Search: A^*

CPSC 322 – Search 5

Textbook §3.6

Lecture Overview

1. Recap
2. A^* Search
3. Optimality 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

Heuristic Search

Definition (search heuristic)

A search heuristic $h(n)$ is an estimate of the cost of the shortest path from node n to a goal node.

- h can be extended to paths: $h(\langle n_0, \ldots, n_k \rangle) = h(n_k)$
- $h(n)$ uses only readily obtainable information (that is easy to compute) about a node.

Definition (admissible heuristic)

A search heuristic $h(n)$ is admissible if it is never an overestimate of the cost from n to a goal.

- there is never a path from n to a goal that has path length less than $h(n)$.
- another way of saying this: $h(n)$ is a lower bound on the cost of getting from n to the nearest goal.

How to Construct a Heuristic

- Overall, a cost-minimizing search problem is a constrained optimization problem
- A relaxed version of the problem is a version of the problem where one or more constraints have been dropped
- It's usually possible to identify constraints which, when dropped, make the problem extremely easy to solve

Lecture Overview

1. Recap
2. A^* Search
3. Optimality of A^*

A cool example

A* search applied to "infinite Mario":

- http://aigamedev.com/open/interviews/mario-ai/
- http://www.doc.ic.ac.uk/~rb1006/projects:marioai

...Thanks to Phillip Mah!

A^* Search

- A^* search uses both path costs and heuristic values
  - $cost(p)$ is the cost of the path p.
  - $h(p)$ estimates the cost from the end of p to a goal.

A^* Search

- A^* search uses both path costs and heuristic values
  - $cost(p)$ is the cost of the path p.
  - $h(p)$ estimates 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

- A^* search uses both path costs and heuristic values
  - $cost(p)$ is the cost of the path p.
  - $h(p)$ estimates 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^* 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.

A^* Example

http://aispace.org/search/

- simple tree graph
- delivery robot (acyclic) graph

Analysis of A^*

Let's assume that arc costs are strictly positive.

- Completeness:

Analysis of A^*

Let's assume that arc costs are strictly positive.

- Completeness: yes.
- Time complexity:

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:

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:

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 strictly positive
- $h(n)$ is an underestimate of the length of the shortest path from n to a goal node, and is non-negative

[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.