# Assignment 11

Nishan Dhoj Karki

11/27/2022

## 1 Consider the following language: LCS = $\{\langle G_1, G_2, k\rangle \mid G_1$ and $G_2$ are graphs that have isomorphic subgraphs with k edges each $\}$

- **(20 points) Prove that LCS $\in$ NP.**
- **(30 points) Prove that LCS is NP-hard.**

### 1.1 Proof: LCS $\in$ NP..

Constructing a poly-time verifier $V$ for LCS:
V= "On input $\langle\langle G_1, G_2, k\rangle, c\rangle$ here c is a certificate with graphs and mapping function between the vertices of the graphs:

1. Check if the certificate $c$ comprises a description of two sub-graphs and a mapping function.
2. Check if graphs $c_1$ and $c_2$ in $c$ have k edges and are subgraphs of $G_1$ and $G_2$ respectively.
3. For two graphs $c_1(V_1, E_1)$ and $c_2(V_2, E_2)$ (here V and E are vertices and edges):

   - Check if the mapping is a bijection i.e. Using mapping function ($f$) from $c$, Check if $f\colon V_1 \to V_2$ such that $(u, v) \in E_1 \iff (f((u), f(v)) \in E_2$
4. If all the above checks are passed, $ACCEPT$. Else, $REJECT$ "

Here, we have constructed a verifier $V$ that accepts if the certificate $c$ has two sub-graphs with $k$ edges and a mapping function for an isomorphic test. If the graphs $c_1$ and $c_2$ satisfy all the conditions mentioned above, then the verifier $V$ accepts or rejects the certificate. The verification process for the graph to have k edges as well as the check for isomorphism can be done in polynomial time. The check for bijection as well as verifying that for every edge in $c_1$ there is an edge in $c_2$ mapped by the mapping function can be done in polynomial time. Thus, the verifier $V$ verifies $LCS$ in polynomial time. So, $LCS \in NP$

### 1.2 LCS is NP-hard.

Let us show a reduction from CLIQUE to LCS.
CLIQUE = $\{\langle G, k\rangle \mid$ graph G has a k-clique $\}$
Let F be the reduction function:
F ="On input $\langle G, n\rangle$ where G is a graph with n vertices.

1. Construct $G_1$ as complete graph of $n$ vertices.

2. Let k be the total edges in graph $G_1$ i.e. k= $(n(n-1)/2)$.

3. Output $\langle G, G_1, k\rangle$"

When the graph $G$ has a $n$ clique the $n$ nodes are isomorphic to $G_1$, this implies that whenever $\langle G, n\rangle \in CLIQUE$ then $\langle G, G_1, k\rangle \in LCS$. Similarly, when $G$ contains a sub graph isomorphic to $G_1$ then, $G_1$ is $n$ clique, so $G$ must have $n$ clique. So, $\langle G, G_1, k\rangle \in LCS \Rightarrow \langle G, n\rangle \in CLIQUE$.

This reduction takes place in polynomial time since the construction of the complete graph $G_1$ takes $O(n^2)$ time. So, we can say that $CLIQUE \leq_P LCS$. So, LCS is NP-hard.

# 2 Consider the following language: IST = $\{\langle G, T\rangle \mid G$ is a graph with a spanning tree isomorphic to tree T $\}$ • (20 points) Prove that IST $\in$ NP. • (30 points) Prove that IST is NP-hard.

## 2.1 Proof: IST $\in$ NP.

Constructing a poly-time verifier $V$ for $IST$.
V = " On input $\langle\langle G, T\rangle, c\rangle$, where $c$ is mapping function $f : V_T \rightarrow V_G$ which maps vertices of Tree $T$ to graph $G$

1. Check if the mapping function given in $c$ is bijective.

2. Check if tree $T$ is connected.

3. Construct $G_1$ using $c$ and $T$ such that the mapping function $f$ is validated i.e. $f : V_T \rightarrow V_G$

4. Check if $G_1$ is a subgraph of $G$ and covers all the vertices of $G$.

5. If all test passes, $ACCEPT$. Else $REJECT$ "

Here, We are trying to generate a spanning tree using the mapping function $f$ provided by $c$ and the tree $T$. If all the checks are passed the new graph $G_1$ generated will be a spanning tree isomorphic to tree $T$ and will cover all vertices in $G$ as the function $f$ provided by $c$ is a mapping function such that $f : V_T \rightarrow V_G$. Since we are generating a new graph using the mapping function $f$ this can be processed in polynomial time. Also, the check for subgraph and coverage for all vertices can be done in polynomial time. Hence, by the construction of poly-time verifier $V$, IST $\in$ NP.

## 2.2 Proof: IST is NP-hard.

Let's show a reduction from HAMPATH to IST. Let F be the reduction function:
F = "On input $\langle G, s, t\rangle$ where Graph G has a Hamilton path from s to t:

1. Select s as marked
2. Mark all adjacent vertex till no vertex is unmarked.
3. Store all paths that end with no further unmarked vertices.
4. From all stored paths select paths that start at vertex $s$ and end at vertex $t$.
5. For all selected paths, select a path with the total number of vertices equal to the total number of vertices in $G$
6. Construct a tree $T$ from the selected path
7. Output $\langle G, T \rangle$"

Here, we transverse through the graph searching for Hamilton paths that start at $s$ and end at $t$. The constructed tree $T$ will start from $s$ and end at $t$ covering all the vertices, so there will at least be one spanning tree isomorphic to $T$. If $\langle G, s, t \rangle \notin HAMPATH$ then it does not have a spanning tree isomorphic to T. Thus, $\langle G, s, t \rangle \in HAMPATH \iff \langle G, T \rangle \in IST$. The above reduction occurs in polynomial time as the path search, selection of a valid path and construction of tree $T$ are done in polynomial time. Hence, HAMPATH $\leq_p$ IST, and IST is NP-hard.