CS3100: Algorithms
Graph Algorithms
In class, October 6

Problems

1. Some friends of yours work on wireless networks, and they're currently studying the properties of a network among n mobile devices. As the devices move around, they define a graph at any point in time as follows: there is a node for each device, and there is an edge between device i and device j if the physical locations are less than 500 meters apart. (If so, we say that i and j are in range of each other.)

   They'd like it to be the case that the network of devices is connected at all times, and so they've constrained the motion to satisfy the following property: at all times, each device i must be in range of at least $n/2$ of the other devices. Our question is: does this property actually guarantee that the network will remain connected? In other words:

   Claim: Let G be a graph on n nodes, where n is an even number. If every node of G has degree at least $n/2$, then G is connected.

2. Decide if you think the claim is true or false, and give a proof or counterexample.

   (a) Suppose we are given an instance of the minimum spanning tree problem on a graph G, with all positive and distinct edge costs. Let T be a minimum spanning tree for this instance. Now suppose we replace each edge weight $w(e)$ by its square, $w(e)^2$, which has the same edge and vertex sets but a different weight function. True or false: T is still a minimum spanning tree for this new graph.

   (b) Now suppose we have a weighted directed graph G with all positive distinct edge weights. Let P be a minimum cost path from vertex s to t in this graph. We now again replace the weight on each edge $w(e)$ with its square $w(e)^2$, creating a new graph with the same edge and vertex sets but new weights. True or false: P must still be a minimum cost s to t path in this new graph.

3. A graph (V, E) is *bipartite* if the vertices V can be partitioned into two subsets L and R such that every edge has one endpoint in L and one in R.

   (a) Prove that every tree is bipartite.

   (b) Describe and analyze an efficient algorithm that determines whether a given undirected graph is bipartite.