Outlook

Challenge Instructions

From donotreply@ultamation.com <donotreply@ultamation.com>
Date Wed 16/10/2024 14:00

Welcome, Bow
Your ID is: 410b8eaa-ddfa-494e-9baa-06aa30d05e6f
THE CHALLENGE

You've been hired by a courier service that need help optimising their delivery routes. Your objective is simple: given a dataset of delivery nodes, find the shortest path that visits all delivery locations and returns the vehicle to the depot.

Your solution will be scored based on its efficiency so you'll need to minimise the total distance travelled. The challenge begins with a manageable number of deliveries, but you'll face stages of increasing difficulty.

Challenge Details
==========

Each stage has delivery nodes which your route must plan to visit exactly once, starting and ending at the depot node 0. The Euclidean distance between all nodes can be calculated using the distance matrix supplied by the /data endpoint.

The way you determine the order of nodes is up to you, but all solutions must generate a list that starts and ends with the depot node and includes all of the delivery nodes from 1 to n, where n is the final delivery node. This should be sent to the /answer endpoint when you're ready to submit (explained below).

Stage 1:
- GOAL: find the shortest route.
- 10 delivery nodes i.e. (1,2,3,4,5,6,7,8,9,10).

Stage 2:
- GOAL: find the shortest route.
- 100 delivery nodes.

Stage 3:
- GOAL: find the shortest route without running out of fuel. Ensure you don't run out, or you'll fail this stage!
- The vehicle has a fuel capacity of 100 miles. To refuel, you must visit a fuel station where your

capacity is reset to 100.
- Fuel stations are located at nodes 20, 40, 60 and 80. Therefore you may visit the same fuel station multiple times but it is NOT mandatory to visit them all.
- 100 nodes: 96 delivery nodes and 4 'fuel station' nodes for refuelling.

Stage 4:
- GOAL: find the shortest path with 3 vehicles.
- You have 3 vehicles available, where each must perform exactly 100 deliveries each. No more, no less!
- Submit the solution as three separate lists, each representing the route for one vehicle. Each list is a partition of delivery routes for the vehicles, where each handles a distinct set of deliveries.
- 300 delivery nodes.

You must complete each stage in order, but you can backtrack and re-submit any stage once you've passed. They're designed to help you build your solutions incrementally.

The winner will be the one who has completed the most stages AND the lowest overall distance across the 4 solutions, so manage your time between each stage wisely!

The Technical Part
==========
You will connect to the challenge using an industry-standard RESTful API.

The leaderboard display will show you the IP address of the challenge server.

When communicating with the server, you must provide your player id (which has been sent to you via email) and current stage as headers with the following key-value pairs:

playerid: <your unique player id>
stage: <stage number, e.g. 1, 2, 3 or 4>

Failure to provide the headers, or providing an invalid value, will return an error.

An *optional* starter project that implements the networking GET/POST is provided in Python and Java. However, you will receive a 10% score penalty for ALL SUBMISSIONS in return for the head start, so choose wisely!

To download the starter project, in a browser go to: http://<server>:8081/starter, and input your player ID sent by email.

Once you've submitted your answers, you can also go to http://<server>:8081/results to find your best score for each stage.

GET Data
==========
GET http://<server>:8081/data

Your dataset is formatted as a distance matrix which is served by this endpoint which you may parse into a suitable data structure. Node 0 represents the depot and the rest represent delivery nodes (unless specified otherwise, such as stage 3).

To retrieve the distance between node i and j, you would access the value at matrix[i][j].

POST Answer
==========
POST http://<server>:8081/answer

The requirements for the solution is that you must submit the order of nodes visited, starting with and ending with 0.

This is a POST request and you should include your solution as application/x-www-form-urlencoded key value, using the key route.

In response to your requests the server returns a message. If the response status code is 200 OK your answer has been accepted, otherwise it has been rejected and will respond with an error message.

For example, your data might be:

route=0,1,2,3,4,5,6,7,8,9,10,0

For the final stage, you'll need to submit three separate numbered routes within the request body. For example, this may look something like:

route1=0,1,2,3,0
route2=0,4,5,6,0
route3=0,7,8,9,0

The response to this request will be the distance travelled for the current stage, where we'll present your best (lowest) score on the leaderboard.

Please don't call this endpoint in a loop with many iterations! While you are working please try to evaluate your solution based on the distance travelled.

Rules
=====
* You can submit as many solutions as you like.
* You must submit the stages in order, but once you have progressed to the next stage, you can backtrack and re-submit previous solutions.
* Only successful routes will be registered.
* Your best route score will be presented on the leaderboard.
* If two players obtain the same route score, the player that submits their solution first will be placed higher.
* Don't cheat or use optimisation libraries. You may only use libraries for data structures such as NumPy or Pandas.
* Don't use LLMs to generate solutions. If you win, we may ask you to explain your solution!
* Don't interfere with any other players, or try to sabotage other player's results - anything suspicious and you will be disqualified. (and we are logging all API activity).
* While this is a competition, the aim is not to hack the service. You may think that's clever - we will not.

--