**“Introduction to Parallel Programming and MapReduce”**

**(15p) What are the basic steps (show all steps) in building a parallel program? Show at least one example.**

There is 4 steps to do when building the parallel program. These are 1) Decomposition 2) Assignment 3) Orchestration and 4) Mapping. One example that can be used to explain the “building a parallel program” is a “Drug design exemplar code”. In “Drug design exemplar code”, the computer must perform a 3 steps. First, Generate the ligands to try for particular protein, Second Compute the score for each ligand, and third, identify the highest scoring ligands.

Comparing those 3 steps with the 4 steps to build the parallel program, we could identify that these steps are similar. For example, Generate ligands can be interprets as Decomposition, Compute a score for each ligand is same as assignment and Orchestration, last, identifying highest scoring ligands(find the pattern) is same as mapping.

To simplify the statement using the Drug design exemplar example,

Decomposition == Generate ligands to try for particular protein

Assignment and Orchestration == Compute a score for each ligand

Mapping == Identify the highest scoring ligands.

Overall, this is the way to build the parallel program.,

**(5p) What is MapReduce?**

MapReduce is a programming model that is used to perform a parallel programming when generating a big data set to the cluster. When user generate the specific map function, the processor will generate user specific key, and those key will generate the intermediate key pairs. This chain reaction will eventually reduce the function because those specific key will merge all the intermediate key values associated with the same intermediate key.

**(10p) What is map and what is reduce?**

Map and reduce are a constructs that user could apply in your data set, when performing a parallel programming. This idea is typically used a HDFS system, meaning you would use map and reduce when you want to apply the operation to each individual element in data set. Map and reduce typical used in very large data set where you map the similar data into one group, then reduce the number of data so that user’s computer would perform the task faster (due to parallel tasking).

**(5p) Why MapReduce?**

Since the MapReduce can write the scalable data using the parallel programming to process/cluster the large amount of data, we could generate the big data sets in small amount of time. These usage on large data set usually used in the scientific application, thus, this is the main reason why the scientist application used a MapReduce (like DNA example)

**(5p) Show an example for MapReduce**

In example from the PDF, the drug Design and DNA is perfect example. In order to perform this, the computer must generate the ligands to define the particular protein. The number of generating ligands is huge and computing these data would takes forever if we are not using the parallel processing. When we compute the similarities of the DNA, processor will score each individual ligand, thus find the pattern/order of each element. Once processor has scored all of the individual ligands, now the processor would sort the values, then identify the highest scoring ligand. These all will be perform using the MapReduce.

**(10p) Explain in your own words how MapReduce model is executed?**

Every MapReduce starts with splitting the task. The processor will perform this task by calling the MapReduce library. The MapReduce library would split the task equally using the parallel distributing algorithm and assigned to cluster. Once the processor received the task, the master will assign the task to the worker either assign map or assign reduce. Once worker assigned a map, the worker would perform the test using a dataset given by a shred, then retrieved the resulted value. Once all worker finished their task, the worker would now write all the answer to the intermediate files (aka local disk), so that the file is accessible to every worker. Now, the generated resulted output would go to the workers again to perform a reduce. Once the reduce is done, all the necessary result will given to the master, then the master write the output to the designated file.

**(6p) List and describe three examples that are expressed as MapReduce computations.**

According to the towardsdatascience website, the one example I could found is “Term-Vector per Host”.

A “Term-Vector per Host” is a term vector that summarized the most important word that occurs in a certain document by using two parameter: word, frequency. Once the map function recived the input of hostname, term vector, it will identify the most shown word/most significant word on the document. Using a MapReduce is a significant since computer would compute individual words, then define the result.

The second is “Inverted Index”. For each individual word, it will creates the list of the document using two parameter: word, document ID. Once parameter is set, the reduce function accepts all pairs of word, then sort it out into certain pattern. Then, it will emits the word, document ID in to sorted pair. Overall, the set of all output will get inverted, then you finish the task.

Last but not least, Distributed Grep is example of Map function that emits the line if the given pattern is matched with a result. The reduce is used when identifying the function that are copied and supplied from intermediate data point.

**(6p) When do we use OpenMP, MPI and, MapReduce (Hadoop), and why?**

The OpenMP is widely spread that is almost set to be a convention when programmer needs a efficient directive based library. The OpenMP is free to use which gave a huge advantage.

The MPI is used to develop the parallel scientific applications. It usually tightly synchronous code and well loaded balance. MPI pass the message and it distribute the memory when performing the parallel model implementation

Last but not Least, The Hadoop MapReduce, like other regular MapReduce, can be used in to identify the pattern/result from the large amount of data (large dataset). This used a HDF system where it distribute the file efficiently, which gave a huge advantage when doing a parallel program

**(14p) In your own words, explain what a Drug Design and DNA problem is in no more than 150 words.**

Our DNA is structed with consist of two RNA, which have a important data to our body system. DNA can be considered as the book of recipes. Here, we have to understand the hundred and millions of data to store in a DNA and form the protein’s shape. This requires a lot of data; which MapReduce is a perfect fits for performing a testing. There is a 3 steps to build the drug design software. First computer must generate the ligands to try for the particular protein. Each ligands will be randomly generated with random character, and length. Some Ligands will carry a bunch of data and some are not, thus only some will fits. Now we must compute the score for each ligand. Computer will compare the maximum match with each other, then scoring solidly base on insertion and deletion. The highest score would represent how the DNA fits in a assigned protein shape. Once the scoring the ligands is done, now the computer would identify the highest scored ligands, and wrote it on the designated file

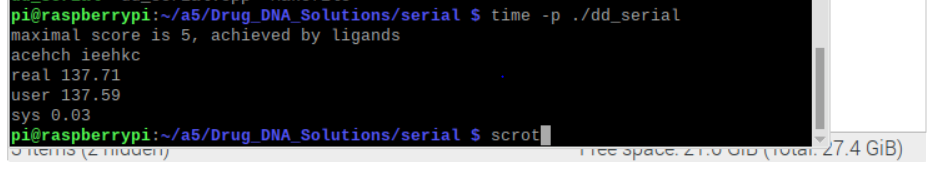
Part B

Drug Design and DNA in Parallel

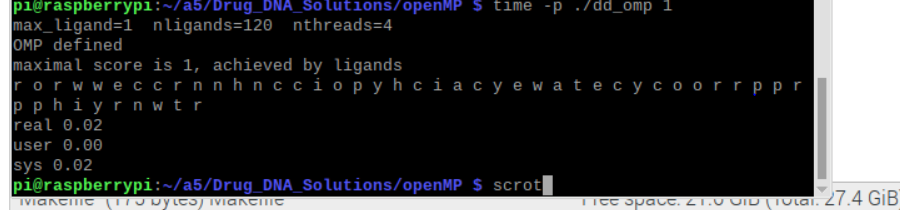
Measure Run time

|  |  |
| --- | --- |
| Implementation | time |
| dd\_serial | 137.71s |
| Dd\_omp | 0.02s |
| Dd\_threads | 0.02s |

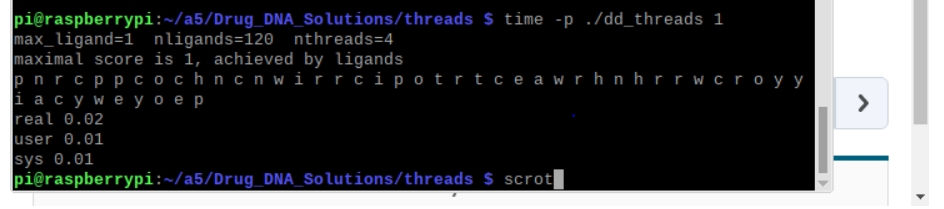
dd\_serial time :



Dd\_omp time:

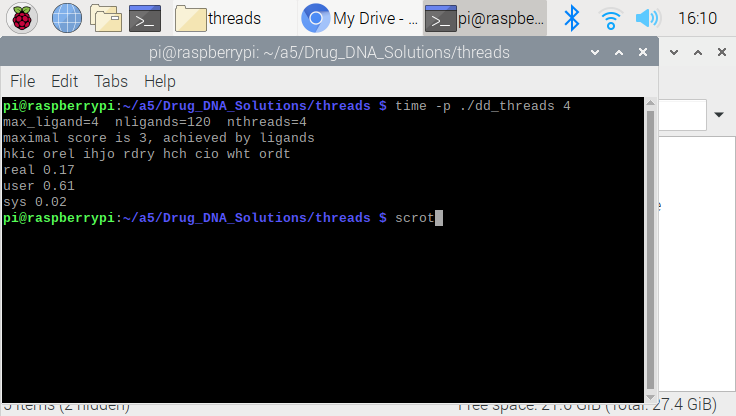


Dd\_threads time:

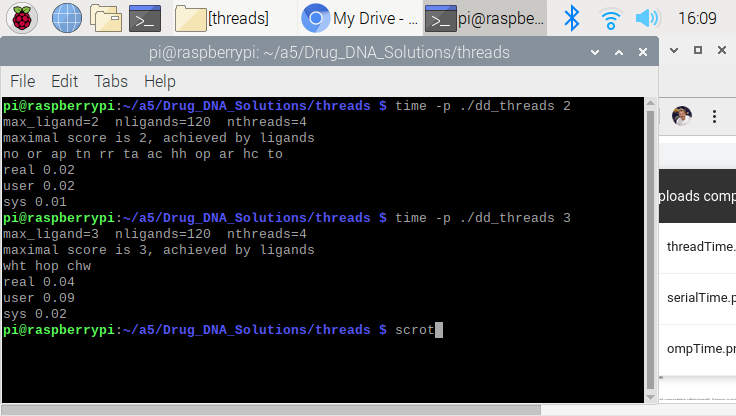


|  |  |  |  |
| --- | --- | --- | --- |
| Implementation | Time(s) 2 Threads | Time(s) 3 Threads | Time(s) 4 Threads |
| Dd\_omp | 0.02s | 0.04s | 0.21s |
| Dd\_threads | 0.02s | 0.04s | 0.17s |

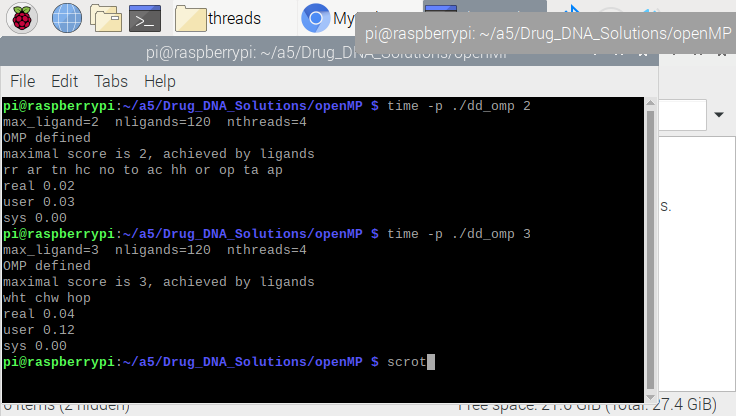
Dd\_thread with 4 threads time:

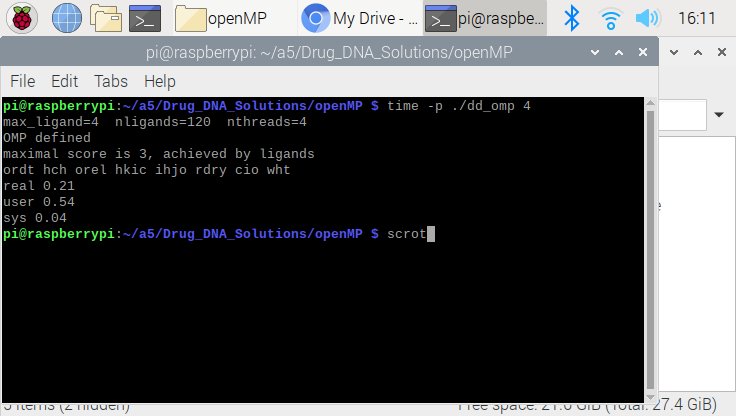


Dd\_thread with thread 2 and 3 time:



Dd\_omp with 2 and 3 threads time:





Discussion Questions

1. Which approach is the fastest?

With only 1 threads, the dd\_omp and dd\_threads are tied each other. However, when it goes up to the 4 thread, dd\_threads method is clearly fastest.

Regardless of number of threads, the serial method is definitely slowest method.

1. Determine the number of lines in each file (use wc-l) How does the C++ 11 implementation compare to the OpenMP implementation?

Dd\_omp’s OpenMP has 152 words Dd\_omp’s C++ has 194 words

Dd\_serial’s OpenMP has 63 words Dd\_serial’s C++ has 170 words

Dd\_threads’s OpenMP has 137 words Dd\_threads’s C++ has 207 words

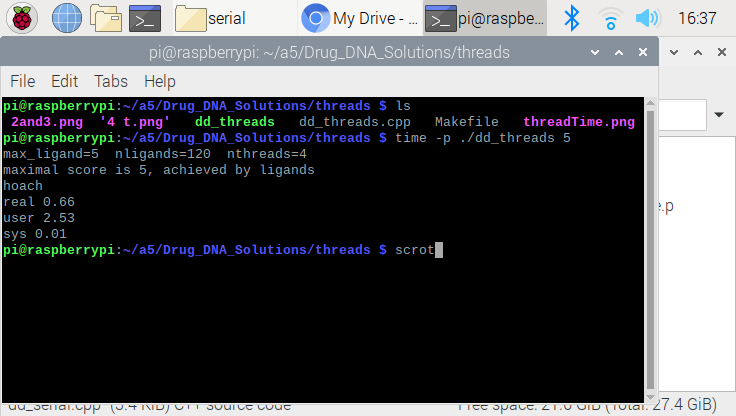
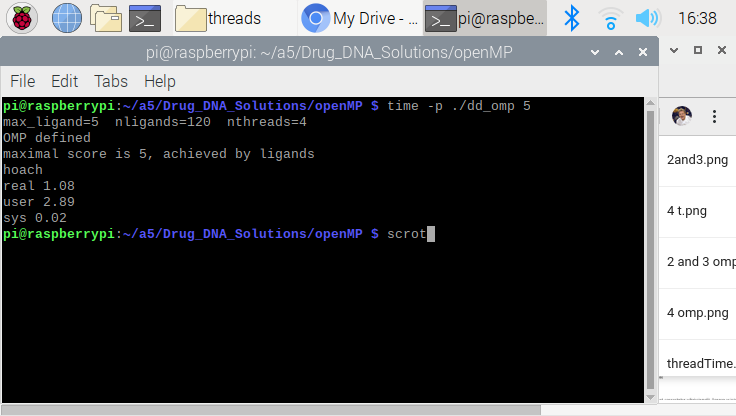
Looking these trends, the C++ requires the more lines of code compare to the OpenMP. This is because the C++ library is general purpose library, whereas the OpenMP is designed for the parallel program. Therefore, in parallel programming, the OpenMP is more efficient.

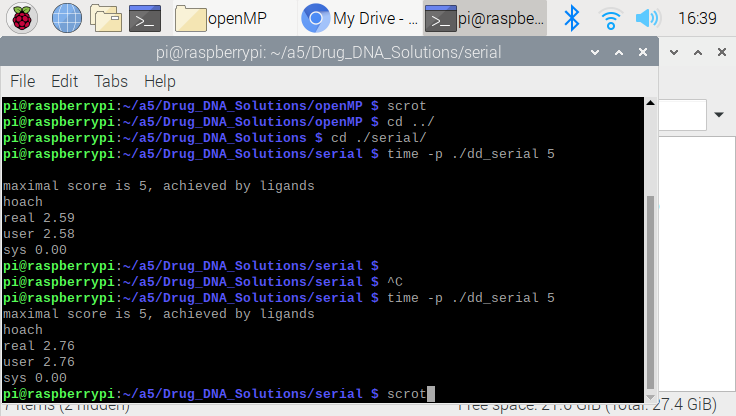
1. Increase the number of threads to 5 threads. What is the run time for each?

Dd\_threads runtime with 5: 0.66s

Dd\_omp runtime with 5: 1.08s

Dd\_serial runtime with 5: 2.76s



1. Increase the maximum ligand length to 7, and rerun each program. What is the run time for each?

The code is already set to be a maximum length of 7. Thus, the answer for the each method is given above (question 1 and 2)