Section 6: Breadth-first Search

SLIDES ADAPTED FROM ALEX MARIAKAKIS,
WITH MATERIAL FROM KRYSTA YOUSOUFIAN, MIKE ERNST, KELLEN DONOHUE
How is Homework 5 going?

Any questions?
Agenda

* Breadth-first search (BFS)
Can I reach B from A?
Breadth-First Search (BFS)

- Often used for discovering connectivity
- Calculates the **shortest path**

  \[
  \text{if and only if all edges have same positive or no weight}
  \]

- Depth-first search (DFS) is commonly mentioned with BFS
Breadth-First Search (BFS)

Starting at A, which nodes will be visited first in a BFS?
Breadth-First Search (BFS)

Starting at A, which nodes will be visited first in a BFS? B, C, D
Breadth-First Search (BFS)

Starting at A, which nodes will be visited second in a BFS?
Starting at A, which nodes will be visited second in a BFS? E, F, G
BFS Pseudocode

boolean bfs(Node start, Node goal):
    put \texttt{start} in a queue
    while (queue is not empty):
        pop node \texttt{N} off queue

        if (\texttt{N} is \texttt{goal}):
            return true
        else:
            for each node \texttt{C} that is child of \texttt{N}:
                push \texttt{C} onto queue

    return false
Breadth-First Search

START:
Q: <A>
Pop: A, Q: <>
Q: <B, C>
Pop: B, Q: <C>
Q: <C>
Pop: C, Q: <C>
Q: <>
DONE

Starting at A
Goal: C
Breadth-First Search with Cycle

START:
Q: <A>
Pop: A, Q: <>
Q: <B>
Pop: B, Q: <>
Q: <C>
Pop: C, Q: <>
Q: <A>
NEVER DONE

Starting at A
Goal: D
boolean bfs(Node start, Node goal):
    put start in a queue
    while (queue is not empty):
        pop node N off queue
        mark node N as visited
        if (N is goal):
            return true
        else:
            for each node C that is child of N:
                if C is not marked
                    push C onto queue
            return false
Breadth-First Search

Problem: Find everything reachable from A

Q: <>
Breadth-First Search

Q: <>
Q: <A>
Breadth-First Search

Q: <>
Q: <A>
Q: <>
Breadth-First Search

Q: <>
Q: <A>
Q: <>
Q: <C>
Breadth-First Search

Q: <>
Q: <A>
Q: <>
Q: <C>
Q: <C, D>
Breadth-First Search

Q: <>
Q: <A>
Q: <>
Q: <C>
Q: <C,D>
Q: <D>
Breadth-First Search

Q: <>
Q: <A>
Q: <>
Q: <C>
Q: <C ,D>
Q: <D>
Q: <D, E>
Breadth-First Search

Q: <>
Q: <A>
Q: <>
Q: <C>
Q: <C,D>
Q: <D>
Q: <D,E>
Q: <E>
Breadth-First Search

Q: <>
Q: <A>
Q: <>
Q: <C>
Q: <C, D>
Q: <D>
Q: <D, E>
Q: <E>
DONE
Shortest Paths with BFS

From Node B

<table>
<thead>
<tr>
<th>Destination</th>
<th>Path</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>&lt;B&gt;</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Shortest path to D? to E? What are the costs?
Shortest Paths with BFS

From Node B

<table>
<thead>
<tr>
<th>Destination</th>
<th>Path</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>&lt;B,A&gt;</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>&lt;B&gt;</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>&lt;B,D&gt;</td>
<td>1</td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Shortest path to D? to E? What are the costs?
Shortest Paths with BFS

<table>
<thead>
<tr>
<th>Destination</th>
<th>Path</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>&lt;B,A&gt;</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>&lt;B&gt;</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>&lt;B,A,C&gt;</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>&lt;B,D&gt;</td>
<td>1</td>
</tr>
<tr>
<td>E</td>
<td>&lt;B,D,E&gt;</td>
<td>2</td>
</tr>
</tbody>
</table>

Shortest path to D? to E? What are the costs?
Shortest Paths with Weights

Weights are not the same! Are the paths?

<table>
<thead>
<tr>
<th>Destination</th>
<th>Path</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Shortest Paths with Weights

We will discuss Dijkstra next section!