HOMEWORK 05 – CS624

DATHRIKA SAICHARITHA

02010404

1. Suppose that, in addition to edge capacities, a flow network has vertex capacities. That is, each vertex v has a limit l(v) on how much flow can pass though v. Show how to transform a flow network G = (V, E) with vertex capacities into an equivalent flow network $G' = (V', E')$ without vertex capacities, such that a maximum flow in G' has the same value as a maximum flow in G. How many vertices and edges does G' have?

   The given flow network G = (V, E) with vertex capacities can be transformed to a flow network without vertex capacities by the following changes:
   - Divide the given vertex v into two separate vertices. Let these vertices be v_i and v_o.
   - If there is an actual edge (u, v) in the given original graph G, then we need to make an edge (u, v_i) in the new graph G'.
   - If there is an actual edge (v, w) in the given original graph G, then we need to make an edge (v_o, w) in the new graph G'.
   - Now we just make an edge between the two vertices (v_i, v_o) and assign this edge capacity to be the vertex capacity of original vertex v.
   - After making these changes, new vertices v_i, v_o do not have any vertex capacity constraints and the original constraint is at the edges of two new vertices.
   - But the network flow is not changed because the overall max flow is not changed as the incoming and outgoing edge capacity is unchanged.
   - The constraint vertex capacity is added by the new edge compared with the original graph, but it is still the same constraint condition as before, so it's the same.
   - So, we can examine the vertex constraints by dividing each vertex into two halves, where the edge between both vertices represents the vertex's capacity.
   - For the edge we need to add |E| to the actual graph G. so that in the new graph $G'$the edges are |V|+|E|.
   - With |V|+|E| edges and 2|V| vertices a new flow network will form.
   - Finally, we can see that this network represents the idea that the vertices have capacities I(V). This is because every flow that passes through the edge V I in the original graph must also pass through the edge 0 x V, 1 x V in the new graph to connect the edges entering V to the edges exiting V.

2. Prove that all problems that are NP-complete are polynomially equivalent, in the sense that if A and B are NP-complete, then $A \leq_P B$ and $B \leq_P A$.
   A decision problem A is NP complete when A is in NP and Every NP problem can be reduced to A in polynomial time.
   Similarly, a decision problem B is NP-complete when B is in NP and every NP problem can be reduced to B in polynomial time.
   So here it is given that all problems are NP-complete that is A and B are NP-complete.

And if A and B are NP-complete then we need to show that there exist polynomial-time reduction algorithms that can transform instances of problem A into instances of problem B and vice versa.

Given A and B are NP-complete we can reduce any problem in Np to either A or B in polynomial time, by creating an instance of A or B that is identical to the original problem.

Consider a polynomial-time reduction algorithm that converts instances of decision problem A into instances of decision problem B.

This indicates that if given an instance of decision problem A, by using the reduction algorithm we can create an instance of problem B which is identical to original instance of decision problem A. because the reduction algorithm takes polynomial time, A is polynomially reducible to B. so we can say $A \leq_P B$.

Similarly, consider a polynomial-time reduction algorithm that converts instances of decision problem B into instances of decision problem A.

This indicates that if given an instance of decision problem B, by using the reduction algorithm we can create an instance of problem B which is identical to original instance of decision problem B. because the reduction algorithm takes polynomial time, B is polynomially reducible to B. so we can say $B \leq_P A$.

3. Summary: Prove that the problem of satisfiability for expressions in disjunctive normal form is in P.
   - Disjunctive normal form (DNF) is standard way to write Boolean functions. It is composed of sum of products which are described as an OR of ANDs.
   - If any of DNF clause is evaluated to 1(true), then DNF is satisfiable.
   - For example, if there is some variable p such that any one clause ha $(p \wedge \sim p)$, whatever the p value is assigned the clause will evaluate to 0(false).
   - If not, the clause will be evaluated to 1(true) depending on the value of variable when given 0 and 1.
   - so, each clause is simple checked deciding if a variable p exist such that p and ~p are in same clause it will return false, or else it will return true.
   - Let there are x number of clauses and every clause is having n literals then each clause is evaluated O(n^2) time and entire algorithm runs in O(xn^2)
   - Therefore, the problem of satisfiability for an expression in DNF is in P.

4. Prove that the following are equivalent: (a) V1 is a vertex cover of G. (b) V − V1 is an independent set in G. and, continuing, prove that the following are equivalent: (a) V2 is an independent set in G. (b) V2 is a clique in G^C.
   - The main goal of the questions is to prove that all problems are equivalent.
   - First, we prove that if V1 is vertex cover of graph then V-V1 is an independent set in the graph and V2 is a clique in the graph.

- If V1 is a vertex cover of G, it contains at least one end of each edge in the graph. This implies that removing V1 from the graph leaves no edges in the remaining vertices V-V1. As a result, V-V1 is an independent set in G.
- Because V1 is a vertex cover of G, it contains one end of every edge in the graph. This means we can add as many edges as we want between the vertices in V1 without causing any conflicts.
- As a result, we can add edges between every pair of vertices in V1, transforming V1 into a clique in G.

- For the second part, we must show that if V-V1 is an independent set in G and V2 is a clique in G, then V1 is a vertex cover of G.
- Let V-V1 be an independent set in G and V2 be a clique in G.
- This means that there are no edges connecting the vertices in V-VI and that every vertex in V2 is connected by an edge to every other vertex in V2.
- Because no edges connect the vertices in V-V1, all edges in the graph must be incident on vertices in VI. That is, V1 is a vertex cover of G.
- Furthermore, because each vertex in V2 is linked to every other vertex in V2 by an edge, V2 must contain at least one end of every edge in the graph. This means that V2 is a vertex cover of G as well.

README:

Done between 1’o clock to 7’0 clock.

Read the slides and took references from there,

Asked some of my classmates to explain and discussed what to write with them.

I took help from these students named Harsha and Nikhila.