Project 1: Delivery Man

Artificial Intelligence (IDL340)

Basics

- This project should be performed alone or in pairs.
- You will use the R programming language.
- The use of third party code libraries/R packages is not permitted.
- The required code is available in the DeliveryMan_1.0.1.zip file on student portal.
- See the R FAQ document on student portal for questions about using R.

Project Overview

You will compete in the Delivery Man game. You will be expected to use the A* algorithm to find optimal routes to work out what moves you should make in the game. Read the runDeliveryMan help documentation in the DeliveryMan package for more detail. A* will permit you to find the optimal route between two points given current traffic conditions. This is important, but not the only important matter in performing well in the game. You will have to think about what else is important and how you can perform you deliveries as well as possible.

You will provide

An R script with a function that can be passed to the runDeliveryMan function.

Pass Requirements

Your function needs to:
- Equal or surpass a performance of 180 (ie get 180 or less)
- Meet the execution time requirement

See documentation on the testDM function in the DeliveryMan package for details.

Important Functions

The important functions in the DeliveryMan package are:

Function Name	**Description**
runDeliveryMan	Runs one game of the Delivery Man game. The help documentation contains important information required for completing this project.

testDM	A simulation of the process that will be used to evaluate you (just with a different random seed), with par performance and time for this simulation given in the help documentation.

Example Control Functions

In addition, there are three control functions included, that you can pass to the runDeliveryMan function:

1. dumbDM
2. basicDM
3. manualDM

Examine their help documentation for details. Note that none of these implement A*.

Dangers

Deliveries and pickups can be placed in the same square. You will need to stay still if you want to pick up a pick up where you dropped off a delivery.

Competition Hints (how to improve on the par function)

The obvious short-coming in the par function is its greedy and sub-optimal determination of the next package to target: When the vehicle has no package, it looks for the (distance) nearest pick-up and heads off. It should improve matters to look for the (cost) nearest package. It would improve matters even more to try and work out a good ordering of pick-ups and deliveries, though this a more difficult matter.

It might also be possible to take advantage of the fact that traffic conditions start off easy (all roads start with cost 1) and then deteriorate until reaching a steady state (it is unclear if such a steady state is reached during the timespan of the game, or if conditions continue to deteriorate during expected game length - I have not investigated it).

At a more advanced level, it would be possible to incorporate these evolution probabilities into the algorithm, either naively or rigorously looking at expected cost when approaching a road. The evolution of the traffic conditions is: Each turn there is a .05 chance the road cost increases by 1, and, if the road does not already have a cost of 1, a .05 chance the road cost decreases by 1. It is the imbalance caused by the presence of the cost floor of 1 that makes it possible to exploit knowledge of these conditions, but it might not make a very large difference (I haven't tried).

Remember: Implement the basic algorithm and ensure you have a passing implementation before trying to be ambitious!