CENG316 Parallel Computing Project-2

Assigned: May 11 2020

Due: May 22, 2020, Wednesday 23:55

Penalty for late submission is 10% per day.

You can ask questions through Aybuzem forum system.

This is a teamwork assignment; you are going to work in the same groups. Due to the coronavirus precautions, team members should collaborate and work remotely at their home. You can use any instant messaging program like Whatsapp or Skype, to arrange video calls for collaboration in the project, i.e., work sharing, figuring out the project requirements etc.

In this project, you are going to implement parallel calculation of gravitational force interaction of stars in a galaxy. You are going to utilize multi-process parallelism via MPI library. You may assume that there are some number of stars in the galaxy occupying a volume of 1000x1000x1000 space. In addition, each star has a mass which is an integer value between 1-20. Your task is to calculate the gravitational pull on each star due to every other star.

Newton's law of universal gravitation states that every body of nonzero mass attracts every other object in the universe. This attractive force is called gravity. It exists between all objects, even though it may seem ridiculous. For example, while you read these words, a tiny force arises between you and the computer screen. This force is too small to cause any visible effect, but if you apply the principle of gravitational force to planets or stars, its effects will begin to show.

Use the following formula to calculate the gravitational force between any two objects:

$$F = (G * M * m) / (R^2)$$

- F stands for gravitational force. It is measured in newtons and is always positive. It means that two objects of a certain mass always attract each other;
- M and m are the masses of two objects;
- R is the distance between these two objects, and can be calculate by the formula:

$$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

  you should take **n** as 3 for 3D space.
- G is the gravitational constant. You can take it as 0.0006674 N*m^2/kg^2.

Implementation Details

1. You will use MPI library in C or C++ programming language.

2. You will read details of stars from a file. The filename is acquired from user by a command line parameter (e.g. using argv[]). Each line with 4 integer represent a star in the file. For example, the following file contains three stars, one is located in (50,11,90) with a mass of 3 kg, the second located in (21,18,71) with a mass of 10 kg and so on.

```
50 11 90 3
21 18 71 10
99 60 50 7
```

The Gravitational force on the first star is the summation of all forces due to every other star. That is:

$F = ((G * M1 * m2) / (R\_12^2)) + ((G * M1 * m3) / R\_13^2)$

$F = (0.0006674 * 3 * 10) / (R\_12^2)) + ((0.0006674 * 3 * 7) / (R\_13^2))$

Where $R\_12 = \sqrt{(50-21)^2 + (11-18)^2 + (90-71)^2}$

Where R_13 = $\sqrt{(50-99)^2 + (11-60)^2 + (90-50)^2}$

3. The number of stars should be taken from user as a command line argument (e.g. using argv[])
4. Therefore, your algorithm with 4 processes should be called like this:
   mpirun –n 4 ./gravity stars.txt 100
   it will read stars.txt which contains details of at least 100 stars.
5. The sum of each Gravitational force on each star should be written out a file named forces.txt. So, for the previous execution, forces.txt should include 100 lines (since we have 100 stars) where each line represents gravitational force on each star.
6. Calculate 2-norm (Euclidean norm) of all forces applied to stars in parallel. Then only master processwill print out this value. We will use this value for checking the correctness of your algorithm. To see more detail use "%.10E" for printf, instead of just "%lf" .
7. Finally, the sum of each Gravitational force on each star should be written out a file named **forces.txt** by the master process. So, for the previous sample execution, forces.txt should include 100 lines (since we have 100 stars) where each line represents gravitational force on each star.

Hybrid Parallelism

Now add one more level parallelism into your code via OpenMP. For each MPI process, you should share the work with 2 threads.
Save your source codes in a file named hybrid_gravity.c.
Measure the executions times with 50000 stars and make a table. It this test, you will use 1,2, and 4 MPI Processes and for all tests use 2 threads.

What is required?

1. Report
   - Small report which includes your name and surname and appropriate title and small description.
   - Very short pseudocode of your **sequential** and **parallel** algorithms with at most **15 lines**.
   - Brief explanation: Steps of your parallel algorithm in terms of 4 steps of Foster's methodology.
     - Furthermore, you should discuss which parallelism you adopted, task or data parallelism, with the reasoning behind it.
   - Weak and strong scalability results of your algorithm by using 1,2, and 4 MPI Processes. For weak scalability test use 10000 stars for 1 core, 20000 stars for 2 cores and 40000 stars for 4 cores. For strong scalability tests use 50000 stars. For accurate timing please take average of at least 3 tests.
   - Results of hybrid parallel code.
2. Code
   Source codes named gravity.c , hybrid_gravity.c written in C programming language.

Notes:

- You can work with your group members, of course, but working together with other groups is prohibited.
- Submit your parallel code together with your report (**PDF format**) in a zip file named exam1-GroupX.zip into uzem. (Replace X with your group number.)
- **One member in each group should upload the exam in behalf of their group. That is, there should be only one submission for each group in the uzem.**