Home Work 4.
Algorithms and Data Structures

M. Pellegrini

May 2003

HW4.1 Given a Directed Graph $G = (V, E)$ represented with Adjacency lists, write in pseudocode an algorithm that computes the *transpose* of a graph. The transpose of $G^T = (V, E')$ of G has the same vertex set as G and $(u, v) \in E'$ if and only if $(v, u) \in E$ (intuitively, every edge of G is reversed). Analyze the asymptotic running time of your algorithm as a function of the number of vertices $|V|$ and the number of edges $|E|$. Repeat with G represented as an adjacency matrix.

HW4.2 Given a Directed Graph $G = (V, E)$ represented with Adjacency lists, write in pseudocode an algorithm that computes the *square* of a graph. The square of $G^2 = (V, E')$ of G has the same vertex set as G and $(v, u) \in E'$ if and only if for some node w $(v, w) \in E$ and $(w, u) \in E$ (intuitively, every edge of G^2 connects two nodes that in G belong to a path of length 2). Analyze the asymptotic running time of your algorithm as a function of the number of vertices $|V|$ and the number of edges $|E|$. Repeat with G represented as an adjacency matrix.

HW4.3 Read proof of correctness of BFS on CLR.

HW4.4 Modify BFS to detect a cycle in an graph (undirected) represented as adjacency list.

HW4.5 (CLR 23.3-8) Singleton example. Build a Graph G and a DFS visit of G so that a node u having both incoming and outgoing edges ends up as a singleton tree in the DFS-forest.

HW4.6 CLR 23.3-6 Descendant example. Build a Graph G and a DFS visit of G so that for two nodes u and v in G there is a directed path form u to v but the DFS-labelling is such that $d[v] \geq f[u]$.

HW4.7 Given a *directed* graph G describe an algorithm based on DFS that detects a cycle in G.

HW4.8 Given an *undirected* graph G describe a method to find a cycle in G based on DFS that uses time $O(|V|)$.

HW4.9 Write pseudo-code for operations extract-min, decrease-key and heapify. Each operation must run in time $O(\log n)$ on a heap implemented in an array + auxiliary direct-address table.