Heuristic Search & A*

CMPUT 366: Intelligent Systems

P&M §3.6

Lecture Outline

1. Recap
2. A* Search
3. Comparing Heuristics

Recap: Heuristics

Definition:
A **heuristic function** is a function $h(n)$ that returns a non-negative estimate of the cost of the cheapest path from n to a goal node.

- e.g., Euclidean distance instead of travelled distance

Definition:
A heuristic function is **admissible** if $h(n)$ is always less than or equal to the cost of the cheapest path from n to a goal node.

- i.e., $h(n)$ is a **lower bound** on cost$(\langle n, \ldots, g \rangle)$ for any **goal node** g

A* Search

- A* search uses **both** path cost information and heuristic information to select paths from the frontier
- Let $f(p) = \text{cost}(p) + h(p)$
  - $f(p)$ **estimates** the total cost to the nearest goal node **starting from** p
- A* removes paths from the frontier with **smallest** $f(p)$
- When h is **admissible**,
  $p^* = \langle s, \ldots, n, \ldots, g \rangle$ is a **solution**, and
  $p = \langle s, \ldots, n \rangle$ is a **prefix** of p^*:
  - $f(p) \leq \text{cost}(p^*)$ (**why?**)

$$\underbrace{\underbrace{\text{start} \xrightarrow{\text{actual}} n}_{\text{cost(p)}} \; \underbrace{\xrightarrow{\text{estimated}} \text{goal}}_{\text{h(n)}}}_{\text{f(p)}}$$

A* Search Algorithm

Input: a *graph*; a set of *start nodes*; a $goal$ function

$frontier$:= $\{\langle s \rangle \mid s$ *is a start node*$\}$
while $frontier$ is not empty:
    select f-minimizing path $\langle n_0, \ldots, n_k \rangle$ from *frontier*
    remove $\langle n_0, \ldots, n_k \rangle$ from $frontier$
    if $goal(n_k)$:
        return $\langle n_0, \ldots, n_k \rangle$
    for each neighbour n of n_k:
        add $\langle n_0, \ldots, n_k, n \rangle$ to *frontier*
end while

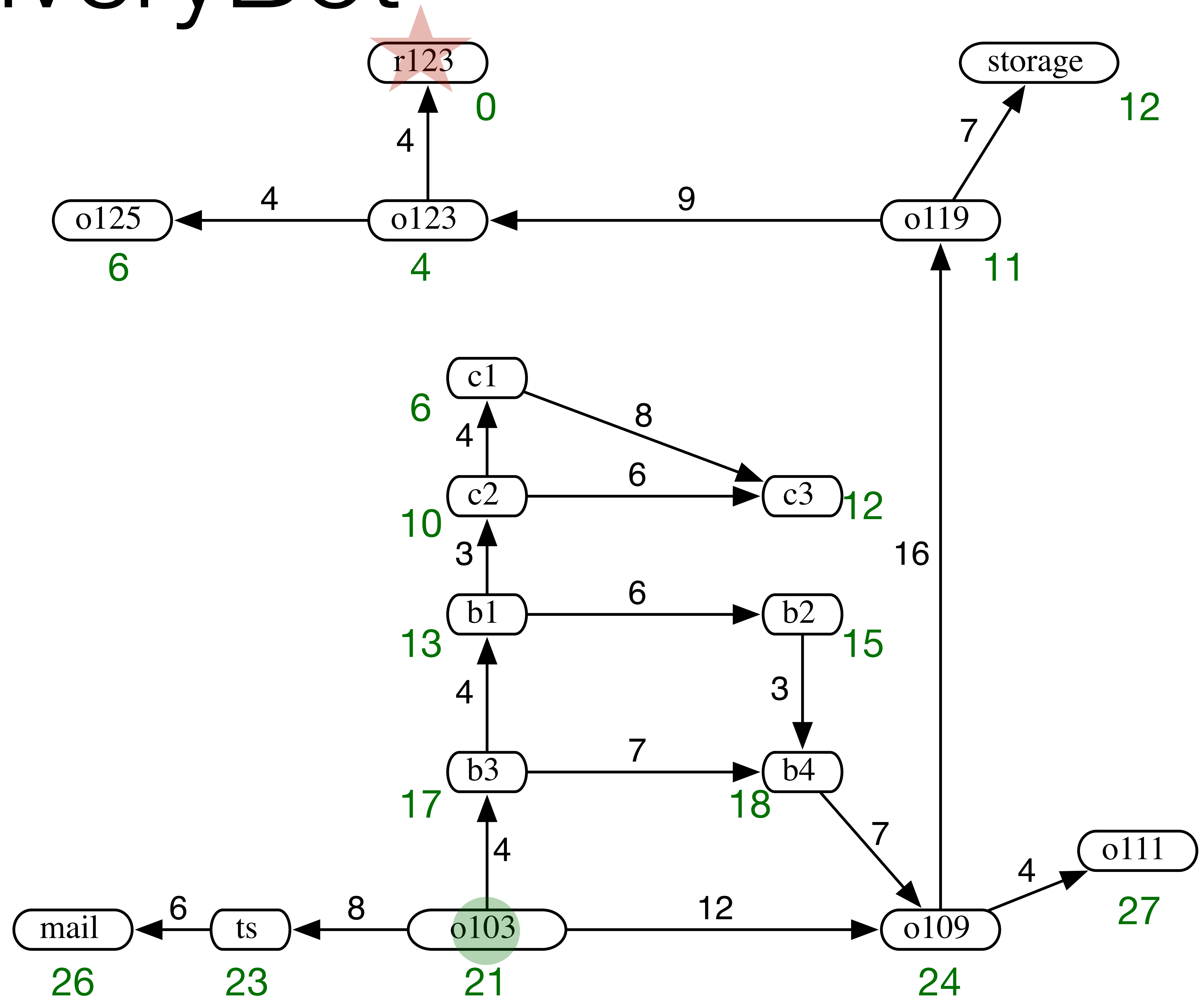
i.e., $f(\langle n_0, \ldots, n_k \rangle) \leq f(p)$ for all other paths $p \in frontier$

Question:

What **data structure** for the frontier implements this search strategy?

A* Search Example: DeliveryBot

- Heuristic: **Euclidean distance**
- **Question:** What is $f(\langle o103, b3 \rangle)$? $f(\langle o103, o109 \rangle)$?
- A* will spend a bit of time exploring paths in the labs before trying to go around via o109
- At that point the heuristic starts helping more
- **Question:** Does breadth-first search explore paths in the lab too?
- **Question:** Does breadth-first search explore any paths that A* does not?



A* Theorem

Theorem:

If there is a solution of finite cost, A* using heuristic function h always returns an **optimal** solution (in **finite time**), if

1. The branching factor is **finite**, and
2. All **arc costs** are greater than some $\epsilon > 0$, and
3. h is an **admissible** heuristic.

Proof:

1. **No suboptimal solution** will be removed from the frontier whenever the frontier contains a **prefix of the optimal solution**
2. The **optimal solution** is guaranteed to be **removed from the frontier** eventually

A* Theorem Proofs: A Lexicon

An **admissible heuristic**: $h(n)$

$f(\langle n_0, \ldots, n_k \rangle) = \text{cost}(\langle n_0, \ldots, n_k \rangle) + h(n_k)$

A **start node**: s

A **goal node**: z (i.e., goal$(z) = 1$)

The **optimal solution**: $p^* = \langle s, \ldots, a, b, \ldots z \rangle$

A **prefix** of the optimal solution: $p' = \langle s, \ldots, a \rangle$

A **suboptimal solution**: $g = \langle s, q, \ldots, z \rangle$

A* Theorem: Optimality

An **admissible heuristic**: $h(n)$

$f(\langle n_0, \ldots, n_k \rangle) = \text{cost}(\langle n_0, \ldots, n_k \rangle) + h(n_k)$

A **start node**: s

A **goal node**: z (i.e., goal$(z) = 1$)

The **optimal solution**: $p^* = \langle s, \ldots, a, b, \ldots z \rangle$

A **prefix** of the optimal solution: $p' = \langle s, \ldots, a \rangle$

A **suboptimal solution**: $g = \langle s, q, \ldots, z \rangle$

Proof part 1: Optimality (no g is removed before p^*)

1. $f(g) = \text{cost}(g)$ and $f(p^*) = \text{cost}(p^*)$

   (i) $f(\langle n_0, \ldots, n_k \rangle) = \text{cost}(\langle n_0, \ldots, n_k \rangle) + h(n_k)$, and $h(z) = 0$

2. $f(p') < f(g)$

   (i) $f(\langle s, \ldots, a \rangle) = \text{cost}(\langle s, \ldots, a \rangle) + h(a)$

   (ii) $f(\langle s, \ldots, a, b, \ldots, z \rangle) = \text{cost}(\langle s, \ldots, a, b, \ldots, z \rangle) + h(z) = \text{cost}(\langle s, \ldots, a \rangle) + \text{cost}(a, b, \ldots, z \rangle)$

   (iii) $h(a) \leq \text{cost}(\langle a, b, \ldots, z \rangle)$

   (iv) $f(p') \leq f(p^*) < f(g)$ ∎

A* Theorem: Completeness

An **admissible heuristic**: $h(n)$

$f(\langle n_0, \ldots, n_k \rangle) = \text{cost}(\langle n_0, \ldots, n_k \rangle) + h(n_k)$

A **start node**: s

A **goal node**: z (i.e., $\text{goal}(z) = 1$)

The **optimal solution**: $p^* = \langle s, \ldots, a, b, \ldots z \rangle$

A **prefix** of the optimal solution: $p' = \langle s, \ldots, a \rangle$

A **suboptimal solution**: $g = \langle s, q, \ldots, z \rangle$

Proof part 2: A* is **complete**

- Every path that is removed from the frontier is only replaced by more-costly paths (**why?**)
- Since individual arc costs are larger than ϵ, every path in the frontier will eventually have cost larger than k, for any finite k
    - Every path with at least $\dfrac{k}{\epsilon}$ arcs will have cost larger than k
- So every path in the frontier will eventually have cost larger than the cost of the optimal solution
- So the optimal solution will eventually be removed from the frontier
- **Question:** Why are we talking about **costs** and not **f-values**?

Comparing Heuristics

- Suppose that we have two **admissible** heuristics, h_1 and h_2
- Suppose that for every node n, $h_2(n) \geq h_1(n)$

Question: Which heuristic is better for search?

Dominating Heuristics

Definition:
A heuristic h_2 **dominates** a heuristic h_1 if

1. $\forall n : h_2(n) \geq h_1(n)$, and
2. $\exists n : h_2(n) > h_1(n)$.

Theorem:
If h_2 dominates h_1, and both heuristics are admissible, then A* using h_2 will never remove more paths from the frontier than A* using h_1.

- i.e., better heuristics remove weakly fewer paths

Question:
Which admissible heuristic dominates **all other** admissible heuristics?

A* Analysis

For a search graph with *finite* maximum branch factor b and *finite* maximum path length m...

1. What is the worst-case **space complexity** of A*?
   [A: $O(m)$] [B: $O(mb)$] [C: $O(b^m)$] [D: it depends]

2. What is the worst-case **time complexity** of A*?
   [A: $O(m)$] [B: $O(mb)$] [C: $O(b^m)$] [D: it depends]

Question: If A* has the same space and time complexity as least cost first search, then what is its advantage?

A* Summary

- **Domain knowledge** can help speed up graph search
- Domain knowledge can be expressed by a **heuristic function**, which **estimates** the cost of a path to the goal from a node
- A* considers both **path cost** and **heuristic cost** when selecting paths:
  $f(p) = \text{cost}(p) + h(p)$
- **Admissible** heuristics guarantee that A* will be **optimal**
- Admissible heuristics can be built from **relaxations** of the original problem
- The more **accurate** the heuristic is, the **fewer** the paths A* will explore