CS540 Introduction to Artificial Intelligence
Lecture 16

Young Wu
Based on lecture slides by Jerry Zhu and Yingyu Liang

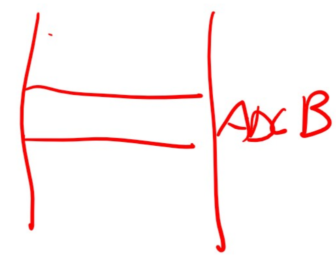
July 18, 2019

Bridge and Torch Game, Part I

Quiz (Participation)

- Four people with one flashlight (torch) want to go across a river. The bridge can hold two people at a time, and they must cross with the flashlight. The time it takes for each person to cross the river:

A	B	C	D
1	2	3	4



- What is the minimum total time required for everyone to cross the river?

$2+1+3+1+4 = 11$

- A: 10, B: 11, C: 12, D: 13, E: 14

Bridge and Torch Game, Part II

Quiz (Participation)

- Four people with one flashlight (torch) want to go across a river. The bridge can hold two people at a time, and they must cross with the flashlight. The time it takes for each person to cross the river:

A	B	C	D
1	2	4	5

2 + 1 + 4 + 1 + 5

13

- What is the minimum total time required for everyone to cross the river?
- A: 10, B: 11, C: 12, D: 13, E: 14

ABCD

12

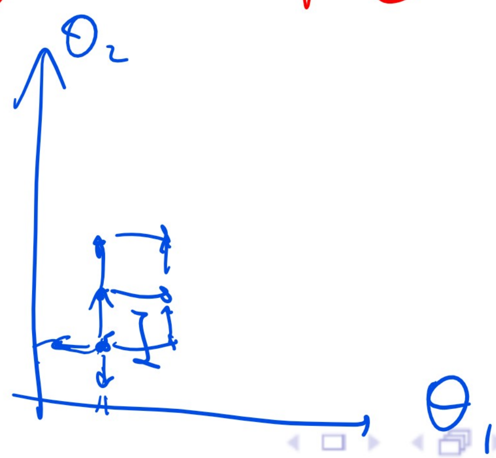
2 + 1 + 5 + 2 + 2

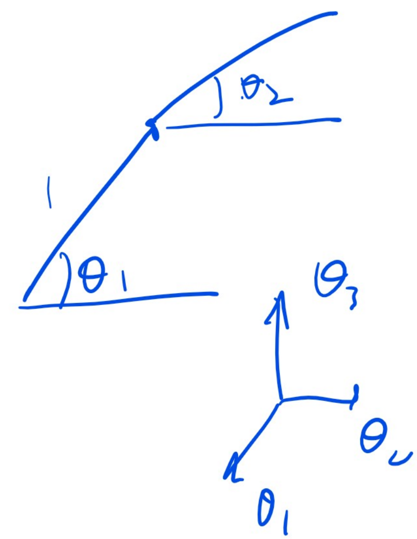
Uniformed vs. Informed Search

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.

Slide: Configuration Space





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

unknown. $h^*(s)$

$$I \longrightarrow S \longrightarrow G$$

sum of cost from I to s $g(s)$

???

$h(s) \approx h^*(s)$

① uniform cost search ; order the Queue by $g(s)$

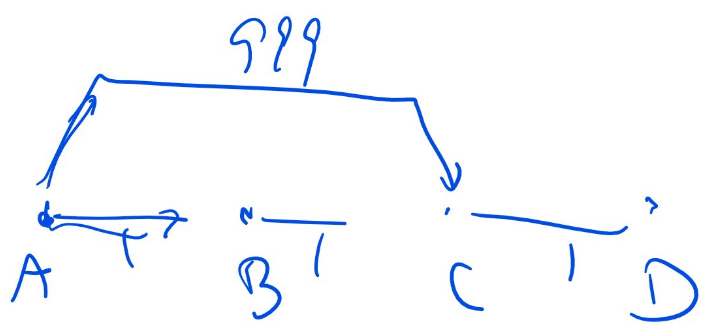
② best first greedy search: —— $h(s)$

③ A search —— $g(s) + h(s)$

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.



$s = \cancel{A} \quad \cancel{B} \; \cancel{C} \; \cancel{D} \; C$

$g(s) = \cancel{0} \quad \cancel{1} \; \cancel{2} \; \cancel{3} \; 999$

UCS Example, Part I

Quiz (Graded)

- Spring 2017 Midterm Q1
- Given the following adjacency matrix. Find UCS expansion path.

–	S	A	B	C	D	E	G
S	$h = 6$	2	1	–	–	–	9
A	–	$h = 0$	–	2	3	–	–
B	–	–	$h = 6$	–	2	4	–
C	–	–	–	$h = 4$	–	–	4
D	–	–	–	–	$h = 1$	–	4
E	–	–	–	–	–	$h = 10$	–
G	–	–	–	–	–	–	$h = 0$

UCS Example, Part II

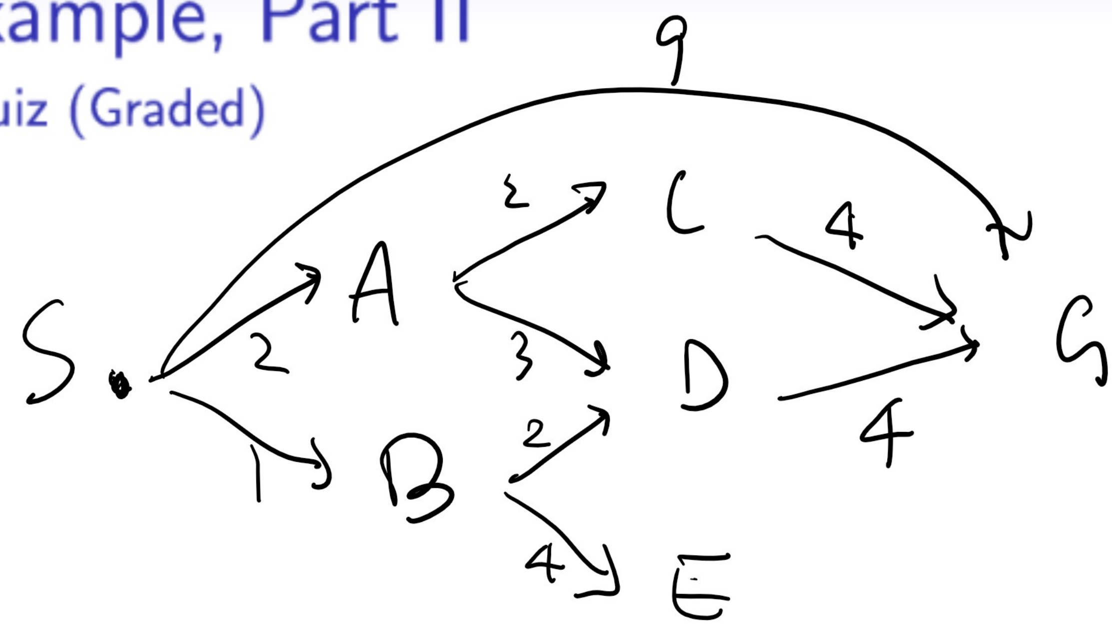
Quiz (Graded)

UCS

expansion path?

Q10

- A: S B A D C E G
- B: S B D G
- C: S A G
- D: S G
- E: S A D B D G



s =	~~S~~	~~A~~	~~B~~	G	D	E	C	D
g(s) =	~~0~~	~~2~~	~~1~~	9	3	5	...	

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.

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

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

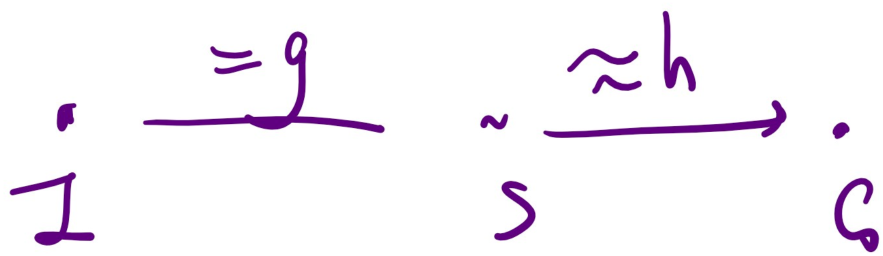
Uniform Cost Search Performance

Discussion

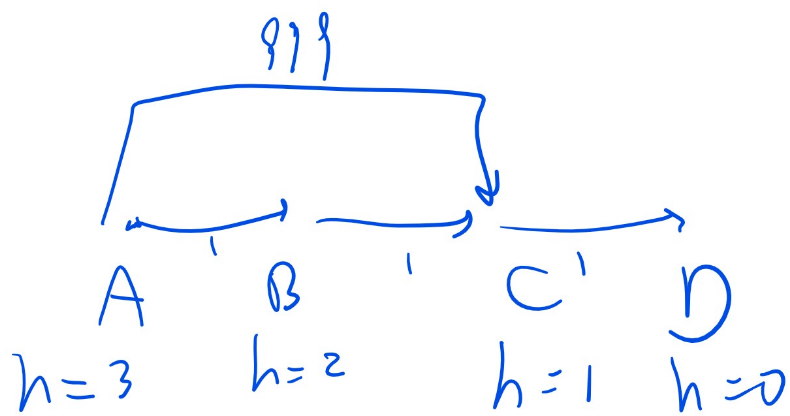
- UCS is complete.
- UCS is optimal with any c.

Best First Greedy Search

Description



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



s	~~A~~	~~D~~	~~C~~ $_{\uparrow A}$	B
h(s)	~~3~~	~~4~~	~~1~~	2

$A \xrightarrow{999} C \xrightarrow{1} D = 1000$

Greedy Example, Part I

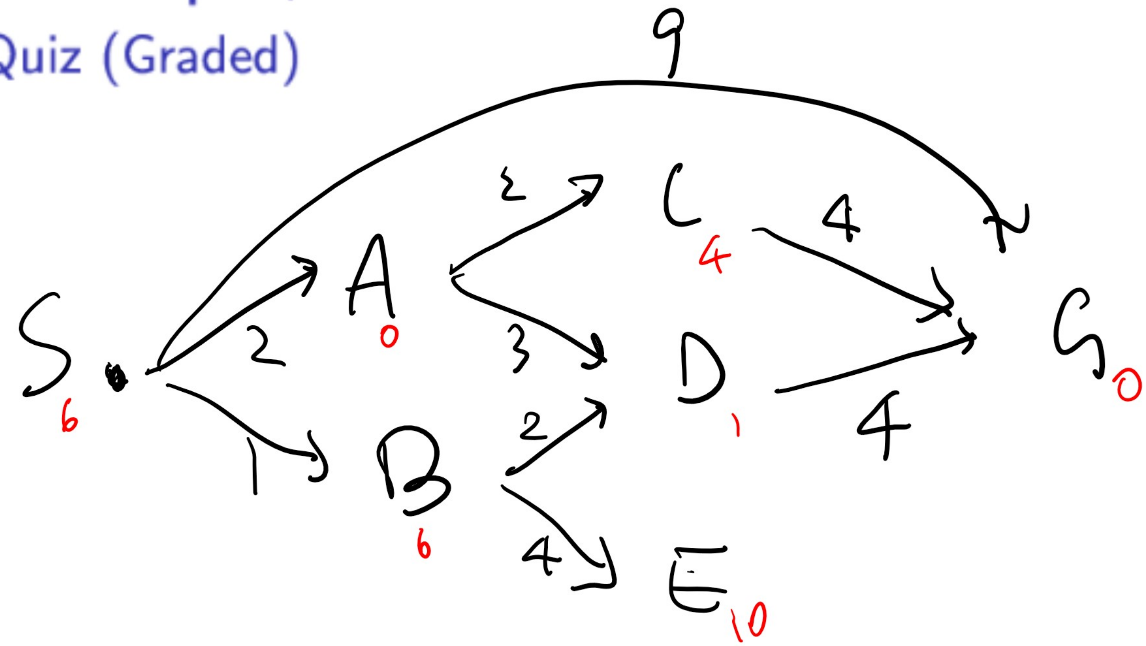
Quiz (Graded)

- Given the following adjacency matrix. Find Greedy Search expansion path.

–	S	A	B	C	D	E	G
S	$h = 6$	2	1	–	–	–	9
A	–	$h = 0$	–	2	3	–	–
B	–	–	$h = 6$	–	2	4	–
C	–	–	–	$h = 4$	–	–	4
D	–	–	–	–	$h = 1$	–	4
E	–	–	–	–	–	$h = 10$	–
G	–	–	–	–	–	–	$h = 0$

Greedy Example, Part II

Quiz (Graded)



- A: S B A D C E G
- B: S B D G
- C: S A G
- D: S G
- E: S A D B D G

~~S~~ ~~A~~ ~~G~~ D C B

~~6~~ ~~0~~ ~~0~~ 1 4 6

Solution. $G \xrightarrow[9]{} G$

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 Example, Part I

Quiz (Graded)

- Given the following adjacency matrix. Find *A* Search expansion path.

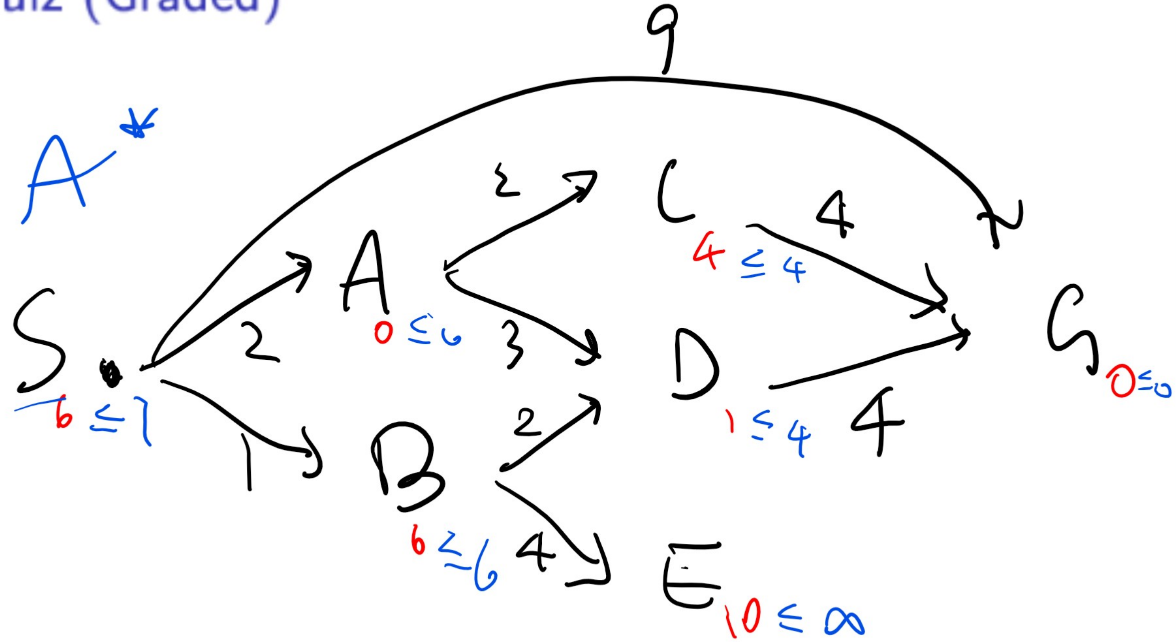
–	S	A	B	C	D	E	G
S	$h = 6$	2	1	–	–	–	9
A	–	$h = 0$	–	2	3	–	–
B	–	–	$h = 6$	–	2	4	–
C	–	–	–	$h = 4$	–	–	4
D	–	–	–	–	$h = 1$	–	4
E	–	–	–	–	–	$h = 10$	–
G	–	–	–	–	–	–	$h = 0$

A Search Example, Part II

Quiz (Graded)

$h \le h^*$

admissible → A^*



- A: S B A D C E G
- B: S B D G
- C: S A G
- D: S G
- E: S A D B D G

$g + h(s) = 0 + 6$

S	A	B	G	C
0+6	2+0	1+6	9+0	4+4

D: 5+1

D	D	B	C	C	G	E
3+1	5+1	1+6	7+0	4+4	9+0	5+10

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. → h is estimate → can be wrong

if h is always optimistic → always underestimating

→ A^* is 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

Quiz (Graded)

Q14 A, B, D, E

- Which ones (select multiple) of the following are admissible heuristic function for the 8 Puzzle?
  - A: $h(s)$ = number of tiles in the wrong position.
  - $\boxed{\text{B}}$: $h(s) = 0.$ $\Longrightarrow$ UCS
  - ~~C: $h(s) = 1.$~~
  - D: $h(s)$ = sum of Manhattan distance between each tile and its goal location.
  - E: $h(s)$ = sum of Euclidean distance between each tile and its goal location.

$h(G) = 1 \not\leq h^*(G) = 0$

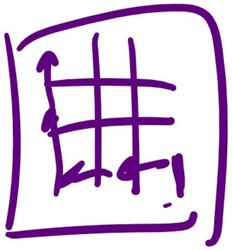
jump anywhere

1	2	3
4	5	6
7	8	

move without other tiles,

$0 \leq h_E \leq h_D \leq h^*$

move diagonally



$h(s) \leq h^*(s)\ \forall s$

Admissible Heuristic General Example

Quiz (Graded)

Q1b A C E

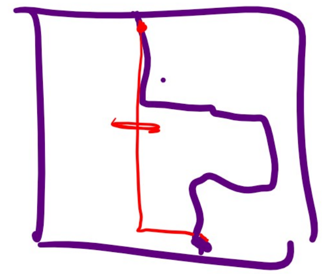
- Which ones (sel~~ect multiple~~) of the following are admissible heuristic function?
- A: $h(s) = h^{\star}(s)$.
- ~~B~~: $h(s) = \max\{2, h^{\star}(s)\}$.
- C: $h(s) = \min\{2, h^{\star}(s)\}$.
- ~~D~~: $h(s) = h^{\star}(s) - 2$.
- E: $h(s) = \sqrt{h^{\star}(s)}$. ← h^{α} must be integer

$0 \leq h(s) \leq h^{*}(s)$

$h^{*}(G) = 0$

Dominated Heuristic

Definition



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

$$h_1(s) \leqslant h_2(s) \leqslant h^{\star}(s), \ \forall\, s \in S$$

h_2 is always closer to h^*

$h(s)$ = Manhattan distance from cell to goal

for HW.

- 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 Maze Example

Quiz (Graded)

A Star Search with Revisit Example, Part I

Quiz (Graded)

- Given the following adjacency matrix. Find $A^\star$ Search expansion path.

–	S	A	B	C	D	E	G
S	$h = 6$	2	1	–	–	–	9
A	–	$h = 0$	–	2	3	–	–
B	–	–	$h = 6$	–	2	4	–
C	–	–	–	$h = 4$	–	–	4
D	–	–	–	–	$h = 1$	–	4
E	–	–	–	–	–	$h = 10$	–
G	–	–	–	–	–	–	$h = 0$

A Star Search with Revisit Example, Part II

Quiz (Graded)

- A: S B A D C E G
- B: S B D G
- C: S A G
- D: S G
- E: S A D B D G

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.

$$P = \emptyset$$

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.

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

$$P = P + s$$

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

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.