# 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 $G = (V, E)$. Let $F_v = \{(u, v) : (u, v) \in E\}$, we call $oracle((V, (E \backslash F_v) \cup \{f\}))$ 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 $G$ has a perfect matching). We put $f$ in $M$ and update $G = (V \backslash \{u, v\}, E \backslash (F_v \cup G))$ where $G = \{(u, w) \in E\}$.
We repeat this last part until $G$ 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.