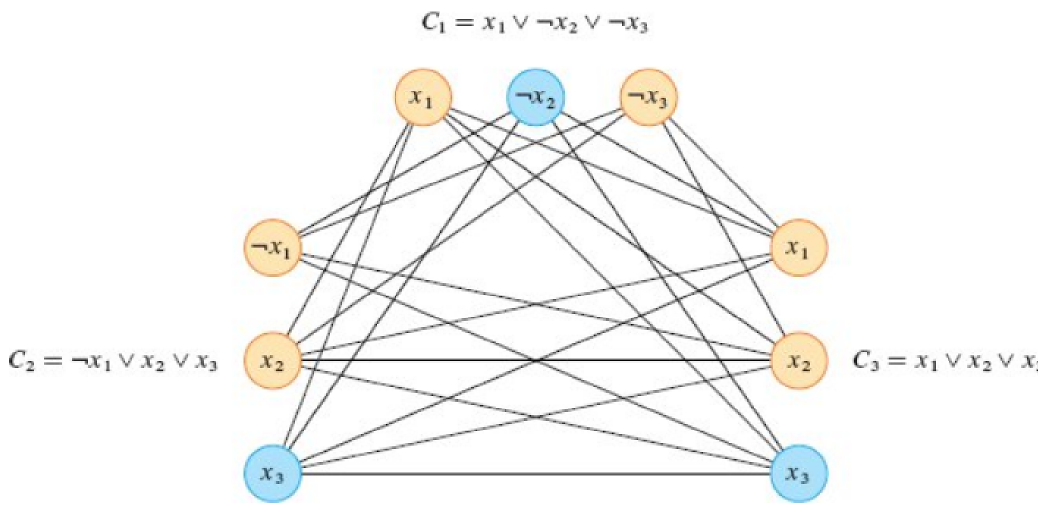
Week 14 Recitation

1. Prove that the CLIQUE problem is NP-Complete by reduction *3-SAT* $\leq_p$ *CLIQUE*.

$$\phi = (x_1 \lor \neg x_2 \lor \neg x_3) \land (\neg x_1 \lor x_2 \lor x_3) \land (x_1 \lor x_2 \lor x_3),$$



NP: We first show that CLIQUE $\in$ NP. For a given graph $G = (V, E)$, use the set $V' \subseteq V$ of vertices in the clique as a certificate for G. To check whether V' is a clique in polynomial time, check whether, for each pair $u, v \in V'$, the edge (u, v) belongs to E.

NP-Hard: We then show that CLIQUE is also in NP-Hard by reduction 3-SAT $\leq_p$ CLIQUE. Assume we have a 3-CNF-SAT formula $\phi = C_1 \land C_2 \land \ldots \land C_k$ with k clauses, then for each $r = 1, 2, \ldots k$, the clause C_r contains exactly three distinct literals: l_r^r, l_2^r, l_3^r. We construct an undirected graph $G = (V, E)$ such that ϕ is satisfiable if and only if G contains a clique of size k as follows. For each clause $C_r = (l_1^r \lor l_2^r \lor l_3^r)$ in ϕ, place a triple of vertices v_1^r, v_2^r, v_3^r into V. Add edge (v_i^r, v_j^s) into E if both of the following are satisfied:

- v_i^r and v_j^s belong to different clauses, that is, $r \neq s$
- their corresponding literals are consistent, that is, l_i^r is not the negation of l_j^s

We can build this graph in polynomial time since there will be exactly $3k$ vertices and at most $9k^2$ edges in the graph.

We now prove that ϕ is satisfiable if and only if G contains a clique of size k:

- First, assume that ϕ has a satisfying assignment, Then each clause C_r contains at least one literal l_i^r that is assigned 1, and each such literal corresponds to a vertex v_i^r. Picking one such "true" literal from each clause yields a set V' of k vertices. We claim that V' is a clique. For any two vertices $v_i^r, v_j^s \in V'$, where $r \neq s$, both corresponding literals, l_i^r, l_j^s map to 1 by the given satisfying assignment, and thus the literals can not be complements. Thus, by the construction of G, the edge (v_i^r, v_j^s) belongs to E.

- Conversely, suppose G contains a clique V' of size k. No edges in G connect vertices in the same triple, and so V' contains exactly one vertex per triple. If $v_i^r \in V'$, then assign 1 to the corresponding literal l_i^r. Since G contains no edges between inconsistent literals, no literal and its complement are both assigned 1. Each clause is satisfied, and so ϕ is satisfied.

Since, CLIQUE is both in NP and NP-Hard, it must also be in NP-Complete.

2. A vertex cover of an undirected graph $G = (V, E)$ is a subset $V' \subseteq V$ such that if $(u, v) \in E$, then $u \in V'$ or $v \in V'$ (or both). A VERTEX-COVER problem determines if there exists a vertex cover of size k in the graph G. Prove that VERTEX-COVER is NP-hard by reducing CLIQUE problem to it.

   Reduction: The reduction algorithm takes as input an instance $\langle G, k\rangle$ of the CLIQUE problem and computes the complement $G' = (V, E')$ in polynomial time, where E' is defined as $E' = \{(u, v) \colon u, v \in V, u \neq v, (u, v) \notin E\}$ (G' contains exactly those edges that are not in G). The output of the reduction algorithm is the instance $\langle G', |V| - k\rangle$ of the VERTEX-COVER problem. To complete the proof, we show that this transformation is indeed a reduction: the graph G contains a clique of size k if and only if G' contains a vertex cover of size $|V| - k$.

   - Suppose that G contains a clique $V' \subseteq V$ with $|V'| = k$. We claim that $V - V'$ is a vertex cover in G'. Let (u, v) be any arbitrary edge in E'. Then, $(u, v) \notin E$, which implies that at least one of u or v does not belong to V', since every pair of vertices in V' must have an edge in between. Equivalently, at least one of u or v belongs to $V - V'$, which means the edge (u, v) is covered by $V - V'$. Since (u, v) was chosen arbitrarily from E', every edge of E' is covered by some vertex in $V - V'$, which forms the vertex cover for G' of size $|V| - k$.
   - Conversely, suppose that G' has a vertex cover $V' \subseteq V$, where $|V'| = |V| - k$. Then for all $u, v \in V$, if $(u, v) \in E'$, then $u \in V'$ or $v \in V'$ or both (by definition of vertex cover). The contrapositive of the same implication is that for all $u, v \in V$, if $u \notin V'$ and $v \notin V'$, then $(u, v) \in E$. In other words, $V - V'$ is a clique and it has size $|V| - |V'| = k$.

3. We define the decision version of a shortest path problem as a problem to determine if there exists a shortest-path of length k from s to t in the given undirected and unweighted graph $G = (V, E)$. Show that the decision shortest path problem can be reduced to a CLIQUE problem in polynomial time.

Reduction: Given an instance of the decision shortest path problem (DSP) $\langle G, s, t, k \rangle$, we can construct the instance $\langle G', k' \rangle$ to the CLIQUE problem in polynomial time as follows. We run BFS to find the shortest paths from the source s to all other vertices $v \in V$. Then we will have two cases:

- Shortest path from s to t has length k. In this case, we construct a complete graph $G' = (V = \{a, b, c\}, E = \{(a, b), (b, c), (a, c)\})$ and assign $k' = 3$ to produce an instance $\langle G', k' \rangle$ for CLIQUE problem.
- Shorets path from s to t does not equal k. Then, we construct any incomplete graph $G' = (V = \{a, b, c\}, E = \emptyset)$ and assign $k' = 3$.

We now show that $DSP(G, s, t, k) = \mathit{yes} \iff CLIQUE(G', k') = \mathit{yes}$:

- Assume the shortest path from s to t equals k. Then, the reduction constructs a complete graph of size 3, that when passed to CLIQUE problem along with $k' = 3$, will produce an affirmative answer.
- If CLIQUE$(G', k') = \mathit{yes}$, then G must have a shortest path of length k from s to t. Assume the opposite, that the shortest-path from s to t is not k. Then, the reduction produces an instance $\langle G', k' \rangle$ without any edges in G' which can not contain a clique of size $k' = 3$. This contradicts our assumption that CLIQUE$(G', k') = \mathit{yes}$.

The construction can be done in polynomial time since BFS takes $O(V + E)$ time to find all shortest paths from s and building G' takes constant time.