UNIVERSITI MALAYA

Managerial Report
WIA1002/WIB1002
Data Structure Assignment
Semester 2 2020/2021

Project 2: 'Always On Time' Delivery

Your friend's delivery company 'Never On Time Sdn Bhd' is receiving tons of complaints from customers as they feel that the delivery process is far too slow. Delivery men in your friend's company are always lost in the middle of their delivery route, don't know where to deliver the parcel and which road they should take to shorten the delivery time. You as a professional engineer are requested to simulate the delivery process and planning in a country to help your friend shorten their delivery time.

Tutorial & Lab Group: 7
Lecturer: Dr. Muhammad Shahreeza Safiruz Kassim
Instructor: Hoe Jiun Tian
Prepared by:

Name	Matric No.
YAU DE MIN	U2005347/1
HONG ZHAO CHENG	U2005280/1
CHIEW ZHE WEI	U2005368/1
WONG YU XUAN	U2005388/1

Task

We are assigned to write a program to simulate the delivery process and planning in a country to help our friend shorten their delivery time. This stimulation is designed with some additional features too.

Task Requirements

There are several basic requirements that are required in this project to simulate the delivery process and planning to shorten the delivery time:

Customer

Entity that has a certain demand with coordinates. This is represented by our Node class which include several fields like coordinates x, y, demand and id etc. This class includes several accessors and mutators for the fields besides methods used for evaluation. Each Node object (Customer) have an id for us to identify each distinct customer and the depot (ID = 0)

Vehicle

Unit that can move between customers and the depot, a unit that initially possesses the demands of the customers represented by Vehicle Class. All vehicles are capacitated so that they can only contain goods (the customer's demands) up to a certain maximum capacity (set by the user). This class contains several fields that specify the state of a vehicle like capacity and list of paths taken (Sequence of Nodes from depot to depot which customers in between). Similarly each vehicle has their own id for identification. This class also contains methods to alter the behaviour and state of a Vehicle object such as adding a Node (Customer) and evaluating whether adding a customer violates the rules (Maximum capacity exceeded).

Tour

A tour is a list of routes of all vehicles to fulfil all customers' demands. The tours are generated by each simulation with information regarding the total cost and details of each vehicle. A tour should include a list of vehicles in which each vehicle has their own distinct path taken. A tour's cost is simply the summation of all routes' costs while a route's cost is simply the total Euclidean distance travelled by the vehicle using that route.

The different types of simulation to find out the best tour with lowest cost are :
Basic Simulation
Greedy Simulation
MCTS Simulation
Extra Features (A*, Best First, Grp123 algo, Site Dependency + Heterogenous Vehicle Capacity)

Approach

We have designed several algorithms to simulate the delivery process and planning in a country to help 'Never On Time Sdn Bhd' to shorten their delivery time. All the requirements are fulfilled and we have implemented several extra features such as A* Search, Best First Search and many more. Each simulation will return a computed Tour Cost which includes the number of vehicles, sequence of nodes (Customers), and total cost path of each vehicle.

To make our system more user friendly , our system will first prompt the users to enter the text file which contains inputs to be tested. After that, there are 8 choices for users to select in which different simulations give different total cost paths (route) due to different implementations. Each simulation has their pros and cons. For example, Greedy Simulation returns good tour cost output in a very short amount of time. However for Basic simulation, although the run time is longer, it will return the best tour cost (cheapest).

Sample:

Choose algorithm to generate best route
1 - Basic Simulation
2 - Greedy Simulation
3 - MCTS Simulation
4 - A* Search Simulation
5 - Best First Search Simulation
6 - Conditional Simulation
7 - Simulation (SiteDependent + Homogeneous Capacity)
8 - Exit

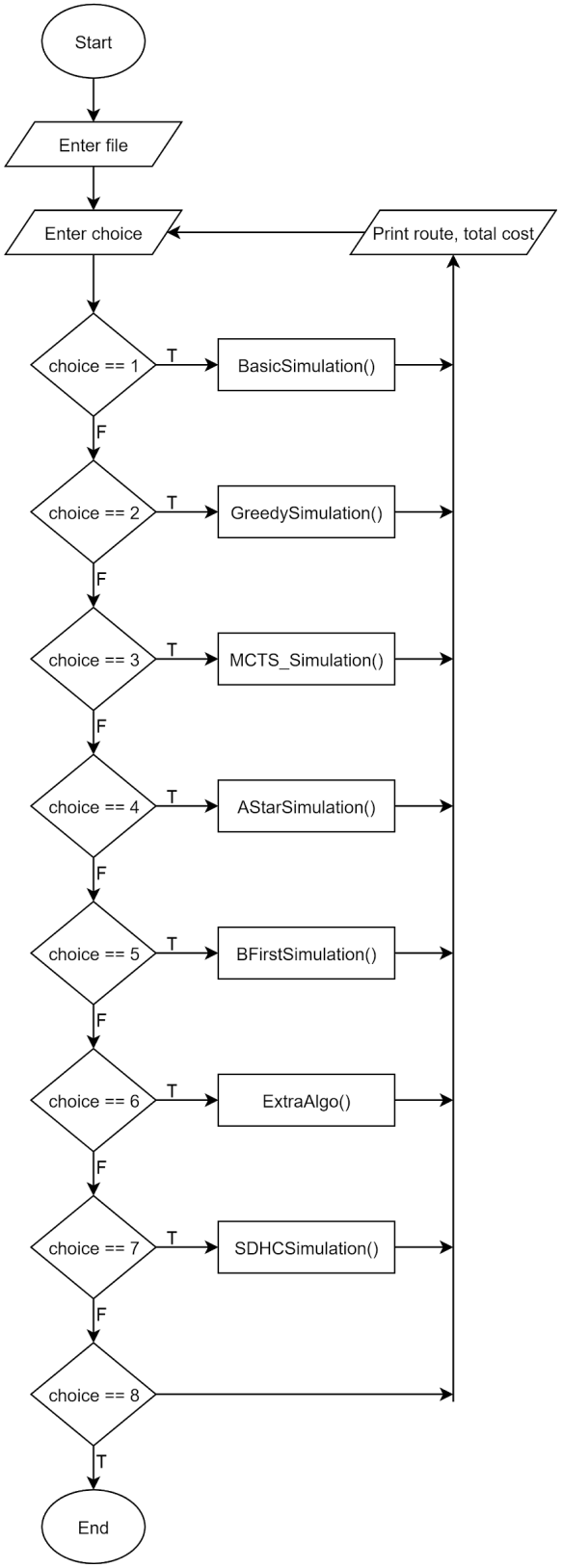
Firstly, we are requested to find the best route for a given case with small N using Breadth-First Search/Depth-First Search traversal implementation. We name this approach 'Basic Simulation'. Secondly, we have implemented a Greedy Search which is able to find a good solution given a case with small or large N. We name our second approach as greedy simulation. Other than that, we implemented a Monte Carlo Tree Search which helped my friend to search for the best tour that he could search in a limited computation time. We name this as MCTS simulation.

Lastly, for extra features, we implemented four types of simulation which are A* Search Simulation, Best First Search Simulation. For Greedy Simulation, we had added two extra features which are Heterogeneous Vehicle Capacity and Site Dependency which are used as important parameters in computing the final tour cost.

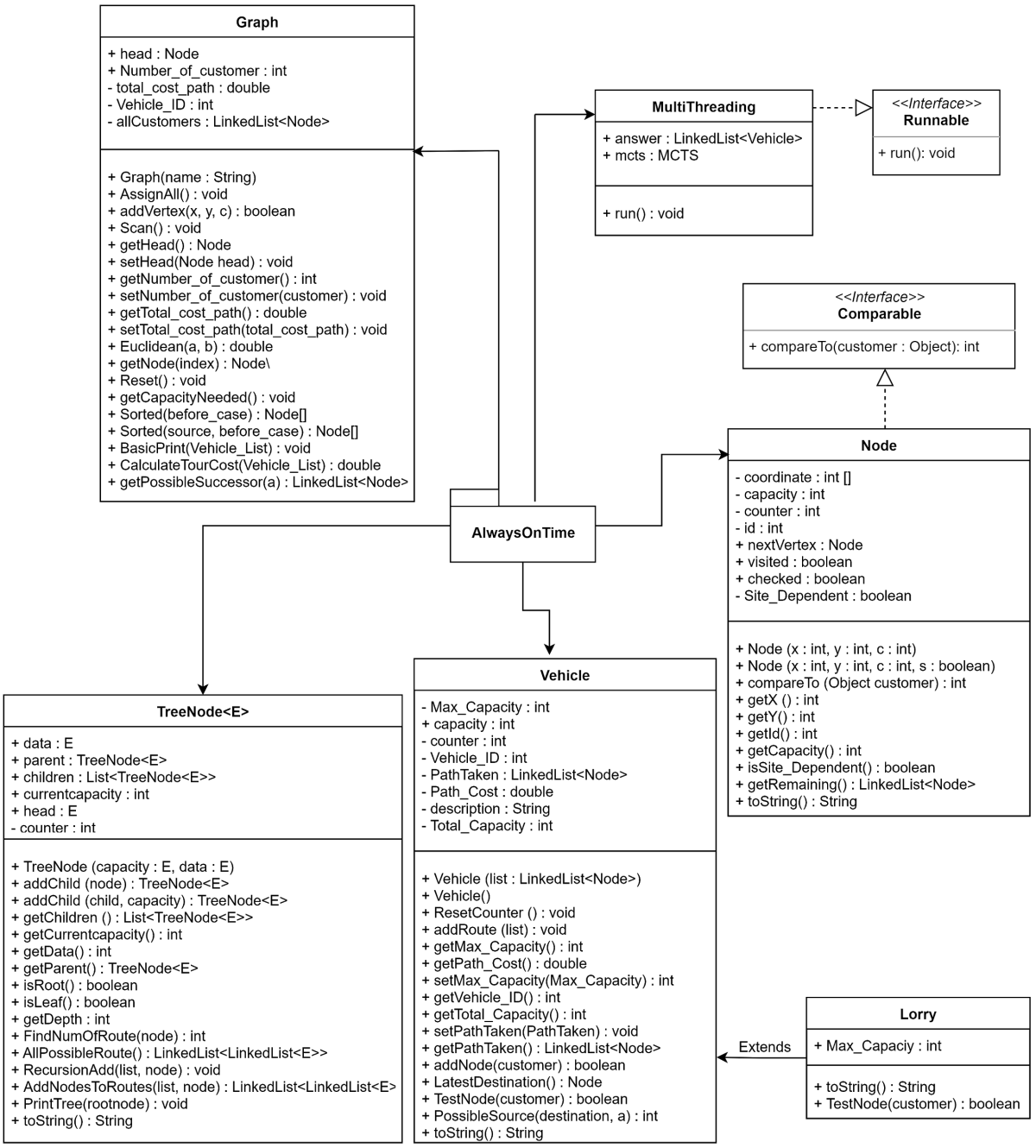
Solution Description

Flow chart

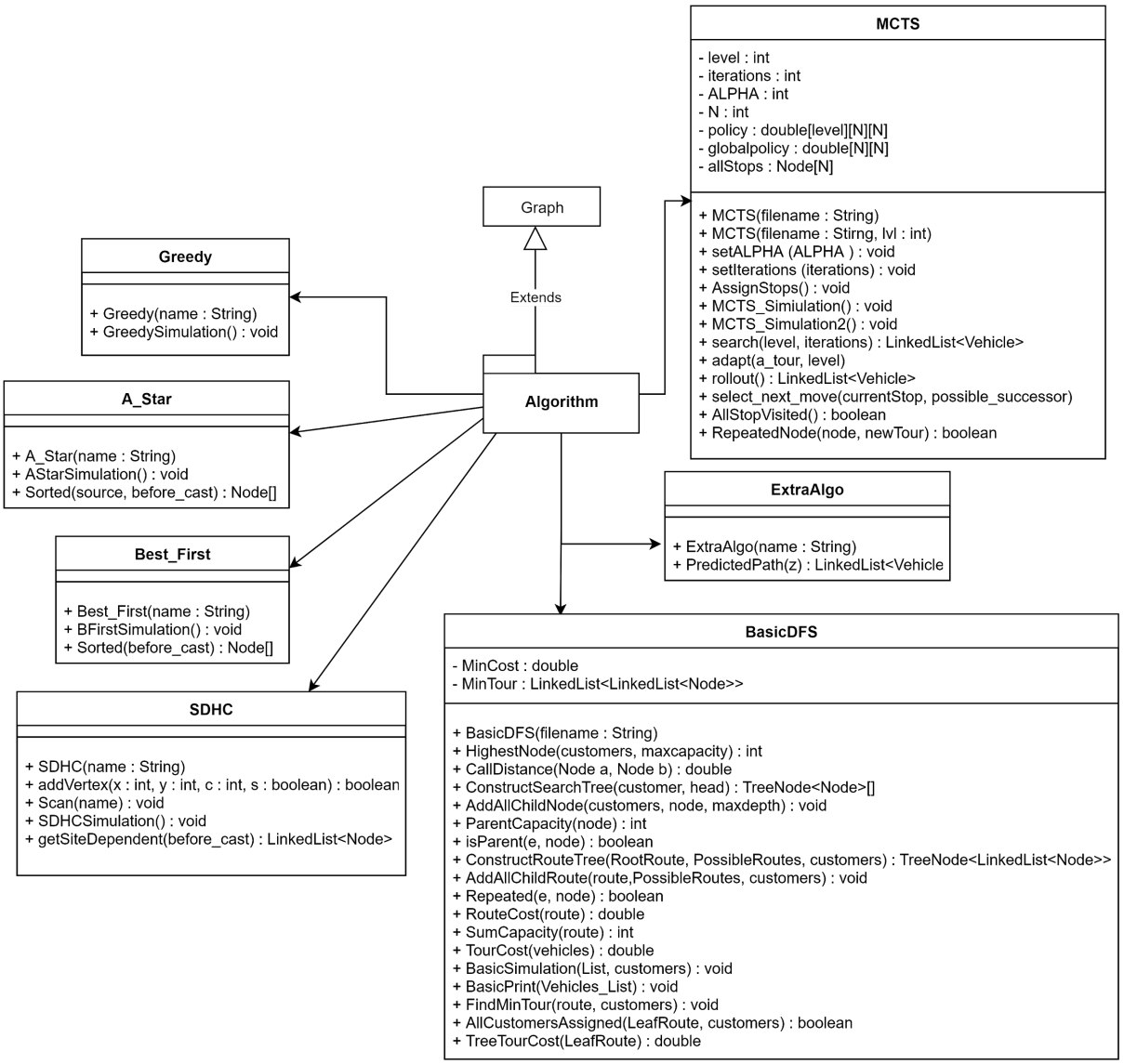
Main Method



Package AlwaysOnTime



Package Algorithm



Technical Difficulties

Difficulties	**Solutions**
Large amount of coding from various group members for each simulation.	Vast amount of code makes it impractical to use copy paste in compiling the whole project. Use Github application for cooperation between members, each member can do their coding in the same project repository.
Some simulations require different formats of txt file. Ex: Greedy Simulation with extra features requires an additional boolean parameter to specify whether a customer is Site Dependent or not.	To prevent index out of bound error when scanning the file (For normal simulations) and passing in parameters, we override the scan method in extra features so that it will only scan and pass in the additional parameter when required.
Different implementation and use of core program's methods and fields for each member in coding different simulations. Ex: method and fields in Graph, Node, Vehicle class.	Use github to collaborate with each other, so that team members can add on the coding so that our data structure can support all types of simulations.
Long run time for BFS simulation which involved generating a tree and tree transversal for best tour.	Set a 60 seconds timer, in which after the 60 seconds will return the latest found tour instead of Best tour (For cases with large N). Removing the timer will guarantee a best tour but run time will exceed 5 minutes.

Sample Input & Output

User menu

```
**********************************
------Never On Time Sdn Bhd------
**********************************
Enter file: 123.txt
**********************************
Choose algorithm to generate best route
1 - Basic Simulation
2 - Greedy Simulation
3 - MCTS Simulation
4 - A* Search Simulation
5 - Best First Search Simulation
6 - Grp123 Simulation
7 - Simulation (SiteDependent + Homegenous Capacity)
8 - Exit
**********************************
```

Basic Simulation

```
-----------------------------
Enter choice to proceed: 1
-----------------------------


Maximum time set is 50 seconds.

Basic Simulation
Tour
Total Cost: 1939.7805837686146
Vehicle 1
0 -> 9 -> 3 -> 0
Capacity:39
Cost: 437.3857488081559
Vehicle 2
0 -> 8 -> 0
Capacity:39
Cost: 306.9397334982879
Vehicle 3
0 -> 6 -> 0
Capacity:42
Cost: 263.21094202179364
Vehicle 4
0 -> 4 -> 0
Capacity:32
Cost: 178.4040358287895
Vehicle 5
0 -> 5 -> 2 -> 0
Capacity:42
Cost: 461.51342131561523
Vehicle 6
0 -> 7 -> 1 -> 0
Capacity:38
Cost: 292.32670229597255

Time Elapsed : 9 s
```

Greedy Simulation

```
-----------------------------
Enter choice to proceed: 2
-----------------------------

No limited time is set since the simulation time is too fast


Greedy simulation
Tour
Total Cost: 2138.005597521426
Vehicle 1
0 -> 1 -> 3 -> 2 -> 0
Capacity:34
Cost: 326.7196961741431
Vehicle 2
0 -> 4 -> 0
Capacity:32
Cost: 178.4040358287895
Vehicle 3
0 -> 7 -> 0
Capacity:24
Cost: 216.90550938138938
Vehicle 4
0 -> 6 -> 0
Capacity:42
Cost: 263.21094202179364
Vehicle 5
0 -> 8 -> 0
Capacity:39
Cost: 306.9397334982879
Vehicle 6
0 -> 9 -> 0
Capacity:34
Cost: 415.24932269661804
Vehicle 7
0 -> 5 -> 0
Capacity:27
Cost: 430.5763579204042

Time Elapsed : 0 s
```

MCTS Simulation (Elapsed time no show because of threading)

```
-----------------------------
Enter choice to proceed: 3
-----------------------------

MCTS Simulation
Tour
Total Cost: 1939.7805837686146
Vehicle 1
0 -> 9 -> 3 -> 0
Capacity:39
Cost: 437.3857488081559
Vehicle 2
0 -> 6 -> 0
Capacity:42
Cost: 263.21094202179364
Vehicle 3
0 -> 4 -> 0
Capacity:32
Cost: 178.4040358287895
Vehicle 4
0 -> 8 -> 0
Capacity:39
Cost: 306.9397334982879
Vehicle 5
0 -> 2 -> 5 -> 0
Capacity:42
Cost: 461.51342131561523
Vehicle 6
0 -> 1 -> 7 -> 0
Capacity:38
Cost: 292.3267022959726
-----------------------------
Would you like to run MCTS all over again? (true/false)
```

A* Simulation

```
-----------------------------
Enter choice to proceed: 4
-----------------------------

No limited time is set since the simulation time is too fast


A* simulation
Tour
Total Cost: 2138.005597521426
Vehicle 1
0 -> 1 -> 3 -> 2 -> 0
Capacity:34
Cost: 326.7196961741431
Vehicle 2
0 -> 4 -> 0
Capacity:32
Cost: 178.4040358287895
Vehicle 3
0 -> 7 -> 0
Capacity:24
Cost: 216.90550938138938
Vehicle 4
0 -> 6 -> 0
Capacity:42
Cost: 263.21094202179364
Vehicle 5
0 -> 8 -> 0
Capacity:39
Cost: 306.9397334982879
Vehicle 6
0 -> 9 -> 0
Capacity:34
Cost: 415.24932269661804
Vehicle 7
0 -> 5 -> 0
Capacity:27
Cost: 430.5763579204042

Time Elapsed : 0 s
```

Best First Simulation

```
------------------------------
Enter choice to proceed: 5
------------------------------

No limited time is set since the simulation time is too fast


Best First simulation
Tour
Total Cost: 2031.5156606058044
Vehicle 1
0 -> 1 -> 7 -> 3 -> 0
Capacity:43
Cost: 406.19820524470003
Vehicle 2
0 -> 4 -> 0
Capacity:32
Cost: 178.4040358287895
Vehicle 3
0 -> 6 -> 0
Capacity:42
Cost: 263.21094202179364
Vehicle 4
0 -> 8 -> 0
Capacity:39
Cost: 306.9397334982879
Vehicle 5
0 -> 2 -> 5 -> 0
Capacity:42
Cost: 461.51342131561523
Vehicle 6
0 -> 9 -> 0
Capacity:34
Cost: 415.24932269661804

Time Elapsed : 0 s
```

Group123 Simulation

```
Enter choice to proceed: 6
------------------------------

No limited time is set since the simulation time is too fast


Grp123 Simulation (Extra Feature)
Tour
Total Cost: 2008.0617803980304
Vehicle 1
0 -> 4 -> 3 -> 0
Capacity:37
Cost: 268.82165856974285
Vehicle 2
0 -> 5 -> 2 -> 0
Capacity:42
Cost: 461.51342131561523
Vehicle 3
0 -> 6 -> 0
Capacity:42
Cost: 263.21094202179364
Vehicle 4
0 -> 7 -> 1 -> 0
Capacity:38
Cost: 292.32670229597255
Vehicle 5
0 -> 8 -> 0
Capacity:39
Cost: 306.9397334982879
Vehicle 6
0 -> 9 -> 0
Capacity:34
Cost: 415.24932269661804

Time Elapsed : 0 s
```

Simulation (SiteDependent + Homogeneous Capacity)

```
------------------------------
Enter choice to proceed: 7
------------------------------

No limited time is set since the simulation time is too fast

Greedy simulation added with extra features
Tour
Total Cost: 2223.7814910887846
Vehicle 3
0 -> 4 -> 3 -> 0
Capacity:37
Cost: 268.82165856974285
Vehicle 5
0 -> 7 -> 0
Capacity:24
Cost: 216.90550938138938
Vehicle 8
0 -> 5 -> 0
Capacity:27
Cost: 430.5763579204042
Lorry: Vehicle 3
0 -> 1 -> 2 -> 0
Capacity:29
Cost: 322.0779670005486
Lorry: Vehicle 6
0 -> 6 -> 0
Capacity:42
Cost: 263.21094202179364
Lorry: Vehicle 7
0 -> 8 -> 0
Capacity:39
Cost: 306.9397334982879
Lorry: Vehicle 8
0 -> 9 -> 0
Capacity:34
Cost: 415.24932269661804

Time Elapsed : 0 s




------------------------------
Enter choice to proceed: 8
------------------------------


Exit successfully
BUILD SUCCESSFUL (total time: 10 minutes 5 seconds)
```

Extra Features

These are a few of our major extra features:

Best First Simulation:
Best First Search means we simply look for the node which looks closest to the goal in the next move when we travel through the whole graph. For illustration, in this context the vehicle will always look for the shortest distance between remaining unserviced customer location (coordinates) and depot (goal) to select the next location to go to.

A* Simulation:
A* Search means we simply look for the node which looks most promising to explore through the whole graph. For illustration, in this context the vehicle will always look for the shortest distance between current location and all next possible locations and also the distance from depot (goal node) to select the next location to go. The implementation is almost similar to Greedy Simulation but we need to take into account the Euclidean distance between a node and depot apart from distance between nodes.

Grp123 Simulation:
This is a custom searching algorithm from our group members. One of its major advantages is that it does not require tree formation and transversal which save a great amount of run time. As for tour cost it is guaranteed to return a cost which is lower than Greedy Simulation and almost similar with the best tour (with a difference of less than 10%).

In this algorithm, we used prediction to first initialise a certain number (minimum) of vehicles required to satisfy the customer demand. Firstly, we will evaluate each customer's demand whether it is bigger than ⅔ the max capacity of a vehicle. Given the former condition is true, we will initialise a vehicle to service this customer with path Depot→ High Demand Customer. The conditions taken into consideration here are these customers cannot share a vehicle with other high demand customers (> ⅔ capacity) which permits them to own each of their own service vehicles.

Next, we will evaluate the current number of vehicles as compared to the total demand of unserviced customers to predict the number of extra vehicles needed. The newly initialised extra vehicles objects will use the closest unserviced customer as their first destination. For the remaining nodes, we will evaluate the distance between the customer and the latest customers of each initialised vehicle to determine the cheapest vehicle (cheapest total path) to add this node given the rules are not violated. Subsequently, if the current number of vehicles is not enough to accommodate the remaining customers, we will initialise new vehicles using the rules mentioned above.

Heterogenous Capacity Vehicle and Site Dependent Customers:

Both of these extra features are only incorporated into Greedy Simulation since Basic simulation and MCTS simulation have a relatively longer run time. Heterogeneous capacity means we might have different types of vehicles that have different capacity (e.g. a lorry can deliver more loads than a van). In this case we have included a Lorry class which inherits from Vehicle class in which the capacity is increased by 5. As for site dependent cases, not every type of vehicle can serve every type of customer because of site-dependent restrictions. For example, customers located in very narrow streets cannot be served by a very big truck, and customers with very high demands require large vehicles. In this context, we use boolean input to specify whether a customer is site dependent and demand > MaxCapcity/2 to specify high demand customers (Must use Lorry).

The difference from Conventional Greedy Simulation is that vehicle object are first used to look for the best option in the next move (Must be site dependent / not high demand) when we travel through the whole graph. Then the remaining nodes (Customers) will all be serviced using the Lorry object using the similar approach mentioned above to find the next best move (Node).

Sample Input: (True indicates it is site dependent)

```
10 43
22 190 0 false
23 101 14 false
54 34 15 false
55 62 5 false
28 101 32 true
137 8 27 true
148 152 42 false
113 131 24 true
175 178 39 false
144 22 34 false
```

Parallelism(Threading)

This extra feature is applied for the MCTS algorithm. Instead of waiting for the result of MCTS simulation which can cost some time, users can actually run other simulations first. A new MCTS thread is created and started using default hyperparameters once the file name is entered. If the MCTS thread has found the result, the user can select MCTS simulation to display the result. If MCTS simulation is called before MCTS simulation manages to produce a result, a simple message will be shown. A time constraint of 50 seconds is set for each MCTS simulation.

```
Enter choice to proceed: 3
----------------------------

MCTS Simulation thread is still searching for the result. Come back later, we will notify you once complete.
```

At the same time, users no need to wait until the MCTS result is given and can directly run other simulations in the program. Once the MCTS simulation is done, messages to remind users will be shown and now users are free to enter 3 to display the result of MCTS simulation.

```
*MCTS simulation running in another thread is done. Choose MCTS Simulation to show result.*
*This MCTS simulation used hypeparameters: level= 3, iterations= 100, ALPHA= 1*
```

Users can also choose whether to run the MCTS all over again and also whether to define new values of levels, iterations and ALPHA after the MCTS simulation runs at least one time.

```
Would you like to run MCTS all over again? (true/false)
true
Would you like to custom the hyperparameters? (true/false)
true
Enter number of levels: (<=3 levels is strongly recommended,level above 3 will exponentially increase time consumption
3
Enter number of iterations:
100
Enter value of ALPHA:
1
A new MCTS thread has been created. We will notify you once the simulation is done
```

Limitations

Limitations	Explanations
Long run-time for Basic Simulation for cases with large N	When the number of nodes is large, a large tree will be generated. Subsequently the time taken for transversal of the whole tree and generation of all possible paths increase exponentially.
A 60 seconds timer is insufficient for Basic Simulation to return a tour for cases with large numbers of N. Thus we need to increase the timer exponentially for these cases.	Due to a large tree with a large number of different branches, 60 seconds is insufficient to traverse the tree to find all possible paths, thus unable to find any tour. (Combinations of the paths)
MCTS simulations with same hyperparameters may produce different result	Due to the nature of MCTS simulation that uses random numbers, the result might deviate a little from the best result. To confirm that a result is the best result, MCTS simulation has to be run more than one time.