Problem Solving and Search

Andrea Danyluk
February 10, 2017

Announcements

• Programming Assignment 1: Search
  – Posted online

Today’s Lecture

• Informed (Heuristic) search
  – Greedy best-first
  – A*

• Will talk a bit more about heuristics on Monday

Uniform Cost Search

- Fringe is a Priority Queue
- Priority = cost so far

Evaluating Uniform Cost Search

- Complete?
  – Yes (if $b$ is finite and step cost $\geq \varepsilon$ for positive $\varepsilon$)
- Optimal?
  – Yes
- Time Complexity?
  – $O(b^{1+C*/\varepsilon})$ ← Can’t check for goal until coming out of PQ!
- Space Complexity?
  – $O(b^{1+C*/\varepsilon})$

Generalized Search

Function $S$\begin{algorithmic}
  \Function \text{Solve}\ $(\text{problem, frontier})$
  \State \text{return solution, or failure}
  \State \text{expanded} \leftarrow \text{an empty set}
  \State \text{frontier} \leftarrow \text{Insert}(\text{Make-Node}(\text{Initial-State}[\text{problem}]), \text{frontier})
  \Loop\While{true}
    \If{\text{frontier} \text{ is empty}}
      \State \text{return failure}
    \EndIf
    \State \text{node} \leftarrow \text{Remove}(\text{frontier})
    \If{\text{Goal-Test}(\text{problem, State}[\text{node}])}
      \State \text{return node}
    \ElseIf{\text{State}(\text{node}) \text{ is not in expanded}}
      \State \text{add State}(\text{node}) \text{ to expanded}
      \State \text{frontier} \leftarrow \text{InsertAll}(\text{Expand}(\text{node}, \text{problem}), \text{frontier})
    \EndIf
  \EndLoop
\EndFunction
\end{algorithmic}
So we’re done, right?
Our search spaces are big.

Informed (Heuristic) Search Strategies
• Use problem-specific knowledge to find solutions more efficiently

Best-first Search
• Choose a node from the frontier based on its “desirability”
  – Frontier is a priority queue
• Requires a search heuristic
  – Any estimate of how close a state is to a goal
  – Examples:
    • Euclidean distance
    • Manhattan distance
    • For the “rush hour” parking problem?

Greedy Best-first Search
\[ h(n) = \text{estimate of cost from } n \text{ to the closest goal} \]
Expand the node with lowest \( h \) value
- i.e., the node that appears to be closest to the goal

Greedy Search with \( h_{SLD} \)

Problems with Greedy Search?
Can be quite good with a high-quality heuristic, but
• Not optimal
• Time and space complexity: \( O(b^m) \)
Cost-Based Searches

- Uniform Cost Search
  - Expands leaf node on path with lowest cost so far
  - Good: Complete and Optimal
  - Bad: Explores “widely”; doesn’t take into account any info about the goal
- Greedy Search
  - Expands node that appears closest to a goal
  - Can take you (quickly) to the wrong goal

A* Search

- Uniform cost search
  - Orders nodes by \textit{backward} cost $g(n)$
- Greedy search
  - Orders nodes by \textit{forward} cost $h(n)$
- A* search
  - Orders nodes by the sum: $f(n) = g(n) + h(n)$

When should A* terminate?

- Should we perform the goal test when
  - Inserting a node into the priority queue?
  - Removing a node from the priority queue?

When removing from the priority queue

Is A* Search Optimal?

A* Conditions for Optimality

- Tree Search
  - Heuristic must be \textit{admissible}
    - Never overestimates the cost to the goal
A* Graph Search Gone Wrong

A* Conditions for Optimality

- Tree Search
  - Heuristic must be admissible
    - Never overestimates the cost to the goal
- Graph Search
  - Heuristic must be consistent
    - If $n'$ is a successor of $n$ generated by action $a$
      - $h(n) \leq c(n, a, n') + h(n')$
    - If an action has cost $c$, then taking that action can only cause a drop in heuristic of at most $c$