Algorithms. Assignment 6

Problem 10

A cycle in an undirected graph is called odd if it has an odd number of nodes and edges. Odd cycles are of special interest, as one can easily show that a graph is bipartite, i.e., 2-colorable, if and only if it does not contain odd cycles. However, this is not the exercise. Instead, the problem is:

Give a polynomial-time algorithm that finds some shortest odd cycle in a given graph (if some exists). Give and motivate your time bound. (Figure out what you can achieve; we do not prescribe some specific time bound.)

Hint: Use BFS. Perhaps the most critical part is to prove that your algorithm cannot fail to really find some *shortest* odd cycle.

Problem 11

Given an undirected connected graph $G = (V, E)$ and a vertex $u \in V$. Let T be the tree that you get by running BFS with root u. Let T' be the tree that you get by running DFS with root u. Now assume you observe $T = T'$.

Prove that $G = T$, that is, if BFS and DFS with root u generate the same tree, then G cannot contain any edges that do not belong to T.

Optional: Will it make any difference if the trees are run from different roots?