1 handout: slides
Search
The ability to think is perhaps the most distinctive of human capacities. Typically, thinking involves mentally representing some aspects of the world (including aspects of ourselves) and manipulating these representations or beliefs so as to yield new beliefs, where the latter may aid in accomplishing a goal. —Edward E. Smith (Psychology, U Michigan)

The ability to solve problems is one of the most important manifestations of human thinking. ... We might therefore suspect that problem solving depends on general cognitive abilities that can potentially be applied to an essentially unlimited range of domains. —Keith Holyoak (Psychology, UCLA)
**State:** hypothetical world state

**Operators:** actions that modify world

**Goal:** desired state or test

VW search space
VW state space
MC representation
Basic Algorithms
open ← an ordered list containing just the initial state.

Loop

If open is empty,
then return failure.

Node ← Pop(open).

If Node is a goal,
then return Node (or path to it).
else

Children ← Expand (Node).
Add Children to front of open.
Assume branching factor $b$ and solution at depth $d$.

Completeness:

Time:

Space:

Admissibility:
open ← an ordered list containing just the initial state.

Loop

If open is empty,
    then return failure.

Node ← Pop(open).

If Node is a goal,
    then return Node (or path to it).
else
    Children ← Expand (Node).
    Add Children to end of open.
Assume branching factor $b$ and solution at depth $d$.

Completeness:

Time:

Space:

Admissibility:
open ← an ordered list containing just the initial state.

Loop
   If open is empty,
      then return failure.
   Node ← Pop(open).
   If Node is a goal,
      then return Node (or path to it).
   else
      Children ← Expand (Node).
      Merge Children into open, keeping sorted by path cost.
Dealing with Graphs

1. Check for cycles with ancestors
2. Maintain closed list (hash table) to detect duplicates
## Comparison

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Time</th>
<th>Space</th>
<th>Complete</th>
<th>Admissible</th>
</tr>
</thead>
<tbody>
<tr>
<td>Depth-first</td>
<td>$b^m$</td>
<td>$bm$</td>
<td>If $m \geq d$</td>
<td>No</td>
</tr>
<tr>
<td>Breadth-first</td>
<td>$b^d$</td>
<td>$b^d$</td>
<td>Yes</td>
<td>If ops cost 1</td>
</tr>
<tr>
<td>Uniform-cost</td>
<td>$b^d$</td>
<td>$b^d$</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

- **branching factor**: $b$
- **maximum depth**: $m$
- **solution depth**: $d$
Assume $b = 10$; 100,000 nodes/sec; 100 bytes/node.

<table>
<thead>
<tr>
<th>Sol. depth</th>
<th>Nodes</th>
<th>Time</th>
<th>Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>11</td>
<td>.11 msec</td>
<td>1.1 Kb</td>
</tr>
<tr>
<td>2</td>
<td>111</td>
<td>1.1 msec</td>
<td>11 Kb</td>
</tr>
<tr>
<td>4</td>
<td>11,111</td>
<td>.11 sec</td>
<td>1 Mb</td>
</tr>
<tr>
<td>6</td>
<td>$10^6$</td>
<td>11 sec</td>
<td>111 Mb</td>
</tr>
<tr>
<td>8</td>
<td>$10^8$</td>
<td>18 min</td>
<td>11 Gb</td>
</tr>
<tr>
<td>10</td>
<td>$10^{10}$</td>
<td>31 hours</td>
<td>1 Tb</td>
</tr>
<tr>
<td>12</td>
<td>$10^{12}$</td>
<td>128 days</td>
<td>111 Tb</td>
</tr>
<tr>
<td>14</td>
<td>$10^{14}$</td>
<td>35 yrs</td>
<td>11 Pb</td>
</tr>
</tbody>
</table>
Breadth-first uses $b^d$ space

but complete and admissible

Depth-first complete only if $\text{limit} > d$, not admissible

but $bd$ space

How can we get the best of both?
A Clever Algorithm
for $d = 1$ to $\infty$ do
    depth-first search to level $d$
    if it succeeds
        then return solution

Could this possibly be efficient?
Assume branching factor $b$ and solution at depth $d$.

**Completeness:**

**Time:**

**Space:**

**Admissibility:**
### Nodes Generated by IDS

#### $b = 2$

<table>
<thead>
<tr>
<th>$d$</th>
<th>at $d$</th>
<th>in prev.</th>
<th>total</th>
<th>IDS</th>
<th>% of opt.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>100.0</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>133.3</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>3</td>
<td>7</td>
<td>11</td>
<td>157.1</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>15</td>
<td>31</td>
<td>57</td>
<td>183.9</td>
</tr>
</tbody>
</table>

#### $b = 10$

<table>
<thead>
<tr>
<th>$d$</th>
<th>at $d$</th>
<th>in prev.</th>
<th>total</th>
<th>IDS</th>
<th>% of opt.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>100.0</td>
</tr>
<tr>
<td>1</td>
<td>10</td>
<td>1</td>
<td>11</td>
<td>12</td>
<td>109.1</td>
</tr>
<tr>
<td>2</td>
<td>100</td>
<td>11</td>
<td>111</td>
<td>123</td>
<td>110.8</td>
</tr>
<tr>
<td>4</td>
<td>10000</td>
<td>1111</td>
<td>11,111</td>
<td>12,345</td>
<td>111.1</td>
</tr>
</tbody>
</table>
Nodes Generated by IDS

\[
b^d + 2b^{d-1} + 3b^{d-2} + \ldots + (d - 1)b^2 + db \\
\approx b^d \left( \frac{b}{b - 1} \right)^2
\]
EOLQs
Please write down the most pressing question you have about the course material covered so far and put it in the box on your way out.

*Thanks!*