Bryanna Hardy

April 15, 2020

Task 3: Parallel Programming Skills

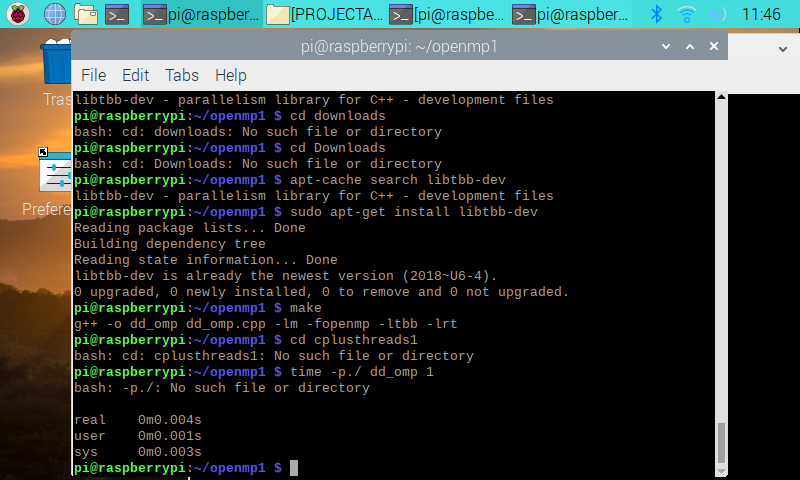
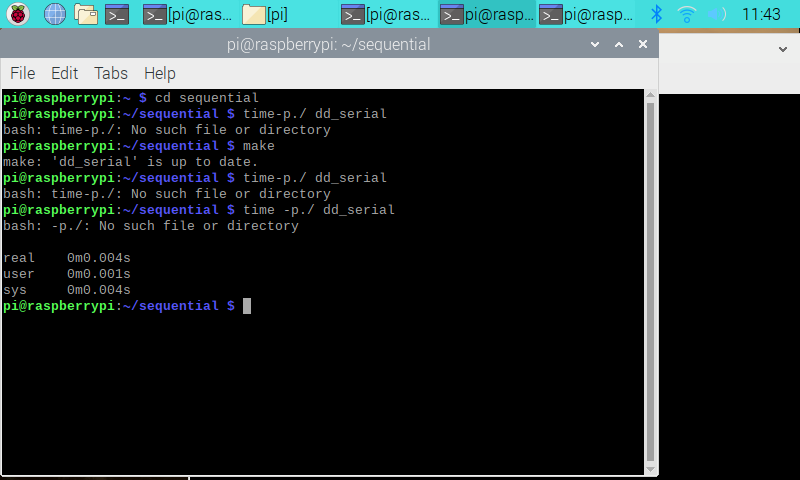
**Questions:**

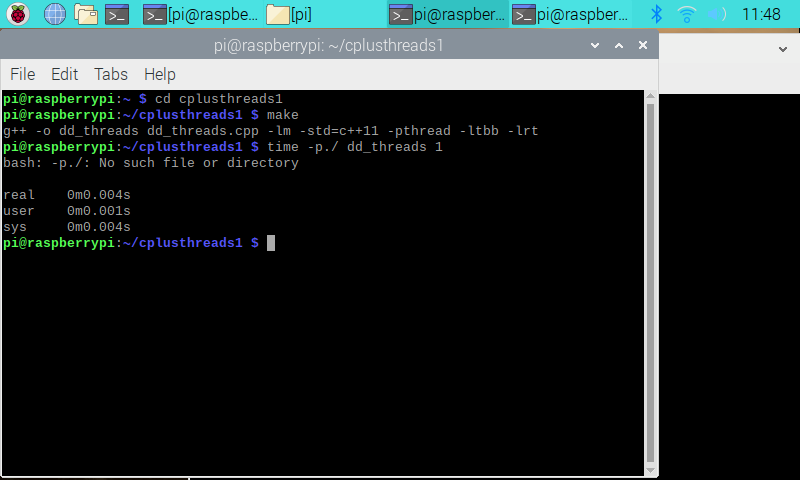
1. What are the basic steps (show all steps) in building a parallel program? Show at least one example.
   1. The basic steps in building a parallel program are
2. What is MapReduce?
   1. A MapReduce is a programming model and an associated implementation for processing and generating big data sets with a parallel, distributed algorithm on a cluster.
3. What is a map and what is reduce?
   1. A **map** processes a key/value pair to generate a set of intermediate key/value pairs, and a **reduce** merges all intermediate values associated with the same intermediate key.
4. Why MapReduce?
   1. Because it can be used to write tightly coupled scientific applications. Also, it processes and generates big data sets.
5. Show an example for MapReduce.
6. Explain in your own words how MapReduce model is executed?
   1. The MapReduce library first shards the input files into M pieces. Then starts up many copies of the program on a cluster of machines. One of the copies of the program is special which is the master. The rest are workers that are assigned work by the master. The master picks workers and assigns each one with a task or a reduce task. The worker who is assigned to a map task reads the contents of the corresponding input shard. It pairs values out of the input data and passes each pair to the user-defined Map function. The buffered pairs are written to the local disk. The locations of these buffered pairs on the local disk are passed back to the master, whose task is to forward these locations to the reduced workers. When a reduce worker is notified by the master, it uses remote procedure calls to read the buffered data from the local disks of the map workers. When it has read all intermediate data, it sorts it by the intermediate keys so all the same key is grouped together. If the data is too large to fit in memory, an external sort is used. The reduce worker iterates over the sorted intermediate data for each unique intermediate key encountered, it passes the key and the corresponding set of intermediate values to the user’s Reduce function. When the map task and reduce tasks have been completed, the master wakes up the user programs. The MapReduce call in the user program returns back to the user code.
7. List and describe three examples that are expressed as MapReduce computations.
   1. Distributed Grep: the map function emits a line if it matches a given pattern. The reduce function is an identity function that copies the supplied intermediate data to the output.
   2. Reverse Web-Link Graph: output <target, source> pairs for each link to a target URL found in a page named “source”. The reduce function concatenates the list of all source URLs associated with a given target URL and emits the pair: <target, list(source)>.
   3. Count of URL Access Frequency: map function processes logs of web page requests and outputs <URL, 1>. The reduce function adds together all values for the same URL and emits a <URL, total count> pair.
8. When do we use OpenMP, MPI, and MapReduce (Hadoop), and why?
   1. OpenMP: It is an efficient directive-based library that you could use.
   2. Hadoop MapReduce: Can use when applying over large data; it distributes. It is a distributed file system designed to run on commodity hardware.
   3. MPI: Message Passing Interface is a distributed memory parallel model implementation, mostly used to develop parallel scientific applications.
9. In your own words, explain what a Drug Design and DNA problem is in no more than 150 words.
   1. DNA is illustrated like a book of recipes. DNA contains the instructions for making proteins in our bodies. The shape of a protein determines the function it performs in an individual's body. For you to design a drug, you need to find ligands, new pieces, to change a protein’s shape. A way to design a drug design software, you first need to generate ligands to try for a particular protein. There are some ligands that will fit and some will not. The second thing is that you need to compute a score for each ligand that simulates how well it will fit in the protein and produces the desired shape change. Thirdly, identify the highest scoring ligands for actual synthesis (production) and testing. In the drug design exemplar code, the command-line arguments: “./drugdesign-static ***threads maxlen count.*** The threads are the number of simultaneous threads, the maxlen is the maximum length of a ligand. Each ligand has a random length up to this max. Then, count the number of ligands to score.

**PART B:**

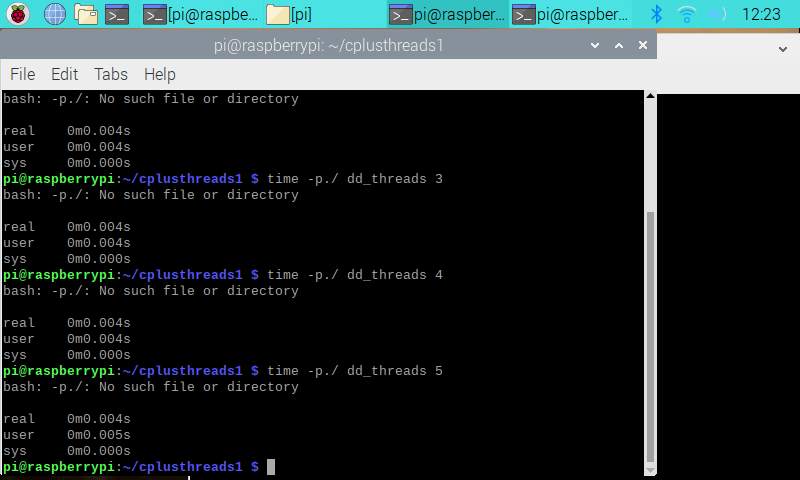
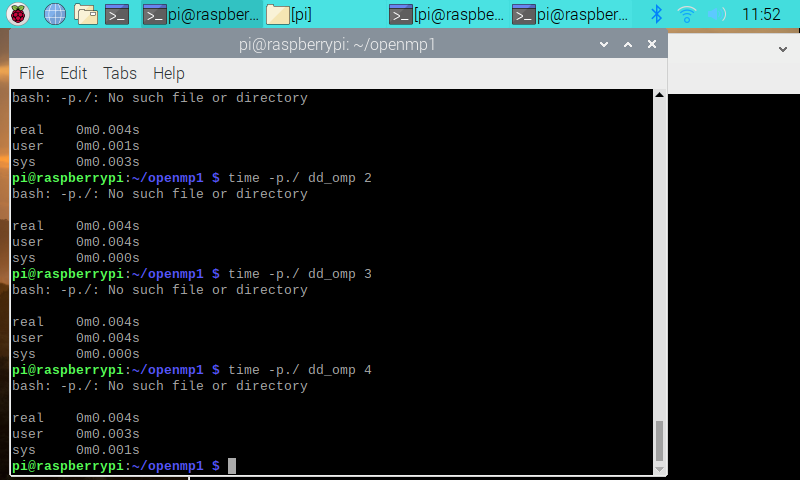
***MEASURE RUN-TIME:***

|  |  |
| --- | --- |
| **IMPLEMENTATION** | **TIME(S)** |
| **dd\_serial** | **0m0.004s** |
| **dd\_omp** | **0m0.004s** |
| **dd\_threads** | **0m0.004s** |

** **

****

|  |  |  |  |
| --- | --- | --- | --- |
| **IMPLEMENTATION** | **Time(s) 2 Threads** | **Time(s) 3 Threads** | **Time(s) 4 Threads** |
| **dd\_omp** | **0m0.004s** | **0m0.004s** | **0m0.004s** |
| **dd\_threads** | **0m0.004s** | **0m0.004s** | **0m0.004s** |

** **

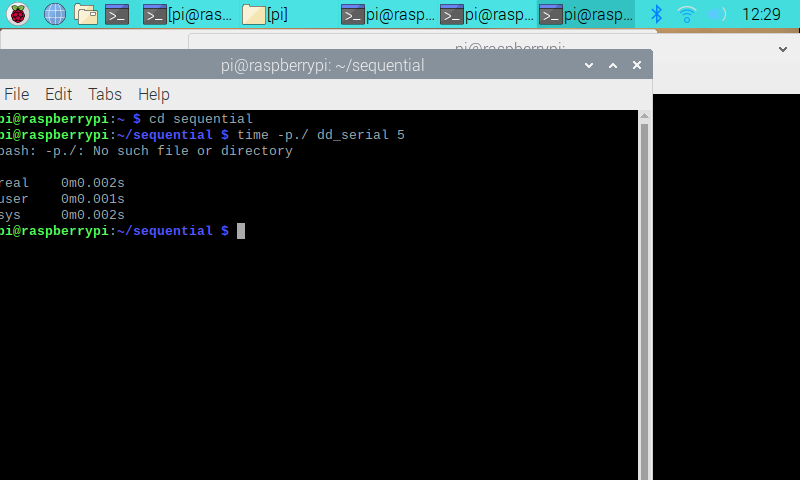
Which approach is the fastest?

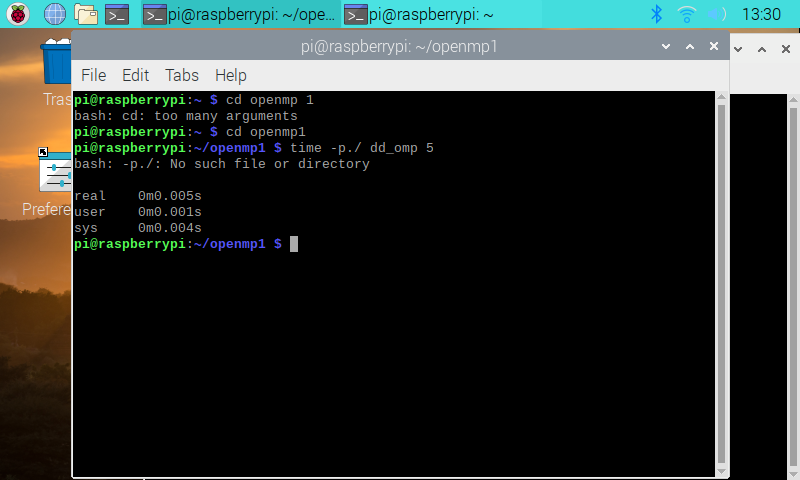
I really can not tell which approach is faster, because I received the same results for the real time. On the other hand, if you look at the other times, such as user and sys, they all have different timings. I would assume that openMp approach is the fastest.

Determine the number of lines in each file. How does the C++11 implementation compare to OpenMp implementations?

Serial had 171 lines in its file, 194 lines in omp file, and 208 lines in threads file. C++ multithreading libraries are general purpose and gives less control over. OpenMp is much efficient than C++ threads. OpenMp is designed to not impact the code and designed for multiple set of problems.

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



As you can see, increasing the number of the threads, you can see the difference in the real time when you compare it with the other times in the previous pictures.

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

When I increased the maximum length to 7, I received the same run time for each, which was 0m0.004s.