Project Report

PROBLEM STATEMENT 3

Team Members :

Ankit Sahoo - 18MI10008

Job Steven James Nandrekar - 18MI3EP02

Gyan Prakash Konhar -18NA10010

Instructor:

Prof. Arijit Mondal,

Prof. P P Chakraborty

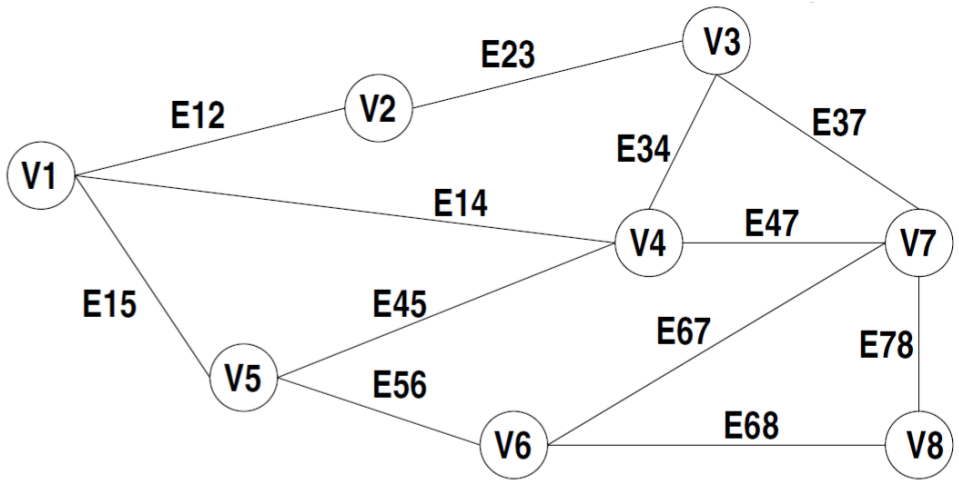
Center for AI, IIT Kharagpur

Problem Statement -- **Electric Vehicles**

"Consider a city network where we need to route a set of electric vehicles which may require to be charged during its journey from some source to some destination. Let us assume that we have n cities (v1; v2; : : : ; vn) and the distance between cities vi and vj be eij (if two cities are not connected directly, then eij = 1and eij = eji): Assume that each city has a single charging station that can charge one EV at a time. Consider a set of k EVs namely P1; P2; : : : ; Pk: For each EV, the following information is provided -

1) Sr - source node
2) Dr - destination node
3) Br - battery charge status initially
4) cr - charging rate for battery at a charging station (energy per unit time)
5) dr - discharging rate of battery while travelling (distance travel per unit charge)
6) Mr - maximum battery capacity
7) sr - average travelling speed (distance per unit time).

Assume that all vehicles start their journey at t = 0 and Pr reaches its destination at t = Tr. We need to route all the vehicles from their respective sources to destinations such that max {Tr} is minimised. You need to develop both optimal as well as heuristic algorithms."



Algorithmic Approach.

The Electric Vehicle Problem is essentially a Multi-Origin-Multi-Destination (MOMD) Constrained Problem.

A simpler version of this problem can be solved by Dijkstra's algorithm. It is an algorithm for finding the shortest paths between nodes in a graph.

Let's consider the cities in the given problem statement to be nodes of a graph and the distances between two nodes as edges of the graph. We can solve this problem by modifying Dijkstra's algorithm in a particular manner.

We have built the following approach in order to solve the problem:

1. Each vehicle P_i has a single optimal from source S_r to destination D_r. The path of the shortest distance (optimal path) will yield the path of shortest time.

2. A modified heuristic of Dijkstra's algorithm is applied to compute the optimum path and its optimum time:

    a. A vehicle would move from one node to the other, every time applying a local heuristic, to proceed from the start node to its end node.
    b. At every current-node, the vehicle will have to make a decision as to which will be the next-node.
    c. The next node will be computed as the closest-node that is at the shortest distance (and consequently, the shortest time) away from the current node.

    d. Once the closest-node is computed, we perform a *check*, which will consist of the following steps:

        i. The current battery capacity of the vehicle is checked to see if it is sufficient to transverse the edge to the closest-node.
            1. *Case I: The battery capacity is sufficient*:
            In this case, the vehicle transverses to the next-closest-node and hence the charging-time at the current node will be zero.

2. *Case II: The battery capacity is insufficient*:
   In this case, the vehicle will have to charge at that particular node for a certain amount of time (charging-time). *The amount of time the vehicle spends at that node charging will depend on the amount of extra charge it requires to* ***just reach*** *the next node.* It is imperative to note here that, the battery is not charged *any more than it is required* to, in order to reach the next-node
   Dong this will ensure that the total time of travel when computed, will be optimum and *no time is wasted in unnecessarily charging* the vehicle.

e. The same check is performed at every node in Dijkstra's optimal path until the vehicle reaches the destination node (end-node). For each point in the path, the time of travelling a particular edge is known:

$$Time\ of\ traveling\ (T_t) = \frac{Average\ speed\ of\ vehicle\ (sr)}{length\ of\ edge\ to\ next-node\ (E_{ij})}$$

Also, if the vehicle has to undergo charging at the current node (Case II), the time of charging can be computed as:

$$Time\ of\ charging\ (T_c) = Extra\ Battery\ charge\ required \times Charging\ Rate\ (cr)$$

Extra Battery required will be only *as much as it is required sufficient enough* to reach the next node. Hence,

$$Extra\ Battery\ Charge\ req. = \frac{Edge\ Length\ (E_{ij}) \times Discharge\ Rate(dr)}{average\ speed\ of\ vehicle\ (sr)} - Current\ Battery\ Charge$$

f. Hence, we then can compute the total time of travel T_R as:

$$Total\ Travelling\ time = \sum_{start\ node}^{end\ node} \left(T_c + T_t\right)$$

3. This time of travel computed for each vehicle and we have initially disregarded the motion of other vehicles in the graph.

4. At the end of the process, we are left with:
    a. A time-optimal path for each vehicle
    b. The amount of time taken by the vehicle charging at each node and travelling each edge in the path. This information is stored as global time-stamps for entry and exit at each node.

5. Now the fact that multiple vehicles can reach the same node in a time span is to be considered, as only one vehicle can be charged at a node at a time, which means that the other vehicle will have to wait for the previous vehicles to get charged. In order to determine the waiting time of each of these subsequent vehicles the following procedure is followed:

    > *"Every node is a queue. And this system can be considered to be a graph of multiple-queues.*"

    a. We start by checking the positions of each vehicle every unit time.
    b. If at any point in time it is found that multiple vehicles are present at a particular node the following rule is to be followed:

        i. The vehicle to enter the node first (primary vehicle) (according to the global time stamps of each vehicle) will charge first according to its charging time (T_c).
        ii. An additional waiting time (T_w) is to be added to every next time-stamp entry of the next vehicle that enters the node (secondary vehicle) while the node is engaged.
        iii. Any other vehicle entering the node subsequently(tertiary, quaternary, etc etc) during the charging period of its preceding vehicle will have to incorporate a waiting time of its own according to the leaving time of the preceding vehicle.
        iv. The time of waiting (T_w) for the nth vehicle entering a node during the charging time of the (n-1)th vehicle at that node can be given by:

$$T_w = Time\ of\ entry\ for\ the\ nth\ vehicle\ -\ Time\ of\ departure\ of\ the\ (n-1)th\ vehicle$$

v. It must be noted that this value of T_w will have to be added to every subsequent time-stamp values of a vehicle in order to refresh its next node entry and exit values.

6. This update will be done for each vehicle meeting another vehicle at a node across time till the last node of the longest travelling time vehicle is reached. Hence, finally, the total travelling time of each vehicle will have the form of:

$$Total\ Travelling\ time = \sum_{start\ node}^{end\ node} \left(T_c + T_t + T_w\right)$$

We can now say that each of the T_w for each vehicle is optimal and hence the maximum T_w of these vehicles is also minimized.

NOTE ON ASSUMPTION:
We have ignored the possibility of *re-routing*. One case might be that a vehicle, knowing that the node it is approaching is already engaged, can reroute to another less-optimal path considering the distance, but might give a more optimal time overall because of the combination of vehicles.