CS 624 - ANALYSIS OF ALGORITHMS

Assignment 6

Aravind Haridas – 02071139

1. Problem 26.1-7 (p714)

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?

Sol: To transform a flow network $G=(V,E)$ with vertex capacities into an equivalent flow network $G'=(V',E')$ without vertex capacities, where each vertex v in G has a capacity limit $l(v)$, you can use the following transformation:

- Transform Each Vertex: For each vertex v in G, split v into two vertices v_{in} and v_{out} in G'.
- Connect the Vertices: Add an edge (v_{in}, v_{ou}) with a capacity of $l(v)$ between the new vertices.
- Adjust Adjacent Edges: For each original edge (u,v) in G, replace it with (v_{in}, v_{ou}) in G'.

This transformation ensures that the flow passing through each vertex does not exceed its original capacity by enforcing the limit on the new edge between v_{in} and v_{out}.

- Vertices and Edges in G': G' will have 2$|V|$ vertices (since each vertex in G is split into two), and $|E|+|V|$ edges (original edges plus the new edge for each vertex).

2. Exercise 1.4 ([aux16] p7)

Prove that all problems that are NP-complete are polynomial equivalent, in the sense that if A and B are NP-complete, then A ≤p B and B ≤p A.

Sol: All NP-complete problems are polynomial equivalent, NP-complete problems are defined such that any problem in NP can be reduced to them in polynomial time. If A and B are both NP-complete, then:

- There exists a polynomial-time reduction from A to B ($A \leq_p B$).
- There exists a polynomial-time reduction from B to A ($B \leq_p A$).
- If A and B are NP-complete problems, then A $\leq_p$ B and B $\leq_p$ A.
- This is because, by definition, an NP-complete problem is one that every other NP problem can be polynomially reduced to, and it itself can be reduced to any other NP-complete problem.
- Thus, $A \leq_p B$ implies there is a polynomial-time reduction from A to B, and similarly, $B \leq_p A$ implies a reduction from B to A.

- Example: Consider two classic NP-complete problems: the 3-SAT Problem and the Traveling Salesman Problem
- 3-SAT Problem: Given a boolean formula in conjunctive normal form with three literals per clause, determine whether there is a truth assignment to the variables that satisfies the formula.
- Traveling Salesman Problem: Given a list of cities and the distances between each pair of cities, find the shortest possible route that visits each city exactly once and returns to the origin city.
- Polynomial Time Reductions: 3-SAT to TSP: Construct a graph where each city corresponds to a possible truth assignment that satisfies one clause of the 3-SAT formula. Connect cities with edges weighted to reflect the number of changes in variable assignments between them. Set a threshold such that the TSP tour's total distance is below this threshold if and only if there is a way to satisfy all clauses of the 3-SAT formula.
- TSP to 3-SAT: For each city and path possibility in the TSP, create variables and clauses that encode the requirement to visit each city once and only once, and to return to the origin. The formula is satisfiable if and only if there exists a tour of the required length or shorter.
- This example shows that solutions to 3-SAT can inform solutions to TSP and vice versa, illustrating their polynomial equivalence.

3. Exercise 2.1 ([aux16] p9)

Summary: Prove that the problem of satisfiability for expressions in disjunctive normal form is in P (that is, it is polynomial time).

Sol: The satisfiability problem for expressions in Disjunctive Normal Form (DNF) is in P because:

- Problem Setup: Disjunctive Normal Form (DNF) is a logical formula consisting of a disjunction of conjunctions. Each conjunction is a set of literals.
- Polynomial-Time Solution: To determine if a DNF expression is satisfiable, check each conjunction to see if it can be true under some assignment. This process involves evaluating each literal in the conjunctions sequentially, which is feasible in polynomial time since the number of operations required is linear with respect to the number of literals and conjunctions.
- A DNF expression is a disjunction (OR) of several conjunctions (ANDs) of literals.
- To determine if a DNF expression is satisfiable, check if any conjunction of literals within the expression is true.
- This can be done in polynomial time by evaluating each conjunction separately to see if it is true under some assignment, which is straightforward as each conjunction is checked independently.

CS 624 - ANALYSIS OF ALGORITHMS

4. Exercise 3.6 ([aux16] p16) Prove that the following are equivalent:

(a) V_1 is a vertex cover of G.

(b) $V - V_1$ is an independent set in G.

and, continuing, prove that the following are equivalent: (a) V_2 is an independent set in G.

(b) V_2 is a clique in G^c.

Sol: Equivalence Proofs:

- V_1 is a vertex cover if and only if every edge in G is incident to at least one vertex in V_1
- $V - V_1$ is an independent set if no two vertices in $V - V_1$ share an edge in G, which is equivalent to saying V_1 covers all edges.

Independent Set and Clique in Complement Equivalence:

- V_2 is an independent set if no two vertices in V_2 share an edge in G.
- V_2 is a clique in G^c (the complement of G) if every pair of vertices in V_2 is connected by an edge in G^c, implying they are not connected in G.

These transformations and proofs cover the fundamental properties of NP-completeness and graph theory, aligning with typical computer science curriculum standards.

Example: Let's consider a simple graph and demonstrate the concepts of vertex cover, independent set, and clique.

- **Graph G**: Vertices: A, B, C and Edges: $(A, B), (B, C)$

Diagram for Graph G and G^c:

```
Graph :  G          A ---- B ---- C

Graph   Gc        A       B     C
                    \   /  \   /
                     \ / \ /
                      \/ \/
```

Vertex Cover ($V1$): Choose $\{B\}$ as it covers both edges (A, B) and (B, C).

Independent Set ($V - V1$): The set $\{A, C\}$ forms an independent set because there is no edge directly connecting A and C in G.

Clique in G^c: In the complement graph G^c, $\{A, C\}$ forms a clique because both possible edges (A, C) exist in G^c.

This graphical representation and example should help visualize how a vertex cover in G relates to an independent set and how this independent set translates into a clique in the complement graph G^c.