Emergency Vehicle Allocation

DAA Project

Git Hub URL

https://github.com/meghanagabhushan/Algorithms-Project

Team Members

Sujitha Puthana - 16233500

Megha Nagabhushan - 16226858

Manvitha Vaduguru - 16239074 Jnana

Gayathri Penumetcha - 16241948

1.

Assumptions

1.1 Designing Data:

- **Request Table:** We store the data of request in the list.

Zip Code	Type	Count of Vehicles.
64110	1	2
64110	2	3

- **Emergency Vehicle**: We store the data of emergency vehicle in the form of hash map.
  Key – Vehicle ID
  Value – Zip Code, Type, availability.

Vehicle Id	Zip Code	Type	Availability
1F	64110	1	2

i.Vehicle Id – Unique ID for each vehicle.

ii. Zip Code – Location zip code. iii. Type –

1	Fire
2	Ambulance
3	Police

iv. Availability-

1	Available
0	Not Available

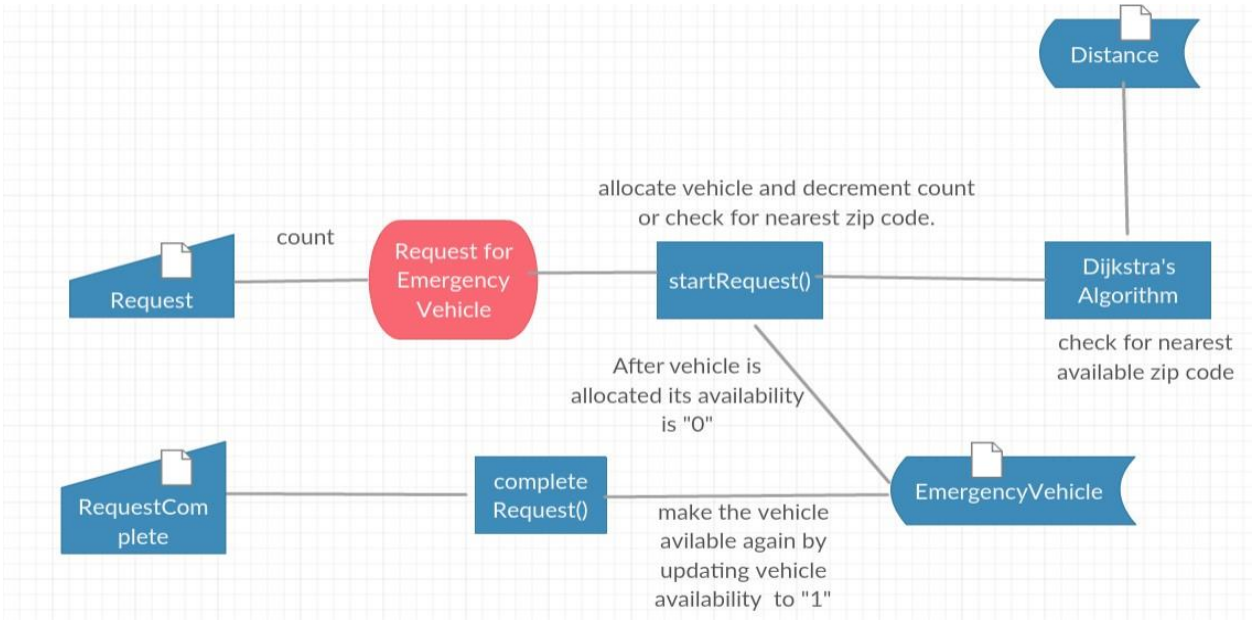
- **Distance**: This table consists of distance between two Zip Code.
- **Request Complete**: When the allocated vehicle is available then we add to this data.

1.2 Two threads:

- To handle multiple requests at a time we are using creating the threads.
- StartRequest () – Count the available vehicle in the requested zip code or else we will use Dijkstra algorithm to find the nearest available vehicle.
- CompleteRequest () – When the request is complete, and the vehicle is available again we add the vehicle ID to the Request Complete data and automatically make the availability of vehicle in Emergency Vehicle as “1”.

2.

Idea of implementation



Input: Emergency Vehicle type and count of vehicles available in the zip code.
Output: Allocated emergency vehicle by calculating the nearest available vehicles.
Algorithm:

Step 1: Request for emergency vehicle is stored in "RequestTable.txt".

Step 2: We are using hash map to store the Emergency Vehicle data and list for request data.

Step 3: startRequest () is used to allocate the emergency vehicle. Count is the number of emergency vehicle requested.

Step 4: We will check if the required emergency vehicle type i.e., fire or police or ambulance is available in the requested zip code. If available, we will allocate the available emergency vehicle and decrement count of required vehicle.

Step 5: If the vehicle is not available in the requested zip code then we should find the nearest available vehicle. To find the shortest path we will use "dijkstra's algorithm".

Step 6: Dijiktra's algorithm will compute the distance between the two zip codes.

Step 7: We will use the quick sort to compute the nearest available zipcode. After finding the nearest zip code we will check if the vehicle is available for allocation else we will compute the next nearest available zip code till all requested count of emergency vehicle are allocated.

Step 8: If count=0 then return the requested zip code and make the availability of emergency vehicle to "0".

Step 9: When the emergency vehicle is available again then the vehicle id will be added into "RequestComplete.txt".

Step 10: completeRequest () is used to update the completed request vehicle ID to "1".

3.

Time Complexity

Let
V – No. of zip codes or vertices
E – No. of Edges

- We are mainly using the Dijkstra's algorithm to find the shortest path and implementing the quick sort on the extracted shortest path.
- O (E log V) is the time complexity with the binary heap.
- O (V log V) is the time complexity of quick sort for best and average case.
- O (V^2) is the time complexity of quick sort for worst case.
- When the requested emergency vehicle is available in the requested zip code then we can allocate the vehicle directly. So, the best case time complexity is O(1).
- For the average case where we cannot find the vehicles in the requested zip code then we need to perform the shortest path and quick sort where time complexity is $O(E*V \log^2 V)$.
- For the worst case considering the worst case of quick sort then time complexity is $O (E*V^2 \log V)$
-

Case	Time Complexity
Best case	O (1)
Average Case	$O (E*V \log^2 V)$
Worst Case	$O (E*V^2 \log V)$

4. References

1. http://www.geeksforgeeks.org/greedy-algorithms-set-7-dijkstras-algorithm-for-adjacency-listrepresentation/
2. http://www.geeksforgeeks.org/quick-sort/
3. Course Content