MO417 – Complexidade de Algoritmos
Segundo Semestre de 2008
Oitava Lista de Exercícios

Árvore Geradora Mínima

1. Let (u, v) be a minimum-weight edge in a graph G. Show that (u, v) belongs to some minimum spanning tree of G.
2. Given a graph G and a minimum spanning tree T, suppose that we decrease the weight of one of the edges not in T. Give an $O(n)$-time algorithm for finding the minimum spanning tree in the modified graph.
3. Is the Fibonacci-heap implementation of Prim's algorithm asymptotically faster than the binary-heap implementation for a sparse graph G = (V, E), where $|E| = \Theta(V)$? What about for a dense graph, where $|E| = \Theta(V^2)$? How must $|E|$ and $|V|$ be related for the Fibonacci-heap implementation to be asymptotically faster than the binary-heap implementation?
4. Suppose that all edge weights in a graph are integers in the range from 1 to $|V|$. How fast can you make Kruskal's algorithm run? What if the edge weights are integers in the range from 1 to W for some constant W?
5. Suppose that all edge weights in a graph are integers in the range from 1 to $|V|$. How fast can you make Prim's algorithm run? What if the edge weights are integers in the range from 1 to W for some constant W?
6. Suppose that the edge weights in a graph are uniformly distributed over the half-open interval $[0, 1)$. Which algorithm, Kruskal's or Prim's, can you make run faster?
7. Suppose that a graph G has a minimum spanning tree already computed. How quickly can the minimum spanning tree be updated if a new vertex and incident edges are added to G?
8. Professor Toole proposes a new divide-and-conquer algorithm for computing minimum spanning trees, which goes as follows. Given a graph G = (V, E), partition the set V of vertices into two sets V_1 and V_2 such that $|V_1|$ and $|V_2|$ differ by at most 1. Let E_1 be the set of edges that are incident only on vertices in V_1, and let E_2 be the set of edges that are incident only on vertices in V_2. Recursively solve a minimum-spanning-tree problem on each of the two subgraphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. Finally, select the minimum-weight edge in E that crosses the cut (V_1, V_2), and use this edge to unite the resulting two minimum spanning trees into a single spanning tree.

   Either argue that the algorithm correctly computes a minimum spanning tree of G, or provide an example for which the algorithm fails.