Artificial Intelligence

Jeff Clune
Assistant Professor
Evolving Artificial Intelligence Laboratory

University of Wyoming
AI Challenge Two!

• Due Sept. 20th
  • this Sunday!
Depth-First Search: Complexity (Tree)

- time:
  - $O(b^m)$
- space:
  - $O(b^m)$

- size of example space:
  - Chess: $\sim10^{120}$ (Shannon 1950)
  - Branching factor 30 (50 moves/100 ply typical)
  - Go: branching factor $\sim361$
**Modified Depth-First Search?**

*Only works for trees, not graphs*

- **time:**
  - $O(b^m)$

- **space:**
  - $O(bm)$

- space: $O(bm)$!!

- Deep Blue
  - $d = 20$
  - $b = \sim 35$
  - 280 vs. 2.2 trillion!!
Backtracking  Depth-First Search

Only works for trees, not graphs

- time:
  - $O(b^m)$
- space:
  - $O(m)$

If each node tells you which child to expand next, you don’t store $b$ nodes at each depth $m$, hence $O(m)$
Breadth-First Search

Depth-First Search
Breadth-First Search

same colors
intentionally vague
shortest path!

put start node in queue, with pathLength 0, mark gray
for node at start of queue:
add non-visited node.neighbors to queue, with node.pathLength +1, mark gray/visited
remove node from queue

Queue:
Shortest Path

DFS

BFS
Breadth-First Search

spanning tree
Breadth-First Search: Complexity

- **time:**
  - \( O(b^d) \)

- **space:**
  - \( O(b^d) \)

- same as DFS! (unmodified)
  - i.e. same on graphs (with cycles)

- space much larger than modified DFS
  - i.e. on trees…DFS has no space advantage on graphs
<table>
<thead>
<tr>
<th>Depth</th>
<th>Nodes</th>
<th>Time</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>110</td>
<td>.11 milliseconds</td>
<td>107 kilobytes</td>
</tr>
<tr>
<td>4</td>
<td>11,110</td>
<td>11 milliseconds</td>
<td>10.6 megabytes</td>
</tr>
<tr>
<td>6</td>
<td>$10^6$</td>
<td>1.1 seconds</td>
<td>1 gigabyte</td>
</tr>
<tr>
<td>8</td>
<td>$10^8$</td>
<td>2 minutes</td>
<td>103 gigabytes</td>
</tr>
<tr>
<td>10</td>
<td>$10^{10}$</td>
<td>3 hours</td>
<td>10 terabytes</td>
</tr>
<tr>
<td>12</td>
<td>$10^{12}$</td>
<td>13 days</td>
<td>1 petabyte</td>
</tr>
<tr>
<td>14</td>
<td>$10^{14}$</td>
<td>3.5 years</td>
<td>99 petabytes</td>
</tr>
<tr>
<td>16</td>
<td>$10^{16}$</td>
<td>350 years</td>
<td>10 exabytes</td>
</tr>
</tbody>
</table>

**Figure 3.13** Time and memory requirements for breadth-first search. The numbers shown assume branching factor $b = 10$; 1 million nodes/second; 1000 bytes/node.

Two lessons can be learned from Figure 3.13. First, the memory requirements are a bigger problem for breadth-first search than is the execution time. One might wait 13 days for the solution to an important problem with search depth 12, but no personal computer has the petabyte of memory it would take. Fortunately, other strategies require less memory.

The second lesson is that time is still a major factor. If your problem has a solution at depth 16, then (given our assumptions) it will take about 350 years for breadth-first search (or indeed any uninformed search) to find it. In general, exponential-complexity search problems cannot be solved by uninformed methods for any but the smallest instances.
## Review of BFS vs. DFS

<table>
<thead>
<tr>
<th></th>
<th>Breadth-First Search</th>
<th>Depth-First Search</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prioritizes</td>
<td>many short paths</td>
<td>fewer long paths</td>
</tr>
<tr>
<td>Time Complexity</td>
<td>$O(b^d)$</td>
<td>$O(b^m)$</td>
</tr>
<tr>
<td>Space Complexity</td>
<td>$O(b^d)$</td>
<td>graph: $O(b^m)$, tree: $O(m)$</td>
</tr>
<tr>
<td>Optimal? (Shortest Path?)</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>Complete? (Guaranteed Solution?)</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>Data Structure</td>
<td>Queue</td>
<td>Stack</td>
</tr>
</tbody>
</table>

Graph traversal for Breadth-First Search and Depth-First Search.
Uniform Cost Search

• Extension of BFS that expands node with cheapest path cost first

Note

• Goal check when expanded, not when added
• Replace in frontier if lower-path cost discovered

Try it in pairs!
Start: Sibiu
Goal: Bucharest