GAME DEVELOPMENT

CSC 3150

Instructor: Dr. Edward Kim
A graph $G(V,E)$ consists of a set of **vertices** $V$ (also called nodes) and a set of **edges** $E$ connecting these vertices.
A simple graph $G(V,E)$ is a graph which contains no multi-edges and no loops.
Path
G contains only edges that can be consecutively traversed
- Node degree $\text{deg}(x)$  The number of edges connected to this node
A directed graph (digraph) is a graph that discerns between the edges $A \rightarrow B$ and $A \rightarrow B$. 
**Additional Terminology**

*Independent Set*
G contains no edges

*Clique*
G contains all possible edges
Tree
G contains no cycles

Network
G contains cycles
Binary Tree

Contains no nodes, or
Is comprised of three disjoint sets of nodes:
  a root node,
  a binary tree called its left subtree, and
  a binary tree called its right subtree
Solving by search

• Search is a goal-based agent

• Uninformed search algorithms
  • Given just the problem formulation

• Informed search algorithms
  • Given some guidance
Route finding
Definition of a problem

• Initial State
• Actions (s) -> {a1, a2, a3, …}
• Result (s,a) -> s’ (transition model)
• Goaltest(s) -> T | F
• Pathcost(s (a) -> s (a) -> s) -> n
• Stepcost(s,a,s’) -> n
Practice once…

Start State

Goal State

states??
actions??
goal test??
path cost??
**states**: integer locations of tiles (ignore intermediate positions)
**actions**: move blank left, right, up, down (ignore unjamming etc.)
**goal test**: = goal state (given)
**path cost**: 1 per move
Search strategies

A strategy is defined by picking the **order of node expansion**

Strategies are evaluated along the following dimensions:
- **completeness**—does it always find a solution if one exists?
- **time complexity**—number of nodes generated/expanded
- **space complexity**—maximum number of nodes in memory
- **optimality**—does it always find a least-cost solution?

Time and space complexity are measured in terms of:
- $b$—maximum branching factor of the search tree
- $d$—depth of the least-cost solution
- $m$—maximum depth of the state space (may be $\infty$)
Breadth first search

- Expand the shallowest unexpanded node
Breadth first search

- Expand the shallowest unexpanded node
Breadth first search

- Expand the shallowest unexpanded node
Breadth first search

- Expand the shallowest unexpanded node
Route finding
Uniform cost search

- Find the pass with cheapest total cost...
Route finding
Depth First search

• Expand the deepest unexpanded node
Depth First search

- Expand the deepest unexpanded node
Depth First search

- Expand the deepest unexpanded node
Depth First search

• Expand the deepest unexpanded node
Depth First search

- Expand the deepest unexpanded node
Depth First search

- Expand the deepest unexpanded node
Depth First search

- Expand the deepest unexpanded node
Depth First search

- Expand the deepest unexpanded node
Depth First search

- Expand the deepest unexpanded node
Depth First search

- Expand the deepest unexpanded node
Depth First search

• Expand the deepest unexpanded node
Route finding
Comparisons of algorithms

<table>
<thead>
<tr>
<th></th>
<th>Breadth First</th>
<th>Uniform Cost</th>
<th>Depth First</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complete?</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Optimal?</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Space</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Comparisons of algorithms

<table>
<thead>
<tr>
<th>Complete?</th>
<th>Optimal?</th>
<th>Time</th>
<th>Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>Breadth First</td>
<td>Yes</td>
<td>$O(b^d)$</td>
<td>$O(b^d)$</td>
</tr>
<tr>
<td>Uniform Cost</td>
<td>Yes</td>
<td>$O(b^{1+[C/e]})$</td>
<td>$O(b^{1+[C/e]})$</td>
</tr>
<tr>
<td>Depth First</td>
<td>No</td>
<td>$O(b^m)$</td>
<td>$O(bm)$</td>
</tr>
</tbody>
</table>
Figure 3.22  Values of $h_{SLD}$—straight-line distances to Bucharest.