Discrete Optimization: Homework #11, Ex. #4

Denis Steffen, Yann Eberhard & Gaëtan Bossy

May 25, 2018

We assume that all vertices of G are connected by at most one edge. If the graph isn't connected, we use the algorithm for every connected components.

Algorithm :
Input : $G = (V, E)$, a graph
Output : M a maximum cardinality matching

While $oracle(G) = false$: We delete the vertex v with minimum degree such that the graph without v remains connected and update $G = (V \backslash \{v\}, E')$ where $E' := \{(a, b) \in E : a \neq v \text{ and } b \neq v\}$.

Then we take a vertex v from the updated graph $\widetilde{G} = (\widetilde{V}, \widetilde{E})$. Let $F_v = \{(u, v) : (u, v) \in \widetilde{E}\}$, we call $oracle((\widetilde{V}, (\widetilde{E} \backslash F_v) \cup \{f\}))$ for every $f \in F_v$ until there is a positive response of the oracle for a particular $f = (u, v) \in F_v$ (there exists a such f because $\widetilde{G}$ has a perfect matching). We put f in M and update $\widetilde{G} = (\widetilde{V} \backslash \{u, v\}, \widetilde{E} \backslash (F_v \cup H))$ where $H = \{(u, w) \in \widetilde{E}\}$.
We repeat this last part until $\widetilde{G}$ is an empty graph.

At the end, M is a maximum cardinality matching of the graph G obtained with at most $|V| + |E|$ calls to the oracle.