Lecture 4: Heuristic Search
Reading: AIAMA 3.5-3.6

Today’s Schedule:
- Review of uniform-cost search
- Best-first search
- A* (Astar) search
- Heuristic functions
Warmup #1

Consider the following graph with initial node A and goal D.

1. What is the depth of the path A-B-D?
2. What is the path A-B-D cost?
3. Is the path optimal?
Uniform-Cost Search

function UNIFORM-COST-SEARCH(problem) returns a solution, or failure

node ← a node with STATE = problem.INITIAL-STATE, PATH-COST = 0
frontier ← a priority queue ordered by PATH-COST, with node as the only element
explored ← an empty set
loop do
  if EMPTY?(frontier) then return failure
  node ← POP(frontier) /* chooses the lowest-cost node in frontier */
  if problem.GOAL-TEST(node.STATE) then return SOLUTION(node)
  add node.STATE to explored
  for each action in problem.ACTIONS(node.STATE) do
    child ← CHILD-NODE(problem, node, action)
    if child.STATE is not in explored or frontier then
      frontier ← INSERT(child, frontier)
    else if child.STATE is in frontier with higher PATH-COST then
      replace that frontier node with child
Exercise: perform uniform-cost search

function \text{UNIFORM-COST-SEARCH}(\text{problem}) \text{ returns} a solution, or failure

\begin{itemize}
  \item \text{node} \leftarrow \text{a node with } \text{STATE} = \text{problem.INITIAL-STATE}, \text{PATH-COST} = 0
  \item \text{frontier} \leftarrow \text{a priority queue ordered by PATH-COST, with } \text{node} \text{ as the only element}
  \item \text{explored} \leftarrow \text{an empty set}
\end{itemize}

loop do
  \begin{itemize}
  \item \textbf{if} \text{EMPTY?}(\text{frontier}) \textbf{then} \textbf{return} failure
  \item \text{node} \leftarrow \text{POP}(\text{frontier}) \quad /* \text{chooses the lowest-cost node in } \text{frontier} */
  \item \textbf{if} \text{problem.GOAL-TEST}(\text{node.STATE}) \textbf{then} \textbf{return} \text{SOLUTION}(\text{node})
  \item \text{add } \text{node.STATE} \text{ to } \text{explored}
  \item \textbf{for each } \text{action} \text{ in } \text{problem.ACTIONS}(\text{node.STATE}) \textbf{do}
    \begin{itemize}
    \item \text{child} \leftarrow \text{CHILD-NODE}(\text{problem}, \text{node}, \text{action})
    \item \textbf{if} \text{child.STATE} \text{ is not in } \text{explored} \text{ or } \text{frontier} \textbf{then}
      \begin{itemize}
      \item \text{frontier} \leftarrow \text{INSERT}(\text{child}, \text{frontier})
      \end{itemize}
    \item \textbf{else if} \text{child.STATE} \text{ is in } \text{frontier} \text{ with higher PATH-COST} \textbf{then}
      \begin{itemize}
      \item replace that \text{frontier} node with \text{child}
      \end{itemize}
    \end{itemize}
  \end{itemize}
\end{itemize}

Informed search uses problem specific information to select which nodes to expand.

- Uniform-Cost search uses the path-cost, which we will denote \( g(n) \), to order nodes, \( n \), in the frontier priority queue.

- **Best-first search** is the same as uniform-cost search, but with a general priority given by the evaluation function, \( f(n) \).

The evaluation function is usually the sum of two terms

\[
f(n) = g(n) + h(n)
\]

- \( g(n) \) as before is the path cost for node \( n \), the sum of the cost from the root to the node along its path (following parents back).

- \( h(n) \) is a *heuristic function*, which is an estimate of the path cost from \( n \) to the goal, and is **problem-dependent**.

Note that \( h \) must be

- non-negative

- evaluate to zero at the goal, i.e. \( h(\text{goal}) = 0 \)
Variations of Best-First Search

Best-first search is uniform-cost search with priority \( f(n) \). When

- \( f(n) = g(n) \), this is uniform-cost
- \( f(n) = h(n) \), this is greedy (best-first) search
- \( f(n) = g(n) + h(n) \), this is A* (A-star) search
Warmup #2

Suppose the graph in Question 1 is augmented with the following heuristic function

<table>
<thead>
<tr>
<th>Node</th>
<th>h(n)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>5</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>0</td>
</tr>
</tbody>
</table>

In what order would nodes be expanded using greedy best-first search?
Exercise

<table>
<thead>
<tr>
<th>Node</th>
<th>n</th>
<th>h(n)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

What would be the contents of the frontier and explored set during A* search?

Denote frontier entries as: (node label, parent, g, f)
Optimality of A*

A* graph search is optimal if the heuristic, $h(n)$, is

1. admissible, it never *overestimates* the distance to the goal, i.e. it is optimistic, and

2. consistent (monotonic), $h(n) \leq c(n, a, n') + h(n')$

Where

- $n'$ is a successor of $n$ generated by action $a$
- $h(n)$ is the heuristic value at $n$
- $h(n')$ is the heuristic value at $n'$
- $c(n, a, n')$ is the step cost from $n$ to $n'$ via $a$
Some remarks about A*

- At any stage of the search the union of the frontier and explored set define an iso-contour of the state space that corresponds to the current maximum of $f$.
- The term *pruning* is used to denote that the effect of the heuristic is to prune from consideration portions of the state space that are unlikely to lead to a solution.
- A* works for inconsistent heuristics, but you have to update the priority of a node when expanded if it is already in the frontier.
- The heuristic $h^*(n)$ is the (hypothetical) *oracle* and gives the exact minimal cost from $n$ to the goal.
- A* is optimally efficient, no other algorithm (using the same information) expands fewer nodes in the search.
- The main limitation of A* is memory, unless you have very short time constraints.
Warmup #3

Consider the 8-puzzle problem with the following initial and goal nodes:

<table>
<thead>
<tr>
<th>Initial</th>
<th>Goal</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 6 2</td>
<td>0 1 2</td>
</tr>
<tr>
<td>7 1 0</td>
<td>3 4 5</td>
</tr>
<tr>
<td>3 8 5</td>
<td>6 7 8</td>
</tr>
</tbody>
</table>

and the number-of-tiles-out-of-place heuristic.

What is the value of $f$ for each child of the initial node in A* search?
Heuristic Functions

So, to summarize, a heuristic function takes a node and returns a number that is

- an estimate of the cost to reach the goal from that node
- is always positive, unless
- is zero at the goal
- ideally is admissible (optimistic) and consistent
- the closer to the oracle $h^*$ the better
Exercise

Consider the following graph with initial node A and goal F, with the step-costs and heuristic values indicated.

<table>
<thead>
<tr>
<th>Node n</th>
<th>h(n)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>3</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>1</td>
</tr>
<tr>
<td>F</td>
<td>0</td>
</tr>
</tbody>
</table>

1. What are the contents of the frontier and explored set during A* search?
2. Is this heuristic admissible?
Next Actions

► Reading on Two-Player Games and min-max
  AIAMA 5.1 and 5.2
► Take warmup before noon on Monday 2/3.

Reminder! PS0 is due Monday at 8am via scholar.