

Due in class: Sep 29.

- (1) Describe an efficient algorithm that given an undirected graph G , determines a spanning tree of G whose largest edge weight is minimum, over all spanning trees of G . Give an argument justifying your algorithm.
- (2) The diameter of a tree $T = (V, E)$ is given by

$$\max_{u, v \in V} \delta(u, v)$$

where $\delta(u, v)$ is the distance between u and v in the tree T . Give an $O(|V|)$ algorithm for computing the diameter of the tree. Write a proof of correctness for your algorithm.

- (3) A directed graph is said to be semi-connected if, for any two vertices $u, v \in V$ we have that u can reach v or v can reach u . Give an efficient algorithm to determine whether or not G is semi-connected. Prove that your algorithm is correct and analyse its running time.
- (4) Assume that we have a network (a connected undirected graph) in which each edge e_i has an associated bandwidth b_i . If we have a path P , from s to v , then the capacity of the path is defined to be the minimum bandwidth of all the edges that belong to the path P . We define $capacity(s, v) = \max_{P(s, v)} capacity(P)$. (Essentially, $capacity(s, v)$ is equal to the maximum capacity path from s to v .) Give an efficient algorithm to compute $capacity(s, v)$, for each vertex v ; where s is some fixed source vertex. Show that your algorithm is “correct”, and analyze its running time.
(Design something that is no more than $O(|V|^2)$, and with the right data structures takes $O(|E| \log |V|)$ time.)
- (5) Let G be a directed graph. The vertices of G have been numbered $1 \dots n$ (where n is the number of vertices in G). Let $small(i) = \min\{j | j \text{ is reachable from } i\}$. In other words, for a vertex numbered i , $small(i)$ is the smallest numbered vertex reachable from it. Design an $O(V + E)$ algorithm to compute $small(i)$ for all vertices in the graph.
- (6) (For Extra Credit) An n -vertex undirected graph is a *wasp* if it has a vertex of degree one (the sting) connected to a vertex of degree two (the tail) connected to a vertex of degree $n - 2$ (the body) connected to the other $n - 3$ vertices (the feet). Some of the feet may be connected to other feet.

Describe an algorithm that decides whether or not a given adjacency matrix represents a wasp by examining only $O(n)$ of the entries. Your algorithm should also run in $O(n)$ time. (Again assume that the input is available in a matrix.)

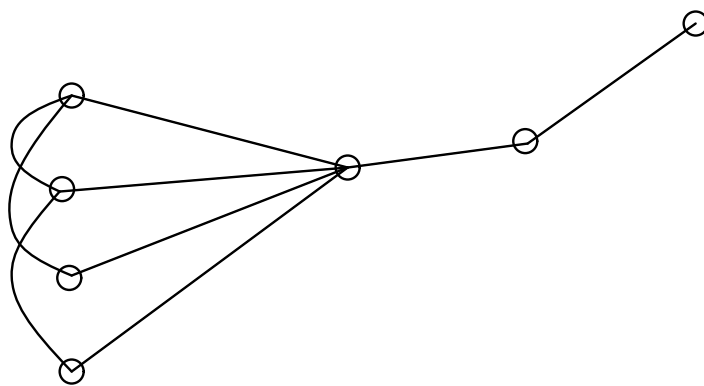


Figure 1: Wasp