MCS 572 Project One : a predator-prey simulation

# Message Passing to distribute time stepping jobs

The goal of the project is to use MPI to distribute the computation of the trajectories of a predator chasing a prey at constant speed. The prey moves on a horizontal line, starting at the origin, traveling with constant speed one. The predator starts chasing the prey at location (0, 10). If the speed of the predator is larger than one, then the predator will eventually catch the prey, see Figure 1.

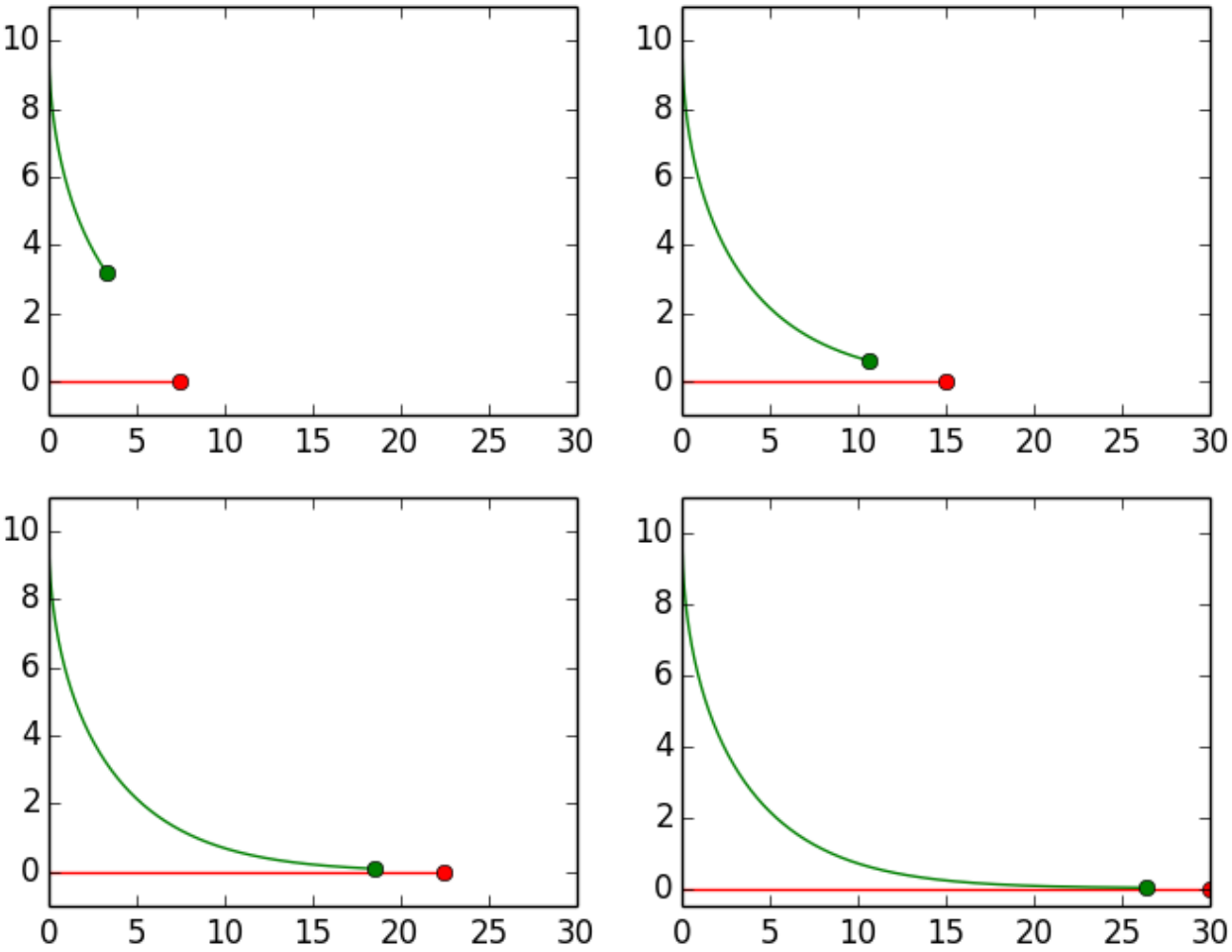



Figure 1: A predator starts at (0, 10), in pursuit of a prey which moves on a straight line. Although the predator approaches the prey at greater speed, at the bottom right, the predator has not caught the prey.

The time interval, which is [0, 30] in Figure 1, is divided in 1,000 time steps. At each time step, the prey advances on its straight line path. The predator advances in the direction of the prey. If the distance between the predator and prey becomes less than 0.1, the predator has caught the prey and the simulation stops. The values 30 and 1,000 are numerical parameters in the simulation.

To help you start the project, a basic Python script `predprey.py` is available for download via the course web site. The script runs in Python 2.6 and 2.7. It uses `matplotlib` for the rendering of the plot. The trajectory of the predator is computed with a time stepping method. Corresponding to `predprey.py`, there is a C program `predprey.c`.

Figure 1 shows just one trajectory. In this simulation we compute many trajectories for a suitable range of values for the speed of the predator. The second parameter is the length of the time interval. With the simulation we would like to answer the following question: given the end of the time interval, what is a lower bound on the speed of the predator so it will catch the prey before the end of time?

For this project you may choose between `mpi4py` or `mpicc`. Of course, FORTRAN is also acceptable.

## Assignment One: define the message passing

Write a parallel program to define the message passing for a program using MPI in a static work load assignment. The root node is responsible for assigning the trajectories to the worker nodes and for writing the final results of the simulation.

## Assignment Two: examine the scalability

Investigate scalability when more worker nodes are available. Take the size of the problem into consideration. How large should the problem size be with 1,024 nodes? Determine the granularity of the parallel code.

Determine experimentally the relation between the communication and the computation cost to determine the scalability of the program developed in the first assignment. In your cost considerations, take the memory usage into account because memory could become a bottleneck for a serial program.

In your study of the communication cost, consider two scenarios in your scalability report:

1. To set the speed of the predator so it catches the prey before the end of the time interval, we need to know only the time of capture (and eventually the location of prey and predator at that time), at the end of the simulation. In this case, the communicated data has a constant size.

2. To verify the results of the simulation, we may need to plot specific trajectories. In this case, the workers need to transfer a variable size of data. The size of the data varies with the discretization of the time steps and the granularity of the visualization.

## Assignment Three: discuss load balancing

The time stepping happens with a maximum number of steps but terminates sooner when the prey is caught. Thus some trajectories may require much less work than others. Describe a situation where static work load assignment does not perform very well.

Develop an algorithm for dynamic load balancing to apply for any number of processors. Decide on the granularity of the size of each job and motivate your choice. Indicate the application of the dynamic load balancing code we have discussed in class to this problem.

## The deadline is Wednesday 28 September 2016 at 10AM

On the deadline, bring to class the answers to the assignments on paper, describing your design decisions made while developing the code and the analysis of communication and computation costs. In addition, also send me by email the programs you wrote.

You may work individually or in pairs on this project. For each pair, please submit only one solution.

If you have questions or difficulties with the assignments, feel free to come to my office for help.