CS301-A4

Görkem Yar (Student)

June 2022

1 Traveler Problem

1.1 Problem Definition

A shortest path problem with a given cities which are required to be a part of the path. From now on the problem will be named as Traveler Problem.

- C represents every vertex/city in the graph.
- s represents the start vertex of the path.
- I_c represents shortest path from s to every city c in C.
- S_c represents the cities that are required to be a part of I_c.

The output of the problem should be the shortest itinerary from s to every c in C which includes desired S_c exactly once.

1.2 Definition of the Decision Problem

The initial problem is an optimization problem; therefore, NP Completeness can not directly address it. To examine the problem in terms of NP Completeness, the problem need to be converted into a decision problem. In order to make a decision problem, an upper bound k will be enforced on the problem.

New Decision Problem: With the same inputs and with an upper bound k, Is there a path exist with from s to every c in C consisting S_c using at most k length?

The new problem is no harder than the initial problem. Different from the initial problem when this problem is solved the length of the path will be compared to k (which is decision parameter). Since the new problem is no harder than the initial problem, if the initial problem is an easy problem then the related decision problem is easy as well. Also, if the related decision problem is hard then this will imply that initial problem is hard as well. From now on, NP membership and completeness of the related decision problem will be proven.

1.3 NP Membership

NP is a complexity class used to classify decision problems. To be a member of NP class a problem either verifiable by an deterministic Turing machine in polynomial time or it should be solvable by non-deterministic Turing machine in polynomial time. For the decision problem that it is created from the Traveler problem, NP membership will be tested by verifiability in deterministic Turing machine.

Assume that there is a shortest itinerary from s to c (c is the each member of C) and there are some intermediate cities (vertexes) between s and c in each path. To verify a solution to a problem there are 3 steps:

- Check the path whether it starts at s and finishes at c
- Check intermediate vertexes, check whether they consist the vertexes S_c.
- Check how much length is covered whether it is bigger or smaller than decision parameter k.

Checking start vertex and finish vertex can be doable in polynomial time (just single if else statement). Checking intermediate vertexes is just a search problem and doable in polynomial time. Lastly, comparison with k parameter doable in polynomial time as well (just a single if else checking).

As a result of these, decision problem related to the Traveler Problem is a member of NP class.

1.4 NP Hardness

To prove a problem is hard, one need to show that the problem can be obtained by a reduction from any NP hard problem. For this particular proof, Hamiltonian Cycle will be used, which is known to be NP complete.

The reduction steps for NP hardness generally as follows:

- Let's use the notation $P- > Q$ to denote that P is polynomial time reducible to Q.
- If $P- > Q$, then in order to solve an instance p of P, we can transform p into an instance q of Q (in polynomial time).
- Solve the instance q which would give us the corresponding solution for the instance p.

To reduce Hamiltonian Cycle to Traveler problem in polynomial time the following steps are required:

- First step is creating a new complete Graph with the same amount of vertexes. Call this new graph W=(V2, E2) where V2 = V
- If $P- > Q$, for edges that both present in G and W set weight of the edge in W as 1. For the other edges that is present in W but not in G, set weight as any number that is bigger than 1 for instance 2.
- Choose any vertex as s in W as start vertex (as Istanbul). If W has Hamiltonian cycle, every vertex can be taken as source (cycle from everywhere).
- Set destination to s (start vertex). For the remaining vertexes, put them into the S_c (the vertexes that are required to be visited). So, problem is now reduced as, from s to s trying to visit all cities only one time.

We will take decision parameter k as number of vertexes V2. As a result, G has a Hamiltonian cycle if and only if there is a cycle in W, which passes each vertex only once and length of the cycle is the number of vertexes (since all valid edges have 1 length). This is an example of the Traveler decision-problem in which k is vertex count, start vertex and end vertex is same, and S_c is every vertex in the W. It can be translated as a path from Istanbul to Istanbul which contains every city and has a weight $\leq$ k = City Count. This particular example of Traveler Problem is converted into Hamiltonian cycle.

If there is a cycle that has length $\leq$ Vertex count in W, this cycle consist of the edges that are already available in the G. So, graph G has Hamiltonian cycle. Furthermore, if G does not have Hamiltonian cycle, then W do not have a Hamiltonian cycle that have length $\leq$ Vertex count.

As it can be seen, Hamiltonian Cycle is reduced to Traveler problem. Hamiltonian cycle is a NP-hard problem therefore Traveler Problem is NP-hard as well. From the previous part, we stated that it is a member of the NP class. Since it is both NP and NP-hard the decision problem that it is corresponds to the Traveler Problem is NP-complete. As it mentioned before, the decision problem that is related to Traveler problem is no harder than the Traveler problem which is an optimization problem. This implies that original Traveler problem is NP-Complete as well.

2 References

1-Lecture Slides