Universität Greifswald
Institute für Mathematik and Informatik
Lecturer: Marc Hellmuth Tutor: Nikolai Nøjgaard

6. Exercise "Datenstrukturen und Effiziente Algorithmen", WS 18/19

Exercise 1: (5 Credits)
A walk of length k in an arbitrary (directed or undirected) graph $G = (V, E)$ is a sequence $(v_1, v_2 \dots, v_k)$ of vertices such that $(v_i, v_{i+1}) \in E$, $1 \leq i \leq k-1$. Note, the vertices in that sequence don't need to be pairwisely distinct.
Consider now the adjacency matrix $A = (a_{ij})$ of G and let $A^k = \prod_{l=1}^{k} A$. Proof that the entries $A^k = (b_{ij})$ coincide with the number of walks of length k between the vertices i and j.

Exercise 2: (12.5 Credits)
The distance $d(u, v)$ of two vertices is the smallest number of edges on a path that connects u and v. The diameter of a (undirected) tree $T = (V, E)$ is defined as $\max\{d(u, v) \mid u, v \in V\}$, that is, the largest of all shortest-path distances in the tree. Give an $O(|V|)$-time algorithm to compute the diameter of a tree. Proof the correctness and runtime of your algorithm.

Exercise 3: (12.5 Credits)
Implement the BFS-algorithm for graphs in adjacency list representation and modify your algorithm to test whether the input graph contains a cycle with an odd number of vertices. Use your algorithm to determine which of the four graphs $G_1, \dots, G_4$ (provided at the homepage, see working material) contain an odd cycle and which do not.
Turn in your code and the output of your program. When handing in programming exercises, always document how to compile and run your program.

Deadline: Wednesday - November 28, 2018 - 12.15pm