University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn
P. Bamberger

Algorithm Theory
Exercise Sheet 5

Exercise 1: Matching & Vertex Cover in Bipartite Graphs *(5+5+2 Points)*

Let $G = (V, E)$ be a graph and assume that $M^* \subseteq E$ is a maximum matching and that $S^* \subseteq V$ is a minimum vertex cover (i.e., M^* is a largest possible matching and S^* a smallest possible vertex cover). We have seen in the lecture that for every graph G, it holds that $|M^*| \leq |S^*|$ because the edges in M^* have to be covered by disjoint nodes in S^*. In this exercise, we assume that G is a *bipartite graph* and our goal is to show that in this case, it always holds that $|M^*| = |S^*|$.

a) Recall that we can solve the maximum bipartite matching problem by reduction to maximum flow. Also recall that if we are given a maximum matching M^* (and thus a maximum flow of the corresponding flow network), we can find a minimum s-t cut by considering the residual graph. Describe how such a minimum cut looks like.

   Hint: Consider the set of all nodes which can be reached from an unmatched node on the left side via an alternating path.

b) Use the above description to show that any bipartite graph G has a vertex cover S^* of size $|M^*|$.

c) Show that the same thing is not true for general graphs by showing that for every $\varepsilon > 0$, there exists a graph $G = (V, E)$ for which $|S^*| \geq (2 - \varepsilon)|M^*|$.

   Hint: First try to find any graph for which $|S^*| > |M^*|$.

Exercise 2: Matching in Regular Graphs *(5+5 Points)*

The degree of a node in a graph is the number of its neighbors. A graph is called r-regular for an $r \in \mathbb{N}$ if all nodes have degree r.

a) Show that any regular bipartite graph has a perfect matching.

b) Show that an n-regular graph with $2n$ nodes has a matching of size at least $n/2$.

Exercise 3: Cover all Edges *(8 Points)*

You are given an undirected graph $G = (V, E)$, a capacity function $c : V \to \mathbb{N}$, and a subset $U \subseteq V$ of nodes. The goal is to cover every edge with the nodes in U, where every node $u \in U$ can cover up to $c(u)$ of its incident edges.
Formally, we are interested in the existence of an assignment of the edges to incident nodes in U such that each node u gets assigned at most $c(u)$ of its incident edges.

Devise an efficient algorithm to determine whether or not such an assignment exists for a given subset U and a given cost function c and state its runtime.