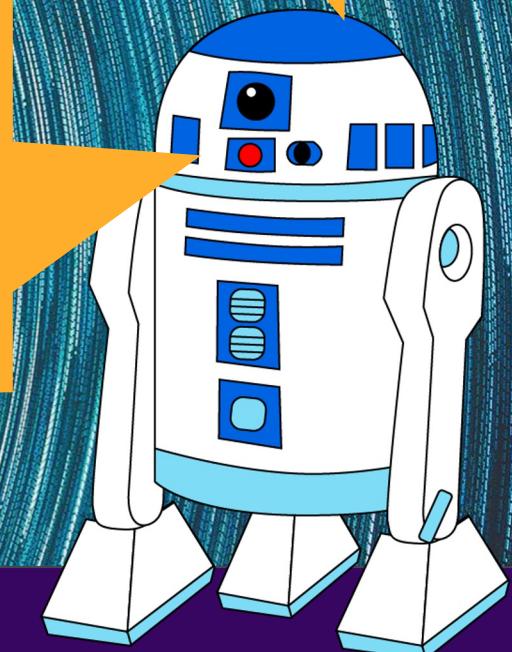


CIS 4210/5210:  
ARTIFICIAL INTELLIGENCE

# Informed Search

**HW2 is due Wednesday by 11:59pm.**

If you need an extension, you must request it 24 hours in advance to avoid a late penalty.



# Review: Search problem definition

*States*: a set  $S$

An *initial state*  $s_i \in S$

*Actions*: a set  $A$

$\forall s \text{ } Actions(s) = \text{the set of actions that can be executed in } s, \text{ that are applicable in } s.$

*Transition Model*:  $\forall s \forall a \in Actions(s) \text{ } Result(s, a) \rightarrow s_r$

$s_r$  is called a *successor* of  $s$

$\{s_i\} \cup Successors(s_i)^*$  = *state space*

*Path cost* (*Performance Measure*): Must be additive

e.g. sum of distances, number of actions executed, ...

$c(x, a, y)$  is the step cost, assumed  $\geq 0$

(where action  $a$  goes from state  $x$  to state  $y$ )

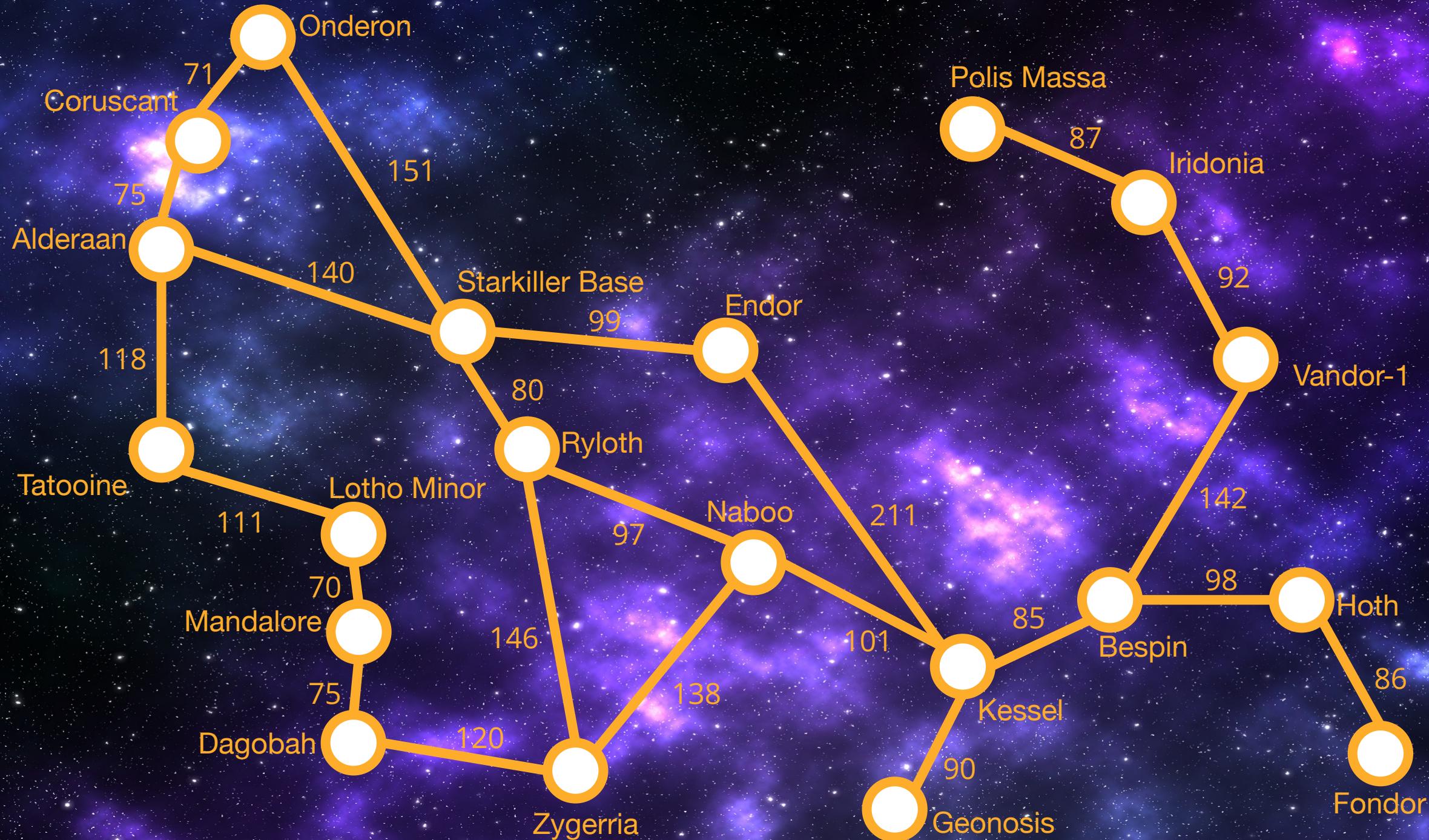
*Goal test*:  $Goal(s)$

Can be implicit, e.g.  $checkmate(s)$

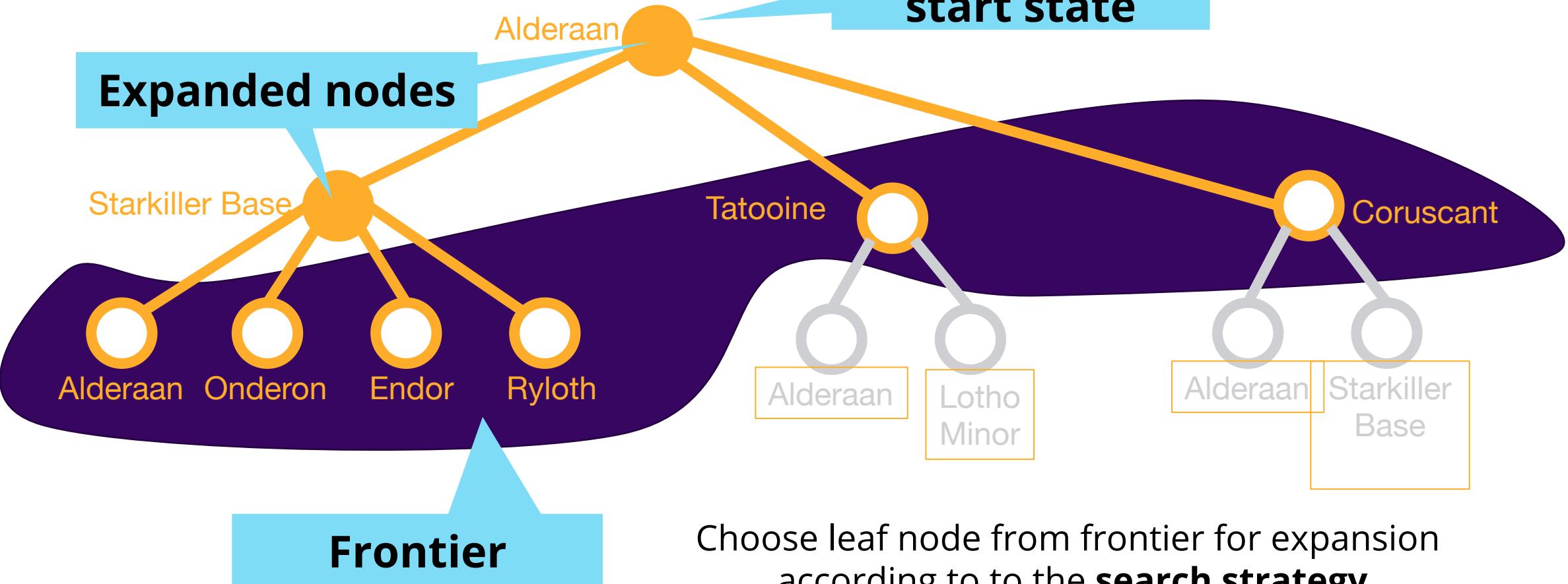
$s$  is a *goal state* if  $Goal(s)$  is true

# Review: Useful Concepts

- *State space*: the set of all states reachable from the initial state by *any* sequence of actions
  - *When several operators can apply to each state, this gets large very quickly*
  - *Might be a proper subset of the set of configurations*
- *Path*: a sequence of actions leading from one state  $s_j$  to another state  $s_k$
- *Frontier*: those states that are available for *expanding* (for applying legal actions to)
- *Solution*: a path from the initial state  $s_i$  to a state  $s_g$  that satisfies the goal test



# Search Tree



# Review: Search Strategies

*Strategy* = order of tree expansion

- Implemented by different **queue structures** (LIFO, FIFO, priority)

Dimensions for evaluation

- *Completeness*- always find the solution?
- *Optimality* - finds a least cost solution (lowest path cost) **first?**
- *Time complexity* - # of nodes generated (*worst case*)
- *Space complexity* - # of nodes simultaneously in memory (*worst case*)

Time/space complexity variables

- $b$ , *maximum branching factor* of search tree
- $d$ , *depth* of the shallowest goal node
- $m$ , maximum length of any path in the state space (potentially  $\infty$ )

Animation of Graph BFS algorithm  
set to music 'flight of bumble bee'

<https://youtu.be/x-VTfcmrLEQ>

Animation of Graph DFS algorithm

Depth First Search of Graph

set to music 'flight of bumble bee'

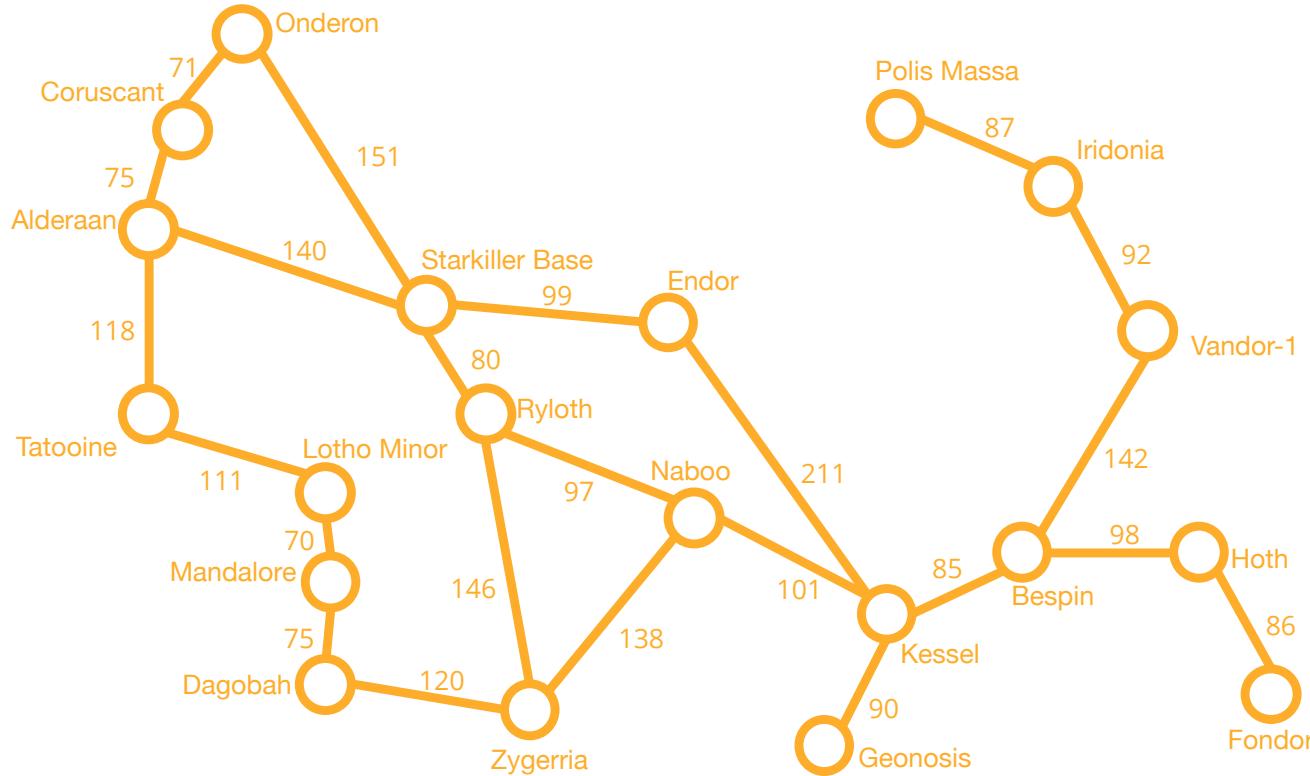
# “Uniform Cost” Search

“In computer science, *uniform-cost* search (UCS) is a tree search algorithm used for traversing or searching a *weighted* tree, tree structure, or graph.” - Wikipedia

# Motivation: Map Navigation Problems

All our search methods so far  
assume *step-cost* = 1

*This is only true for some problems*



# **$g(N)$ : the path cost function**

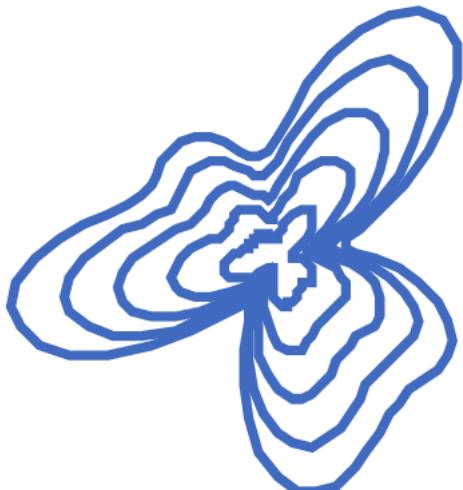
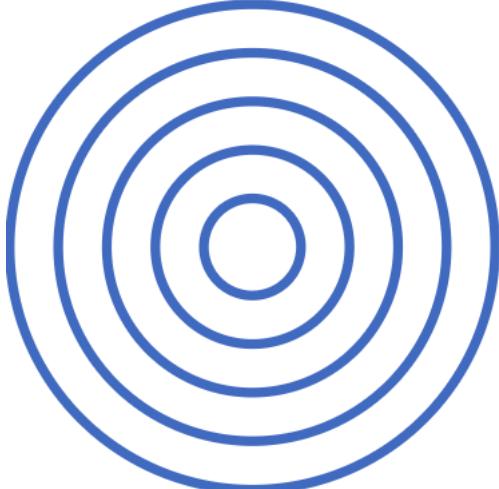
- Our assumption so far: All moves equal in cost
  - Cost = # of nodes in path-1
  - $g(N) = \text{depth}(N)$  in the search tree
- More general: Assigning a (potentially) unique cost to each step
  - $N_0, N_1, N_2, N_3$  = nodes visited on path  $p$  from  $N_0$  to  $N_3$
  - $C(i,j)$ : Cost of going from  $N_i$  to  $N_j$
  - If  $N_0$  the root of the search tree,  
$$g(N_3) = C(0,1) + C(1,2) + C(2,3)$$

# Uniform-cost search (UCS)

- Extension of BF-search:
  - **Expand node with *lowest path cost***
- Implementation:
  - *frontier* = priority queue ordered by  $g(n)$
- Subtle but significant difference from BFS:
  - Tests if a node is a goal state when it is selected for expansion, **not when it is added to the frontier.**
  - Updates a node on the frontier if a better path to the same state is found.
  - So always enqueues a node *before checking whether it is a goal.*

*WHY???*

# Shape of Search



- **Breadth First Search** explores equally in all directions. Its frontier is implemented as a FIFO queue. This results in smooth contours or “plies”.
- **Uniform Cost Search** lets us prioritize which paths to explore. Instead of exploring all possible paths equally, it favors lower cost paths. Its frontier is a priority queue. This results in “cost contours”.

# This may remind you of something...



Which algorithm from your algorithms class does Uniform Cost Search remind you of?

[PollEv.com/ccb](https://PollEv.com/ccb)

# Uniform Cost Search (UCS) vs. Dijkstra's algorithm

## Foundational Concepts:

- Dijkstra's: Shortest path from initial to all vertices in a weighted graph.
- UCS: AI's pathfinding. Shortest path from initial state to a goal state.

## Similarities:

- Use of **Priority Queue** to pick lowest cost node.
- **Greedy Nature**: Always choose optimal local solution.
- Both operate on a **weighted graph**.
- Guarantee **Optimality** for non-negative weights.

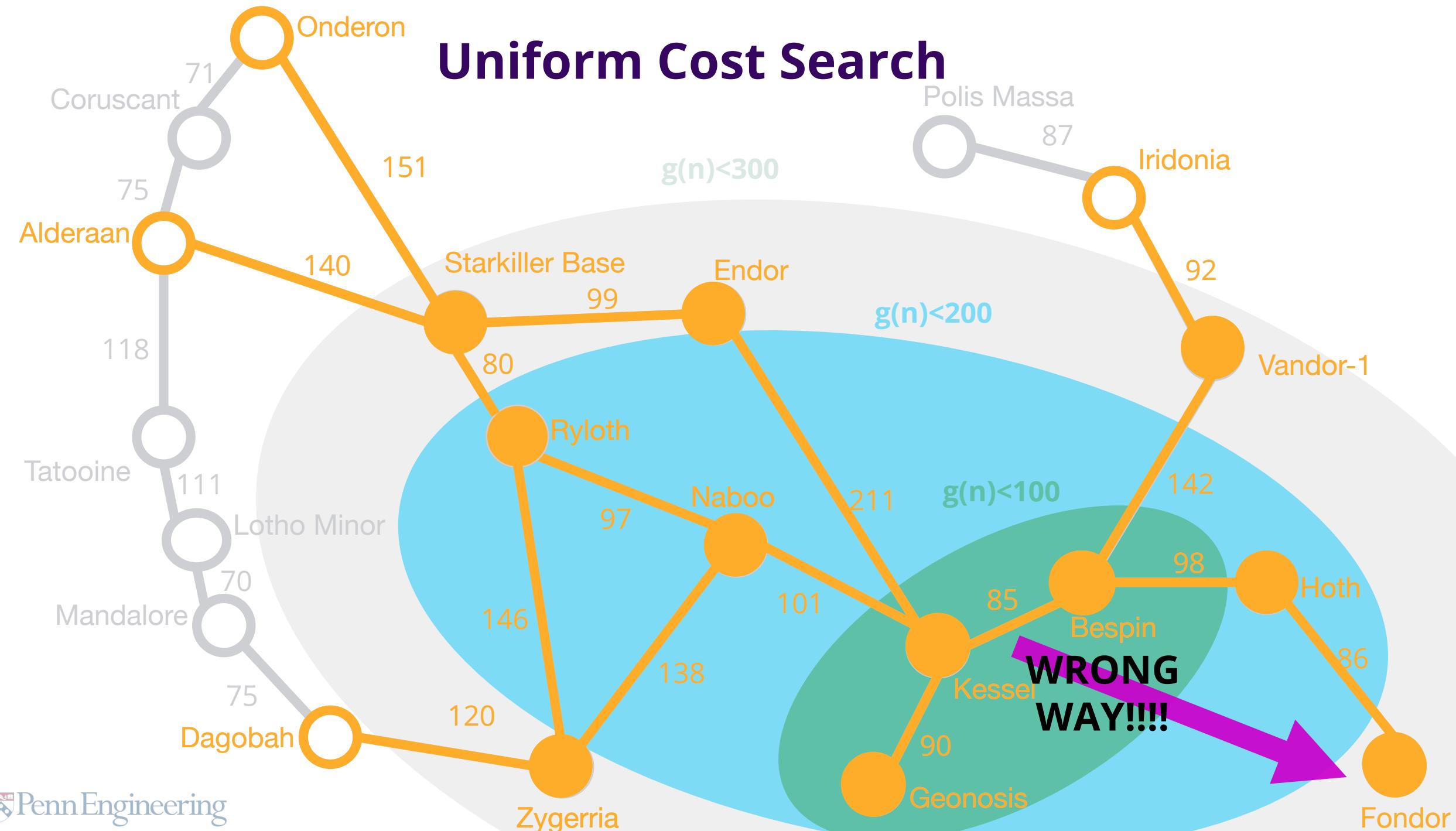
## Differences:

- **Objective**: UCS targets specific goal node; Dijkstra's finds paths to all nodes.
- **Termination**: UCS stops at goal; Dijkstra's stops after all nodes have been reached.

## Relation:

- UCS is a "targeted" Dijkstra's: it seeks a path to a specific node, not all nodes.

# Uniform Cost Search



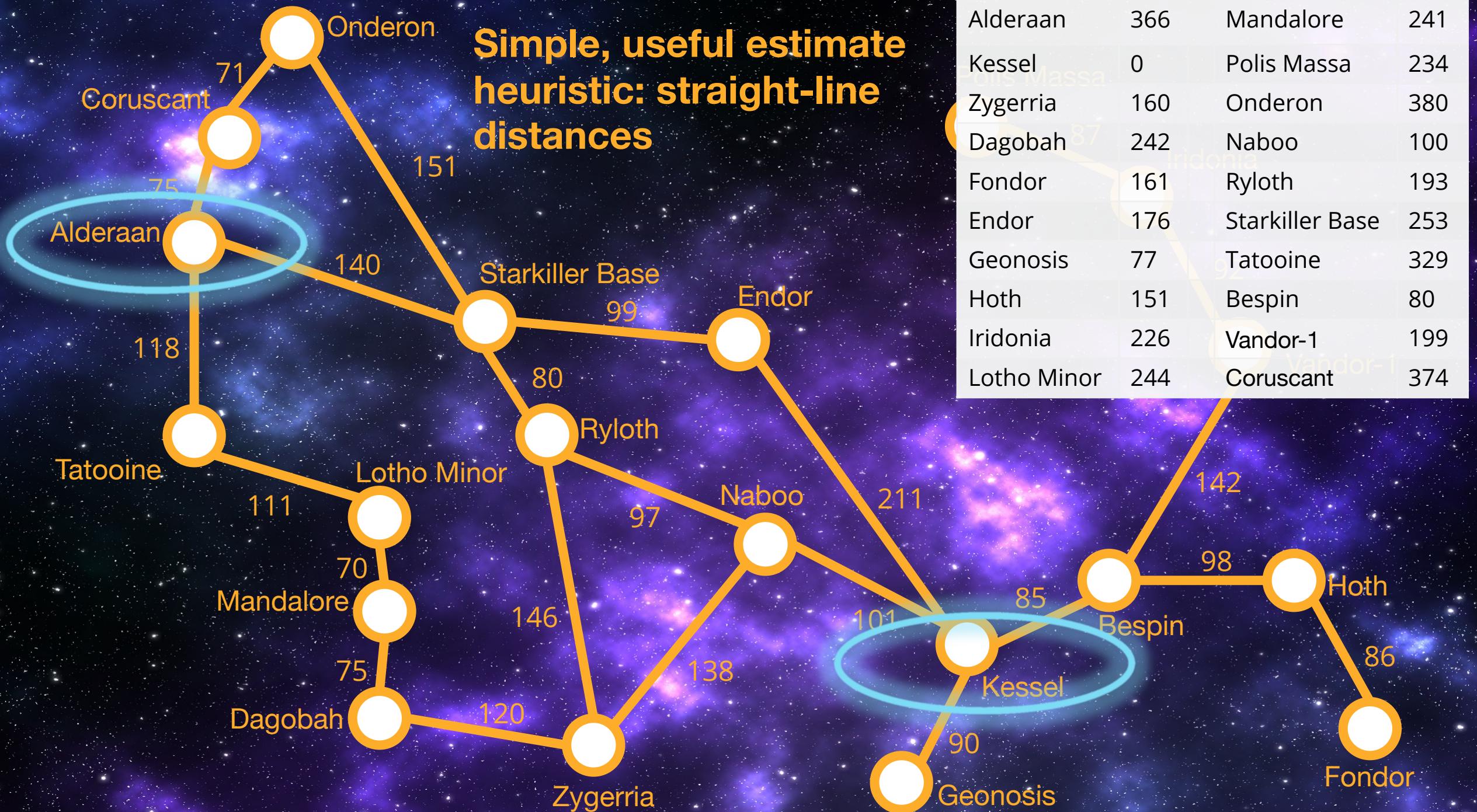
# A better idea than UCS...

- Node expansion based on *an estimate* which *includes distance to the goal*
- General approach of **informed search**:
  - **Best-first search**: node selected for expansion based on an *evaluation function  $f(n)$* 
    - ✓  $f(n)$  includes estimate of distance to goal (*new idea!*)
- Implementation: **Sort our frontier queue** by this **new  $f(n)$** .

Instances of best-first search:

- **greedy search**
- **$A^*$  search**

# Simple, useful estimate heuristic: straight-line distances



# Heuristic (estimate) functions



*Heureka! ---Archimedes*

[dictionary] “*A rule of thumb, simplification, or educated guess that reduces or limits the search for solutions in domains that are difficult and poorly understood.*”

*Heuristic knowledge* is useful, but not necessarily correct.

*Heuristic algorithms* use heuristic knowledge to solve a problem.

A *heuristic function*  $h(n)$  takes a state  $n$  and returns an *estimate* of the distance from  $n$  to the goal.

# Greedy Best-First Search

First attempt at integrating heuristic knowledge

# Best-first search

*Basic idea:*

*Select node for expansion* with minimal *evaluation function  $f(n)$*

- where  $f(n)$  is some function that includes *estimate heuristic  $h(n)$*  of the remaining distance to goal

Implement using priority queue

Exactly UCS with  $f(n)$  replacing  $g(n)$

# Greedy best-first search: $f(n) = h(n)$

Expands the node that *is estimated* to be closest to goal

Completely ignores  $g(n)$ : the cost to get to  $n$

In our Romanian map,  $h(n) = h_{SLD}(n)$  = straight-line distance from  $n$  to Bucharest

In a grid, the heuristic distance can be calculated using the “Manhattan distance”:

---

```
def heuristic(a, b):
    # Manhattan distance on a square grid
    return abs(a.x - b.x) + abs(a.y - b.y)
```

---

# Greedy best-first search

```
frontier = PriorityQueue()
frontier.put(start, 0)
came_from = {}
came_from[start] = None

while not frontier.empty():
    current = frontier.get()

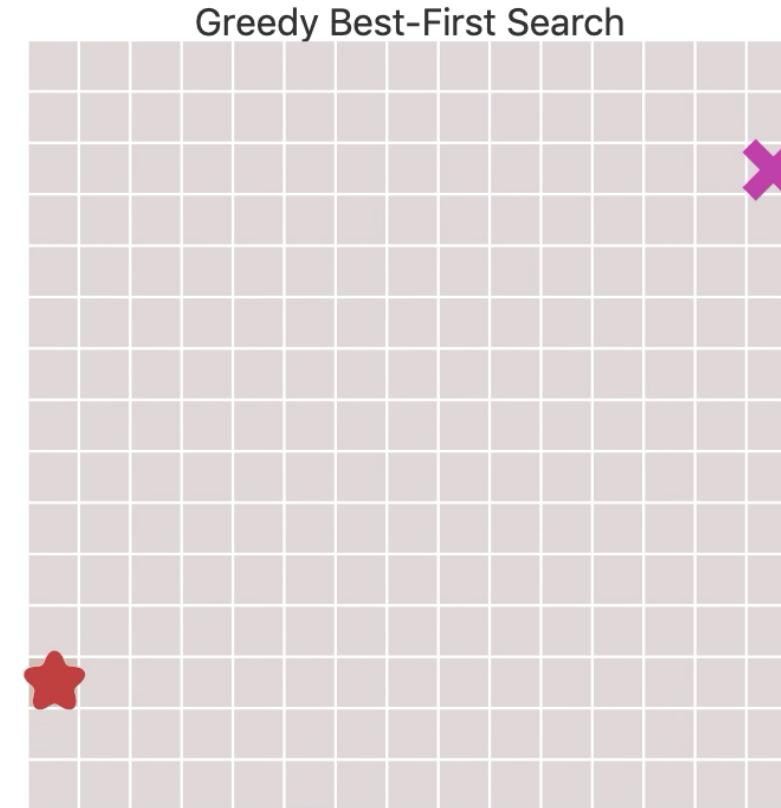
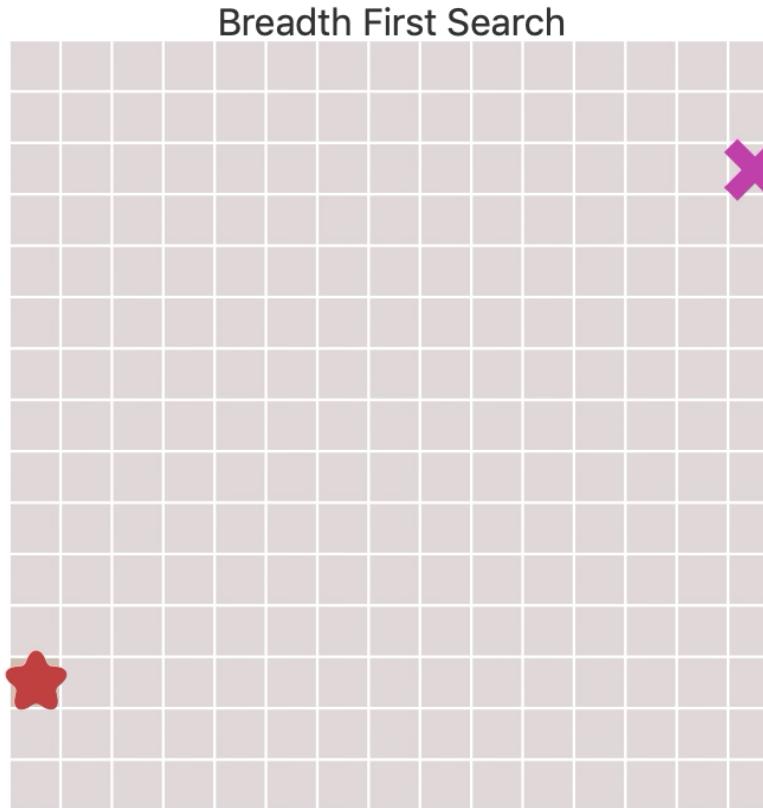
    if current == goal:
        break

    for next in graph.neighbors(current):
        if next not in came_from:
            priority = heuristic(goal, next)
            frontier.put(next, priority)
            came_from[next] = current
```

Code from Amit Patel  
of Red Blob Games

This is not the solution  
to the HW assignment.

# BFS v. Greedy Best-First Search

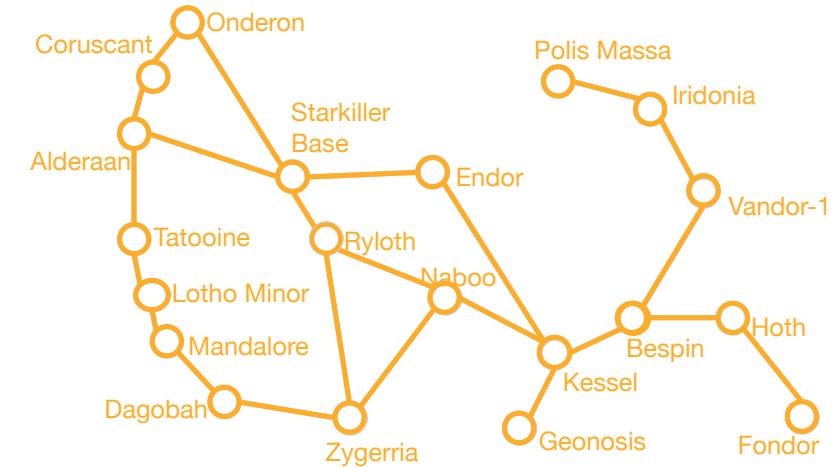


<https://www.redblobgames.com/pathfinding/a-star/introduction.html>

# Greedy best-first search example

Initial State: **Alderaan**  
Goal State: **Kessel**

# Alderaan



## Frontier queue:

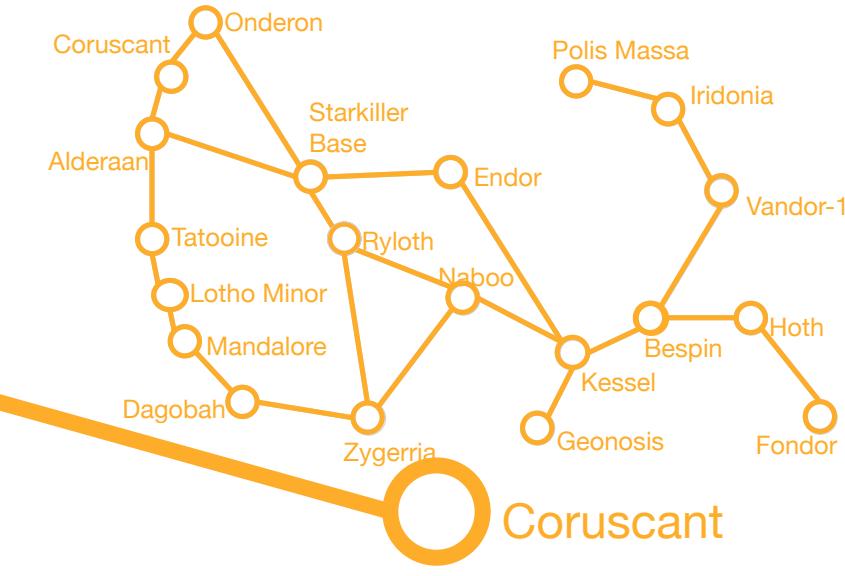
# Alderon 366

| Planet    | SLD | Planet          | SLD |
|-----------|-----|-----------------|-----|
| Alderaan  | 366 | Lotho Minor     | 244 |
| Bespin    | 80  | Mandalore       | 241 |
| Coruscant | 374 | Naboo           | 100 |
| Dagobah   | 242 | Onderon         | 380 |
| Endor     | 176 | Polis Massa     | 234 |
| Fondor    | 161 | Ryloth          | 193 |
| Geonosis  | 77  | Starkiller Base | 253 |
| Hoth      | 151 | Tatooine        | 329 |
| Iridonia  | 226 | Vandor-1        | 199 |
| Kessel    | 0   | Zygerria        | 160 |

# Greedy best-first search example

Initial State: Alderaan

Goal State: Kessel



Frontier queue:

**Starkiller Base** 253

**Tatooine** 329

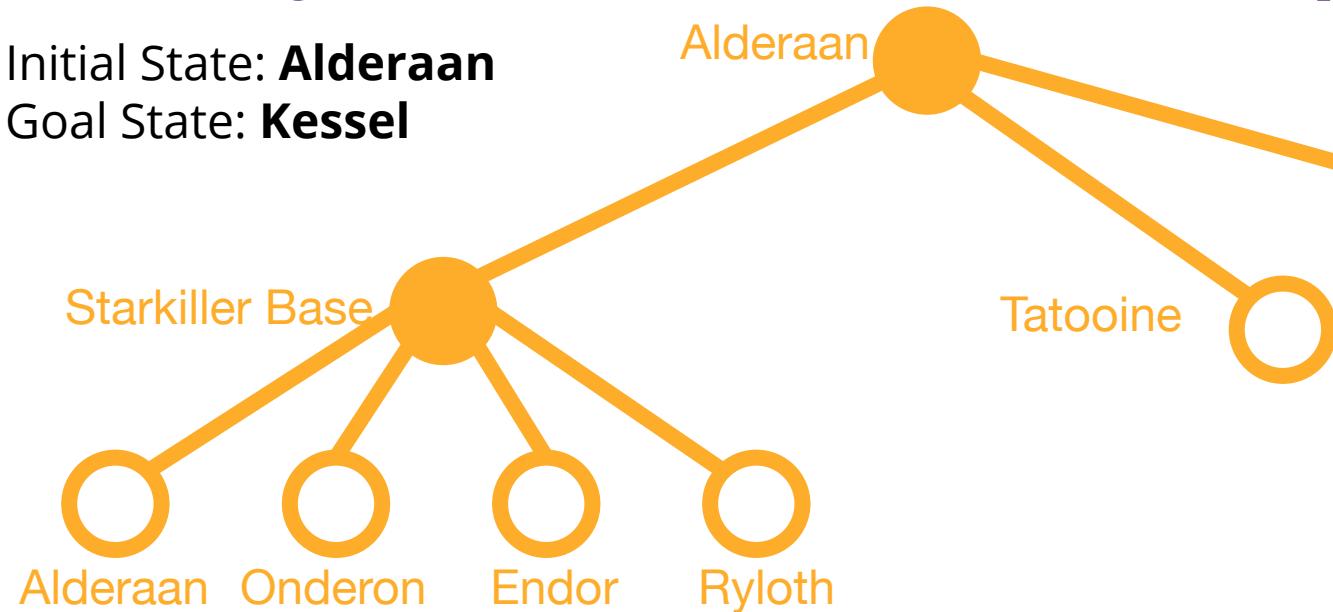
**Coruscant** 374

| Planet    | SLD | Planet          | SLD |
|-----------|-----|-----------------|-----|
| Alderaan  | 366 | Lotho Minor     | 244 |
| Bespin    | 80  | Mandalore       | 241 |
| Coruscant | 374 | Naboo           | 100 |
| Dagobah   | 242 | Onderon         | 380 |
| Endor     | 176 | Polis Massa     | 234 |
| Fondor    | 161 | Ryleth          | 193 |
| Geonosis  | 77  | Starkiller Base | 253 |
| Hoth      | 151 | Tatooine        | 329 |
| Iridonia  | 226 | Vandor-1        | 199 |
| Kessel    | 0   | Zygerria        | 160 |

# Greedy best-first search example

Initial State: Alderaan

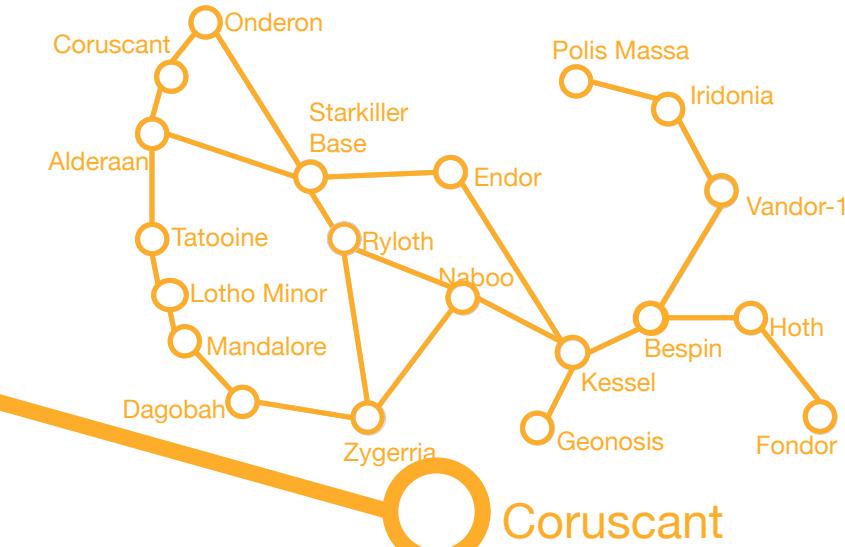
Goal State: Kessel



Are we doing **graph** search or **tree** search?

Frontier queue:

|           |     |
|-----------|-----|
| Tatooine  | 329 |
| Coruscant | 374 |
| Alderaan  | 366 |
| Onderon   | 380 |
| Endor     | 176 |
| RylOTH    | 193 |



| Planet    | SLD | Planet          | SLD |
|-----------|-----|-----------------|-----|
| Alderaan  | 366 | Lothal Minor    | 244 |
| Bespin    | 80  | Mandalore       | 241 |
| Coruscant | 374 | Naboo           | 100 |
| Dagobah   | 242 | Onderon         | 380 |
| Endor     | 176 | Polis Massa     | 234 |
| Fondor    | 161 | Ryloth          | 193 |
| Geonosis  | 77  | Starkiller Base | 253 |
| Hoth      | 151 | Tatooine        | 329 |
| Iridonia  | 226 | Vandor-1        | 199 |
| Kessel    | 0   | Zygerria        | 160 |

# Graph search versus tree search



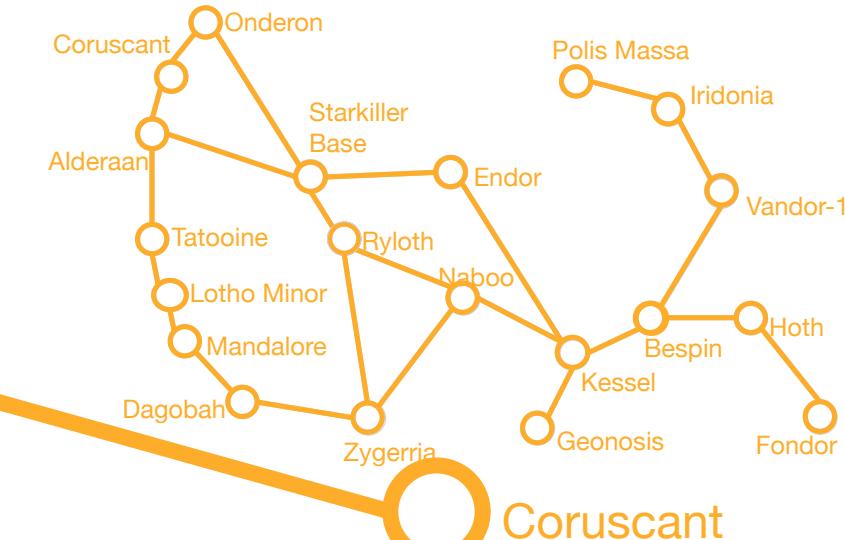
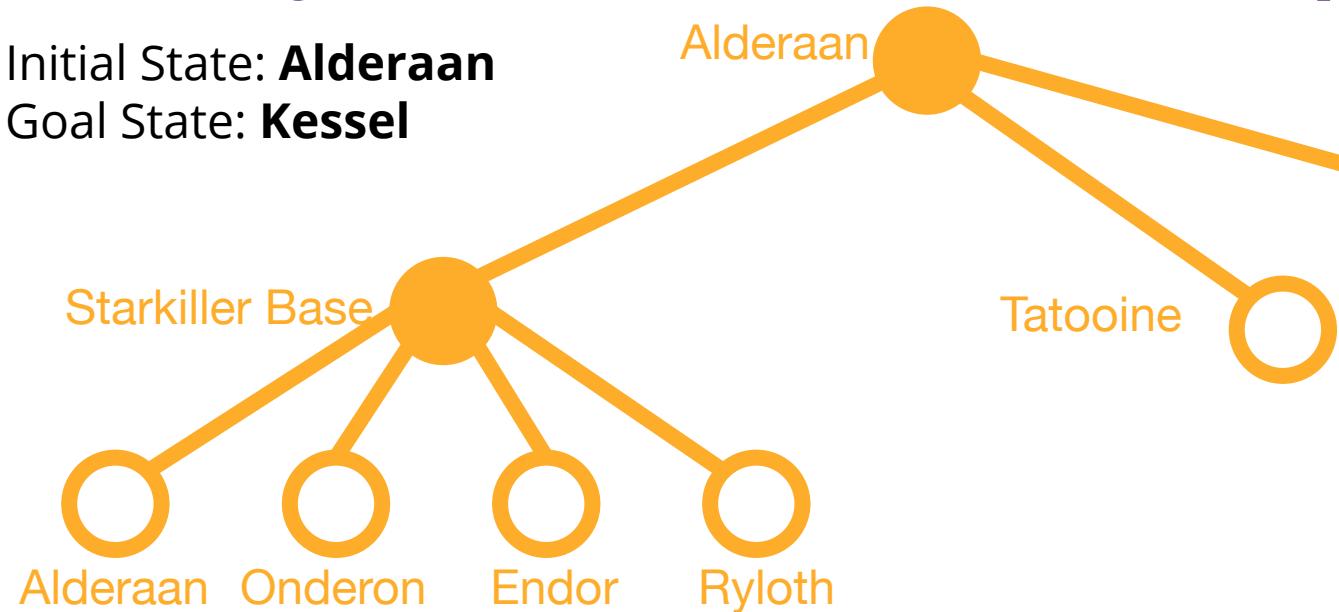
What is the difference between graph search and tree search?

[PollEv.com/ccb](https://PollEv.com/ccb)

# Greedy best-first search example

## Initial State: Alderaan

## Goal State: Kessel



## Frontier queue:

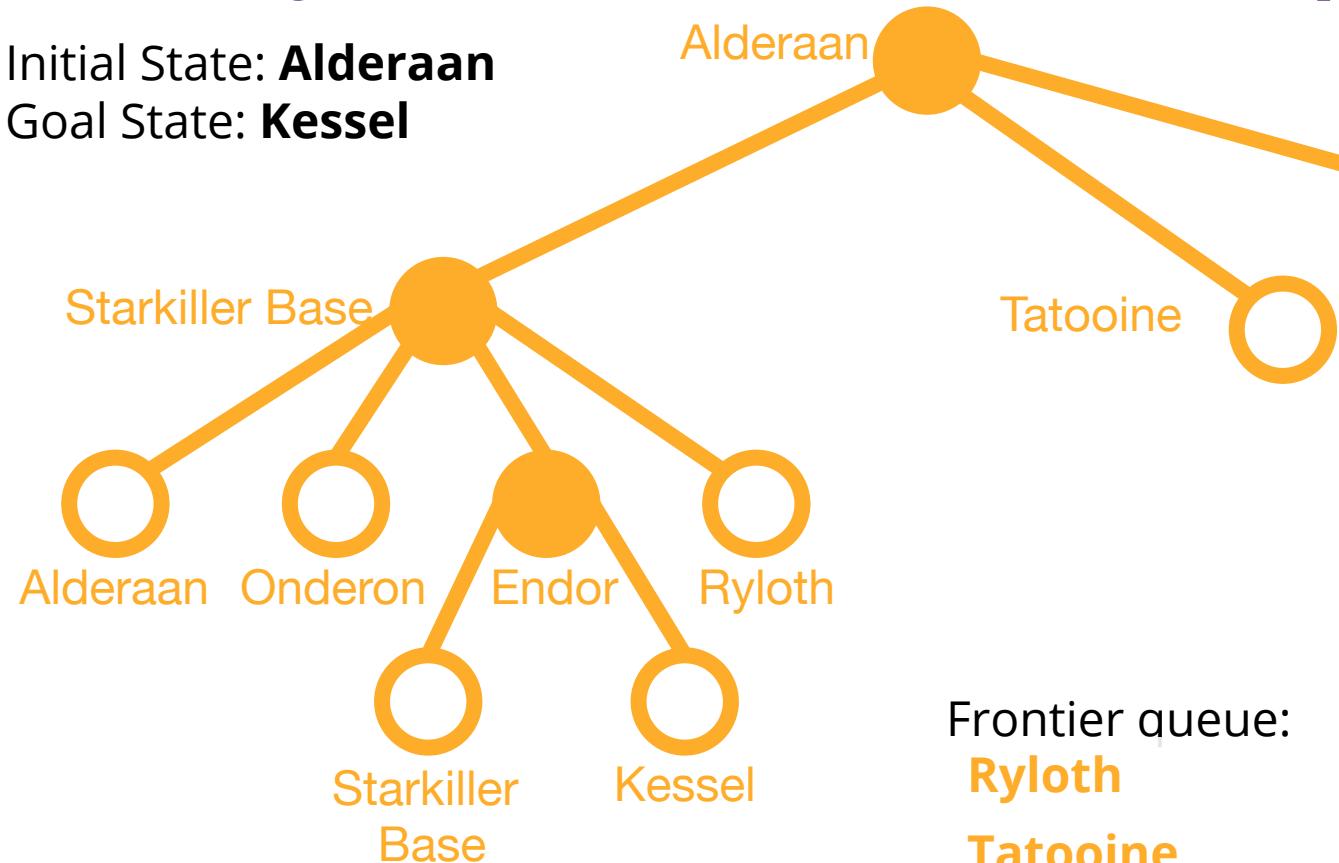
|           |     |
|-----------|-----|
| Endor     | 176 |
| Ryloth    | 193 |
| Tatooine  | 329 |
| Alderaan  | 366 |
| Coruscant | 374 |
| Onderon   | 380 |

| Planet    | SLD | Planet          | SLD |
|-----------|-----|-----------------|-----|
| Alderaan  | 366 | Lotho Minor     | 244 |
| Bespin    | 80  | Mandalore       | 241 |
| Coruscant | 374 | Naboo           | 100 |
| Dagobah   | 242 | Onderon         | 380 |
| Endor     | 176 | Polis Massa     | 234 |
| Fondor    | 161 | Ryloth          | 193 |
| Geonosis  | 77  | Starkiller Base | 253 |
| Hoth      | 151 | Tatooine        | 329 |
| Iridonia  | 226 | Vandor-1        | 199 |
| Kessel    | 0   | Zygerria        | 160 |

# Greedy best-first search example

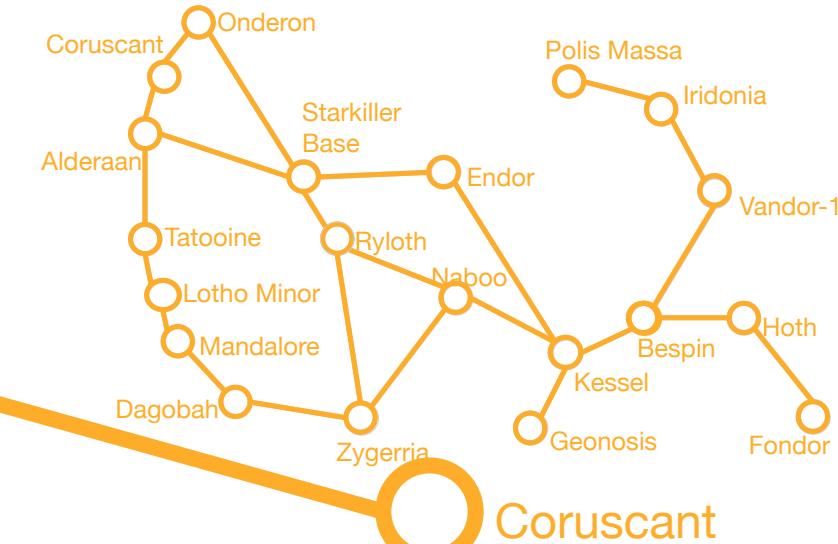
Initial State: Alderaan

Goal State: Kessel



Frontier queue:

|                 |     |
|-----------------|-----|
| RylOTH          | 193 |
| Tatooine        | 329 |
| Alderaan        | 366 |
| Coruscant       | 374 |
| Onderon         | 380 |
| Starkiller Base | 253 |
| Kessel          | 0   |

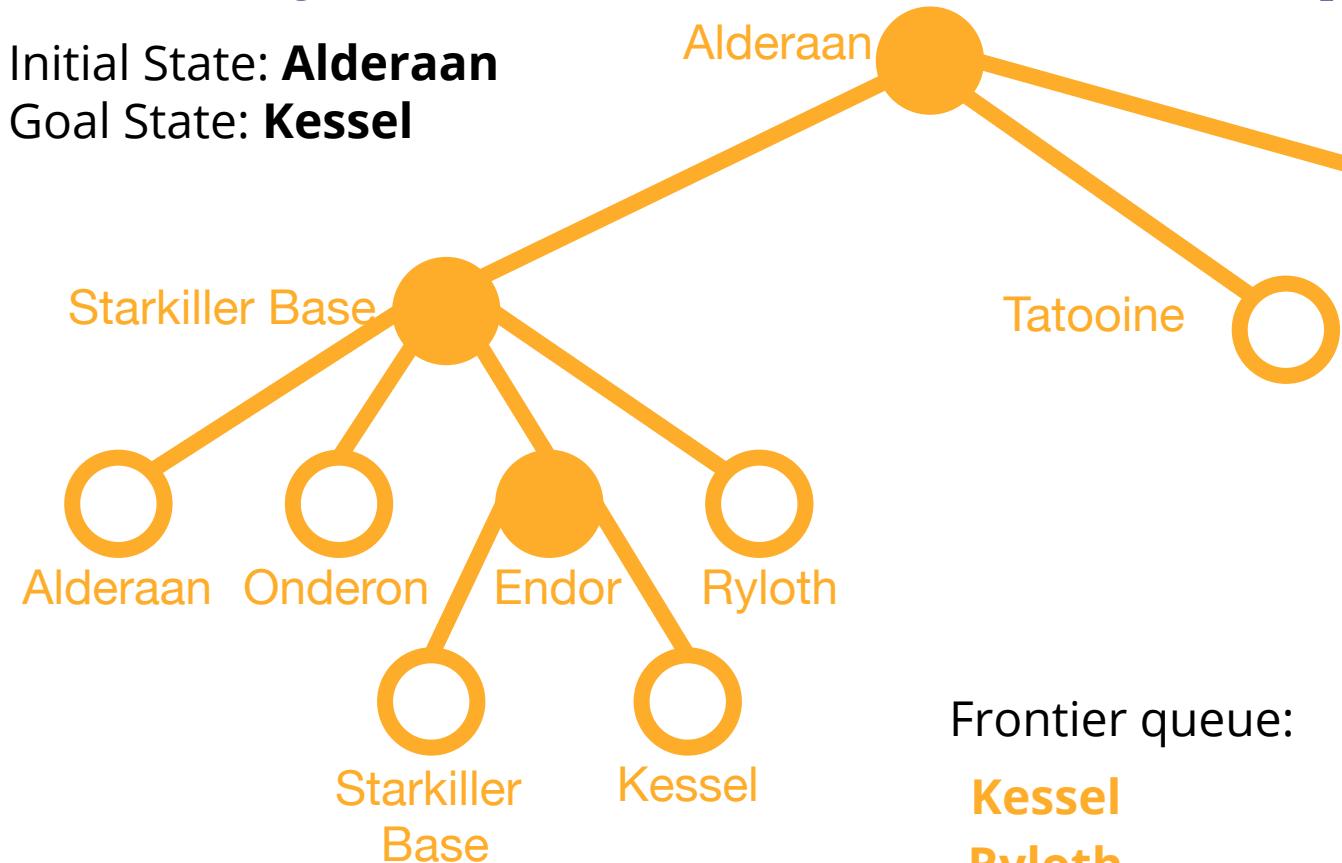


| Planet    | SLD | Planet          | SLD |
|-----------|-----|-----------------|-----|
| Alderaan  | 366 | Lothal Minor    | 244 |
| Bespin    | 80  | Mandalore       | 241 |
| Coruscant | 374 | Naboo           | 100 |
| Dagobah   | 242 | Onderon         | 380 |
| Endor     | 176 | Polis Massa     | 234 |
| Fondor    | 161 | RylOTH          | 193 |
| Geonosis  | 77  | Starkiller Base | 253 |
| Hoth      | 151 | Tatooine        | 329 |
| Iridonia  | 226 | Vandor-1        | 199 |
| Kessel    | 0   | Zygerria        | 160 |

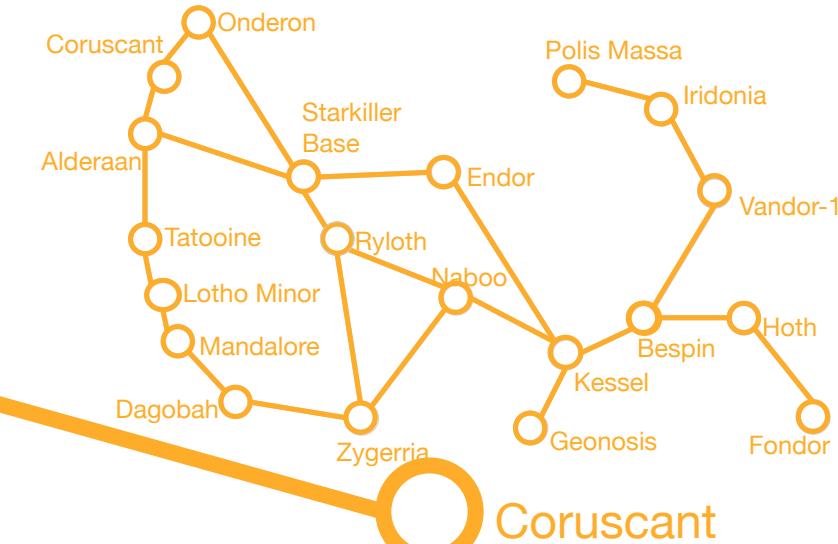
# Greedy best-first search example

Initial State: Alderaan

Goal State: Kessel



|                 |     |
|-----------------|-----|
| Kessel          | 0   |
| RylOTH          | 193 |
| Starkiller Base | 253 |
| Tatooine        | 329 |
| Alderaan        | 366 |
| Coruscant       | 374 |
| Onderon         | 380 |

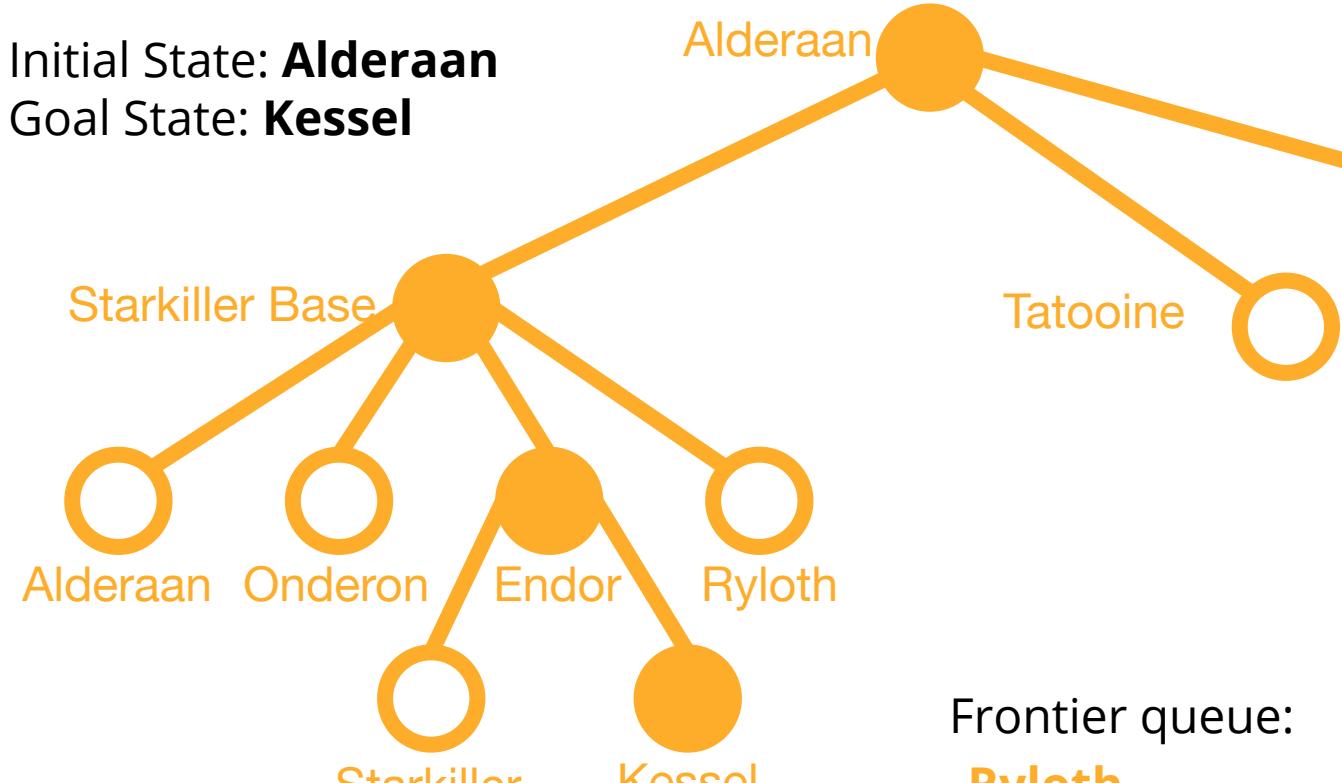


| Planet    | SLD | Planet          | SLD |
|-----------|-----|-----------------|-----|
| Alderaan  | 366 | Lothal Minor    | 244 |
| Bespin    | 80  | Mandalore       | 241 |
| Coruscant | 374 | Naboo           | 100 |
| Dagobah   | 242 | Onderon         | 380 |
| Endor     | 176 | Polis Massa     | 234 |
| Fondor    | 161 | RylOTH          | 193 |
| Geonosis  | 77  | Starkiller Base | 253 |
| Hoth      | 151 | Tatooine        | 329 |
| Iridonia  | 226 | Vandor-1        | 199 |
| Kessel    | 0   | Zygerria        | 160 |

# Greedy best-first search example

Initial State: Alderaan

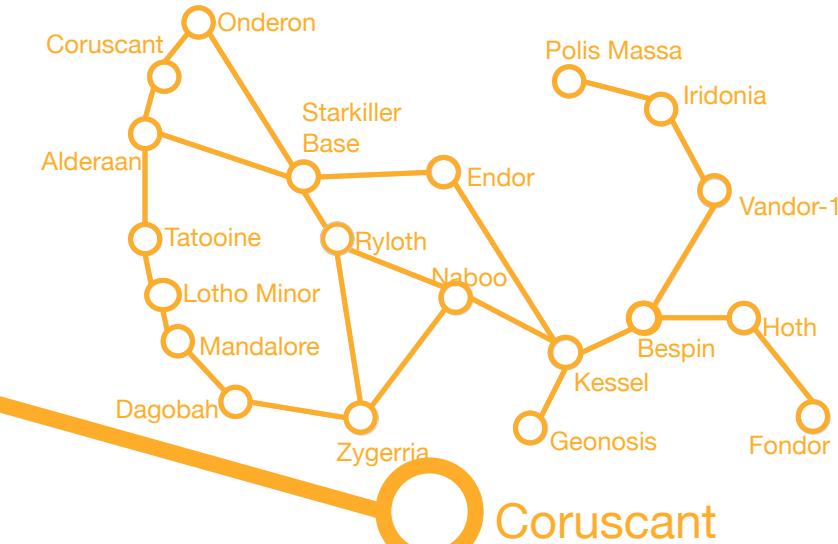
Goal State: Kessel



Goal reached !!

Frontier queue:

|                 |     |
|-----------------|-----|
| RylOTH          | 193 |
| Starkiller Base | 253 |
| Tatooine        | 329 |
| Alderaan        | 366 |
| Coruscant       | 374 |
| Onderon         | 380 |

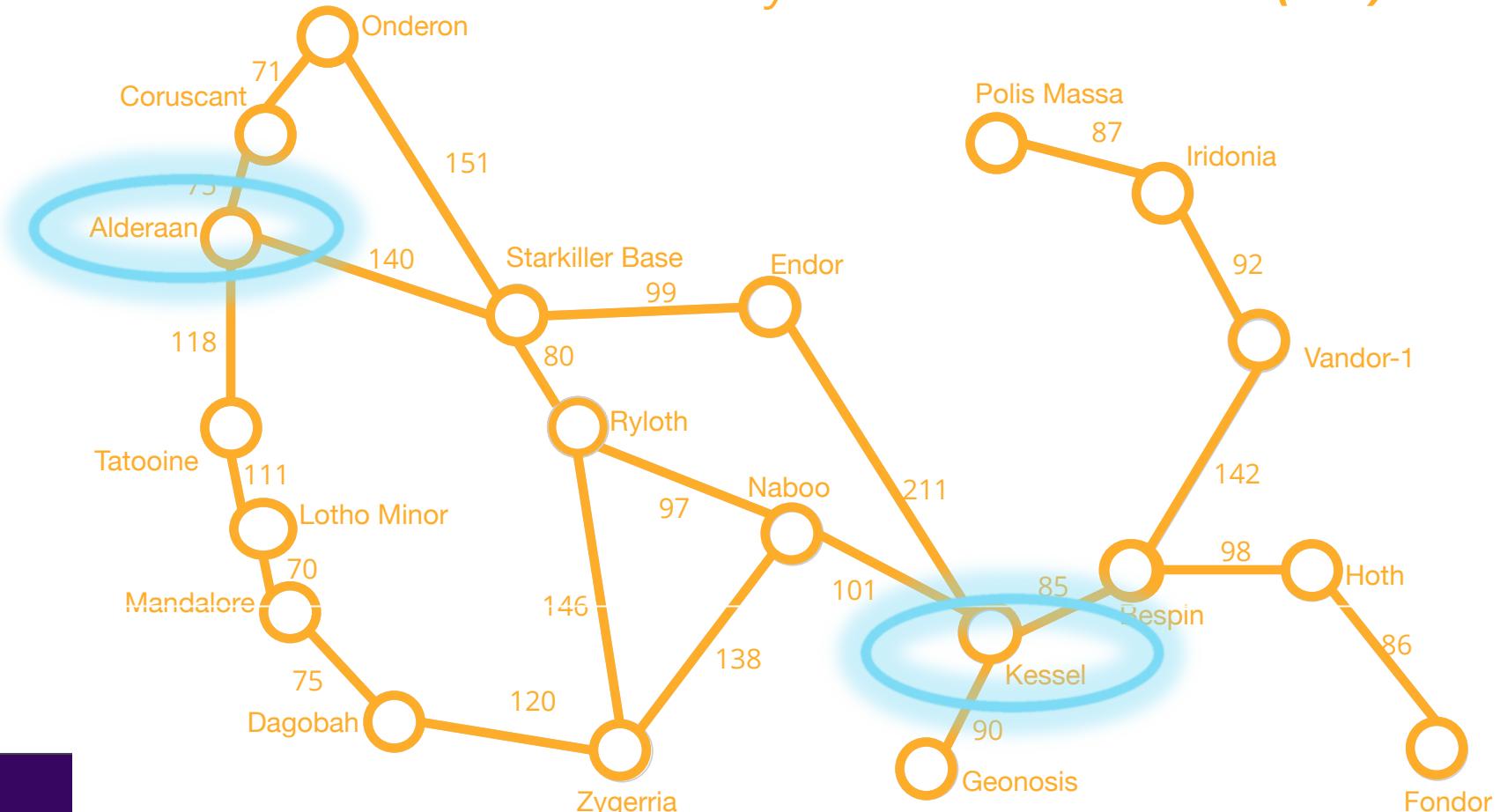


| Planet    | SLD | Planet          | SLD |
|-----------|-----|-----------------|-----|
| Alderaan  | 366 | Lothal Minor    | 244 |
| Bespin    | 80  | Mandalore       | 241 |
| Coruscant | 374 | Naboo           | 100 |
| Dagobah   | 242 | Onderon         | 380 |
| Endor     | 176 | Polis Massa     | 234 |
| Fondor    | 161 | RylOTH          | 193 |
| Geonosis  | 77  | Starkiller Base | 253 |
| Hoth      | 151 | Tatooine        | 329 |
| Iridonia  | 226 | Vandor-1        | 199 |
| Kessel    | 0   | Zygerria        | 160 |

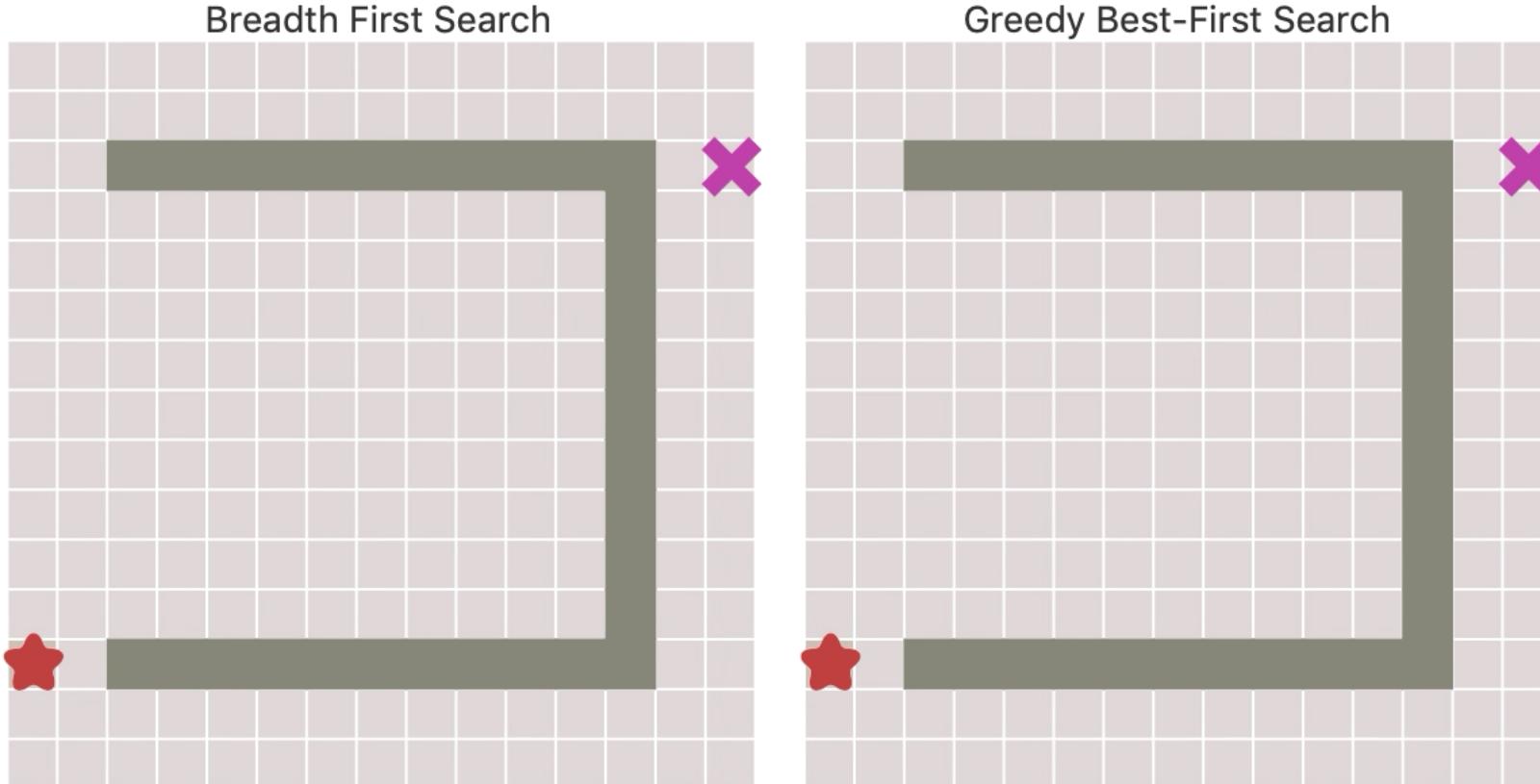
# Properties of greedy best-first search

Optimal?

- No!
  - Found: *Alderaan → Starkiller Base → Endor → Kessel (450)*
  - Shorter: *Alderaan → Starkiller Base → Ryloth → Naboo → Kessel (418)*



# BFS v. Greedy Best-First Search

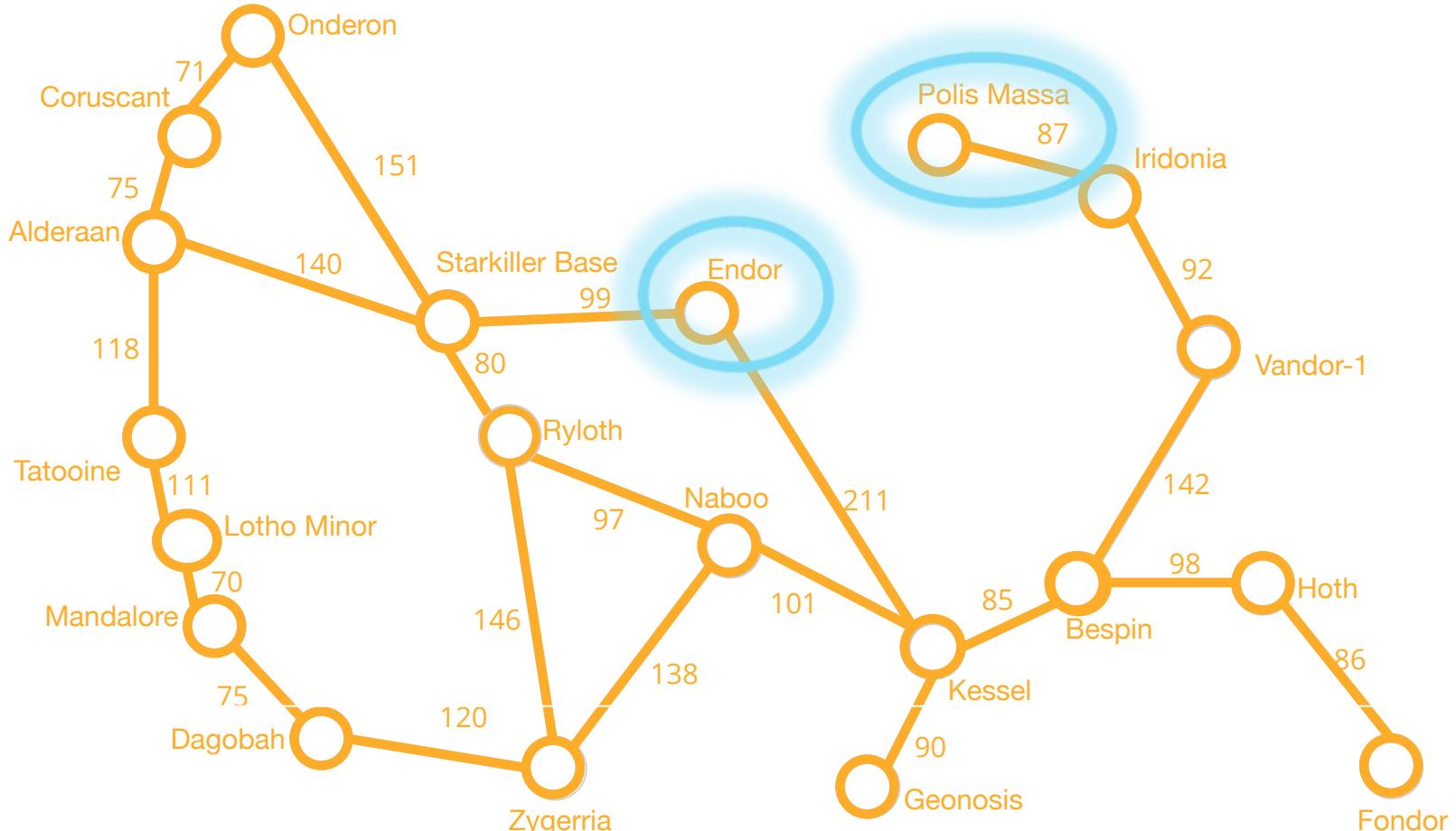


<https://www.redblobgames.com/pathfinding/a-star/introduction.html>

# Properties of greedy best-first search

Complete?

- No – can get stuck in loops,
- e.g., Iridonia → Polis Massa → Iridonia → Polis Massa →...

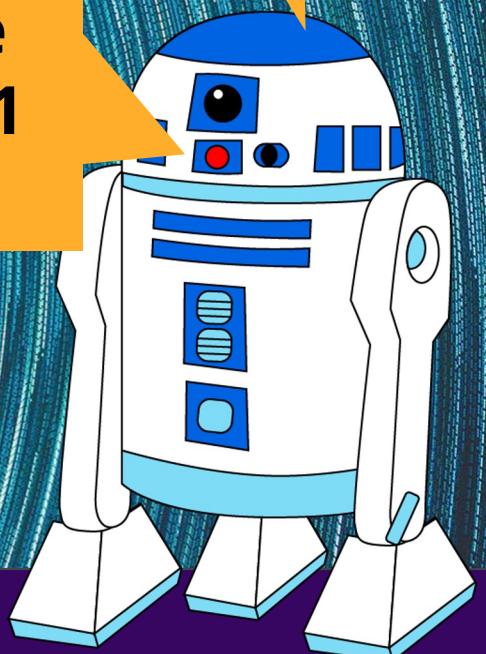


CIS 4210/5210:  
ARTIFICIAL INTELLIGENCE

# Informed Search

Quiz due on Sunday. HW3 is  
due on Wednesday.

If you need an extension,  
request is 24 hours in  
advance via Canvas (no need  
to email me). Usually, we  
grant a max extension of 1  
or 2 days.





# A\* search

AIMA 3.5



# A\* search

Best-known form of best-first search.

Key Idea: avoid expanding paths that are already expensive but expand most promising first.

*Simple idea:  $f(n) = g(n) + h(n)$*

- $g(n)$  the actual cost (so far) to *reach* the node
- $h(n)$  estimated cost to *get from the node to the goal*
- $f(n)$  estimated *total cost* of path through  $n$  to goal

Implementation: Frontier queue as priority queue by increasing  $f(n)$  (*as expected...*)

# Key concept: Admissible heuristics

A heuristic  $h(n)$  is *admissible* if it *never overestimates* the cost to reach the goal; i.e. it is *optimistic*

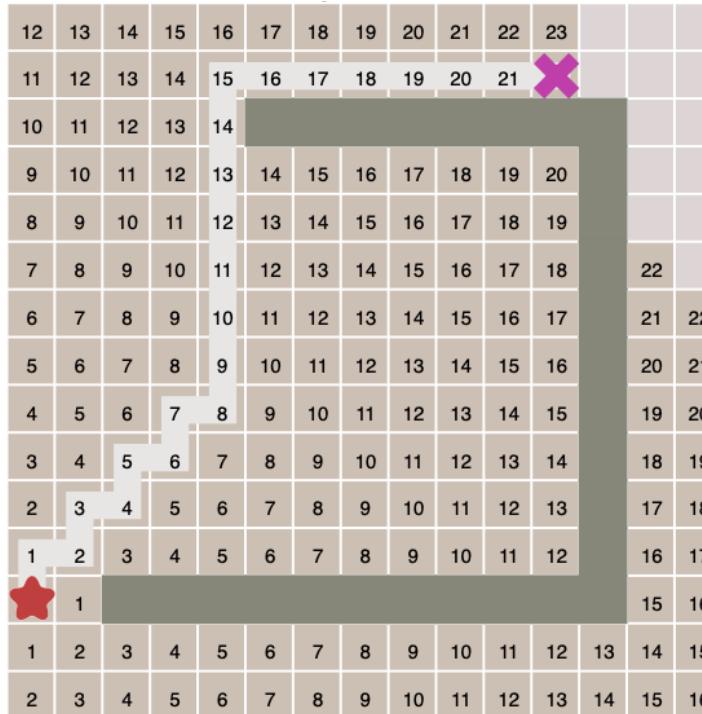
- Formally:  $\forall n$ ,  $n$  a node:
  - $h(n) \leq h^*(n)$  where  $h^*(n)$  is the true cost from  $n$
  - $h(n) \geq 0$  so  $h(G)=0$  for any goal  $G$ .

*Example:*  $h_{SLD}(n)$  never overestimates the actual road distance

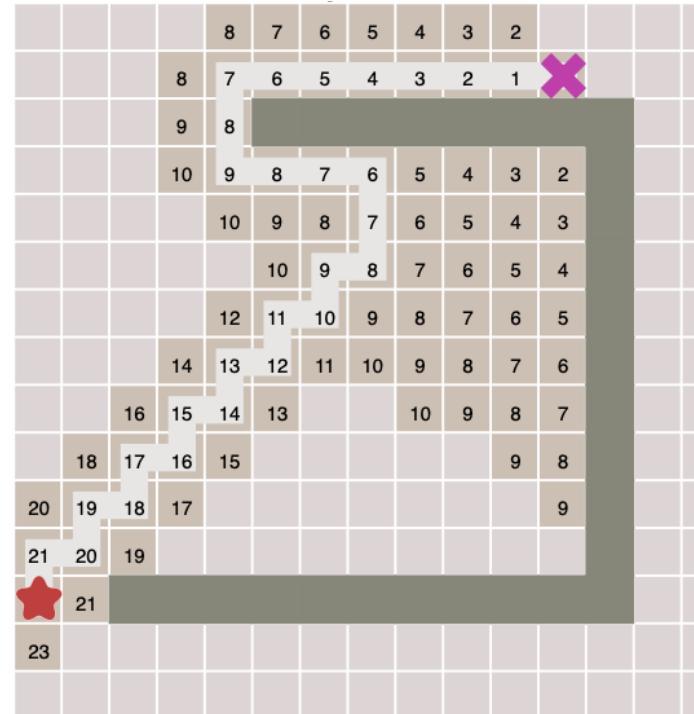
*Theorem:* If  $h(n)$  is *admissible*, A\* using **Tree Search** is *optimal*

# A\* is optimal with admissible heuristic

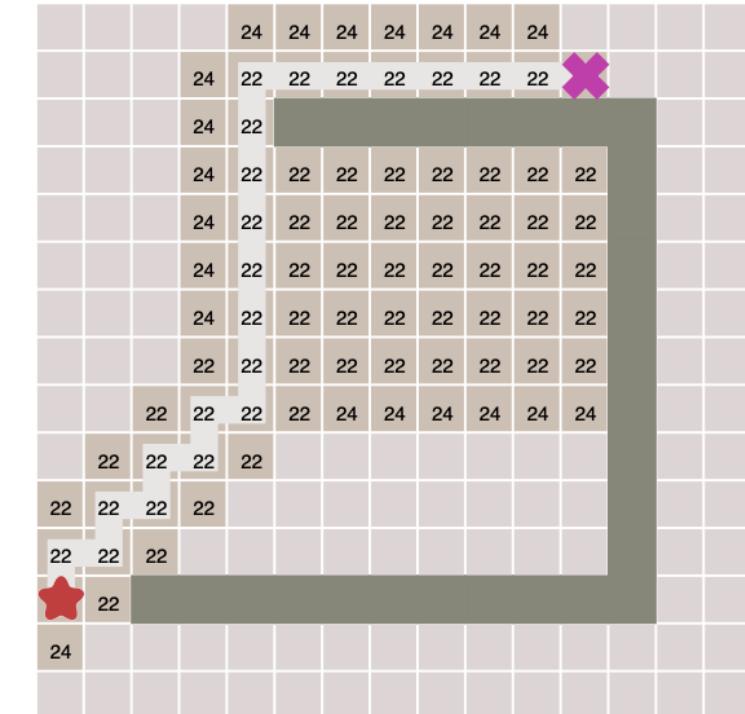
Uniform Cost Search



Greedy Best-First Search



A\* Search



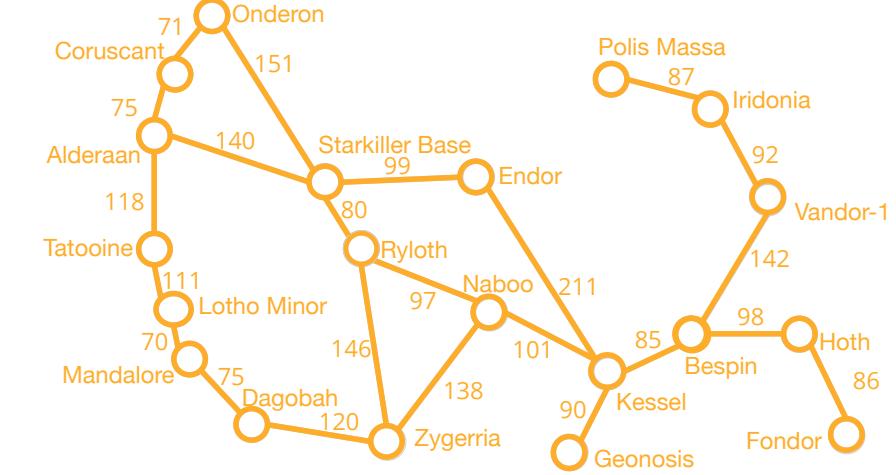
<https://www.redblobgames.com/pathfinding/a-star/introduction.html>

# A\* search example

Initial State: **Alderaan**



Goal State: **Kessel**



Frontier queue:

**Alderon 366 = 0 + 366**

| Planet    | SLD | Planet          | SLD |
|-----------|-----|-----------------|-----|
| Alderaan  | 366 | Lothal Minor    | 244 |
| Bespin    | 80  | Mandalore       | 241 |
| Coruscant | 374 | Naboo           | 100 |
| Dagobah   | 242 | Onderon         | 380 |
| Endor     | 176 | Polis Massa     | 234 |
| Fondor    | 161 | Ryleth          | 193 |
| Geonosis  | 77  | Starkiller Base | 253 |
| Hoth      | 151 | Tatooine        | 329 |
| Iridonia  | 226 | Vandor-1        | 199 |
| Kessel    | 0   | Zygerria        | 160 |

# A\* search example

Initial State: **Alderaan**  
Goal State: **Kessel**



$$f(n) = g(n) + h(n)$$

Frontier queue:

**Starkiller Base**  $393 = 140 + 253$

**Tatooine**  $447 = 118 + 329$

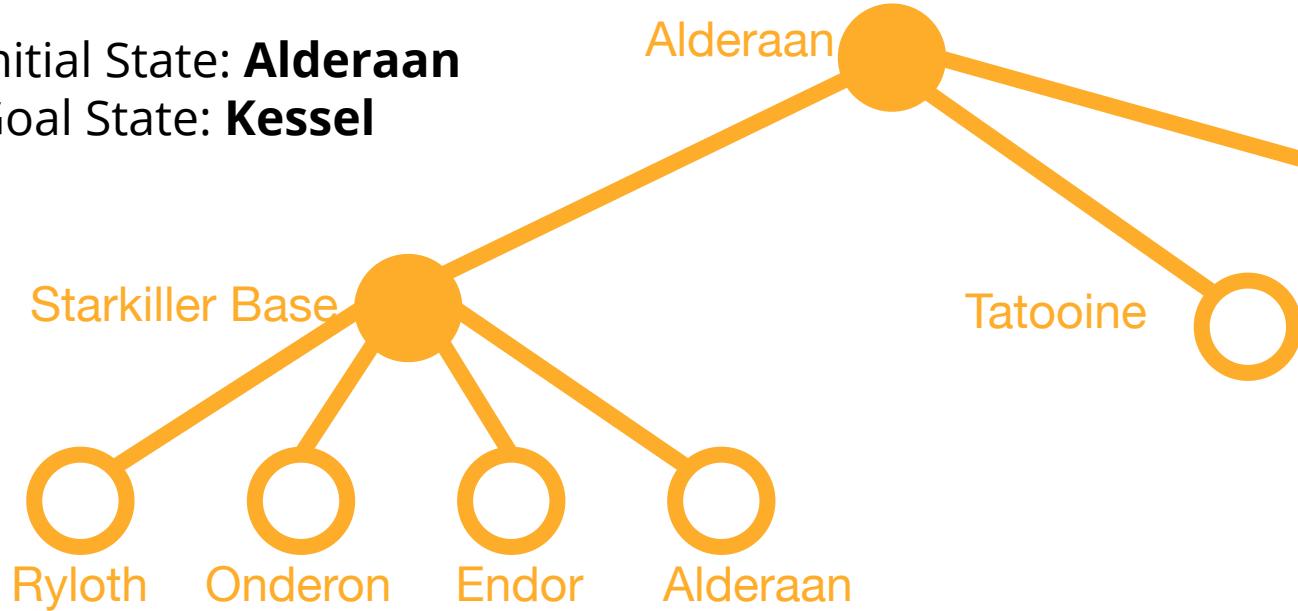
**Coruscant**  $449 = 75 + 374$

| Planet    | SLD | Planet          | SLD |
|-----------|-----|-----------------|-----|
| Alderaan  | 366 | Lothal Minor    | 244 |
| Bespin    | 80  | Mandalore       | 241 |
| Coruscant | 374 | Naboo           | 100 |
| Dagobah   | 242 | Onderon         | 380 |
| Endor     | 176 | Polis Massa     | 234 |
| Fondor    | 161 | Ryloth          | 193 |
| Geonosis  | 77  | Starkiller Base | 253 |
| Hoth      | 151 | Tatooine        | 329 |
| Iridonia  | 226 | Vandor-1        | 199 |
| Kessel    | 0   | Zygerria        | 160 |

# A\* search example

Initial State: Alderaan

Goal State: Kessel



Are we doing **graph**  
**search or tree search?**

Frontier queue:

Tatooine  $447 = 118 + 329$

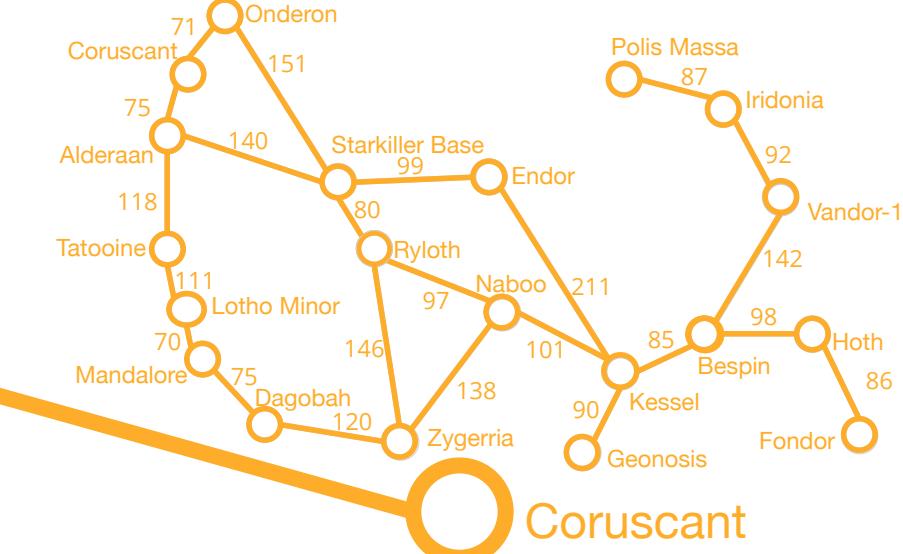
Coruscant  $449 = 75 + 374$

Alderaan  $646 = 280 + 366$

Onderon  $526 = 146 + 380$

Endor  $415 = 239 + 176$

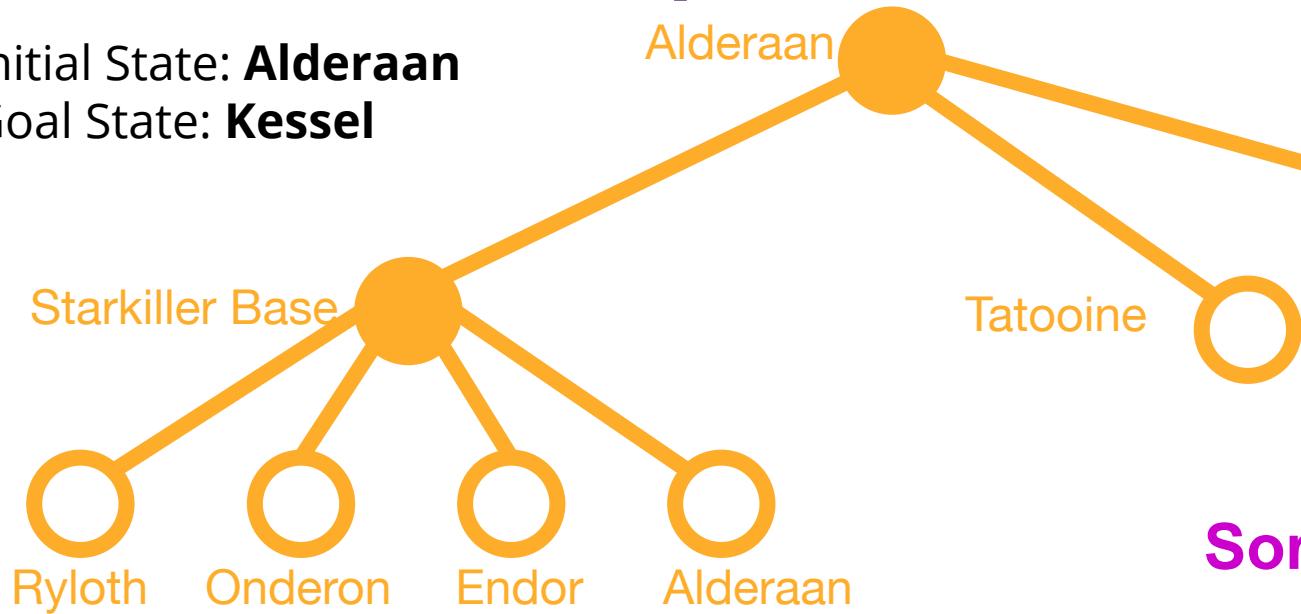
RylOTH  $413 = 220 + 193$



| Planet    | SLD | Planet          | SLD |
|-----------|-----|-----------------|-----|
| Alderaan  | 366 | Lothal Minor    | 244 |
| Bespin    | 80  | Mandalore       | 241 |
| Coruscant | 374 | Naboo           | 100 |
| Dagobah   | 242 | Onderon         | 380 |
| Endor     | 176 | Polis Massa     | 234 |
| Fondor    | 161 | RylOTH          | 193 |
| Geonosis  | 77  | Starkiller Base | 253 |
| Hoth      | 151 | Tatooine        | 329 |
| Iridonia  | 226 | Vandor-1        | 199 |
| Kessel    | 0   | Zygerria        | 160 |

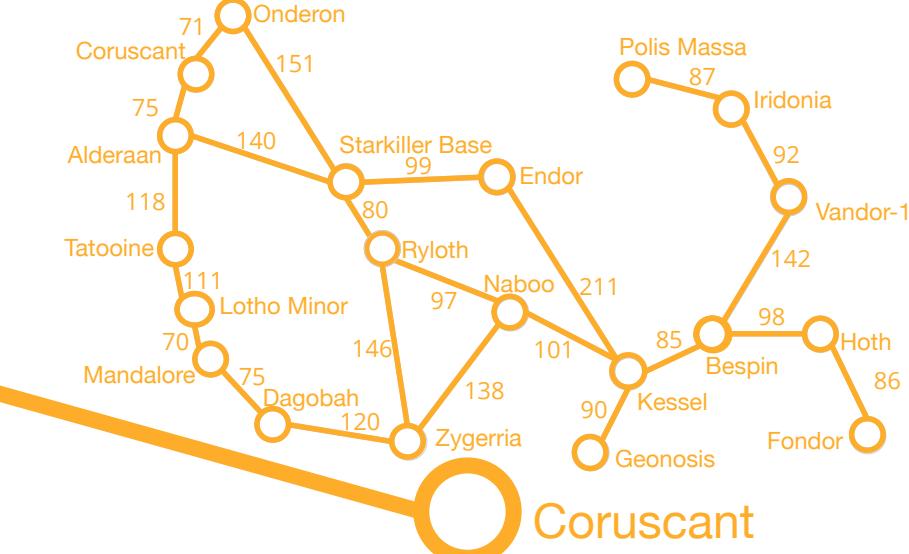
# A\* search example

Initial State: Alderaan  
Goal State: Kessel



Frontier queue:

|           |                   |
|-----------|-------------------|
| Ryloth    | $413 = 220 + 193$ |
| Endor     | $415 = 239 + 176$ |
| Tatooine  | $447 = 118 + 329$ |
| Coruscant | $449 = 75 + 374$  |
| Onderon   | $526 = 146 + 380$ |
| Alderaan  | $646 = 280 + 366$ |



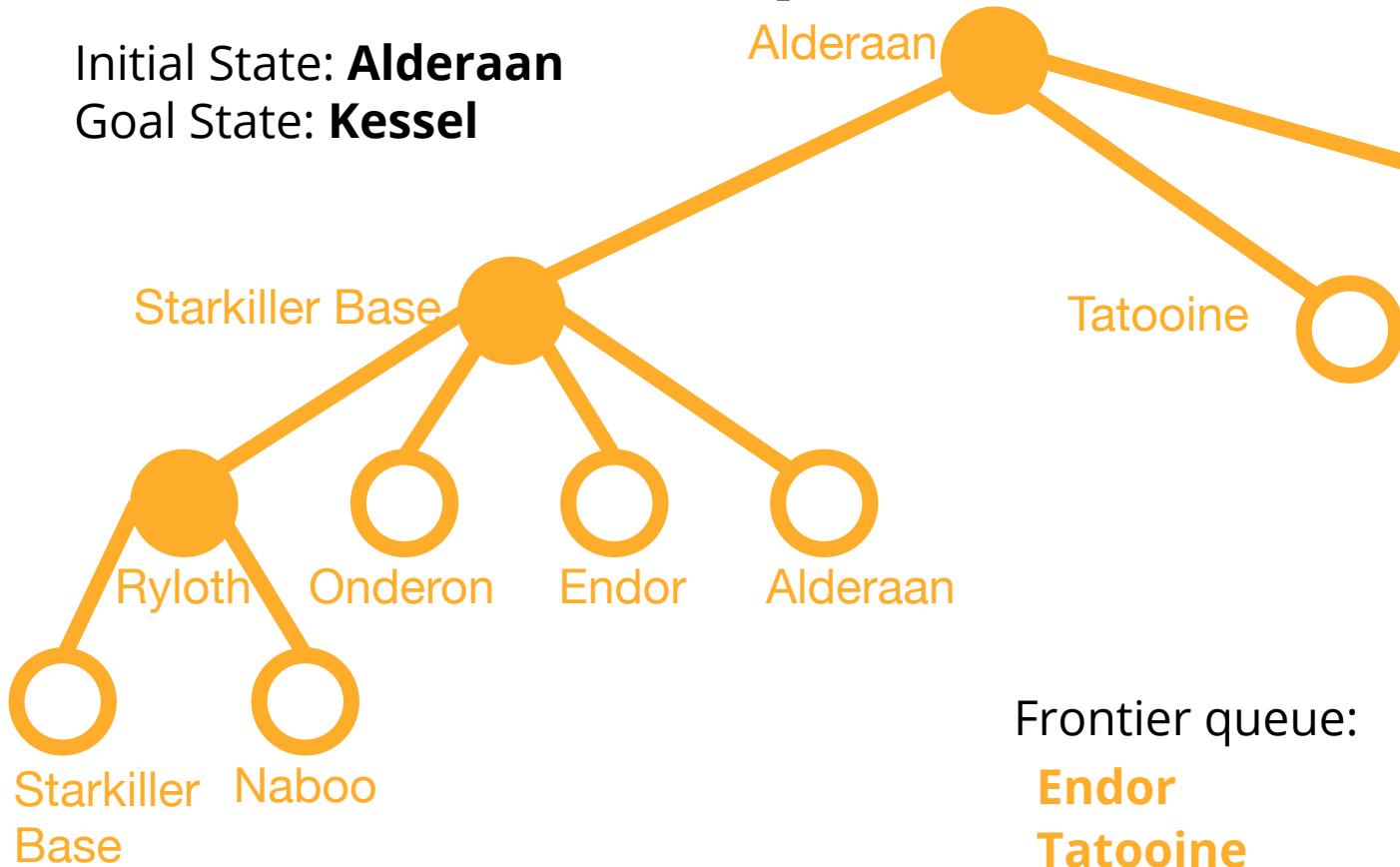
Sort queue by  $f(n)$ .

$$f(n) = g(n) + h(n)$$

| Planet    | SLD | Planet          | SLD |
|-----------|-----|-----------------|-----|
| Alderaan  | 366 | Lothal Minor    | 244 |
| Bespin    | 80  | Mandalore       | 241 |
| Coruscant | 374 | Naboo           | 100 |
| Dagobah   | 242 | Onderon         | 380 |
| Endor     | 176 | Polis Massa     | 234 |
| Fondor    | 161 | Ryleth          | 193 |
| Geonosis  | 77  | Starkiller Base | 253 |
| Hoth      | 151 | Tatooine        | 329 |
| Iridonia  | 226 | Vandor-1        | 199 |
| Kessel    | 0   | Zygerria        | 160 |

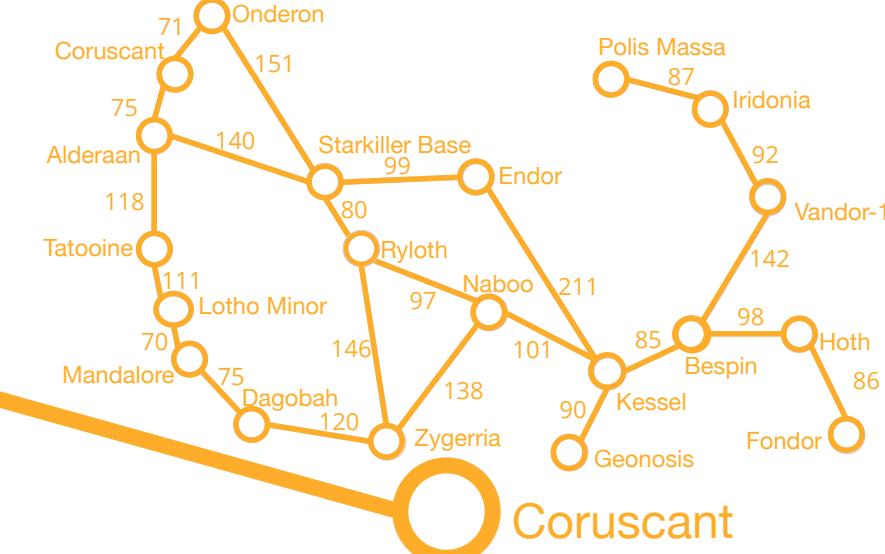
# A\* search example

Initial State: **Alderaan**  
Goal State: **Kessel**



## Frontier queue:

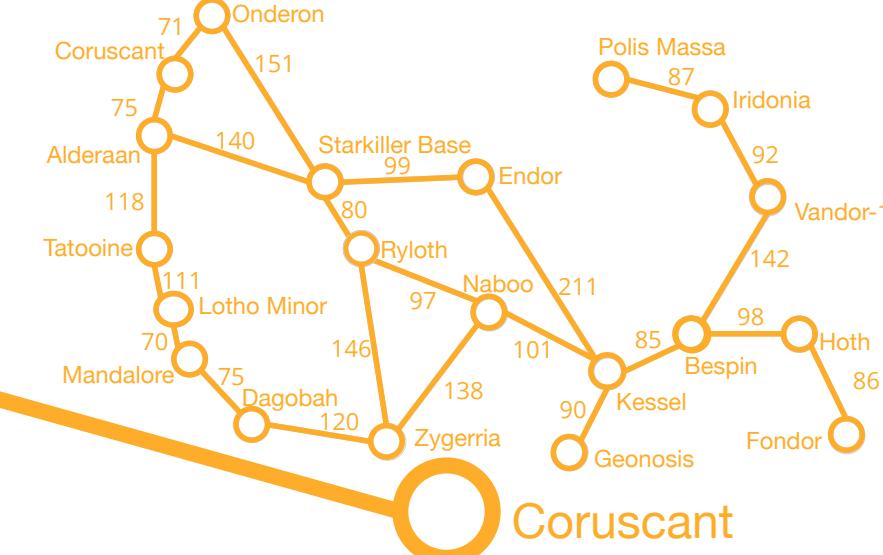
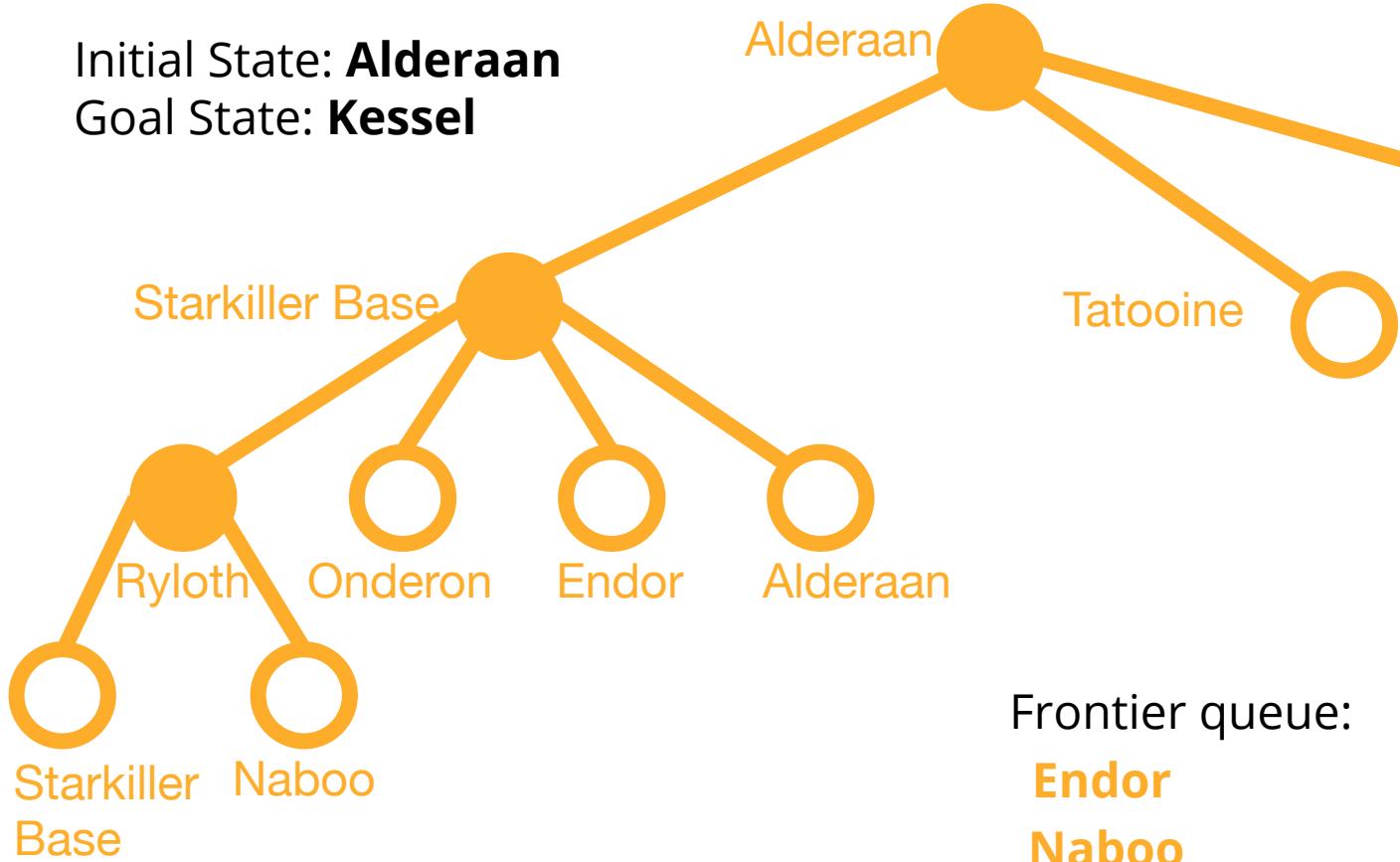
|                        |                        |
|------------------------|------------------------|
| <b>Endor</b>           | <b>415 = 239 + 176</b> |
| <b>Tatooine</b>        | <b>447 = 118 + 329</b> |
| <b>Coruscant</b>       | <b>449 = 75 + 374</b>  |
| <b>Onderon</b>         | <b>526 = 146 + 380</b> |
| <b>Alderaan</b>        | <b>646 = 280 + 366</b> |
| <b>Starkiller Base</b> | <b>532 = 300 + 253</b> |
| <b>Naboo</b>           | <b>417 = 317 + 100</b> |



| Planet    | SLD | Planet          | SLD |
|-----------|-----|-----------------|-----|
| Alderaan  | 366 | Lotho Minor     | 244 |
| Bespin    | 80  | Mandalore       | 241 |
| Coruscant | 374 | Naboo           | 100 |
| Dagobah   | 242 | Onderon         | 380 |
| Endor     | 176 | Polis Massa     | 234 |
| Fondor    | 161 | Ryloth          | 193 |
| Geonosis  | 77  | Starkiller Base | 253 |
| Hoth      | 151 | Tatooine        | 329 |
| Iridonia  | 226 | Vandor-1        | 199 |
| Kessel    | 0   | Zygerria        | 160 |

# A\* search example

Initial State: Alderaan  
Goal State: Kessel



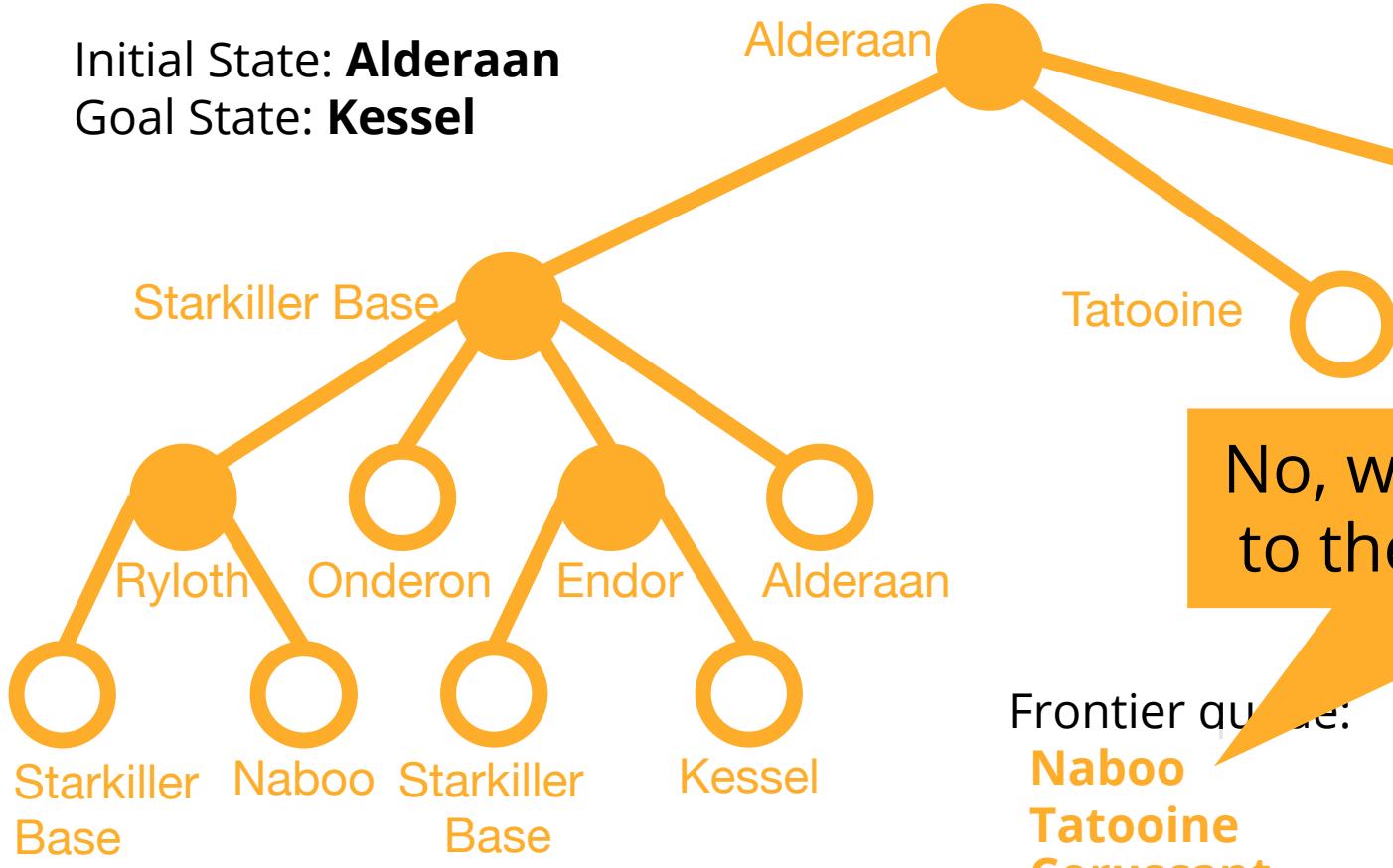
Frontier queue:

|                        |                        |
|------------------------|------------------------|
| <b>Alderaan</b>        | <b>646 = 280 + 366</b> |
| <b>Starkiller Base</b> | <b>532 = 300 + 253</b> |
| <b>Onderon</b>         | <b>526 = 146 + 380</b> |
| <b>Coruscant</b>       | <b>449 = 75 + 374</b>  |
| <b>Tatooine</b>        | <b>447 = 118 + 329</b> |
| <b>Naboo</b>           | <b>417 = 317 + 100</b> |
| <b>Rylloth</b>         | <b>415 = 239 + 176</b> |
| <b>Endor</b>           |                        |

| Planet    | SLD | Planet          | SLD |
|-----------|-----|-----------------|-----|
| Alderaan  | 366 | Lothal Minor    | 244 |
| Bespin    | 80  | Mandalore       | 241 |
| Coruscant | 374 | Naboo           | 100 |
| Dagobah   | 242 | Onderon         | 380 |
| Endor     | 176 | Polis Massa     | 234 |
| Fondor    | 161 | Rylloth         | 193 |
| Geonosis  | 77  | Starkiller Base | 253 |
| Hoth      | 151 | Tatooine        | 329 |
| Iridonia  | 226 | Vandor-1        | 199 |
| Kessel    | 0   | Zygerria        | 160 |

# A\* search example

Initial State: **Alderaan**  
Goal State: **Kessel**



Do we apply the goal test  
to the **children** like in BFS?

No, we apply the goal test to the **top of the queue**.

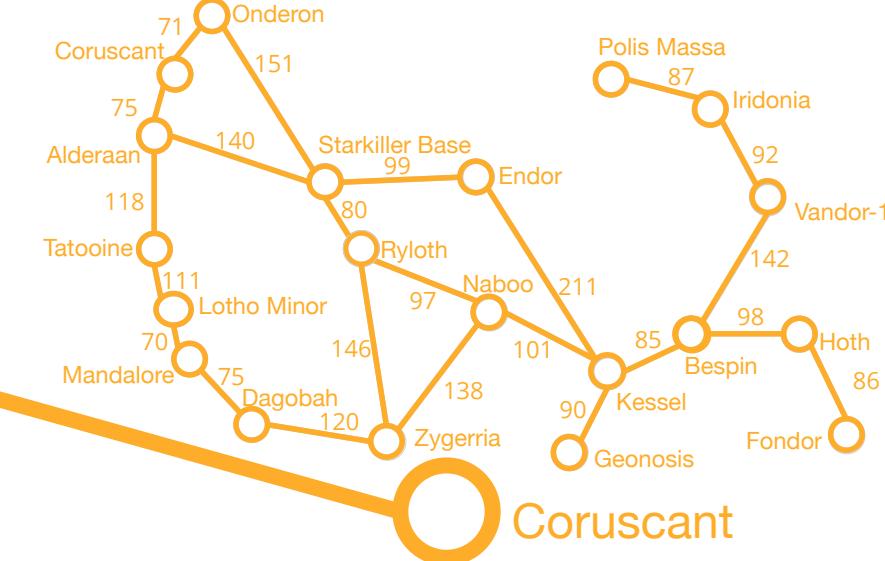
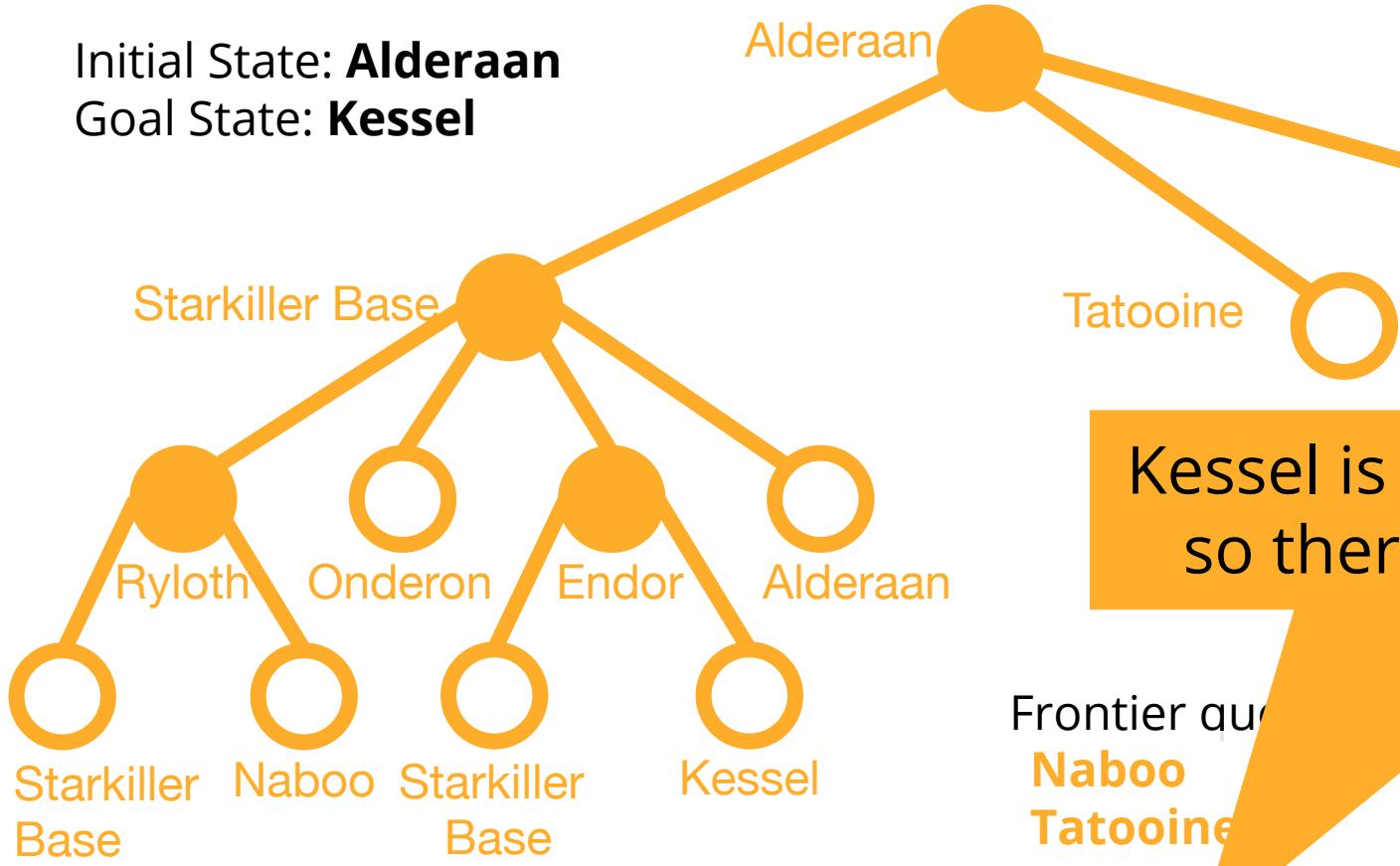
# Frontier audience:

|                 |                   |
|-----------------|-------------------|
| Naboo           | $417 = 317 + 100$ |
| Tatooine        | $447 = 118 + 329$ |
| Coruscant       | $449 = 75 + 374$  |
| Onderon         | $526 = 146 + 380$ |
| Starkiller Base | $532 = 300 + 253$ |
| Alderaan        | $646 = 280 + 366$ |
| Starkiller Base | $591 = 338 + 253$ |
| Kessel          | $450 = 450 + 0$   |

| Planet    | SLD | Planet          | SLD |
|-----------|-----|-----------------|-----|
| Alderaan  | 366 | Lotho Minor     | 244 |
| Bespin    | 80  | Mandalore       | 241 |
| Coruscant | 374 | Naboo           | 100 |
| Dagobah   | 242 | Onderon         | 380 |
| Endor     | 176 | Polis Massa     | 234 |
| Fondor    | 161 | Ryloth          | 193 |
| Geonosis  | 77  | Starkiller Base | 253 |
| Hoth      | 151 | Tatooine        | 329 |
| Iridonia  | 226 | Vandor-1        | 199 |
| Kessel    | 0   | Zygerria        | 160 |

# A\* search example

Initial State: Alderaan  
Goal State: Kessel



Kessel is not yet at **top of the queue**,  
so there may be a **shorter path...**

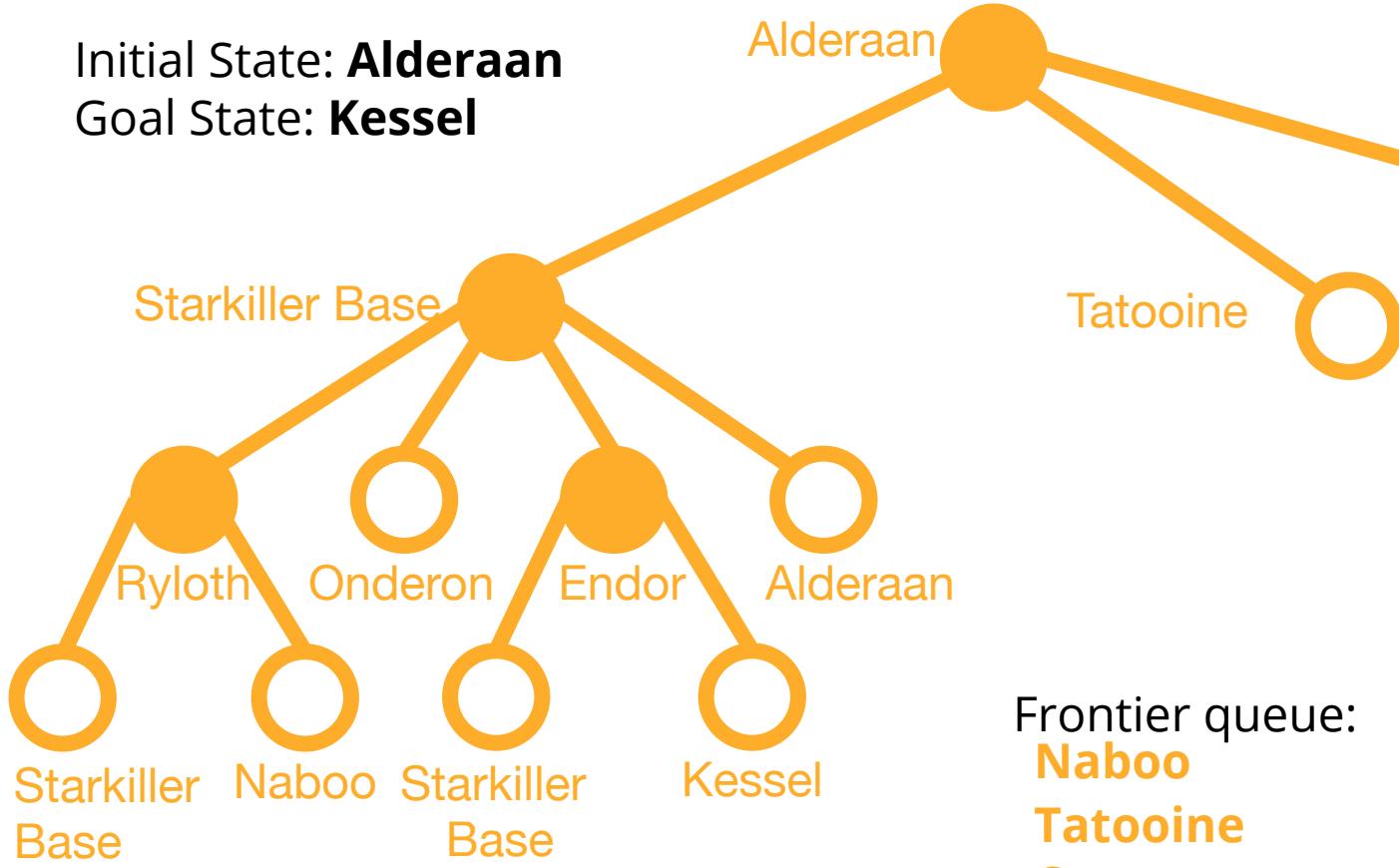
Frontier queue

| Planet          | SLD             |
|-----------------|-----------------|
| Naboo           | 417 = 317 + 100 |
| Tatooine        | 447 = 118 + 329 |
| Coruscant       | 449 = 75 + 374  |
| Kessel          | 450 = 450 + 0   |
| Onderon         | 526 = 146 + 380 |
| Starkiller Base | 532 = 300 + 253 |
| Starkiller Base | 591 = 338 + 253 |
| Alderaan        | 646 = 280 + 366 |

| Planet          | SLD |
|-----------------|-----|
| Alderaan        | 366 |
| Bespin          | 80  |
| Coruscant       | 374 |
| Dagobah         | 242 |
| Endor           | 176 |
| Fondor          | 161 |
| Geonosis        | 77  |
| Hoth            | 151 |
| Iridonia        | 226 |
| Kessel          | 0   |
| Lothal Minor    | 244 |
| Mandalore       | 241 |
| Naboo           | 100 |
| Onderon         | 380 |
| Polis Massa     | 234 |
| RylOTH          | 193 |
| Starkiller Base | 253 |
| Tatooine        | 329 |
| Vandor-1        | 199 |
| Zygerria        | 160 |

# A\* search example

Initial State: **Alderaan**  
Goal State: **Kessel**



## Frontier queue:

# Naboo

$$417 = 317 + 100$$

# Tatooine

$$447 = 118 + 329$$

Corus

$$449 = 75 + 374$$

# Kessel

$$450 = 450 + 0$$

# Onderon

$$526 \equiv 146 + 380$$

## Starkiller Base

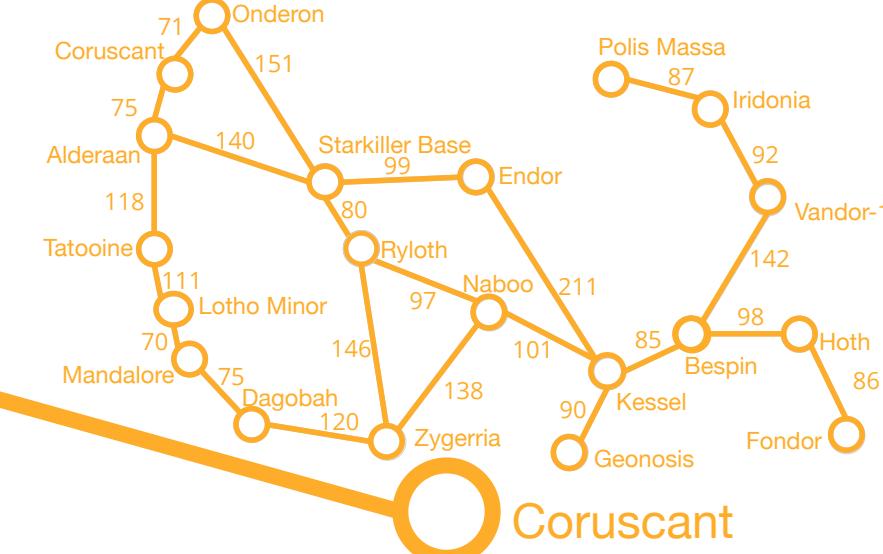
532 = 300 ± 253

## **Starkiller Base**

**592 500 + 253**

**Starkiller  
Alderson**

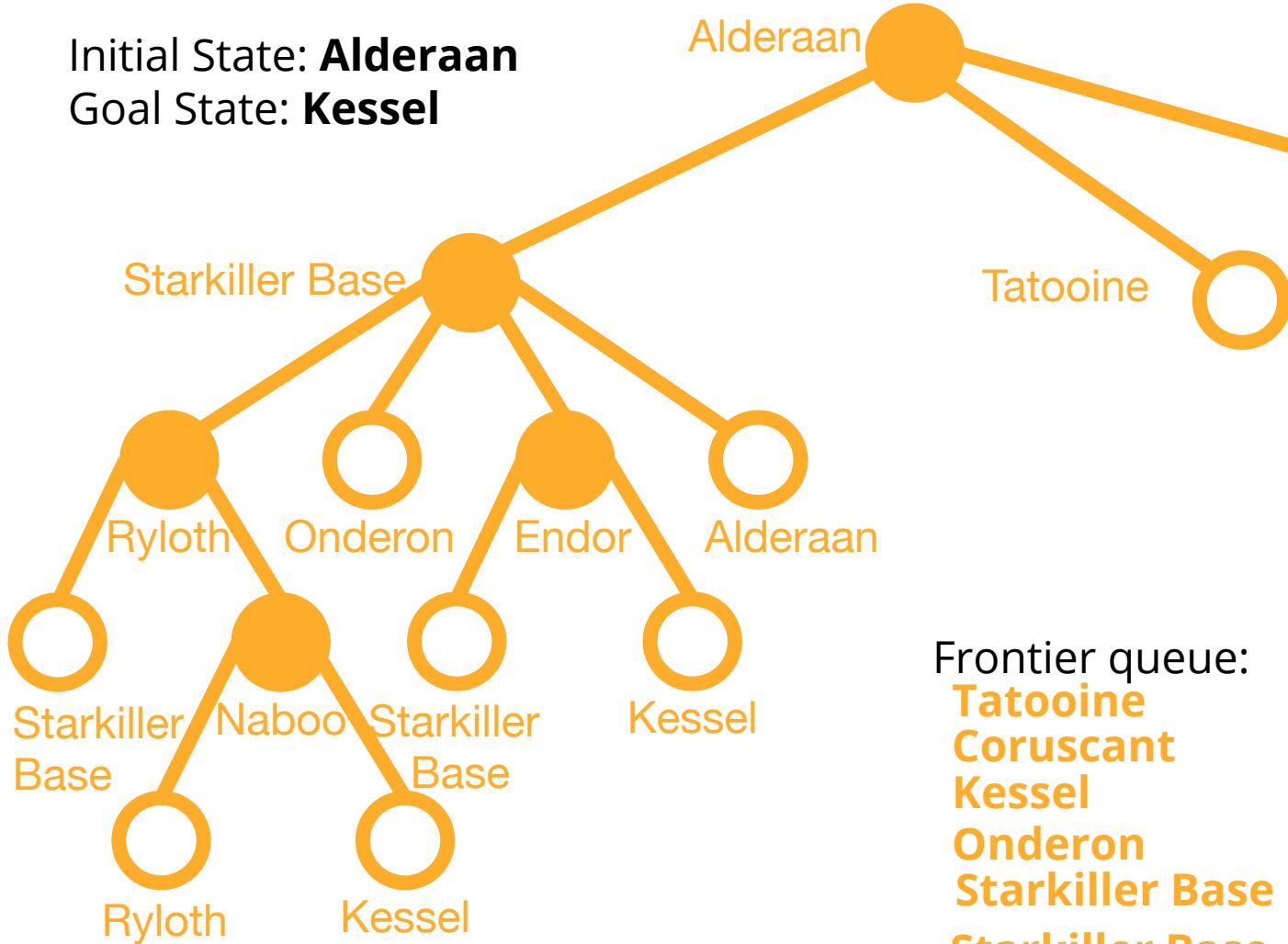
**646-380-266**



| Planet    | SLD | Planet          | SLD |
|-----------|-----|-----------------|-----|
| Alderaan  | 366 | Lotho Minor     | 244 |
| Bespin    | 80  | Mandalore       | 241 |
| Coruscant | 374 | Naboo           | 100 |
| Dagobah   | 242 | Onderon         | 380 |
| Endor     | 176 | Polis Massa     | 234 |
| Fondor    | 161 | Ryloth          | 193 |
| Geonosis  | 77  | Starkiller Base | 253 |
| Hoth      | 151 | Tatooine        | 329 |
| Iridonia  | 226 | Vandor-1        | 199 |
| Kessel    | 0   | Zygerria        | 160 |

# A\* search example

Initial State: **Alderaan**  
Goal State: **Kessel**



## Frontier queue:

$$447 = 118 + 329$$

$$449 = 75 + 374$$

$$450 = 450 + 0$$

$$526 = 146 + 380$$

$$532 = 300 + 253$$

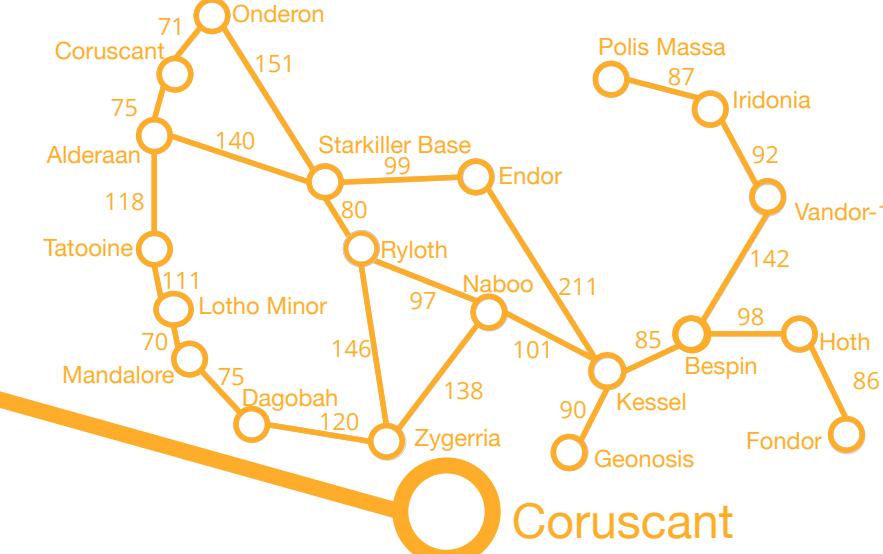
E01 - 338 + 2E3

$$641 = 338 + 253$$

$$646 = 280 + 366$$

$$607 = 414 + 193$$

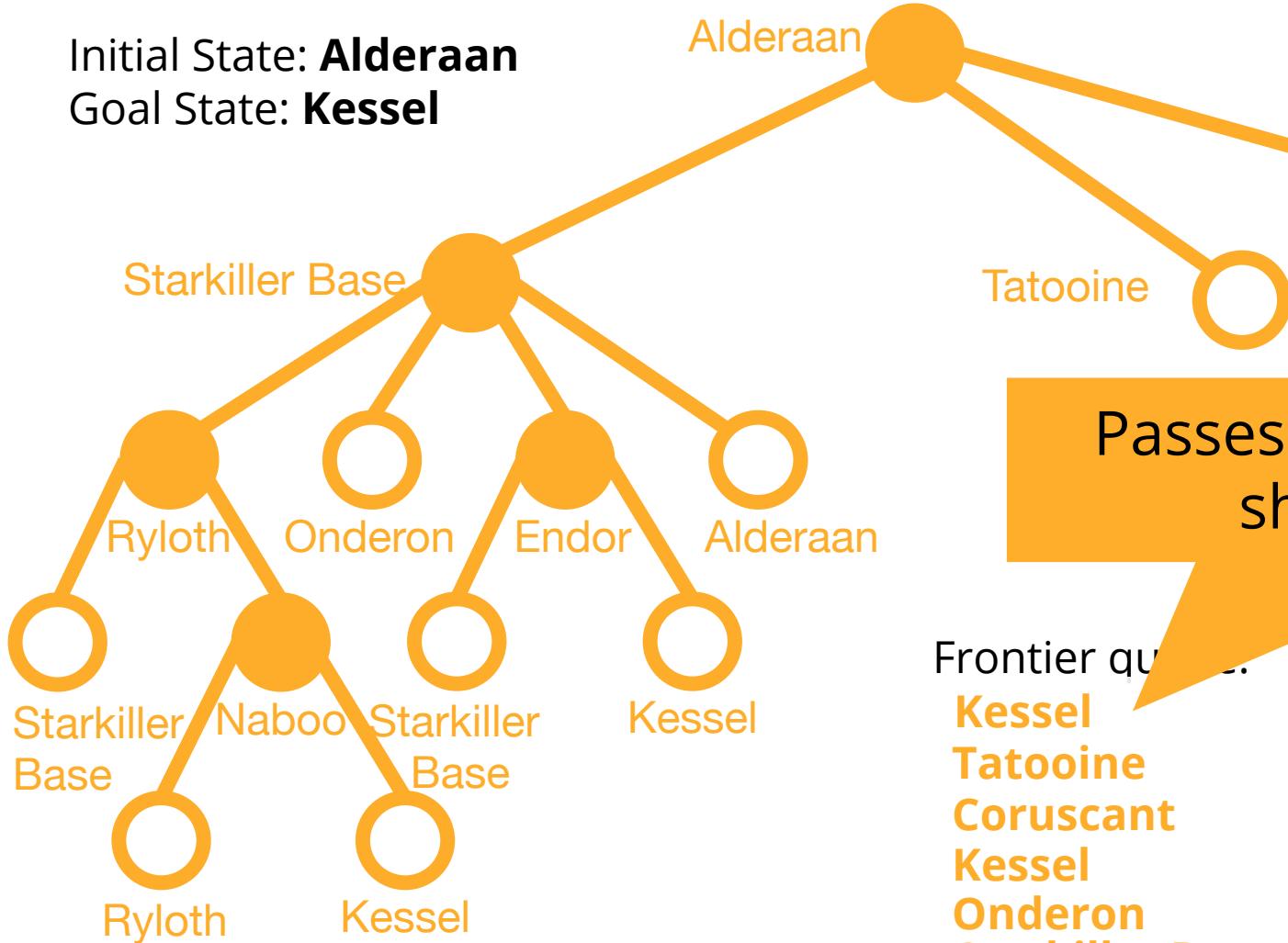
$$418 = 418 + 0$$



| Planet    | SLD | Planet          | SLD |
|-----------|-----|-----------------|-----|
| Alderaan  | 366 | Lotho Minor     | 244 |
| Bespin    | 80  | Mandalore       | 241 |
| Coruscant | 374 | Naboo           | 100 |
| Dagobah   | 242 | Onderon         | 380 |
| Endor     | 176 | Polis Massa     | 234 |
| Fondor    | 161 | Ryloth          | 193 |
| Geonosis  | 77  | Starkiller Base | 253 |
| Hoth      | 151 | Tatooine        | 329 |
| Iridonia  | 226 | Vandor-1        | 199 |
| Kessel    | 0   | Zygerria        | 160 |

# A\* search example

Initial State: **Alderaan**  
Goal State: **Kessel**



Passes the goal test, so **this** is the shortest path to Kessel!

# Frontier qu...

|                 |                   |
|-----------------|-------------------|
| Kessel          | $418 = 418 + 0$   |
| Tatooine        | $447 = 118 + 329$ |
| Coruscant       | $449 = 75 + 374$  |
| Kessel          | $450 = 450 + 0$   |
| Onderon         | $526 = 146 + 380$ |
| Starkiller Base | $532 = 300 + 253$ |
| Starkiller Base | $591 = 338 + 253$ |
| Ryloth          | $607 = 414 + 193$ |
| Alderaan        | $646 = 280 + 366$ |

| Planet    | SLD | Planet          | SLD |
|-----------|-----|-----------------|-----|
| Alderaan  | 366 | Lotho Minor     | 244 |
| Bespin    | 80  | Mandalore       | 241 |
| Coruscant | 374 | Naboo           | 100 |
| Dagobah   | 242 | Onderon         | 380 |
| Endor     | 176 | Polis Massa     | 234 |
| Fondor    | 161 | Ryloth          | 193 |
| Geonosis  | 77  | Starkiller Base | 253 |
| Hoth      | 151 | Tatooine        | 329 |
| Iridonia  | 226 | Vandor-1        | 199 |
| Kessel    | 0   | Zygerria        | 160 |

# Idea: Admissibility



Inadmissible  
(pessimistic) heuristics  
break optimality by  
trapping good plans on  
the frontier



Admissible (optimistic)  
heuristics slow down  
bad plans but never  
outweigh true costs

# Heuristic functions

For the 8-puzzle

- **Avg. solution cost is about 22 steps**
  - (branching factor  $\leq 3$ )
  - (branching factor  $\leq 3$ )
  - A good heuristic function can reduce the search process



Start State



Goal State

# Example Admissible heuristics

For the 8-puzzle:

$h_{oop}(n)$  = number of out of place tiles

$h_{md}(n)$  = total Manhattan distance (i.e., #  
of moves from desired location of  
each tile)

$$h_{oop}(S) = 8$$

$$h_{md}(S) = 3+1+2+2+2+3+3+2 = 18$$



Start State



Goal State

# Relaxed problems

A problem with fewer restrictions on the actions than the original is called a *relaxed problem*

*The cost of an optimal solution to a relaxed problem is an admissible heuristic for the original problem*

If the rules of the 8-puzzle are relaxed so that a tile can move *anywhere*, then  $h_{oop}(n)$  gives the shortest solution

If the rules are relaxed so that a tile can move to *any adjacent square*, then  $h_{md}(n)$  gives the shortest solution

# Defining Heuristics: $h(n)$

Cost of an exact solution to a *relaxed* problem (fewer restrictions on operator)

Constraints on *Full* Problem:

A tile can move from square A to square B *if A is adjacent to B and B is blank.*

- Constraints on *relaxed* problems:
  - A tile can move from square A to square B *if A is adjacent to B.* ( $h_{md}$ )
  - A tile can move from square A to square B *if B is blank.*
  - A tile can move from square A to square B. ( $h_{oop}$ )

# Dominance: A metric on better heuristics

If  $h_2(n) \geq h_1(n)$  for all  $n$  (both admissible)

- then  $h_2$  *dominates*  $h_1$

So  $h_2$  is optimistic, but more accurate than  $h_1$

- $h_2$  is therefore better for search
- Notice:  $h_{md}$  dominates  $h_{oop}$

Typical search costs (average number of nodes expanded):

$d=12$  Iterative Deepening Search = 3,644,035 nodes

$A^*(h_{oop}) = 227$  nodes,  $A^*(h_{md}) = 73$  nodes

$d=24$  IDS = too many nodes

$A^*(h_{oop}) = 39,135$  nodes,  $A^*(h_{md}) = 1,641$  nodes

# The best and worst admissible heuristics

$h^*(n)$  - the (unachievable) Oracle heuristic

- $h^*(n)$  = the true distance from the n to goal

$h_{\text{we're here already}}(n) = h_{\text{teleportation}}(n) = 0$

Admissible: both yes!!!

$h^*(n)$  *dominates all other heuristics*

$h_{\text{teleportation}}(n)$  *is dominated by all heuristics*

# A\* search is Optimal

AIMA 3.5



# A\* search

Best-known form of best-first search.

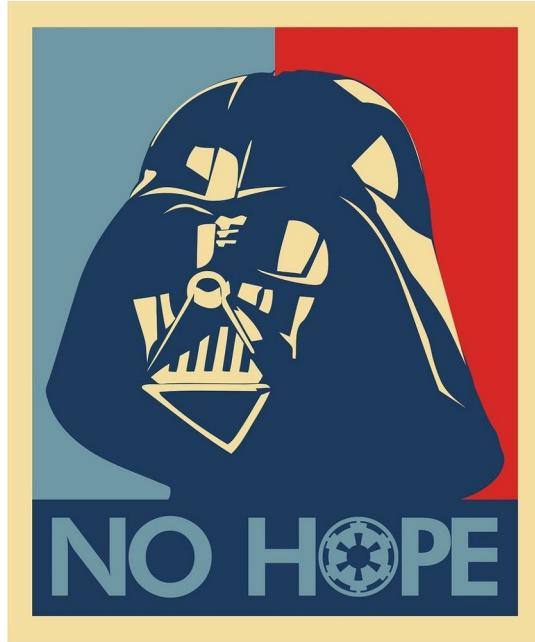
Key Idea: avoid expanding paths that are already expensive, but expand most promising first.

*Simple idea:  $f(n) = g(n) + h(n)$*

- $g(n)$  the actual cost (so far) to *reach* the node
- $h(n)$  estimated cost to *get from the node to the goal*
- $f(n)$  estimated *total cost* of path through  $n$  to goal

Implementation: Frontier queue as priority queue by increasing  $f(n)$  (*as expected...*)

# Key: Admissibility



Inadmissible (pessimistic) heuristics break optimality by pushing good plans too far back on the frontier, which means they may never get expanded.



Admissible (optimistic) heuristics slow down bad plans but never outweigh true costs. That means that the true best plan will always be expanded.

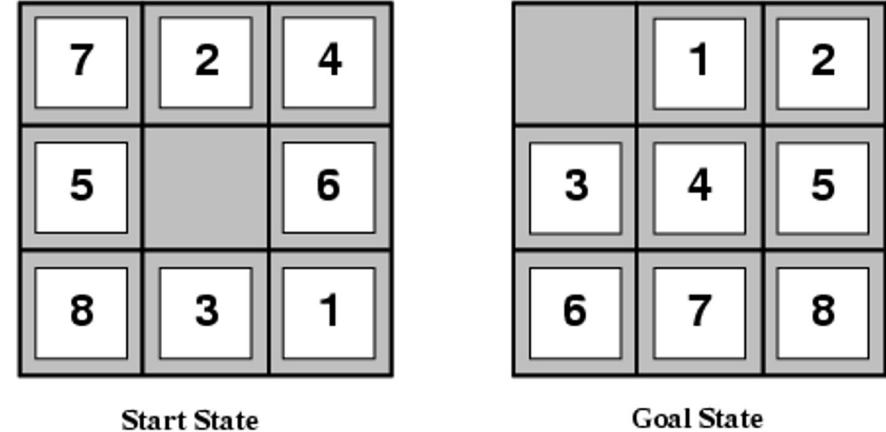
# Admissible Heuristics

A heuristic  $h$  is *admissible* (optimistic) if:

$$0 \leq h(n) \leq h^*(n)$$

where  $h^*(n)$  is the true cost to a nearest goal

Is Manhattan Distance admissible?



Coming up with admissible heuristics is most of what's involved in using A\* in practice.

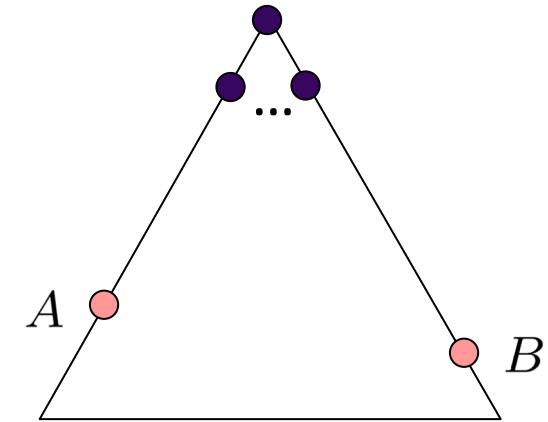
# Optimality of A\* Tree Search

Assume:

A is an optimal goal node

B is a suboptimal goal node

$h$  is admissible



Claim:

A will exit the frontier before B

Slide credit: Dan Klein and Pieter  
Abbeel  
<http://ai.berkeley.edu>

# Optimality of A\* Tree Search

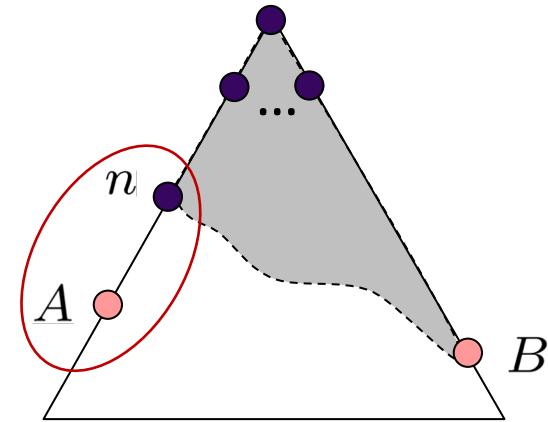
Proof:

Imagine B is on the frontier

Some ancestor  $n$  of A is on the frontier, too (maybe A!)

Claim:  $n$  will be expanded before B

- $f(n)$  is less or equal to  $f(A)$



$$f(n) = g(n) + h(n) \quad \text{Definition of f-cost}$$

$$f(n) \leq g(A) \quad \text{Admissibility of } h$$

$$g(A) = f(A) \quad h = 0 \text{ at a goal}$$

Slide credit: Dan Klein and Pieter

Abbeel

<http://ai.berkeley.edu>

# Optimality of A\* Tree Search

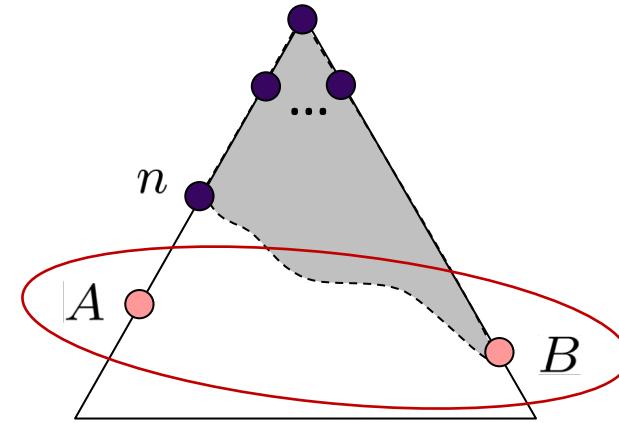
Proof:

Imagine B is on the frontier

Some ancestor  $n$  of A is on the frontier, too (maybe A!)

Claim:  $n$  will be expanded before B

- $f(n)$  is less or equal to  $f(A)$
- $f(A)$  is less than  $f(B)$



$g(A) < g(B)$     B is suboptimal  
 $f(A) < f(B)$      $h = 0$  at a goal

Slide credit: Dan Klein and Pieter

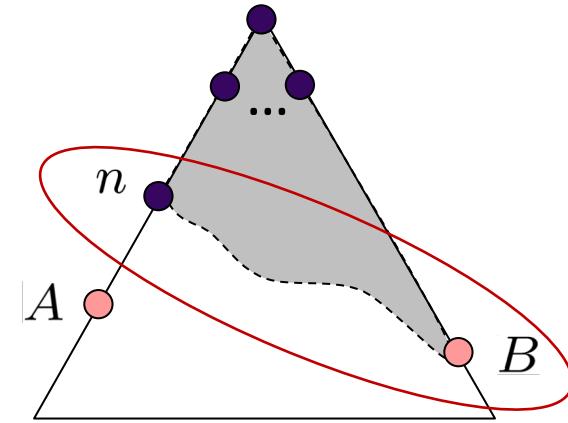
Abbeel

<http://ai.berkeley.edu>

# Optimality of A\* Tree Search

Proof:

- Imagine B is on the frontier
- Some ancestor  $n$  of A is on the frontier, too (maybe A!)
- Claim:  $n$  will be expanded before B
  - $f(n)$  is less or equal to  $f(A)$
  - $f(A)$  is less than  $f(B)$
  - $n$  expands before B
- All ancestors of A expand before B
- A expands before B
- A\* search is optimal



$$f(n) \leq f(A) < f(B)$$

Slide credit: Dan Klein and Pieter

Abbeel

<http://ai.berkeley.edu>

# Consistent Heuristics

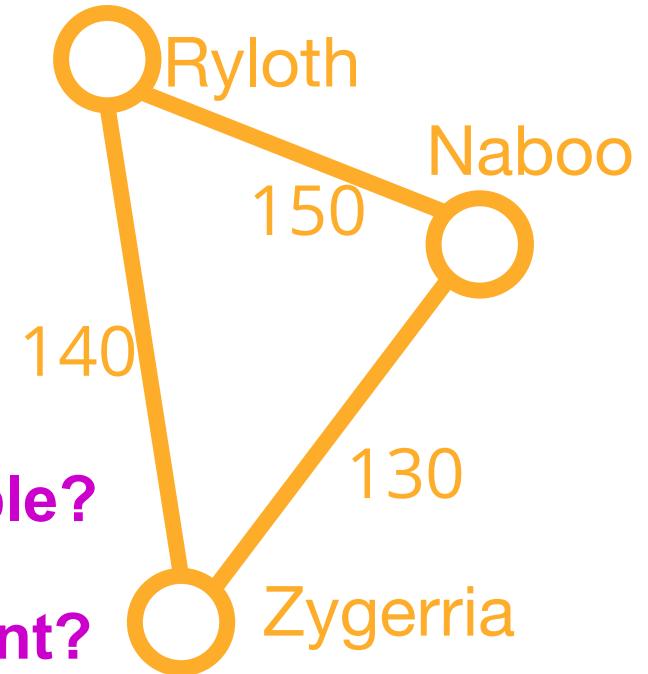
This ensures that A\* is optimal in **graph search** in addition to tree search.

**Consistent heuristics** are a subset of admissible heuristics, which provide even stronger guarantees.

They obey the **triangle inequality**.

| Planet   | $h_1$ |
|----------|-------|
| Ryloth   | 150   |
| Zygerria | 5     |
| Naboo    | 0     |

| Planet   | $h_2$ |
|----------|-------|
| Ryloth   | 150   |
| Zygerria | 120   |
| Naboo    | 0     |



Is  $h_1$  admissible?

Is  $h_1$  consistent?

Is  $h_2$  admissible?

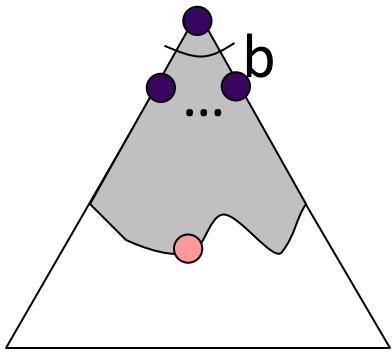
Is  $h_2$  consistent?

# Properties of A\*

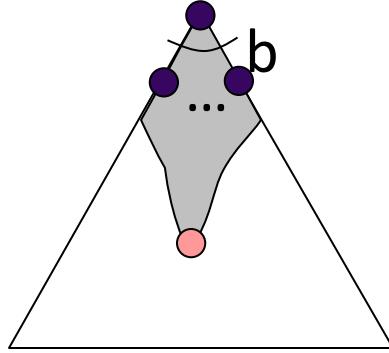
Slide credit: Dan Klein and Pieter  
Abbeel  
<http://ai.berkeley.edu>

# Properties of A\*

Uniform-Cost



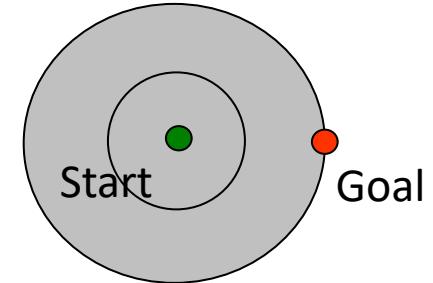
A\*



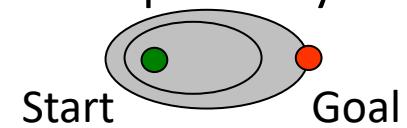
Slide credit: Dan Klein and Pieter  
Abbeel  
<http://ai.berkeley.edu>

# UCS vs A\* Contours

Uniform-cost expands equally in all “directions”



A\* expands mainly toward the goal, but does hedge its bets to ensure optimality



Slide credit: Dan Klein and Pieter  
Abbeel  
<http://ai.berkeley.edu>

# A\* Applications

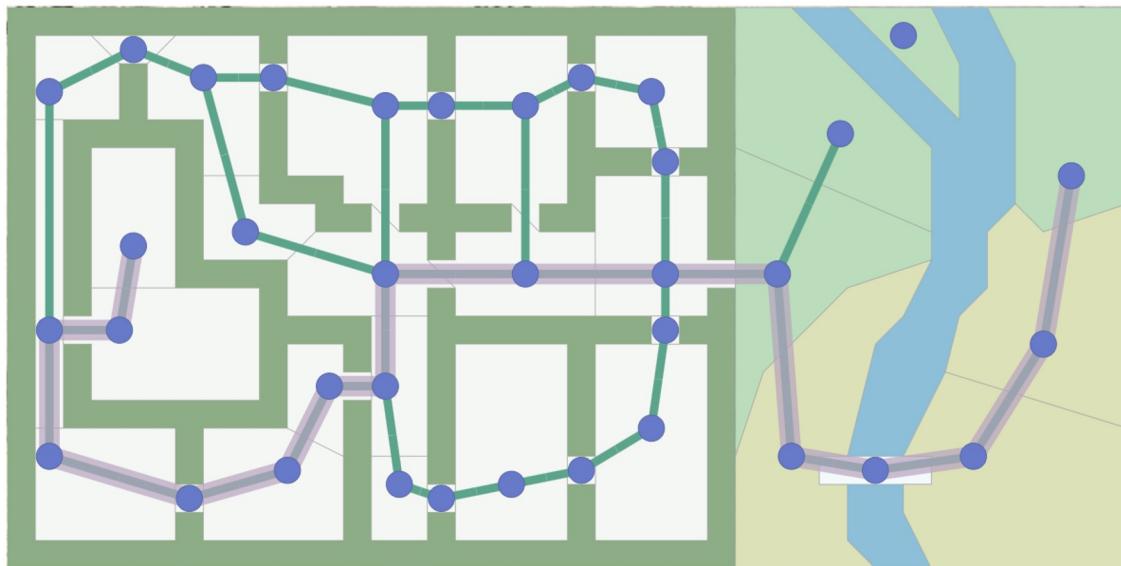
Pathing / routing problems (A\* is in your GPS!)

Video games

Robot motion planning

Resource planning problems

...



# Supplemental Reading

I recommend this A\* tutorial by Amit Patel of Red Blob Games

This is not the solution to the HW assignment.

<https://www.redblobgames.com/pathfinding/a-star/introduction.html>

## Introduction to the A\* Algorithm

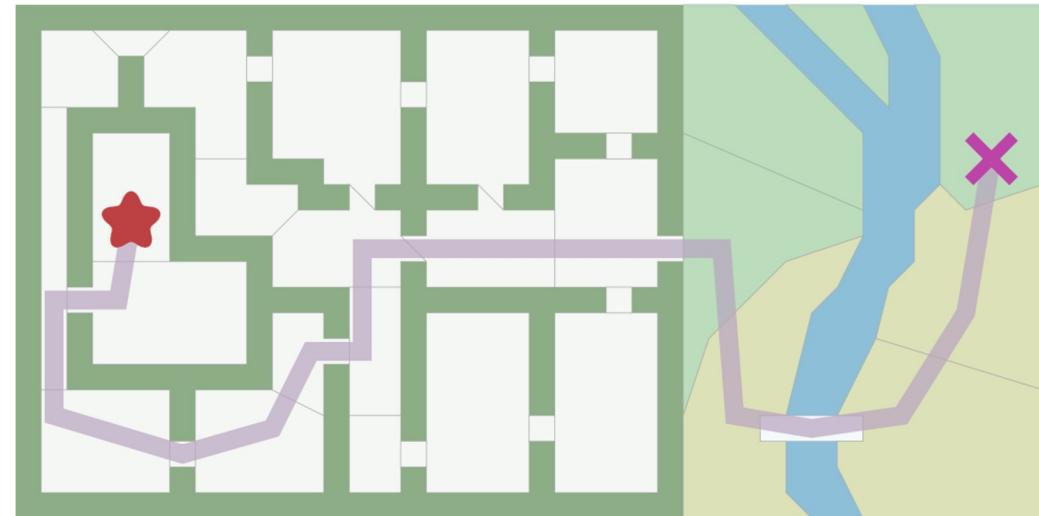
from Red Blob Games

[Home](#) [Blog](#) [Links](#) [Twitter](#) [About](#)

Search

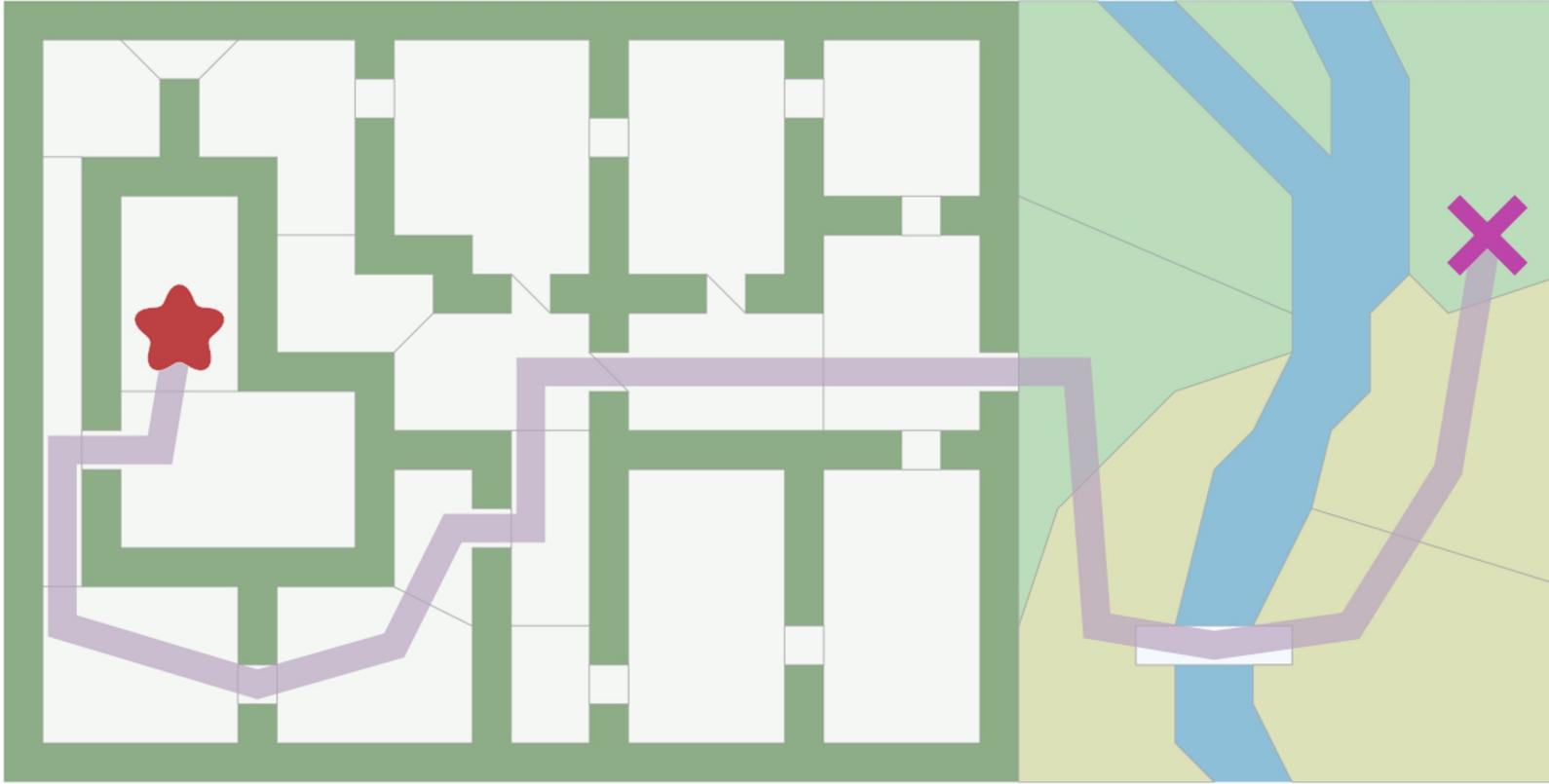
Created 26 May 2014, updated Aug 2014, Feb 2016, Jun 2016

In games we often want to find paths from one location to another. We're not only trying to find the shortest distance; we also want to take into account travel time. Move the blob  (start point) and cross  (end point) to see the shortest path.



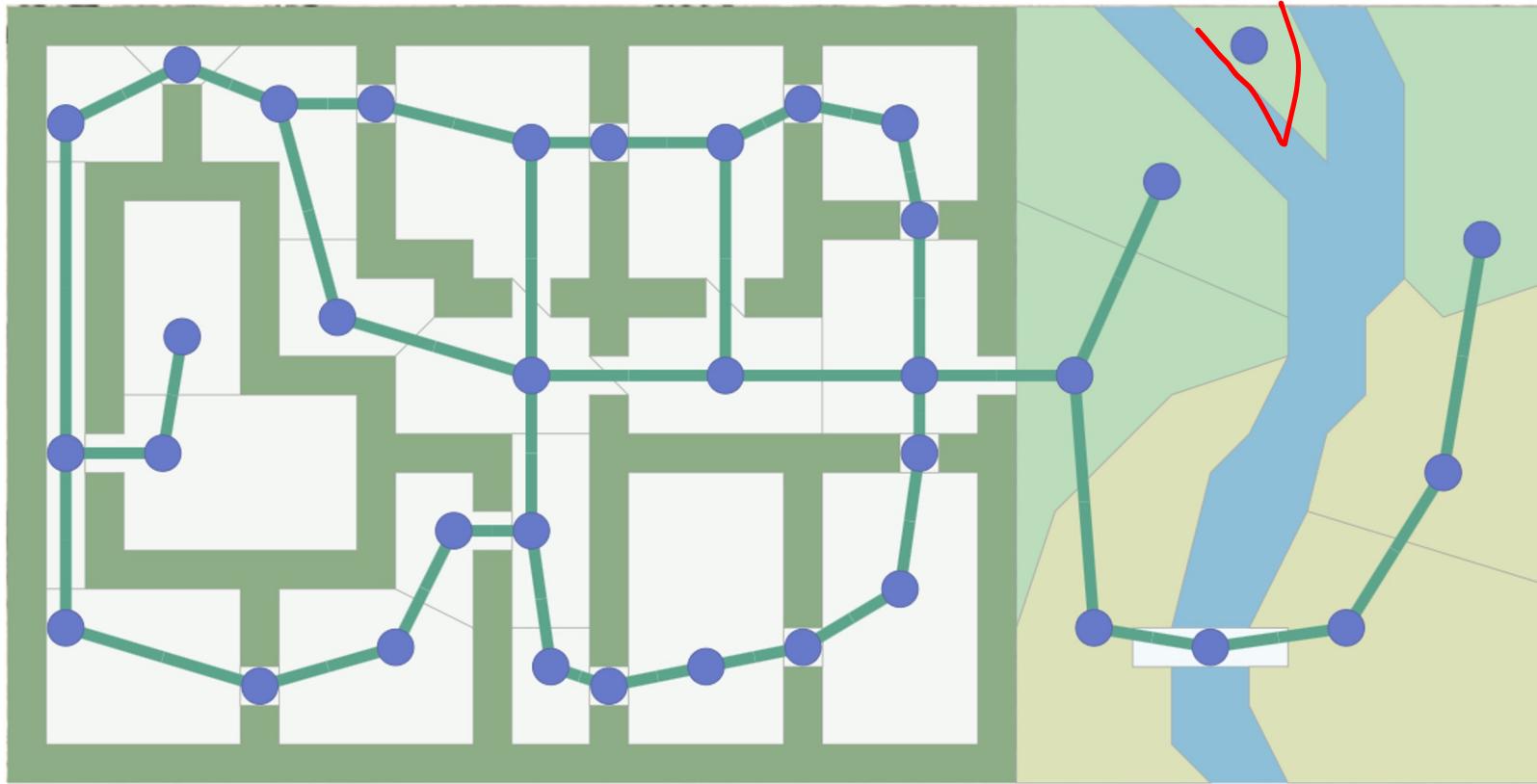
To find this path we can use a *graph search* algorithm, which works when the map is represented as a graph. A\* is a popular choice for graph search. **Breadth First Search** is the simplest of the graph search algorithms, so let's start there, and we'll work our way up to A\*.

# Pathfinding in Games



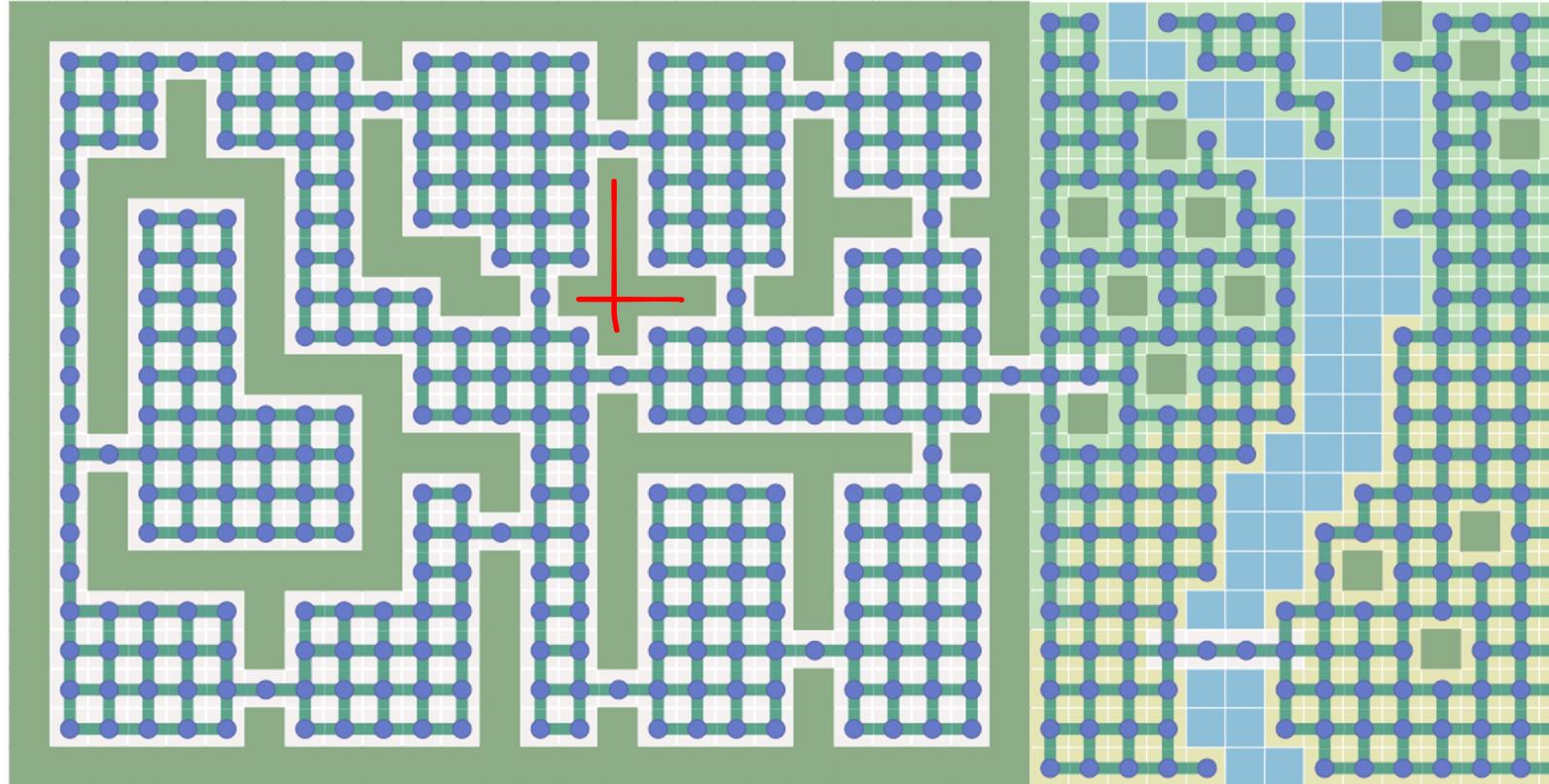
<https://www.redblobgames.com/pathfinding/a-star/introduction.html>

# Pathfinding in Games



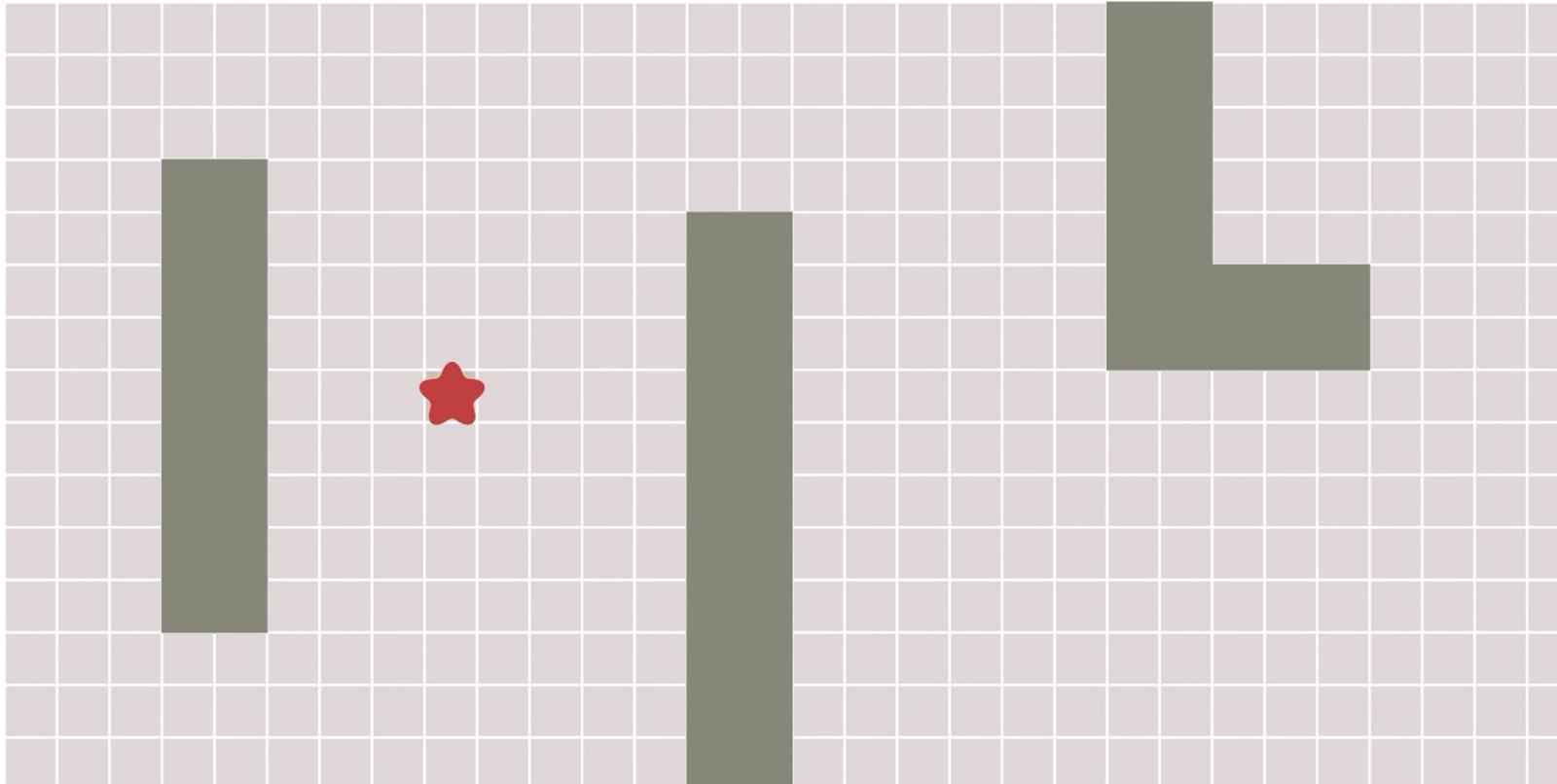
<https://www.redblobgames.com/pathfinding/a-star/introduction.html>

# Pathfinding in Games



<https://www.redblobgames.com/pathfinding/a-star/introduction.html>

# Breadth First Search



<https://www.redblobgames.com/pathfinding/a-star/introduction.html>

# BFS in 10 lines of Python

---

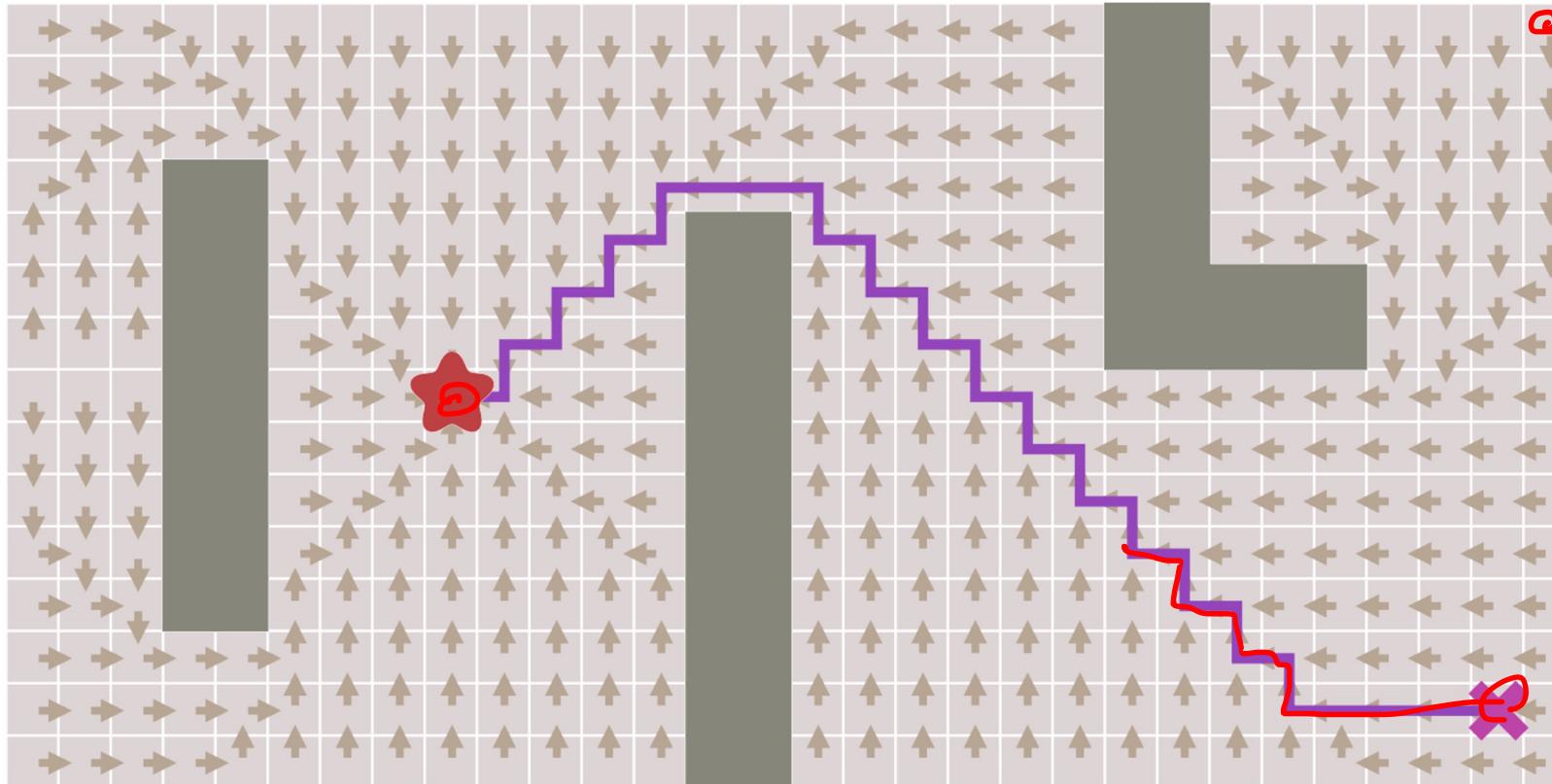
```
frontier = Queue()
frontier.put(start ★)
visited = {}
visited[start] = True

while not frontier.empty():
    current = frontier.get()
    for next in graph.neighbors(current):
        if next not in visited:
            frontier.put(next)
            visited[next] = True
```

---

<https://www.redblobgames.com/pathfinding/a-star/introduction.html>

# Finding the shortest path



<https://www.redblobgames.com/pathfinding/a-star/introduction.html>

# Finding the shortest path

---

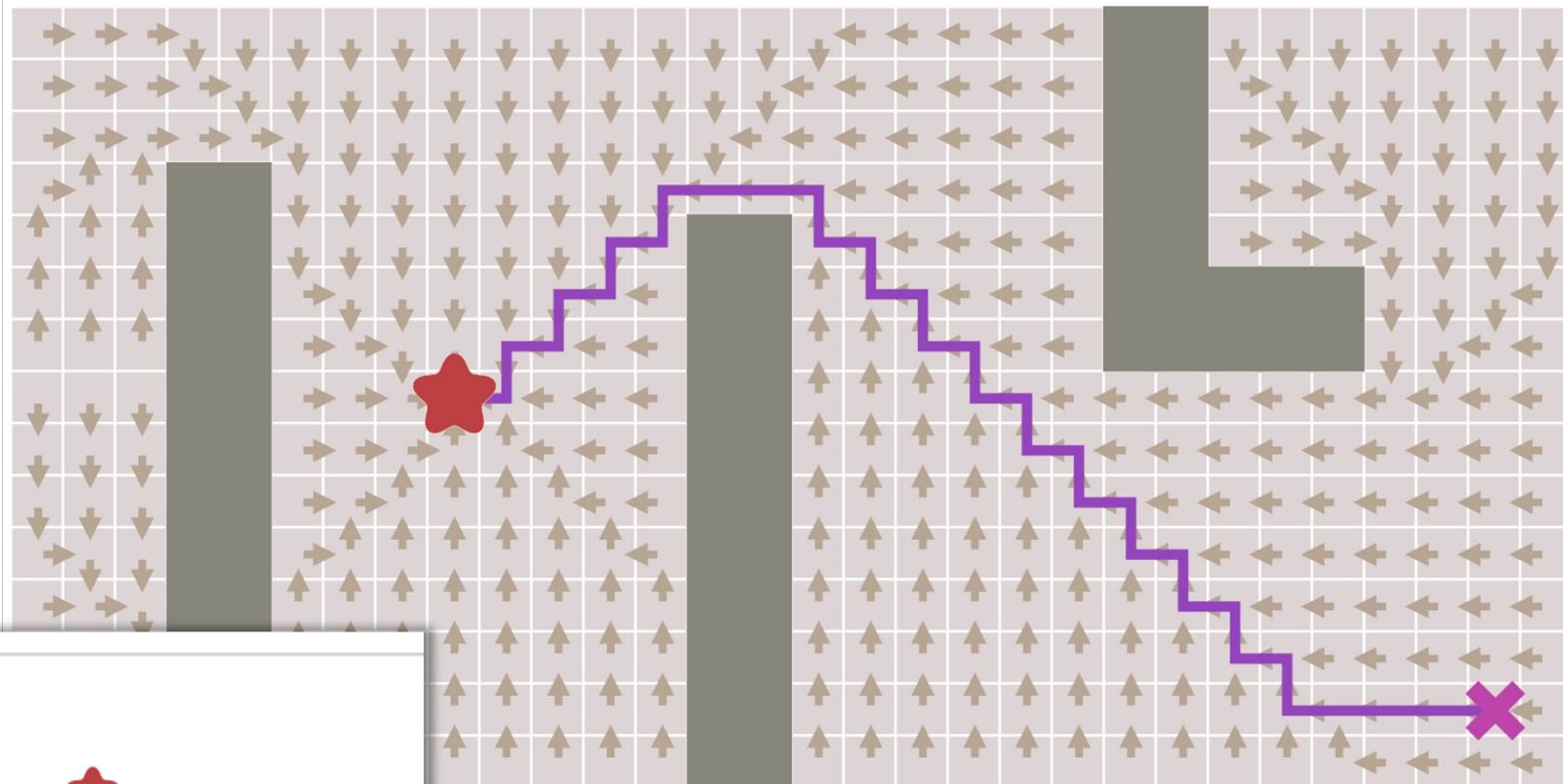
```
frontier = Queue()
frontier.put(start ★)
came_from = {}
came_from[start] = None

while not frontier.empty():
    current = frontier.get()
    for next in graph.neighbors(current):
        if next not in came_from:
            frontier.put(next)
            came_from[next] = current
```

---

<https://www.redblobgames.com/pathfinding/a-star/introduction.html>

# Finding the shortest path



```
current = goal ✎  
path = []  
while current != start: ★  
    path.append(current)  
    current = came_from[current]  
path.append(start) # optional  
path.reverse() # optional
```