Jason Lin and Aaron Li
Period 10
fireFighters
Group 74

Systems Final Project: Forest Fire

Project Description

We plan on making a simulator for forest fires that will utilize systems networking to distribute the workload of doing such a task. The user will be able to set parameters such as the percentage of trees, map size, and initial fire location to obtain data on which percentages are optimal for the longest burn time.

Minimum Viable Product:

- The minimum product should have an algorithm for simulating the forest fire burns using the different parameters mentioned above. It should be able to distribute the workload between processes in the same computer.

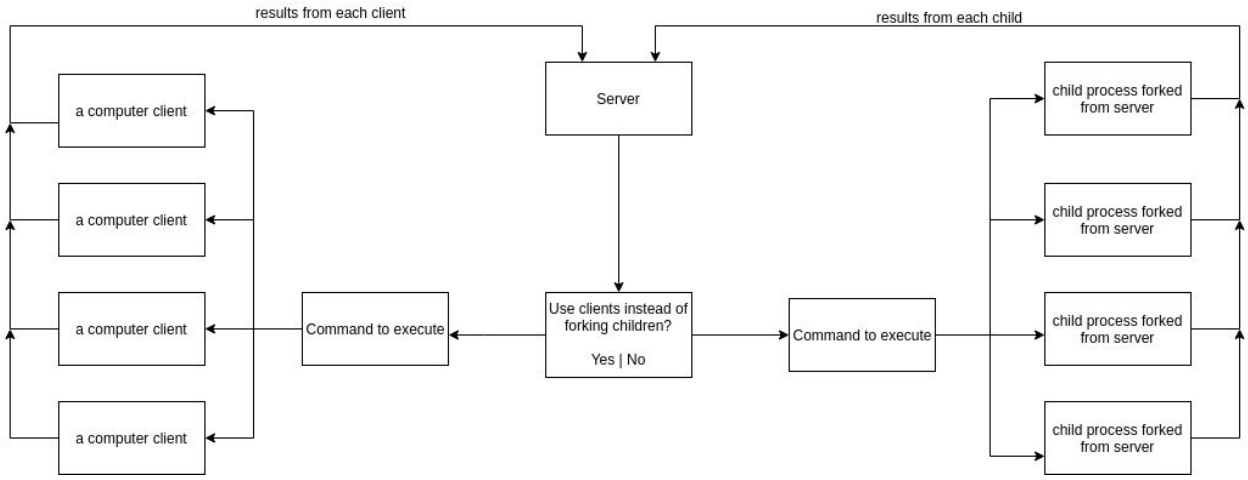
Expected Product:

- The expected product should be able to use the algorithm mentioned above and split it between different computers using networking.

If We Have Extra Time:

- If we have extra time, we will provide a bar graph at the end of the simulation to show the data that was collected. It will use the x-axis as percentage and y-axis as burn time. We can also have a live progress bar of the percentage until completion and also the time it took for the entire simulation at the end.

Map of Server-Client Communication



1) If no clients are connected, the server can run the command(forest density) on the computer the code is running on using the forking method.
2) When a client(s) is connected(before the command is executed on the server):
    a) The server waits for command to execute
    b) Server will relay this command to all connected clients using networking, and ask them to send back their results(time it took for forest of density *x* to burn)
    c) Server will give statistical responses(mean, median, max, graph etc) of the time it took for density *x* forest to burn.

Development Stages

Algorithm Stage:

- Have a simulate function that takes in parameters such as map_size, initial_fire_location, and percentage_trees.
- The algorithm will go as follows:
    1. Create the 2D array of trees and start the fire by adding it to the linked list of all fire locations
    2. Loop through the linked list of all fire locations and see if there are any trees that touch (up, down, left, or right) the fire. If there is, add the location of those trees to a temporary linked list of fire locations. These trees will become the fires in the next iteration.
    3. As it loops through the linked list of all fires, turn those locations on the 2D array to dirt so future fires will not spread to that area.
    4. Change the temporary linked list of all fire locations to the real one.
    5. Increment the time counter.
    6. Repeat steps 2-5 until the temporary linked list is empty after an iteration.
    7. Return the time it took to burn.

Forking Stage:

- The simulator will be able to fork off children and run multiple simulations in order to give an average time it takes for a forest of density *x* to burn given an *n* by *n* grid(these numbers may be hard-coded at first).

Distribution Stage (requires Algorithm and Forking Stage to be completed):

- Create a function that distributes the work of simulating the forest fires to connected clients. This function will work as follows:
    1. Based on how many clients are connected, distribute work to those clients by giving them a number (percentage of trees).
    2. Use semaphores to prevent too many clients from communicating to the server their results at once.
    3. Give another number (percentage of trees) to the clients that are done with their simulation until there are no more percentages.

Networking Stage (requires Distribution Stage to be completed):

- Add the clients that connect with other computers. Use networking to communicate between the clients and the master server.

Extra Stage (Requires Distribution Stage to be completed):

- Create a bar graph from the results. The x-axis can be the percentage and the y-axis can be the burn time. There can be two graphs: one with intervals 10%, 20%, … , 90%, 100%, and another one with the maximum in the center but with intervals of 1%. (E.g. If 58% was the maximum, the graph will go from 53% to 63%)
- Print out the real-time it took for the entire simulation using some time library.