Overview: In this exercise, we want to introduce the idea of computational modeling — the modeling of real-world processes or phenomena in software.

Computational modeling is useful in situations where experimenting with a real-world physical process is impractical, costly, or dangerous. A computer model of the physical phenomenon lets us experiment with the phenomenon, and perhaps come to understand how the process works, without the limitations, cost, or danger found in the real world.

The physical process we'll be modeling is a forest fire. The model we'll use is simplified so that we can understand it easily, which reduces its representational accuracy. This simplicity-vs-accuracy tradeoff is common in modeling: models are (by definition) imperfect representations of the phenomena they represent, and the more simplified the model, the less accurately it represents the complexities of the real world.

Getting started: Begin by experimenting with this fire applet from the Shodor Foundation's Interactivate Project. This model lets you start a forest fire by clicking on a tree, which lights it on fire. The fire then spreads probabilistically to neighboring trees, based upon the burn-probability, which models the likelihood of a fire spreading from a tree to its neighbors (e.g., perhaps reflecting local moisture levels).

When the fire burns out, the model displays the percentage of the forest that burned, and the time (number of iterations) the fire burned. Once your forest fire has burned itself out, you can "regrow" the forest by clicking the Regrow Forest button. Take a few minutes to experiment with the model, to get a sense of how it works.

Exercise: We want to use this model to explore these questions:

- What is the relationship between the burn probability and the percentage of the forest that burns?
- What is the relationship between the burn probability and how long the fire burns?

Click the Highlight Center Tree button to make it clear which tree is the center tree. Then for each of the burn probabilities 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, and 0.9,

1. Start a forest fire at the center tree, noting the percentage of the forest that burns, and the time (iterations) the forest burns in a spreadsheet.
2. Regrow the forest.

When you have finished, use the spreadsheet to create a chart in which you plot the percentage-burned and time (iterations) for each burn-probability. Compare your chart to the charts of your classmates. How do they compare?

This model is an example of a class of computations called Monte Carlo simulations, which use statistical probabilities and pseudo-random numbers to represent physical phenomena. As a result, the model is non-deterministic: if you rerun the model several times with the exact same inputs (burn probability, center tree), you are likely to get different outputs (percentage-burned, time). For this reason, it is usually necessary to do many trials or repetitions of a Monte Carlo simulation, and then average the results, to get a sense of what is most likely to happen (on average).

Start a forest fire two more times for each burn probability, and add the percent-burned and time results to your spreadsheet so that you have six measurements (three percent-burned and three time) for each burn-probability. Then calculate the average of the three percent-burned and time...
measurements for each burn-probability, and create a new chart plotting these averages against
the burn-probabilities. How does your new chart compare to your first one? How does it compare
to your classmates' charts? Are they more or less similar than before?

Homework: As you can see, Monte Carlo models need to compute average results over many
repetitions or trials to produce results that are "accurate." However these trials can turn a fast
program into a time-consuming one, especially as the size of the problem (e.g., the forest size)
increases.

Your homework assignment this week is to take a sequential version of this model (written in C),
and "parallelize" it using MPI. Download the zip file of the sequential model, extract its contents
(gunzip firestarter.tgz; tar xvf firestarter.tar), compile it using the included
Makefile, and then run the program. This version (by default) uses 101 probabilities (0, 1, 2, ..., 100) and 5000 trials to burn a forest of size 20x20. When it is done, it displays the burn-
probabilities and their corresponding percentage-burned values, plus a graph plotting these
values.

You have been given an account on the IBM Power 5 shared-memory machine. To access it,
ssh to p5login.hpc.ncsu.edu, and log in with your Unity ID and password. There is some
basic Power 5 usage information on the HPC web site at
http://www.ncsu.edu/itd/hpc/Documents/SharedMemory/GettingStartedp5.php

1. Modify the program so that in addition to the percentage-burned values, it also calculates
and displays the times (iterations) for each burn.

2. Even with a "small" (20×20) forest, this program takes "a long time" to compute its
results. Rewrite this program using MPI commands to spread the computation of these
burn-percentage and time values across N processes. Try to develop an "embarrassingly
parallel" solution to the problem. (Hint: Use the example from class for assistance.)

3. Add MPI timing commands to benchmark how fast your parallel program runs in "wall
clock" time.

4. Increase the default size of the forest to a "larger" size (i.e., 50x50, 100x100, etc.) (which
still is not all that big for a forest). On the system, plot your program's run-times for 1, 2,
4, 8, 12, and 16 CPUs.

Hand in:
• Your spreadsheet data and chart from the Exercise above;
• Your source code for your "parallelized" program;
• A script file showing your program's textual output;
• A screenshot of your program's graphical output.
• Your spreadsheet data and chart of your parallel program's run times on the system.
• A 1-page analysis of what this model tells us, including:
  o The relationship between the burn-probability and the percentage of the forest
    that burns (on average).
  o The relationship between the burn-probability and the duration of the forest fire
    (according to this model).
  o The performance of your program on the system, as the number of CPUs
    increases.