CS540 Introduction to Artificial Intelligence
Lecture 16

Young Wu

Based on lecture slides by Jerry Zhu, Yingyu Liang, and Charles Dyer

July 4, 2020

Uniformed vs. Informed Search
Motivation

- Uninformed search means only the goal G and the successor functions s' are given.

- Informed search means which non-goal states are better is also known.

Heuristic
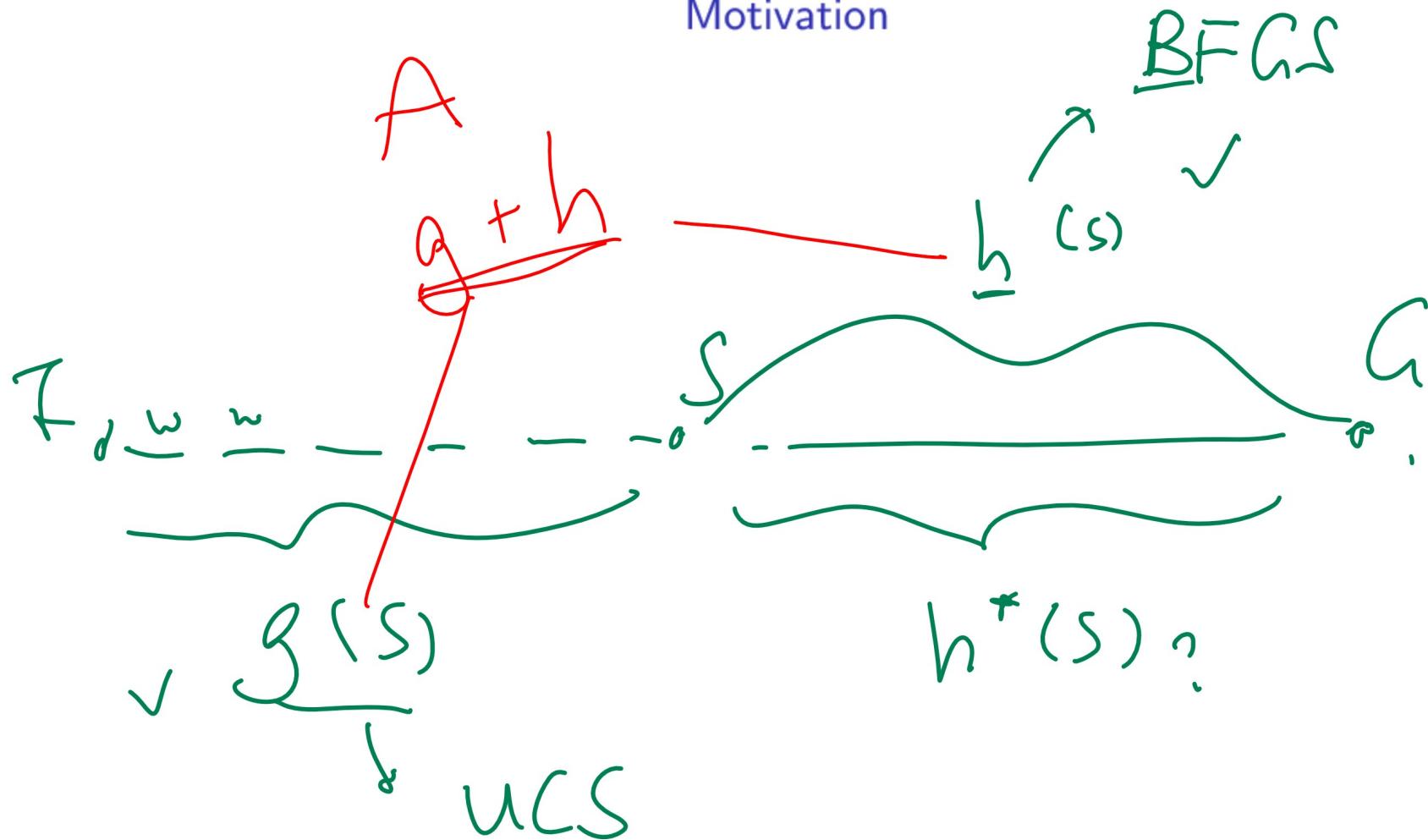
Motivation

- The additional information is usually given as a heuristic cost from a state s to the goal.

- The cost of the path from the start to a vertex s in the frontier is $g(s)$.

- The cost from s to the goal, $h^{\star}(s)$, is estimated by $h(s)$. This estimate may not be accurate.

$$h(s) \approx h^{\star}(s)$$

Heuristic Diagram

## Motivation

Uniform Cost Search
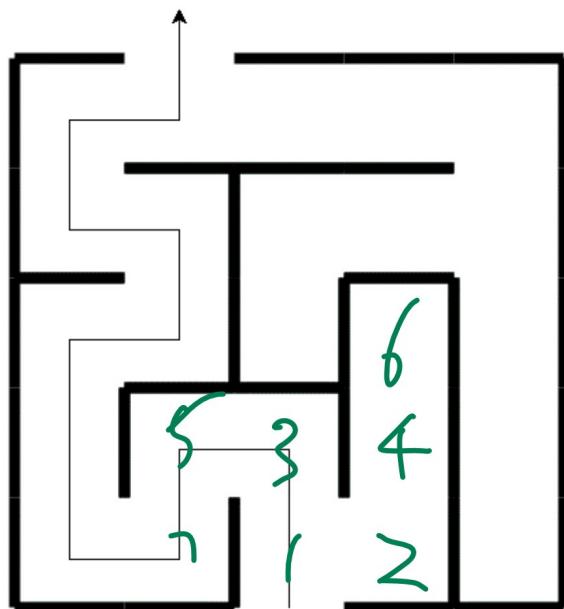
Description

- Expand the vertices with the lowest current path cost $g(s)$ first.

- It is BFS with a priority queue based on $g(s)$.

- It is equivalent to BFS if $c = 1$ is constant on all edges.

- It is also called Dijkstra's Algorithm.
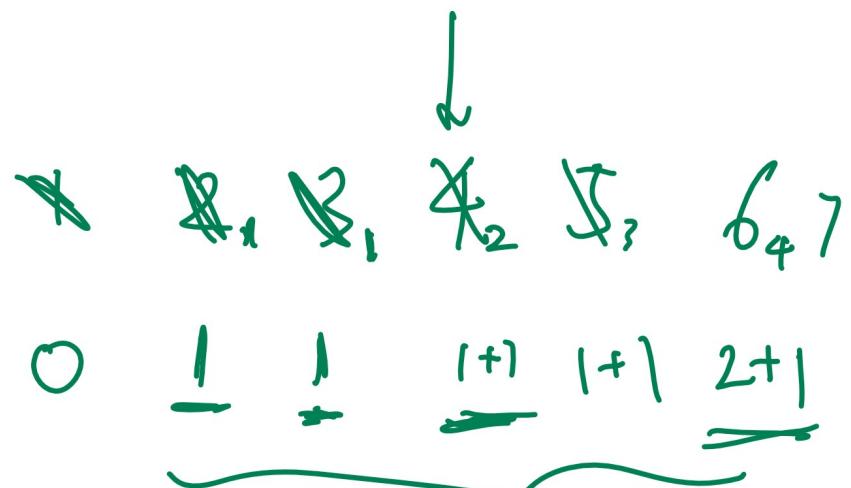
577

Uniform Cost Search Maze Example

Definition

PQ

$g(s)$

$A \quad B_1, \quad B_1, \quad A_2 \quad J_3 \quad 6_4 \; 7$

$0 \quad 1 \quad 1 \quad 1+1 \quad 1+1 \quad 2+1$
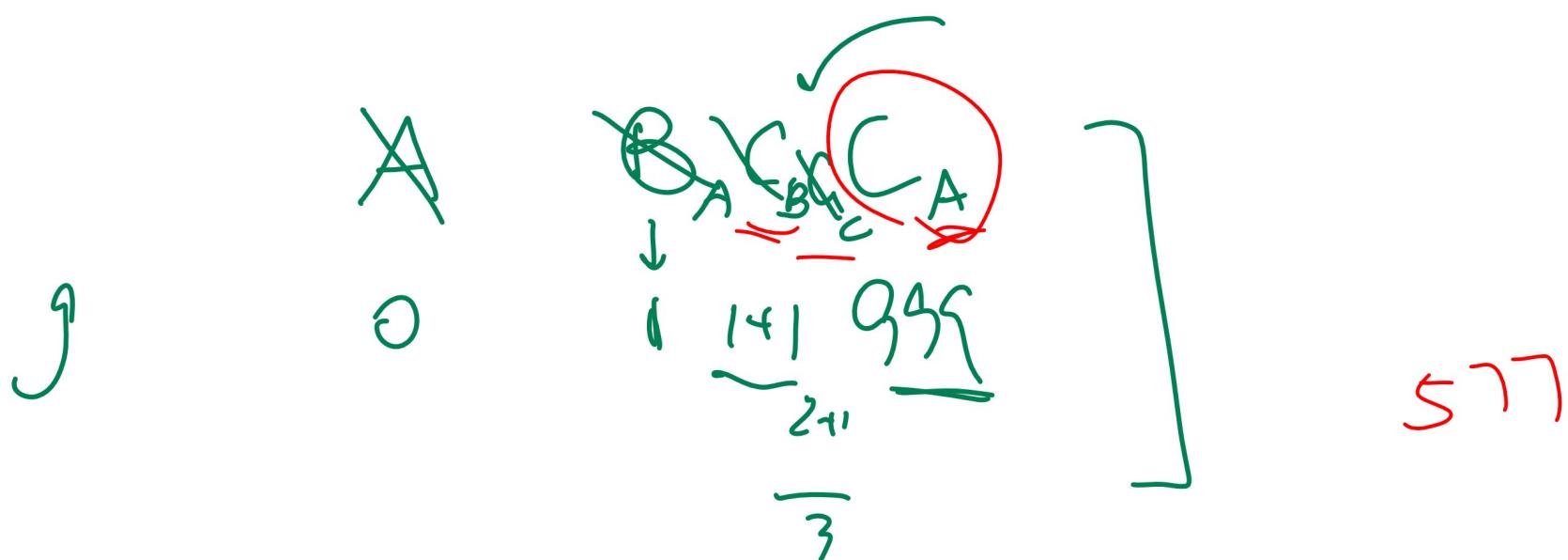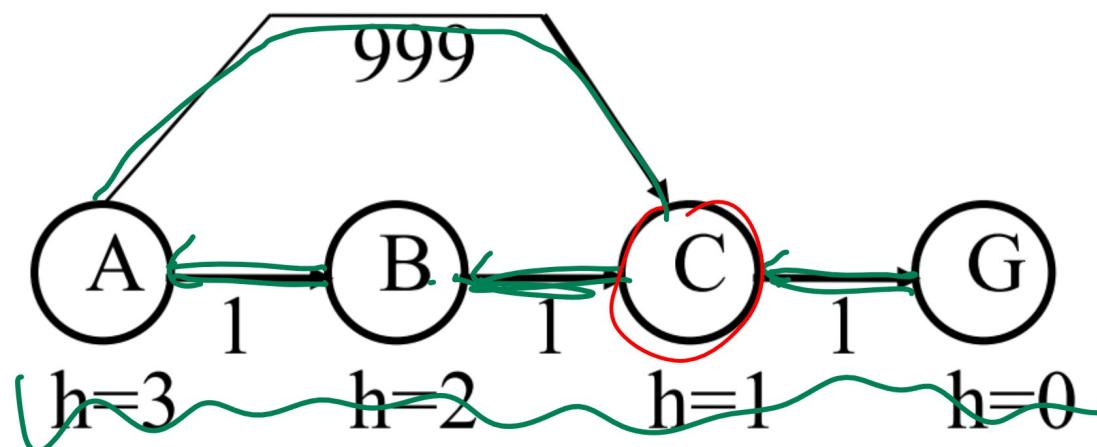
Uniform Cost Search Simple Example

Definition

Uniform Cost Search
Algorithm

- Input: a weighted digraph (V, E, c), initial states I and goal states G.

- Output: a path from I to G.

- EnQueue initial states into a priority queue Q. Here, Q is ordered by $g(s)$ for $s \in Q$.

$$Q = I$$

- While Q is not empty and goal is not deQueued, deQueue Q and enQueue its successors.

$$s = Q_{(0)} = \arg \min_{s \in Q} g(s)$$
$$Q = Q + s'(s)$$

Uniform Cost Search Performance
Discussion

- UCS is complete.

- UCS is optimal with any c.

Best First Greedy Search
Description

- Expand the vertices with the lowest heuristic cost $h(s)$ first.
- Use a priority queue based on $h(s)$.

Greedy Search Maze Example
Definition

Greedy Search Simple Example

Definition

Best First Greedy Search

Algorithm

- Input: a weighted digraph (V, E, c), initial states I and goal states G, and the heuristic function $h(s), s \in V$.

- Output: a path from I to G.

- EnQueue initial states into a priority queue Q. Here, Q is ordered by $h(s)$ for $s \in Q$.

$$Q = I$$

- While Q is not empty and goal is not deQueued, deQueue Q and enQueue its successors.

$$s = Q_{(0)} = \arg \min_{s \in Q} h(s)$$
$$Q = Q + s'(s)$$

Best First Greedy Search Performance
Discussion

- Greedy is incomplete.
- Greedy is not optimal.

A Search

Description

- Expand the vertices with the lowest total cost $g(s) + h(s)$ first.

- Use a priority queue based on $g(s) + h(s)$.

- A stands for Always be optimistic?

A Search Maze Example

Definition

A Search Simple Example 1

Definition

A Search Simple Example 2
Definition

A Search Simple Example 3

Definition

A Search

Algorithm

- Input: a weighted digraph (V, E, c), initial states I and goal states G, and the heuristic function $h(s), s \in V$.

- Output: a path from I to G.

- EnQueue initial states into a priority queue Q. Here, Q is ordered by $g(s) + h(s)$ for $s \in Q$.

$$Q = I$$

- While Q is not empty and goal is not deQueued, deQueue Q and enQueue its successors.

$$s = Q_{(0)} = \arg\min_{s \in Q} g(s) + h(s)$$

$$Q = Q + s'(s)$$

A Search Performance
Discussion

- A is complete.
- A is not optimal.

A Star Search

Description

- $A^\star$ search is A search with an admissible heuristic.

Admissible Heuristic

Definition

- A heuristic is admissible if it never over estimates the true cost.

$$0 \leqslant h(s) \leqslant h^\star(s)$$

Admissible Heuristic 8 Puzzle Example
Definition

Dominated Heuristic
Definition

- One heuristic, h_1, is dominated by another, h_2, if:

$$h_1\left(s\right) \leqslant h_2\left(s\right) \leqslant h^\star\left(s\right), \ \forall\, s \in S$$

- If h_2 dominates h_1, then h_2 is better than h_1 since $A^\star$ using h_1 expands at least as many states (or more) than $A^\star$ using h_2.

- If h_2 dominated h_1, $A^\star$ with h_2 is better informed than $A^\star$ with h_1.

Non-Optimal Heuristic
Definition

- If optimality is not required and a satisfying solution is acceptable, then the heuristic should be as close as possible, either under or over, to the actual cost.

- This results in fewer states being expanded compared to using poor but admissible heuristics.

A Star Search with Revisit, Part I

Algorithm

- Input: a weighted digraph (V, E, c), initial states I and goal states G, and the heuristic function $h(s), s \in V$.

- Output: a path with minimum cost from I to G.

- EnQueue initial states into a priority queue Q. Here, Q is ordered by $g(s) + h(s)$ for $s \in Q$.

$$Q = I$$
$$g(I) = 0$$
$$g(s) = \infty, \text{ for } s \notin I$$

- Initialize the list of visited vertices, P.

$$P = \varnothing$$

A Star Search with Revisit, Part II
Algorithm

- While Q is not empty and goal is not deQueued, deQueue Q, put it on P and enQueue its successors to Q, and update the cost functions.

$$s = Q_{(0)} = \arg\min_{s \in Q} g(s) + h(s)$$

$$P = P + s$$

$$Q = Q + s'(s), \text{ update } g(s') = \min\{g(s'), g(s) + c(s, s')\}$$

A Search Performance

Discussion

- $A^\star$ is complete.
- $A^\star$ is optimal.

Iterative Deepening A Star Search
Discussion

- $A^\star$ can use a lot of memory.

- Do path checking without expanding any vertex with $g(s) + h(s) > 1$.

- Do path checking without expanding any vertex with $g(s) + h(s) > 2$.

- ...

- Do path checking without expanding any vertex with $g(s) + h(s) > d$.

Iterative Deepening A Star Search Performance
Discussion

- IDA* is complete.

- IDA* is optimal.

- IDA* is more costly than $A^\star$.

Beam Search

Discussion

- Version 1: Keep a priority queue with fixed size k. Only keep the top k vertices and discard the rest.

- Version 2: Only keep the vertices that are at most ε worse than the best vertex in the queue. ε is called the beam width.

Beam Search Performance

Discussion

- Beam is incomplete.

- Beam is not optimal.