Citadel Challenge Problem 2015

CMU Maps

Overview

It is the year 2020 and Citadel is starting a passenger service with self-driving cars. Design an algorithm to figure out the fastest way to get passengers to their destination taking into consideration traffic congestion.

1st, 2nd, 3rd places prizes are $1000, $500, $200, respectively
Submissions to autolab are open at 6pm on Thursday, January 22nd and are due by 6pm on Sunday, January 25th.

Terminology

Map

- A map is defined as a set of intersections connected by roads.
- A road has a length and is one-way.
- The length of a road is an integer always greater than zero.

Cars

- You have a fleet of cars which are controlled by your algorithm.
- Initially, cars are distributed among intersections.
- Multiple cars are allowed on an intersection.

Passengers

- Passengers spawn at intersections at random times with their destinations known.
- Passengers are picked up by explicitly specifying the passenger id to a car.
- Cars can only pick up a passenger on the same intersection on which they reside..
- Cars can only pick up one passenger at a time.
- When a car carrying a passenger reaches its destination, the passenger is automatically dropped off.

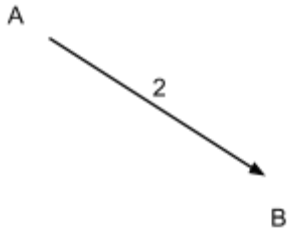
Simulation

- At the start of each simulation, you will be provided with complete information of intersections and roads.
- Subsequent simulation is turn-based.
- Each turn, multiple passengers may spawn at intersections. You will have information about their current intersection and their destination intersection.
- Each turn, you can control cars at intersections. You cannot control cars on the roads as they are considered “moving”.
- Each turn, you can assign one or both actions to a car.
    - Pickup a passenger. Passenger must be at the same location as the car.

    - Assign the next intersection which the car should move to. Next intersection must be reachable by one hop.

- Simulation ends when you have delivered all passengers to their destinations. If there are undelivered passengers, the program continues running.

Congestion Logic



- At time T, if a car is trying to enter an empty road, its time of arrival is T + L, where L is the length of the road.
    - Time t = 0, car_0 is at intersection A. Player sets car_0's next intersection to be B (ETA: t=2).
    - Time t = 1, car_0 is travelling on the road
    - Time t = 2, car_0 has reached intersection B. Player may assign next action.

- At time T, if a car is trying to enter a road which has existing cars on it, its time of arrival is T + L + C, where L is the length of the road and C is the number of cars on the road on the turn when the move command is issued. Only consider cars travelling in the same direction.
    - Time t = 0, car_0 and car_1 are at intersection A. Player sets car_0's next intersection to be B (ETA: t =2).
    - Time t = 1, car_0 is on the road. Player sets car_1's next intersection to be B (ETA: t = 4 because car_0 is on the road).
    - Time t = 2, car_0 reached intersection B. car_1 is on the road.
    - Time t = 3, car_1 is on the road.
    - Time t = 4, car_1 reached intersection B.

Scoring

- You must deliver all your passengers to their destinations to get a non-zero score. A 300 seconds timeout is set on autolab as a time limit restriction.
- Your score is calculated using the following.
    - There are 3 maps (small, medium, large) and your total score is the sum of scores from all 3 maps
    - Score for each map is 50000 - (Sum of all passenger travel times) / (number of passengers).
    - Passenger travel time is (Turn which passenger reached destination) - (Turn which passenger Spawned)

- i.e. if passenger spawned on turn 0 and reached destination on turn 5, travel time is 5.
- To get a higher score, pickup passengers and get them to their destinations fast!

Logistics

- Getting Started
  - Download the handout (citadel_challenge.tar.gz) from the autolab website at https://autolab.cs.cmu.edu/courses/41/assessments.
  - If you are not enrolled in this course, please contact citadelchallenge2015@gmail.com.
  - Extract the gzipped tar archive to obtain the `cmu_maps` directory along with its contents, e.g.

```
$ tar xzvf handout.tar.gz
cmu_maps/
cmu_maps/Makefile
cmu_maps/src/
cmu_maps/src/CarDispatcher.h
cmu_maps/src/RouteLib.h
cmu_maps/src/RouteLib.cpp
cmu_maps/src/TransportationTypes.h
cmu_maps/input/
cmu_maps/input/simple_map.txt
cmu_maps/obj/
cmu_maps/obj/maps_entry.o
cmu_maps/CitadelChallenge2015.pdf
```

    - The `Makefile` is extermely similar to the one we will use to test your code. Note taht any additional source files you add must have a `.cpp` suffix to be compiled. We will be using `g++` to compile your code.
    - `CarDispatcher.h` contains the starter interface of the `CarDispatcher` class you are required to implement. Feel free to add to this class and/or add new classes.
    - `RouteLib.cpp/h` contains code which is used to pass call your car dispatcher. do not edit these files
    - `TransportationTypes.h` contains structs which are passed to your car dispatcher..
    - `simple_map.txt` is a sample input file used for testing
    - `CitadelChallenge2015.pdf` is this file
    - `obj/maps_entry.o` is an object file with a main function which reads the input file, parses it and runs the game loop against your car dispatcher. It is used when compiling your binary.

  - Start implementing your solution in the cmu_maps directory. Remember that our Makefile will compile and link all .C files.

- Submitting your solution

- Writing to STDOUT or STDERR will result in a score of zero. These file descriptors are reserved for our main() function.
- Your submission must be a gzipped tar archive named `handin.tar.gz`.
- When extracted, the tar archive must expand into a directory named `cmu_maps`.
- Any additional source files you have created must be in the src directory.
- Include a copy of your resume and name it `resume.pdf`.
- In your handin tar archive that is named `handin.tar.gz`, we would expect to see at least the following(in addition to any other source files you have added) when extracting your handin.

```
$ tar xzvf handin.tar.gz
cmu_maps/
cmu_maps/src/
cmu_maps/src/CarDispatcher.h
cmu_maps/src/RouteLib.h
cmu_maps/src/RouteLib.cpp
cmu_maps/src/TransportationTypes.h
cmu_maps/resume.pdf
cmu_maps/obj/
cmu_maps/obj/maps_entry.o
cmu_maps/Makefile
```

- Submissions must be made to autolab. Entries will receive a score of zero if it does not pass the tests or if it does not compile. Otherwise, score will be based on our scoring function above.

- Testing your implementation.
  - To compile your code, simply type ‘make’ on your command line.
  - Test your code with the sample simple_maps.txt file.

```
[cmu_maps] $ make
      [...output ...]
[cmu_maps] $ ./cmu_maps input/simple_map.txt
```

- Entries must be submitted before 6:00PM Sunday, January 25th
- Winners will be revealed following a tech talk led by Citadel concerning this challenge. Participants must be present in order to be eligible to win. The discussion will be in Gates & Hillman - Reddy conference room 4405 at 5pm on Monday, January 26th. In the event of a tie, winners will be selected at random..
- Email any questions to citadelchallenge2015@gmail.com.