CS301 Assignment 4

Dilara Nur Tekinoğlu

September 28, 2021

1 Computational Problem

Given an undirected graph $G = (V, E)$, a source node s, a set of nodes C where $C \subseteq V$ and for every node c in C a set of nodes S_c where $S_c \subseteq V$; find the shortest path I_c from s to every node c in C which includes all nodes in S_c exactly once.

In other words: Input: $G = (V, E)$, a source node s, a set of nodes C where $C \subseteq V$ and for every node c in C a set of nodes S_c where $S_c \subseteq V$ Output: Shortest itinerary I_c from s to every node c in C which includes all nodes in S_c exactly once.

2 NP-Completeness of the Problem

First of all, this problem is an optimization problem. As it is stated in our textbook "Introduction to Algorithms", NP-completeness applies directly not to optimization problems but to decision problems. Therefore, we need to cast the given optimization problem as a related decision problem by imposing a bound value to be optimized.

Related decision problem for our assignment can be as follows: Given the same inputs and an integer k, does a path exist from s to every c in C of length of at most k?

This decision problem is in a sense "easier," or at least "no harder" than our original problem. We can solve this modified problem by solving the original problem and then comparing the number of edges in the shortest path found to the value of the decision-problem parameter k.

In other words, if our original problem is easy, its related decision problem is easy as well. Now, if we can provide evidence that a decision problem is hard, we also provide evidence that its related optimization problem is hard. **Therefore, we will prove the NP-completeness of the modified decision problem in the following section. Let us call this modified problem TRAVEL-AND-VISIT.**

By definition, a problem is NP-complete if it is in NP and as hard as any other problem in NP. So, there are two conditions we need to analyze: Membership and Hardness.

2.1 Membership

NP is the class of decision problems that are verifiable in polynomial time. A decision problem can be verified in polynomial time if for every "yes" instance of the problem there is a proof (a witness) that can be checked in polynomial time.

Given shortest paths from s to each c in C, S_c and integer k for each shortest path, we can just check in polynomial time that (1) it is a path from s to c, (2) it does contain all vertices in S_c exactly once, i.e. the nodes that must be visited, and (3) the total length is at most k.

For the first step, we will just follow the given path and check whether it starts at s and ends at c, and whether there are edges between intermediate vertices. It takes polynomial time.

For the second step, we will check whether the given path contains all the vertices in S_c exactly once, this can be thought as a simple search problem which takes polynomial time.

The last step takes again polynomial time since we will only add edges and compare the total with k.

Therefore, our problem TRAVEL-AND-VISIT is in NP.

2.2 NP-Hardness

In order to prove NP-Hardness, we will apply reduction. If a problem that we know to be NP-hard can be reduced to TRAVEL-AND-VISIT problem in polynomial time, then TRAVEL-AND-VISIT is NP-Hard as well.

According to our textbook:

- Given an instance α of problem A, use a polynomial-time reduction algorithm to transform it to an instance β of problem B.
- Run the polynomial-time decision algorithm for B on the instance β.
- Use the answer for β as the answer for α.

We can reduce Hamiltonian Cycle problem to TRAVEL-AND-VISIT in polynomial time as follows:

- Complete G by adding edges between all pairs of vertices that were not connected in G. New graph G2 = (V2,E2) where V2 = V and E2 = {(u,v)} for any u,v in V2.
- For edges in G2 that were also present in G, assign a weight 0. For new edges, assign weight 1.
- Select any vertex s in graph G2 as city Istanbul because if Graph G2 = (V2,E2) has Hamiltonian cycle then any vertex can be the source vertex and hence any vertex can be set as city Istanbul.
- Set C ={s}. For all the remaining vertices in set V2-{s}, put them into set S_v. So, we will find the path from Istanbul to Istanbul which visits all other cities exactly once.

The graph G has a Hamiltonian Cycle if and only if there exists a cycle in G2 passing through all vertices exactly once and that has length ≤ 0. (Here instance α of TRAVEL-AND-VISIT problem where $k \leq 0$, C contains only vertex s and S_v contains all the remaining vertices. I.e. A path from

Istanbul to Istanbul that visits every other city exactly once and weights less than k. This particular instance of TRAVEL-AND-VISIT is transformed to an instance of Hamiltonian cycle problem.) If there is such a cycle of length ≤ 0 in G2, then this cycle contains only edges that were present in original graph G. So, there is a Hamiltonian cycle in G. Also if graph G does not have Hamiltonian path, then there is no such Hamiltonian cycle in G2, then there is no path $\leq$ k in G.

So, we reduced Hamiltonian Cycle Problem to TRAVEL-AND-VISIT. Since we know that Hamiltonian Cycle is NP-Hard, TRAVEL-AND-VISIT is NP-hard too. Since it is in NP and in NP-hard, TRAVEL-AND-VISIT problem is NP-complete. But it is not the original problem which was an optimization problem. Instead, it is the related decision problem of the original one. As it is stated above, if related decision problem is hard, it shows that the optimization problem is also hard. Therefore, we can conclude that the original problem is NP-complete.