DIAL A RIDE PROBLEM

PROBLEM SCENARIO:

Given here are different locations in the whole city. Each of the location is provided with its distance to every other location across the city. There are passengers present at these locations at different point of time for a specific period. These passengers are waiting for certain cab to arrive which could take them to their destination location. Also there are cabs which have the responsibility of carrying passenger to their respective destination.

AIM :

The main aim of the problem is to schedule the cabs to move across different location in the city in such a way that they could satisfy requests of the passengers and generate a large revenue out of them.

CONSTRAINTS :

1. At any given instant of time one cab can carry as many passengers as specified by capacity of taxi.
2. Passengers have to be picked up from their source location (location of boarding the cab) within a defined period of time.
3. No restriction over the time at which passengers are dropped to their destination location.
4. The passenger pays according to the shortest distance between the source and destination location.

ASSUMPTIONS :

1. Each of the taxi is considered as an independent entity. The updation of information in one taxi is independent of the other. Eg. At a given time the time associated with one taxi is independent of the other taxis.
2. The speed of taxi is 1km/2minutes.
3. All the time is calculated and given in terms of minutes.

4. All the taxi time is 0 in the beginning indicating mid night.

REPRESENTATION OF INPUT DATA :

The different locations present in the city are provided in the form of adjacency matrix, here each of the location of the city is considered as one node and the distance in between the locations is edge of the two city location nodes in the graph.

The various attributes like capacity defined for a cab, total number of requests in the problem statement, total cabs in the city, Total number of locations in the city are stored within different variables.
As the two main entities in the above problem are cab and passenger which will interact with each other using all other information provided, in order to represent them structures are used.

Taxi Structure:

The taxi structure consists of information as

1. Taxi id → to uniquely identify a taxi.
2. Taxi location → Current location of the taxi.
3. Taxi Time → Time at which a request is processed
4. Capacity → Current capacity of the taxi

Request Structure:

The request structure consists of following information :

1. Request id → To uniquely identify a request.
2. Waiting start time → the time at which the request starts waiting at its source location
3. Waiting end time → the time at which the request ends waiting at its source location.
4. Source → The source location.
5. Destination → The destination location of the request.

Output Structure:

The output structure consists of following information :

1. Taxi id: the taxi which processed a certain request.
2. Request: The request processed by a particular taxi.
3. Revenue : The revenue of a particular taxi

Shortest distance array:

2 Dimensional arrays is used for representing all the locations and the distance in between them.

APPROACH:

Dropping while Picking:

Main approach used here is to keep on adding passenger to the taxi if the taxi satisfy the time constraints specified by the passenger request, until taxi capacity is full meanwhile, keep on updating the revenue time, capacity , location of the taxi.
Also while adding passenger to the taxi check if there is any request already present in the taxi whose destination location is same as current request source location. If it is so drop that passenger to that location and also update the information.

Picking while Dropping :

If the number of the passengers in the cab is equal to the capacity of the cab then sort all the request on the basis of there shortest distance from the current cab position.

Request with shortest distance is chosen and dropped to its destination while updating current cab position to its destination then continue with processing remaining requests performing above operation until the entire cab is empty. As the request is allocated to taxi, information about the current location of the cab, current time of the cab, total revenue associated with the cab is updated.

While dropping the passenger to their destination if any request is identified in the passenger's destination location during that time then pick that request and update information associated with that request.

ALGORITHM :

1. Take input from the file the variables like capacity of cab, number of locations in the city, number of requests, number of cabs.
2. Take input from the file the input containing distance between the different locations in the city and store it in a 2 dimensional array.
3. Apply Floyd Warshall algorithm to calculate the minimum distance in between all the locations in the city.
4. Create structure for both the cabs and the passenger.
5. The taxi structure have taxi id (for unique identification of the taxi), taxi time (each taxi is associated with a unique which keeps on incrementing for each of the taxi as it keeps on moving), current location of the taxi (the point at which taxi is present at current time period).
6. Passenger structure have request id(unique identification of each passenger), start time of waiting, end time of waiting, source location , destination location.
7. Sort the request on the basis of start time of waiting of the passenger.
8. For each of the request
    1. sort the cabs according to the shortest distance between the cab and the source location of the request in non decreasing order.
    2. for each cab

b1. if the taxi is able to reach to passenger within its specified time limit update the current time and location associated with the taxi also update the revenue and add the request to the cab.

Meanwhile if there is any request in the taxi whose destination is the source location of the current request which is been processed then drop him to its destination decrease the total number passenger in the taxi.

Otherwise Repeat step 8

b2. If at any time the number of passengers exceeds the capacity of taxi then for that taxi keep on dropping the passenger to their respective destination until no passenger is left in the taxi. Also while dropping the passengers check for every dropping location if there is any request which can be picked at the time within which taxi is dropping any specific request

a If there is any such request and taxi reaches within request's specified time of waiting then pickup that request.

otherwise continue to drop passengers.

OPTIMIZATIONS :

1. Using the concept of Drop While Pick. This will cause to drop the passenger request sitting already in taxi while picking another request from a specified location and decrease overhead of coming back to that location again to drop the same request leading to unnecessary increase in time which could be utilized to pick up some another request

2. Dropping in ascending order of distance from current location of taxi: Once the capacity of taxi is full then sort all the destination location of the request in the cab from current location of the taxi in increasing order of their distance . Drop the request at nearest distance from the taxi's current location and update the taxi location as the destination location of the dropped request. Again sort remaining request until all the request are dropped to their destination.

3. Wait for the request if taxi reached before starting wait time of the request : In order to increase possibility of adding a request to a taxi if a taxi reaches a location before waiting start time of the request then wait for the request to arrive at the its source location.

4. Check for the nearest taxi first and if it satisfies the request constraints laid by the request allocate request to the taxi: This is done to allocate the passenger to a taxi so that minimum update to the time of taxi has to be done. This time in turn could be used for identifying another request for more revenue.

5. While dropping the passenger to their destination location identify if there is any request which is present within the destination location of the request to be dropped. If there is such request and it is waiting during the time cab is dropping passenger then take the request and update the information about the request within the structure.

REFERENCES :

1. www.stackoverflow.com
2. Venkatesh Vishwarup
3. Devesh Singh Rawat
4. Ajay Tiwari
5. ALGORITHMS COURSE BOOK (CORMEN)