Prob. 1	Prob. 2	Prob. 3	Prob. 4

Problem 1. /25

Recall that the subgraph G' of G induced by a subset V' of V is the graph $G' = (V', E')$, where E' is the subset of E containing exactly those edges with both endpoints in V'. You are given an undirected graph $G = (V, E)$ (as an array of adjacency lists) and a positive integer K and asked to find a maximum subset of vertices such that the subgraph of G induced by these vertices has degree at least K (i.e., every vertex in the induced subgraph has degree of at least K).

Consider the following greedy algorithm:

> repeatedly remove the remaining vertex of smallest (updated) degree, until all remaining vertices have degree at least K.

We use a priority queue to retrieve the remaining vertex of smallest degree; since removing a vertex decreases the degree of its neighbors, we use a Fibonacci heap to take advantage of its efficient `DecreaseKey` operation. Thus our algorithm begins by building a Fibonacci heap containing all vertices, using the degree of each vertex as the priority key. Our main loop then simply runs a `DeleteMin` operation on the heap. If the heap is empty, the algorithm stops and returns the empty set; otherwise, it tests the key (degree) of the returned vertex v against K. If the degree is at least K, the algorithm stops and returns the collection of vertices still in the heap. Otherwise, the algorithm updates the degrees of the neighbors of v (listed in the adjacency list of v) using `DecreaseKey` and begins the next iteration.

Prove the correctness of this algorithm and that it runs in $O(|E| + |V| \cdot \log(|V|))$ worst-case time.

Solution

Proof of correctness

The proposed greedy method transforms the original objective function that maximizes the subset of vertices with at least K degree to one that minimize the subset of vertices such that the sub-graph induced by these vertices has degree smaller than K. Therefore, the left vertices construct the maximal sub-graph of G that includes the vertices with at least K degree.

I will use the "cut and paste" method to prove that the removed vertices by the greedy method constitute the minimal subset so that the induced sub-graph the vertices of which all have degree smaller than K.

Suppose there is an optimal solution, a vertex sub-set V^* to be removed. $V^* = v_1^*, v_2^*, \cdots, v_n^*$ are the sequential deleted vertices with degree smaller than K and the size of this subset is N.

The function $d(v)$ represents the degree of vertex v. According to the proposed greedy algorithm, initially the vertex with smallest degree is denoted as v_0 and its degree is smaller than K(Otherwise, the greedy algorithm won't run), hence $d(v_1^*) \leq d(v_0)$.

Case 1. If v_0 is the v_i^* in V^*, namely, $V^* = \{v_1^*, v_2^*, \cdots, v_{i-1}^*, v_i^*, \cdots, v_n^*\}$.

During the sequential deletion of $v_1^*, v_2^*, \cdots, v_{i-1}^*$, the degree of v_i^* may be updated because some of vertices in $\{v_1^*, v_2^*, \cdots, v_{i-1}^*\}$ have edges linking v_i^*. Also, the degree of $\{v_{i+1}^*, \cdots, v_n^*\}$ will be updated because of the deletion of v_i^*.

If I exchange the position of v_1^* and v_i^*, the degree of all vertices in V^*/v_i^* connecting v_i^* would be updated after the v_i^* is removed first. Moreover, these vertices' degree would be the same as the those in vertex removing process of original V^* where v_i^* is the i-th deleted. Therefore, the optimal solution V^* can be transformed to anther optimal solution with the first greedy choice.

Case 2. When v_0 is not in V^*, the v_1^* is replaced by v_0 and $V_\alpha = V^* - v_1^* \bigcap v_0$, we will see if V_α still hold the optimality, namely, $|V_\alpha| = V^*$.

Here, I define another function, $R(v)$ stands for the a set of vertices that have edges with v as one of end-points.

$r(v)$ is a sub-set of $R(v)$ including the vertices both in $R(v)$ and V^*, namely, $r(v) = R(v) \cup V^*$

If $R(v_1^*)) \cup R(v_0) = \phi$,
If $R(v_1^*)) \cup R(v_0) \neq \phi$

Time complexity

Problem 2. /25

Give a polynomial-time algorithm for each of the following problems—sketch a proof of their correctness and analyze their running time.

You are given an undirected graph $G = (V, E)$ with (not necessarily distinct) positive integral weights for the edges, and an edge $e_0 \in E$.

1. Decide whether *every* minimum spanning tree of G contains e_0.
2. Decide whether *some* minimum spanning tree of G contains e_0.

Solution

Problem 3.

/25

Give a polynomial-time algorithm for each of the following problems—sketch a proof of their correctness and analyze their running time.

You are given a bipartite graph $G = (V_1, V_2, E)$, $|V_1| = |V_2|$, and an edge $e_0 \in E$.

1. Decide whether *every* perfect matching of G contains e_0.
2. Decide whether *some* perfect matching of G contains e_0.

Problem 4. /25

You are given an undirected network $N = (V, E)$ with integral edge capacities, a source, s, and a sink, t. Define N to be *k-stable* if and only if the value of the maximum s-t flow does not increase under any k-edge alteration. A k-edge alteration consists of picking k edges of the network and assigning them arbitrary new (positive integral) capacities.

1. Design an algorithm to test whether N is 1-stable; your algorithm should run in $O(|V|^2 \cdot |E|)$ time.

2. Design an algorithm to test whether N is 2-stable; your algorithm should also run in $O(|V|^2 \cdot |E|)$ time, although it is likely to involve significantly more work.