CSCE 435 Spring 2020

HW 3: Pizza Drones

Due date: 11:59pm Friday, March 6, 2020

A spate of hackjackings of pizza delivery drones has resulted in drones deviating from the delivery path to unknown destinations. Although these drones are equipped with beacons to determine their location, hackers have rendered this capability inoperative. The only means of querying a drone's path that remains active is via a hardwired interface, put in by astute far-sighted Aggie designers. When queried if the drone is at cell (x,y) in a two-dimensional grid of cells covering the region, the drone responds with the number of steps it has taken from (x,y) to its current location.

The goal of this assignment is to develop a multithreaded code to find the location of a drone in a two-dimensional grid in the shortest time possible. You are provided with the program `drone.c` that has serial code to initialize a grid with a drone in an unknown location. The drone is placed at a random location and subsequently allowed to move a fixed number of steps taken randomly. You are expected to develop a multithreaded code to find the location of the drone. The goal of your parallel implementation should be to minimize the execution time.

A header file `drone.h` is also provided that has data structures and routines associated with the grid. A summary of the important routines provided in the header file is given below.

- `initialize_grid`: initialize the size of the two-dimensional grid, place the drone in an initial random location on the grid and subsequently call `move_drone` a number of times to move the drone to one of the four neighboring cells chosen randomly each time; initialize the delay after each call to `check_grid`;
- `check_grid(x,y)`: check if the drone is present at grid location `(x,y)`, return 0 if the drone is found at `(x,y)`, otherwise return the number of steps the drone has taken since the last time it was at `(x,y)`; move the drone to a neighboring cell if there have been `move_freq` number of consecutive hits to the drone path via `check_grid` (**see note below**); return MAX_PATH_LENGTH+1 if the drone has never been to `(x,y)` and remain at its current position; introduces delay on each call to `check_grid`;
  Note: For this assignment, you may assume that the drone is not allowed to move in the `check_grid` routine.
- `check_drone_location(x,y)`: check if the location you determined is correct (to be used to determine if the computed solution is correct).

The header file should not be modified in any way. You are to assume that you cannot access the drone location from your code directly and must use `check_grid` to search for the drone.

1. (80 points) Develop a multithreaded implementation to determine the location of the drone using the interface routines provided in the header file.
2. (20 points) Your code should compute the solution quickly and efficiently, utilizing all available processors on a single compute node of `ada` or `terra`. 20 points are reserved for an efficient parallel implementation. Testing will include problems with a variety of grid sizes. The delay in `check_grid` may also be varied to test how the code performs.

Submission: You need to upload the following to eCampus:

1. Submit the file `drone.c`; you do not need to submit `drone.h`