Today’s Outline

• Announcements:
  › HW 5 due Friday, May 31.

• Today’s Topics:
  › Weiss 9.5, 9.6

Graph Searching Methodology

Breadth-First Search (BFS)

• Breadth-First Search (BFS)
  › Use a queue to explore neighbors of source vertex, then neighbors of neighbors etc.
  › All nodes at a given distance (in number of edges) are explored before we go further

Depth First Search Algorithm

• Recursive marking algorithm
• Initially every vertex is unmarked

DFS(i; vertex)
mark i;
for each j adjacent to i do
  if j is unmarked then DFS(j)
end(DFS)

Marks all vertices reachable from i
DFS Application: Spanning Tree

- Given a (undirected) graph \( G(V,E) \) a spanning tree of \( G \) is a graph \( G'(V',E') \)
  - \( V' = V \), the tree touches all vertices (spans) the graph
  - \( E' \) is a subset of \( E \) such \( G' \) is connected and there is no cycle in \( G' \)
  - A graph is connected if given any two vertices \( u \) and \( v \), there is a path from \( u \) to \( v \)

Example of DFS: Graph connectivity and spanning tree

Example Step 2

Example Step 5

Example Steps 6 and 7

Example Steps 8 and 9

Red links will define the spanning tree if the graph is connected.

Now back up.
Example Step 10 (backtrack)

DFS(1)
DFS(2)
DFS(3)
DFS(4)
DFS(5)

Back to 5, but it has no more neighbors.

Example Step 12

DFS(1)
DFS(2)
DFS(3)
DFS(4)
DFS(6)

Back up to 4. From 4 we can get to 6.

Example Step 13

DFS(1)
DFS(2)
DFS(3)
DFS(4)
DFS(6)

From 6 there is nowhere new to go. Back up.

Example Step 14

DFS(1)
DFS(2)
DFS(3)
DFS(4)
DFS(6)

Back to 4. Keep backing up.

Example Step 17

DFS(1)

All the way back to 1. Done.

Adjacency List Implementation

- Adjacency lists

<table>
<thead>
<tr>
<th>M</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0 1 4 5 6</td>
</tr>
<tr>
<td>0</td>
<td>2 3 11 6</td>
</tr>
<tr>
<td>2</td>
<td>3 4 6</td>
</tr>
<tr>
<td>3</td>
<td>5 6 1</td>
</tr>
<tr>
<td>4</td>
<td>1 5 4</td>
</tr>
<tr>
<td>5</td>
<td>1 6</td>
</tr>
<tr>
<td>6</td>
<td>5 1 2</td>
</tr>
</tbody>
</table>

Index next
Another Use for Depth First Search: Connected Components

Connected Components

Depth-first Search for Labeling Connected components

Main:
   i : integer
   for i = 1 to n do M[i] := 0; //initial label is zero
   label := 1;
   for i = 1 to n do
      if M[i] = 0 then DFS(G,M,i,label);

Connected Components for Image Analysis

Performance DFS

• n vertices and m edges
• Storage complexity \(O(n + m)\)
• Time complexity \(O(n + m)\)
• Linear Time!

Breadth-First Search

BFS
   Initialize Q to be empty;
   Enqueue(Q,1) and mark 1;
   while Q is not empty do
      i := Dequeue(Q);
      for each j adjacent to i do
         if j is not marked then
            Enqueue(Q,j) and mark j;
      end(BFS)
Can do Connectivity using BFS
• Uses a queue to order search

Beginning of example

Next

Depth-First vs Breadth-First
• Depth-First
  › Stack or recursion
  › Many applications
• Breadth-First
  › Queue (recursion no help)
  › Can be used to find shortest paths from the start vertex
  › Can be used to find short alternating paths for matching

Minimum Spanning Tree
• Edges are weighted: find minimum cost spanning tree
• Applications
  › Find cheapest way to wire your house
  › Find minimum cost to wire a message on the Internet