COMPSCI / SFWRENG 4F03
- Project -

Due April 9 @ 11:59pm

Thomas Gwosdz

February 7, 2018

1 Part A

The project may be done in groups of 3 or 4. Use MPI (potentially in conjunction with OpenMP) to accelerate the computation for the N-Body simulation. Use the provided project.zip boilerplate available on avenue as a start. It contains methods for saving images of the simulation steps. You may then use any program capable of combining images into a video file (such as ffmpeg).

For the math behind N-Body simulation, please refer to the excellent slides by Dr. Nedialkov (also available on avenue).

2 Part B [25]

Program [20]

When the program is called, it should randomly generate light, medium, and heavy particles. Each particle will have a random velocity. Then simulate how these particles will interact with each other. It will report step timings as well as produce image snapshot of the simulation at the time.

Tuning [5]

We will select a set of parameters, the timings will be used to assign 5 marks for fastest, and 1 for slowest group. **You are competing with your peers**

Specifications

- The program is to be called project.x
- The program accepts the following parameters: project.x numParticlesLight numParticleMedium numParticleHeavy numSteps subSteps timeSubStep imageWidth imageHeight imageFilenamePrefix

  numParticlesLight The number of light particles

  numParticleMedium The number of medium particles

  numParticleHeavy The number of heavy particles

  numSteps Number of major steps. These will be the frames in the video and be written to images

  subSteps The number of simulation steps between major steps. These provide the timing data

  timeSubStep The time in seconds between each minor timestep

  imageWidth The width of our image

  imageHeight The height of our image

  imageFilenamePrefix The prefix to use for our images (i.e., image will produce image_00000.bmb)
- Use functions in savebmp.c to generate the images
- Use double precision for decimals
- Use drand48 as the random number generator
- It should be a C/C++ program, and make use of MPI
- Use MPI_Wtime() for timing, ensure that only the simulation substep is timed, not I/O
- The program should output the following to stdout: <min time of substeps> <max time of substeps> <average time of substeps>
- The program should also save an image representation of each Step (not sub-step)
- The program will use mpirun to specify the number of cores to use
- Use "properties.h" to define min/max velocities and mass of each particle group.

Example

To simulate 10,000 particles (1,000 heavy particles, 3,000 medium particles, and 6,000 light particles) with 8 nodes we can call:
> mpirun -np 8 ./project.x 6000 3000 1000 900 50 0.5 1920 1080 simulation
> 12.433 22.2132 16.34

The above program would produce 900 "frames" for our video with resolution of 1920x1080. Between each frame we will have 50 simulation sub-steps with 0.5 seconds between substeps.

Although we only save 900 frames (simulation_00000.bmp - simulation_00900.bmp), we will have calculated 900 x 50 = 45,000 simulation steps.

3 Part C [24]

Report

Part of the report will be tuning and squeezing as much performance out of the program as possible.

- [6] Provide timing, speedup, and efficiency plots. For total particle numbers of 2,000 4,000, 8,000, 16,000, 32,000 particles using 1,2,4,8,16,32 nodes.
- [4] Discuss weather or not the algorithm is strongly scalable and/or weakly scalable. Why/why not?
- [4] Explain any anomalies in the plots, what could have caused them?
- [5]Explain your modifications/findings when tuning your program for best performance (provide plots for comparision i.e., MPI vs. MPI with OpenMP)

Video

Part of the report will be tuning and squeezing as much performance out of the program as possible.

- [5] Generate and submit a video of 2 minute simulation using 16,000 particles (play with the breakdown for interesting results), your video should have 2550x1440 resolution.

4 SVN Submission

Submit your source code, makefile, readme, report, and any other resources to Project folder of your SVN trunk. All source code and makefile should be located in Project and not a subfolder. When 'make' is called, it should produce programs named: "project.x"

5 Note

- Make sure your programs can be fully compiled and executed on the departmental machines (i.e. Mills, mpihost)
- Familiarize yourself with the department's policy on plagiarism and the university regulations on plagiarism and academic misconduct. Plagiarism will not be tolerated, and will be dealt with harshly.