Network and Integer Optimization - Problem Set 1

Sarp Sümer

February 18, 2025

6 Checking bipartiteness

(a) We first run BFS on the given input graph $G = (V, E)$, starting from an arbitrary vertex $s \in V$. We rerun BFS from an unvisited vertex until all vertices have been eventually visited. For every vertex $v \in V$, there is some vertex $s_v \in V$ that is in the same component as v such that some BFS run was initiated from s_v and led to the discovery in v. We record the value $d(s_v, v)$. We then iterate through every edge $\{u, v\} \in E$. If any of these edges satisfies $d(s_u, u) = d(s_v, v)$, then we have certified the existence of an odd cycle (see proof in previous task). To reconstruct this cycle, we follow a two-pointer approach. We initialize two pointers at p_u at u and p_v at v and simultaneously move them to their predecessors in every iteration, as long as $p_u \neq p_v$. This gives us the closest common ancestor s^* of u and v in the BFS tree. We then concatenate the resulting $s^* - u$ and $s^* - v$ paths, which make up the odd cycle (again, see proof in previous task).

If no such edge is found, then we have proof that the sets:

$$X = \{v \in V \mid d(s_v, v) \text{ is even}\}$$
$$Y = \{v \in V \mid d(s_v, v) \text{ is odd}\}$$

are a partition of V satisfying the requirements, so we simply return them. Correctness follows directly as we turned the proof above into an algorithm. Running BFS takes linear time, and iterating over every edge also takes $\mathcal{O}(|E|)$ time. Constructing the two sets of the partition takes $\mathcal{O}(|V|)$ time and in the other case, constructing the odd cycle takes also $\mathcal{O}(|V|)$ time, as we simply backtrack to construct two paths in G, which can have length at most $|V| - 1$. Overall the algorithm runs in linear time.

(b) See attached notebook.