A^* and Branch-and-Bound Search

CPSC 322 Lecture 7

January 22, 2007
Textbook §2.5

Lecture Overview

1. Recap
2. A^* Search
3. Optimality of A^*
4. Optimal Efficiency of A^*

Search with Costs

- Sometimes there are costs associated with arcs.
    - The cost of a path is the sum of the costs of its arcs.
- In this setting we often don't just want to find just any solution
    - Instead, we usually want to find the solution that minimizes cost
- We call a search algorithm which always finds such a solution optimal
- Lowest-Cost-First Search: expand paths from the frontier in order of their costs.

Heuristic Search

- $h(n)$ is an estimate of the cost of the shortest path from node n to a goal node.
- $h(n)$ uses only readily obtainable information (that is easy to compute) about a node.
- Admissible heuristic: $h(n)$ is an underestimate if there is no path from n to a goal that has path length less than $h(n)$.
- How to make a heuristic: generally, drop or relax constraints from the original problem.

Best-First Search

- Best-First search selects a path on the frontier with minimal h-value.
- It treats the frontier as a priority queue ordered by h.
- This is a greedy approach: it always takes the path which appears locally best
- It is neither complete nor optimal.

Lecture Overview

1. Recap
2. A^* Search
3. Optimality of A^*
4. Optimal Efficiency of A^*

A^* Search

- A^* search uses both path cost and heuristic values
  - $cost(p)$ is the cost of the path p.
  - $h(p)$ estimates of the cost from the end of p to a goal.
- Let $f(p) = cost(p) + h(p)$.
  - $f(p)$ estimates the total path cost of going from a start node to a goal via p.

$$\underbrace{\underbrace{start \xrightarrow{\text{path } p} n}_{cost(p)} \underbrace{\xrightarrow{\text{estimate}} goal}_{h(p)}}_{f(p)}$$

A^* Search Algorithm

- A^* is a mix of lowest-cost-first and Best-First search.
- It treats the frontier as a priority queue ordered by $f(p)$.
- It always selects the node on the frontier with the lowest estimated total distance.

Analysis of A^*

Let's assume that arc costs are strictly positive.

- Completeness: yes.
- Time complexity: $O(b^m)$
  - the heuristic could be completely uninformative and the edge costs could all be the same, meaning that A^* does the same thing as BFS
- Space complexity: $O(b^m)$
  - like BFS, A^* maintains a frontier which grows with the size of the tree
- Optimality: yes.

Lecture Overview

Optimality[1] of A^*

If A^* returns a solution, that solution is guaranteed to be optimal, as long as

- the branching factor is finite
- arc costs are non-negative
- $h(n)$ is an underestimate of the length of the shortest path from n to a goal node.

[1] Some literature, and the textbook, uses the word "admissiblity" here.

Why is A^* optimal?

Theorem

If A^* selects a path p, p is the shortest (i.e., lowest-cost) path.

- Assume for contradiction that some other path p' is actually the shortest path to a goal
- Consider the moment just before p is chosen from the frontier. Some part of path p' will also be on the frontier; let's call this partial path p''.
- Because p was expanded before p'', $f(p) \leq f(p'')$.
- Because p is a goal, $h(p) = 0$. Thus $cost(p) \leq cost(p'') + h(p'')$.
- Because h is admissible, $cost(p'') + h(p'') \leq cost(p')$ for any path p' to a goal that extends p''
- Thus $cost(p) \leq cost(p')$ for any other path p' to a goal. This contradicts our assumption that p' is the shortest path.

Lecture Overview

1. Recap
2. A^* Search
3. Optimality of A^*
4. Optimal Efficiency of A^*

Optimal Efficiency of A^*

- In fact, we can prove something even stronger about A^*: in a sense (given the particular heuristic that is available) no search algorithm could do better!

- Optimal Efficiency: Among all optimal algorithms that start from the same start node and use the same heuristic h, A^* expands the minimal number of nodes.
  - problem: A^* could be unlucky about how it breaks ties.
  - So let's define optimal efficiency as expanding the minimal number of nodes n for which $f(n) \neq f^*$, where f^* is the cost of the shortest path.

Why is A^* optimally efficient?

Theorem

A^* is optimally efficient.

- Let f^* be the cost of the shortest path to a goal. Consider any algorithm A' which has the same start node as A^*, uses the same heuristic and fails to expand some node n' expanded by A^* for which $cost(n') + h(n') < f^*$. Assume that A' is optimal.
- Consider a different search problem which is identical to the original and on which h returns the same estimate for each node, except that n' has a child node n'' which is a goal node, and the true cost of the path to n'' is $f(n')$.
  - that is, the edge from n' to n'' has a cost of $h(n')$: the heuristic is exactly right about the cost of getting from n' to a goal.
- A' would behave identically on this new problem.
  - The only difference between the new problem and the original problem is beyond node n', which A' does not expand.
- Cost of the path to n'' is lower than cost of the path found by A'.
- This violates our assumption that A' is optimal.