The subgraph-isomorphism problem takes two undirected graphs $G_1$ and $G_2$, and it asks whether $G_1$ is isomorphic to a subgraph of $G_2$. Show that the subgraph-isomorphism problem is NP-complete.

Solution:

To see that the problem is in NP, we observe that a certificate is a mapping $\phi$ from the nodes of $G_1$ to (a subset of) the nodes of $G_2$, describing which vertices of $G_2$ correspond to vertices of $G_1$. The certifier then needs to make sure that for each edge $e = (u,v)$ in $G_1$, the edge $(\phi(u), \phi(v))$ is also in $G_2$, and whenever $(u,v)$ is not an edge of $G_1$, then $(\phi(u), \phi(v))$ is not an edge of $G_2$. This can be done with two simple nested loops, and takes at most $O(n^2)$ time, i.e., polynomial.

To prove NP-hardness, we show that, for instance, the Clique problem is a special case. Given an instance of Clique, consisting of a graph $G$ and a number $k$, we generate an instance of Subgraph Isomorphism by setting $G_2 = G$, and $G_1$ a clique on $k$ vertices. This reduction clearly takes polynomial time, since all we do is copy a graph, and write down a complete graph in time $O(k^2)$.

Conversely, if $G_2$ contains a subgraph isomorphic to $G_1$, because $G_1$ is a $k$-clique, $G_2$ must contain a $k$-clique. Thus, $(G, k)$ is a “Yes” instance.

34-1

An Independent set of a graph $G=(V,E)$ is a subset $V'$ of vertices such that each edge in $E$ is incident on at most one vertex in $V' \subseteq V$. The independent-set problem is to find a maximum-size independent set in $G$.

(a) Formulate a related decision problem for the independent-set problem, and prove that it is NP-complete. (Hint: Reduce from the clique problem.)

(b) Suppose that you are given a black-box subroutine to solve the decision problem you defined in part (a). Given an algorithm to find an independent set of maximum size. The running time of your algorithm should be polynomial in $|V|$ and $|E|$, counting queries to the black box as a single step.

Solution:

(a) Decision problem:

Input: Given an undirected graph $G$ with $V$ vertices, $E$ edges and a lower bound on an integer $N$.

Question: Does a Graph $G$ contain an integer $N$ such that there exits at least a set $P$ of non-adjacent or independent set of vertices of cardinality of max size $N$.

Output: Yes, there exists such an independent set $P$ of vertices of size at most $N$.

To prove Independent Set Problem is NP complete, it is required to show that it is NP as well as NP-hard.

NP: Given an independent set $S$ of an undirected graph $G$ is NP if there exists polynomial time algorithm taking an instance of the problem as a certificate. The algorithm considered will check whether all the vertices of the certificate are available in the graph and also it will check that there should not be any edge
connecting the pair of vertices of the certificate.

The running time complexity of the algorithm is $O(V+E)$ which is simply a polynomial time and thus, it can be stated that an independent $S$ belongs to NP.

NP-Hard: To prove it is NP-hard, an instance of Clique problem is reduced to an instance of an independent set $S$. Consider the instance of Clique problem as $G(V,E)$. This problem is independent set problem having set $G'(V',E')$ where the $E'$ is the complement of $E$. And the set of vertices $V$ represents a clique having size $x$ is in the graph $G$ only in a case if $V$ is an independent set of graph $G'$ having the size of $x$. Construction of $G'$ from $G$ also can be done in polynomial time. It can be concluded that independent set problem is also NP-hard.

(b) Consider the black-box as $B(G,x)$, the algorithm is as follows:

**Step1:** Start binary search on $B(G,x)$ to determine the maximum size $x^*$ for the independent set.

**Step2:** Set the independent set $I$ to be an empty set.

**Step3:** for each $v$ in $V$

Construct $G'(V',E')$ by removal of $v$ and its related edges from the graph $G(V,E)$

if $B(G',x)$ is true

Assign $G'$ to $G$

else

Assign $G''$ to $G$

$G'$ is obtained by removing all vertices which are connected to $v$ and the edges which are linked to it from the graph $G$.

35.1-1

Given an example of a graph for which APPROX-VERTEX-COVER always yields a suboptimal solution.

**Solution:**

A graph with two vertices and an edge between them has the optimal vertex cover consisting one vertex. However APPROX-VERTEX-COVER returns both vertices in this case.