Graph Representation - Adjacency List, Adjacency Matrix

Problem 1 (25 points)

The square of a directed graph $G = (V, E)$ is the graph $G^2 = (V, E^2)$ such that $(u, w) \in E^2$ iff for some $v \in V$, both $(u, v) \in E$ and $(v, w) \in E$. In other words, $G^2$ contains an edge $(u, w)$ whenever there is a path of length 2 between $u$ and $w$ in $G$.

a) Describe an efficient algorithm for computing $G^2$ from $G$ for both adjacency list and adjacency matrix representations of $G$.

b) Analyze the running times of your algorithms in both cases.

c) Which representation would you use for a graph with very few edges, and why? If the graph was completely connected would that change your answer?

Graph Search Algorithms

Problem 2 (25 points)

Write down the output order of nodes if the graph in the figure is traversed using:

- A depth first search
- A breadth first search
Problem 3 (25 points)

a) Consider the DFS tree in the figure above. List the output order of the nodes:
   • for a post-order traversal
   • for a pre-order traversal

b) Consider the bipartite graph below. The thick arrows indicate a matching. As you can see it is maximal. The dashed arrows indicate other edges in the graph not in the matching. Find an augmenting path that will transform the maximal matching into a maximum matching.

Problem 4 (25 points)

Give a counterexample to the conjecture that if there is a path from $u$ to $v$ in a directed graph $G$, and if $d[u] < d[v]$ in a depth-first search of $G$, then $v$ is a descendant of $u$ in the depth-first forest produced.
$d[w]$ is the discovery time of vertex $w$ in a depth-first search.
Spanning Trees

Problem 5 (25 points)

A maximum spanning tree \((V', E')\) is a subset of a graph \(G = (V, E)\) such that \(V' \in V, E' \in E\) and the sum of the edge-weights in \(E'\) is maximized.

a) How would you change Kruskal’s algorithm (for finding the minimum spanning tree) to find a maximum spanning tree of a \(G\). What is the change in the running time? (if there is any).

b) Suppose that all the edge weights in \(G\) are integers in the range \([1, |V|]\). How fast can you make your algorithm run? What if the weights are integers in the range \([1, W]\) for some constant \(W\)?

Extra Credit (20 points)

Given a graph \(G = (V, E)\). Describe a linear time algorithm to determine whether \(G\) is bipartite.