COMP 6651: Assignment 6

Fall 2020

Submission through Moodle is due by November 1st at 23:55

1. You are given a simple undirected graph $G = (V, E)$ with two edge weight functions: $w_1, w_2 : E \to \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$. That is, each edge $e \in E$ has two weights associated with it: $w_1(e) \in \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ and $w_2(e) \in \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$. You are also given a source vertex $s \in V$ and a target vertex $t \in V$. For a path $p = \langle v_0, v_1, \ldots, v_k \rangle$ from $s = v_0$ to $v = v_k$ we can measure the weight of this path with respect to w_1 and with respect to w_2 separately. The two weights are denoted, respectively, by $w_1(p)$ and $w_2(p)$. We define the **product-weight** of a path p by $w_{\text{prod}}(p) = w_1(p) \cdot w_2(p)$. The goal is to design a polynomial time algorithm to find minimum product-weight paths from s to t in G.

   (a) Prove that every minimum product-weight path is acyclic.

   (b) Let p be a minimum product-weight path. What is the maximum possible value of $w_1(p)$? What about $w_2(p)$? State it as a function of basic parameters of G.

   (c) Consider the following algorithm: create a new weight function $w'(e) = w_1(e) + w_2(e)$. Run Dijkstra's algorithm with respect to w'. Return the paths found by Dijkstra as minimum product-weight paths in G. Give a small example on which this algorithm fails.

   (d) Consider defining a new graph $G' = (V', E')$ such that each vertex $v' \in V'$ is described by a label (v, c_1, c_2) where $v \in V$ and c_1, c_2 are non-negative integers. Define the source vertex in the new graph to be $s' = (s, 0, 0)$. Finish the definition of this graph (finish specifying V' and specify the edge set E') so as to satisfy the following property: $v' = (v, c_1, c_2)$ is reachable from s' in G' if and only if v is reachable from s in G via a path p with $w_1(p) = c_1$ and $w_2(p) = c_2$.

   Observe that you don't need to consider nodes with values of c_1 and c_2 exceeding the bounds from part (b).

   (e) State upper bounds on V' and E' in terms of $|V|$ and $|E|$.

   (f) Explain how you can use G' in order to solve the minimum product-weight paths problems in G.

   (g) State the running time of the resulting algorithm in terms of $|V|$ and $|E|$. Briefly justify it.

2. Let $G = (V, E)$ be a simple undirected graph. Recall that a subgraph $H = (W, F)$ of G is called spanning if $V = W$ and $F \subseteq E$. Recall that a graph is called bipartite if its set of vertices can be partitioned into two blocks such that all edges have exactly one endpoint in each block.

   Given $G = (V, E)$ with $n = |V|$ vertices and $m = |E|$ edges, we wish to find a spanning bipartite subgraph with at least $m/2$ edges. Consider the following algorithm:

   Initially, color vertices of V with two colors, red and blue, arbitrarily. Call a vertex v **bad** if v has more neighbors of its own color than of the opposite color. Call an edge $\{u, v\}$ **monochromatic** if u and v have the same color; otherwise, call $\{u, v\}$ a **bichromatic** edge. The algorithm can be summarized as: **while** there exists a bad vertex, pick an arbitrary bad vertex and swap its color **end while**.

Answer the following:

(a) Count the spanning subgraphs of G. Your answer should be a simple formula in terms of basic parameters of G. Provide a brief justification (at most 1 sentence).

(b) Write down pseudocode for the above algorithm. Show low level implementation details. For example, what data structure do use to maintain coloring? How do you decide if a vertex is bad or good?

(c) Let A be a set of bichromatic edges in G at any point during the algorithm. Prove that (V, A) is a spanning bipartite subgraph of G.

(d) Prove: if there are no bad vertices then there are at least $m/2$ bichromatic edges.

(e) Prove that the following statement is false: "The number of bad vertices decreases in each iteration of the while loop." Give an example where in a single iteration of the while loop the number of bad vertices increases by 10.

(f) Prove that the algorithm terminates in at most m rounds.

(g) Combine the previous parts to prove correctness of the algorithm.

(h) State the running time of the algorithm and briefly justify it.