2 Administrivia

- HW#5 submission trouble!
- HW#6 will be out shortly
- HW#4 will be returned next week; solutions are up now
- HW#3 Q1a grading (it is 24), programming grading
- Scheduling final exam?
- Read the webboard!
- I beg of you, start earlier!

4 Connectivity in directed graphs

- Can’t just do an arbitrary BFS or DFS
  - Connectivity depends on starting node, i.e., “what can you reach from node X?”
  - Do DFS from every vertex!
- Alternative: develop connectivity matrix from adjacency matrix
  - Transitive closure of adjacency matrix
  - If \( L \rightarrow M \) and \( M \rightarrow N \), \( L \rightarrow N \)

5 Warshall’s Algorithm

- For all rows \( y \),
  - For all columns \( x \) in row \( y \),
    - If any value \( (x,y) \) is 1,
    - For all rows \( z \) in column \( y \),
      - If \( (y,z) \) is 1, then \( (x,z) \) should be 1

  - That’s it!
    - Remember array references are “backwards” \([y][x]\)
  - Yes, this actually works in one pass – all the holes are filled
  - What’s the complexity of this algorithm?

6 Weighted graphs

- How to represent? Not just 0s and 1s in the adjacency matrix; weight instead
- Example
  - Roadmap!
- Can be directed or undirected

7 MSTs with weights

- Many possible STs; how do we figure out the minimum?
- Simple idea: grow the tree from one node
  - Pick smallest edge from vertices that we know to nodes not in tree
  - Add edge and corresponding destination vertex to tree
• Add edges from new vertex to unknown nodes into priority queue
• Picking smallest edges: priority queue
• Applications
  – Minimizing wiring given multiple choices
  – In general, undirected graphs

8 However...
• If an edge to a destination vertex already exists in PQ, and we find a shorter path, need to replace the existing entry with shorter path
  – Simplest way: scan through PQ, see if any such edges exist, remove them, and insert the new one
  – Slicker ways of doing it include backpointers from vertices
• By the way, this is “Prim”

9 Shortest-path problem
• Given a graph with weighted edges, and a starting vertex, find shortest path to a target
• Dijkstra’s algorithm most canonical way of doing it
• So turns out you get shortest paths to all remote vertices from that starting vertex
• Can handle both directed and undirected graphs
  – Produces a directed tree
• Cannot handle negative weights

10 Dijkstra’s Algorithm: Basic idea
• Initialize an array of distances from starting node to each vertex – if there doesn’t exist a direct edge to a vertex, consider it at “infinite” distance
• Add the closest node not already in the shortest-path tree
• Update weights based on edges from newest node plus distance from starting to new – and keep track of the node we used to get to that target
• Repeat
• To find a path to a node, go backwards through the parent nodes

11 Next time
• Finish Dijkstra’s algorithm
• Floyd’s algorithm
• Putting things together, HW6 discussion