D_4_180014_180022

Abhay 180014 Abhishek Mittal 180022

21 October 2020

1 Problem Statement

Your friend X has recently purchased a **THUNDERBIRD** motorcycle from Royal Enfield in India. He is very excited and wants to drive it from some town s to another town d in Thar desert (think of Sahara desert if you are more adventurous). He is provided with the complete road map of the desert - the roads, their lengths and junctions (where two or more roads meet). The mobike has a very natural limitation - it can drive for c kilometers with full fuel tank. Since the destination is very far, X must plan his route so that he can refill the fuel tank along the way. Note that the shortest route may not necessarily be the feasible route. You friend is bit confused and scared - what if he gets lost in Thar desert due to improper planning.

Your friend is very proud of you since you are a student of IITK. He also knows that you have done a course on algorithms. He approaches you with full faith that you will help him find the shortest feasible route from s to d if it exists. Model the problem in terms of a weighted, undirected graph in which roads are edges, junctions are vertices, and some junctions have fuel stations, and design an efficient algorithm to find shortest feasible route from s to d and inform if no route is feasible. Assume that both s and d are also at junctions and the source s has a fuel station. Your algorithm should take time of the order of O(mn log n) where m is the number of edges and n is the number of vertices. You should not exploit the planarity of the input graph while designing your algorithm. In other words, your algorithm should work for any arbitrary undirected graph with positive edge weights and under the fuel constraint mentioned above. Also note that the shortest feasible route need not be a simple path. For example, consider the following situation : There are five towns : s, a, b, e, d where fuel stations are available at s, b, e only. The connecting roads are : (s, a) of length 200 km, (s, d) of length 300 km, (a, b) of length 40 km, (a, d) of length 150 km, (b, e) of length 200 km, (e, d) of length 200 km; and motor bike can travel 250 km with full fuel tank. For this road network, shortest feasible route from s to d is is s $\rightarrow$ a $\rightarrow$ b $\rightarrow$ a $\rightarrow$ d, and has length 430 km.

2 Solution

2.1 Algorithm

We assume that there are k fuel stations in the given graph G. Also, let us denote the vertex set of graph G by V.

Procedure : ShortestFeasiblePath(G, s, d, c)

---

1. Calculate and store shortest distances and thier respective paths from each fuel station to every other fuel station and to the destination vertex by running dijkstra's algorithm from each fuel station in graph G.
2. Construct a graph G' defined as :

   $$V' = \{F_1, F_2, F_3, ..., F_k\} \cup \{d\}$$

   where F_i denotes a fuel station

   $$\text{weight of } E'(u,v) = \begin{cases} \infty, & \text{if } \delta(u,v) > c \\ \delta(u,v), & \text{otherwise} \end{cases} \tag{1}$$

   where $\delta(u,v)$ denotes shortest distance(calculated in step 1) between vertices u, v $\in V'$.
3. Calculate shortest distance path from source vertex s in graph G' to the destination vertex d with dijkstra's algorithm.
4. If the shortest distance calculated is infinity, return NULL.
5. Else suppose the shortest path returned from Djikstra's is given by $s \rightarrow F_{k1} \rightarrow F_{k2} \rightarrow F_{k3} \ldots \rightarrow d$. Now for each edge in this path we expand the path stored in step1. After expanding all edges, the final route is returned.

2.2 Proof of Correctness

NOTE : Here, the "$-$" used in description of paths below, denotes that the vertices connected by this are reachable from each other and come in the path in the given order. $(s - x)$ denotes x comes after s (not necessarily immediately) in the path and other vertices might lie in between s and x in the path. Also, "$leftarrow$" edge connecting two vertices.

Lemma 1 : In the shortest feasible route from s to d, no fuel station shall be visited more than once.

Proof Suppose, in the shortest feasible route from s to d, a fuel station(say x) was visited twice (i.e the path looks like $s - x - p - x - d$) then, we could omit the path p in between x and x and directly move from s to x to d. Since, x is a fuel station and fuel shall be refilled upon first reaching x due to which, we don't need to come back to x again. Now, we need to show our algorithm and construction of graph G' takes care of this property. According to the construction of new graph G', we have all fuel stations in the graph as vertices along with the destination vertex. Now, in order to return the shortest feasible route, we are running a dijkstra's algorithm over graph G', which gives us a shortest "path" composed of fuel stations and destination vertex at its end. Hence, before expansion of paths(step 5) there won't be any repeated fuel station. Assume the (unexpanded)path(say P) to be ($s - F_{k_i} - d$) where F_{k_i} is any fuel station. Then, after expansion, suppose that there is another occurrence of F_{k_i} in the route. Suppose the other occurence of F_{k_i} occurs due to expansion of edge F_{k_p} and $F_{k_{p+1}}$. Consider 2 cases:

1. if p < i: Thus our unexpanded path(P) returned from step 3 looks like $(s - F_{k_p} \to F_{k_{p+1}} - F_{k_i} - d)$. Since the shortest path from F_{k_p} to $F_{k_{p+1}}$ goes through F_{k_i}, therefore edge between F_{k_p} to F_{k_i} must exist in G' and $E'(F_{kp}, F_{ki})$ in G' must be less than $E'(F_{k_p}, F_{k_{p+1}})$. Now consider a path(say Q) in G' as $(s - F_{k_p} \to F_{k_i} - d)$, where $s - F_{k_p}$ and $F_{k_i} - d$ are same as in P. Since $E'(F_{k_p}, F_{k_i}) < E'(F_{k_p}, F_{k_{p+1}})$, thus distance calculated in path Q is less than the distance calculated from path P which contradicts the property of Djikstra's algorithm since it must return the shortest possible path.

2. if p > i : For this case, the unexpanded path(P2) returned from step 3 looks like $(s - F_{k_i} - F_{k_p} \to F_{k_{p+1}} - d)$. Since the shortest path from F_{k_p} to $F_{k_{p+1}}$ goes through F_{k_i}, therefore edge between (F_{k_i} and $F_{k_{p+1}}$) must exist in G' and $E'(F_{k_i}, F_{k_{p+1}})$ must be less than $E'(F_{k_p}, F_{k_{p+1}})$. Now consider a path(say Q2) in G' as $(s - F_{k_i} \to F_{k_{p+1}} - d)$, where $s - F_{k_i}$ and $F_{k_{p+1}} - d$ are same as in P2. Since $E'(F_{k_i}, F_{k_{p+1}}) < E'(F_{k_p}, F_{k_{p+1}})$, thus distance calculated in path Q2 is less than the distance calculated from path P2 which contradicts the property of Dijkstra's algorithm since it must return the shortest possible path.

Both the above cases prove that there must be atmost one occurence of any fuel station in the route.

Lemma 2 : Consider a feasible route with vertices as $s - F_1 - F_2 - ... - F_k - d$ where F_i denotes a fuel station and F_i and F_{i+1} are consecutive fuel stations. Then, there exists a feasible route of length less than equal to length of given route with shortest distance between F_1, F_2 in G.

Proof : Considering the given route in lemma, let the distance between two consecutive fuel stations(F_i, F_{i+1}) in the feasible route be x. Now, as we have considered two consecutive fuel stations, there isn't any other fuel station in between them. Hence, we have enough fuel at F_i to reach F_{i+1} or the distance x is less than equal to c. Now, if we replace the path between F_i to F_{i+1} with the shortest path between them, then also the distance between these fuel stations must be less than c since it is shortest path. Hence, lemma is proved.

Now, with the lemma 2, we can say that in the shortest feasible route between s and d, then between each consecutive fuel station, we shall have the shortest possible path between them. Thus, the final shortest feasible route must be a sequence of fuel stations ended with destination vertex where between each fuel station there is shortest path of vertices used. Now, from lemma 1 since any fuelling station can't be repeated in the route and by lemma 2, we can say that by using dijkstra's algorithm over constructed graph G' returns the shortest possible distance of a feasible route.

2.3 Time Complexity :

We analyse the time complexity of the above algorithm step by step:

1. For step 1, we run Dijkstra's for every fueling station which are O(n). For each station Dijkstra's take $O(m \log n)$ time using Binary Heap, thus step 1 takes $O(mn \log n)$.
2. Step 2 is construction of graph G' which contains atmost O(n^2) edges and O(n) vertices. Thus it takes $O(n^2)$ time.
3. Step 3 is essentially a run of Dijkstra's algorithm. Hence it is done in $O(m \log n)$ time.
4. Step 4 is essentially an O(1) check.
5. In Step 5, since we have stored the shortest path, expanding each edge in path return from Dijkstra's take O(n) time. Now since it is a path,we have to expand O(n) edges. Therefore total time taken for step 2 is O(n^2)

Since the highest time taking step is step1. Therefore the overall time complexity for the above algorithm is $O((m + n)n \log n)$ but if m ¿ n then the time is essentially $O(mn \log n)$.