales/Desktop/ngram/Output/"
```

“Don’t forget to change the location for your computer.”

To use the code in Ubuntu, go to the directory where it is saved. After that, to execute the code.

use “./nggram.sh numberofprocess” command in terminal.

e.g “./nggram.sh 4”

Each Results for Each Number of Process

MPI With 1 Process

```
thales@thales-VirtualBox:~/Desktop/ngram$ ./ngram.sh 1
file loading time 1.6806999610707862e-05
Executing for /home/thales/Desktop/ngram/Data/text/news.txt
Compute time 62.630442250000215
Total time to complete 62.63043177899999
```

Figure 1.2 Result with MPI

MPI With 2 Process

```
thales@thales-VirtualBox:~/Desktop/ngram$ ./ngram.sh 2
file loading time 1.2814000001526438e-05
file loading time 1.0052000106952619e-05
Executing for /home/thales/Desktop/ngram/Data/text/news.txt
Compute time 55.97790236799983
Total time to complete 55.977912597999875
```

Figure 1.3 Result with MPI

MPI With 3 Process

```
thales@thales-VirtualBox:~/Desktop/ngram$ ./ngram.sh 3
file loading time 1.1493999863887439e-05
file loading time 1.1008000001311302e-05
file loading time 1.0685000233934261e-05
Executing for /home/thales/Desktop/ngram/Data/text/news.txt
Compute time 49.921755974999996
Total time to complete 49.92176394299986
```

Figure 1.4 Result with MPI

MPI With 4 Process

```
thales@thales-VirtualBox:~/Desktop/ngram$ ./ngram.sh 4
file loading time 0.0001264150000679365
file loading time 1.0937999832094647e-05
file loading time 1.3143999694875674e-05
file loading time 1.0537999969528755e-05
Executing for /home/thales/Desktop/ngram/Data/text/news.txt
Compute time 47.70798519300024
Total time to complete 47.7079894819999
```

Figure 1.5 Result with MPI

CONCLUSION

When we look at the result obtained MPI with 1 process. File loading time is very less because my HDD are fast enough to load in small portion of time and all the data is about 35MB.

When we look at the results obtained MPI with 2 process. File loading time is less than MPI with the 1 process. But most dramatic change in the compute part. Time spent in the compute part is goes to 63 to 56. Increasing number of process to 2 to 3 also decreases the time spent in the compute part 56 to 50. Increasing the number of process to 3 to 4 also decreases the time spent in the compute part 50 to 47

When we look at the all the result of each number process. Nearly all the file reloading time are very fast but when we increase the workers file reloading time is getting faster and faster each time because there are less data to load for each worker. When we increase the workers, we see that compute time and the total time to complete are getting faster. We also see that most of the time is spend in the compute process. This concludes that increasing workers decrease the spent time in compute process dramatically. For example total time spend in with the 4 thread and with the 1 process are 47 seconds to 63 seconds.

p.s I used my Ubuntu operating system via VirtualBox. While running ubuntu, there were other programs and another operating system was running. Because of that you can get different and better results while running the code in the real ubuntu operating system.