High-Performance Scientific Computing (MATH-GA 2011/ CSCI-GA 2945)

Homework Set 5

Out: October 10, 2012 · Due: October 17, 2012

Problem 1: Pitch a project

Please form an initial idea of what final project you would like to do. The scope should at least amount to about four homework sets in terms of overall work, and you'll need to be able to present your progress in a 15-minute talk towards the end of the class. The project has to involve some non-trivial form of parallel programming. Note that you don't need to have the idea fully developed right now. This pitch is about me helping you to figure out whether your proposed project is suitable. At this stage, "no" is an acceptable answer to that question, and if that happens, we'll figure out a plan together.

If you'd like to work in a team, please also find your teammates now, and pitch a project together.

a) Please make an appointment with me for the week of October 15–October 19. During this appointment (which will last for about 10 minutes), I'll ask you to present, within three to five minutes, what your project is about, and what makes it interesting as a computational problem. You should also be prepared to answer the following questions:

   - What type of parallelization do you intend to apply? Why do you think that type would be a good fit?
   - What software exists for this problem? What existing components do you intend to use?

   I'll provide feedback, and we'll be on our way to narrowing down what your project will be.

Problem 2: Simulate fish in a long tank

In this problem, you'll be using MPI to model a large school of fish that are swimming around a two-dimensional tank. If N is the number of ranks in your simulation, then the dimension of the tank is $50N \times 50$. Each rank simulates fish in a portion of the tank. Rank i manages the portion $[50i, 50(i+1)) \times [0, 50)$, i.e. from x-coordinate $50i$ up to (but not including) x-coordinate $50(i+1)$.

a) Create a structure representing a fish. A fish has

   - A position (x, y), represented as two double-precision numbers.
   - A velocity (v_x, v_y), represented as two double-precision numbers.

   The simulation proceeds in time steps. A fish moves from (x, y) to $(x + v_x, y + v_y)$ every time step.

   The upper and lower walls of the tank are solid. If a fish would hit or penetrate a wall in a time step, it stays where it is and changes the sign of its vertical direction instead.

   Examine the first command line argument (call it m) and have each rank place m fish randomly and uniformly in its part of the tank. Pick each of the velocities uniformly in the interval $[-2, 2]$.

b) Write a time stepping procedure. Note that the tank is periodic in the horizontal direction. I.e. if a fish swims out of the left boundary of the tank area managed by rank 0, it reappears on the right-hand side of the tank of rank $N - 1$.

Use MPI to realize the inter-rank communication necessary to ensure that fish keep swimming as if nothing had happened if they encounter the boundary between areas of the tank that are managed by different ranks. The presence of the computational boundaries should not affect the simulation at all.

You may want to use non-blocking communication to avoid deadlocks and communication ordering issues. Your code has to work for any number of ranks. Also make sure to send all fish that go from one rank to another in *one* message to avoid incurring message send cost repeatedly.

You'll also have to deal with the fact that a variable number of fish may wander from one subtank to the next in a time step. You may use fixed-size buffers for this. If you do, your code *must* notice if it's overrunning a buffer, drop the fish that don't fit into the send buffer from the simulation, and print a comment to the console about this fact.

If you like, you may investigate the use one of two possible solutions to this issue:

- `MPI_Probe` can get the size of a message without actually receiving it.
- You can send a message containing the number of fish to be sent *before* actually sending the fish data.

To send the fish, you may either send a 'bag of bytes' or use MPI's facilities for structured data types.

c) Add code so that each rank outputs the current state of its fish to a 50x50 pixel PPM image file with a name as output by

```
sprintf(ppm_file_name, "fish-step%04d-rank-%04d.ppm", step, rank);
```

Fill the image with a black background and draw a white pixel for every fish, where the extent of your rank's part of the tank is mapped to the 50x50 pixel image.

You may then use the following command to glue these images together horizontally:

```
for step in $( seq -f "%04g" 0 100) ; do
  convert fish-step$step-rank*.ppm +append fish-step$step.png
done
```

Note that this is *Shell* code, i.e. it goes into the window where you compile and run. You may have to change the number 100 to the actual number of time steps.

Then delete all the PPMs and view the `.png` files using the `ristretto` image viewer on your VM. This allows you to step forward through the images in the style of a flip book.

Use the second command line argument to determine when to output these image files. If that argument (call it n) is zero, do not output anything. If it is nonzero, output the images every n steps.

d) Use the third command line argument to determine how many time steps to simulate. Insert barriers around your code, and have rank 0 output a simulation timing at the end, in units of 'fish updates per second', where this counts the total number of fish.

e) Time on multiple ranks

Turn in your code as `fish.c` along with a `Makefile`.