Heuristic Search
&
Local Search

CS171 Week 3 Discussion
July 7, 2016

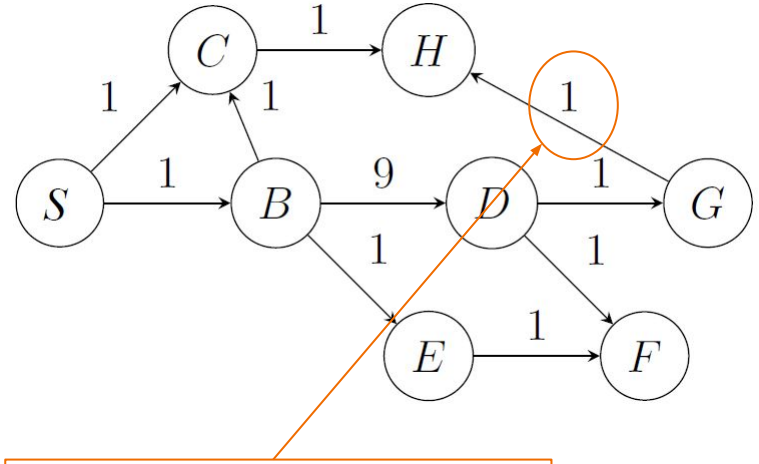
Heuristic Search: Exercise

Consider the following graph, with initial state **S** and goal **G**, and the heuristic function h.

Fill in the form using **greedy best-first search.** Indicate the f value in parenthesis after the node label, e.g. A(8).

Assume the algorithm **does not re-visit each node.**

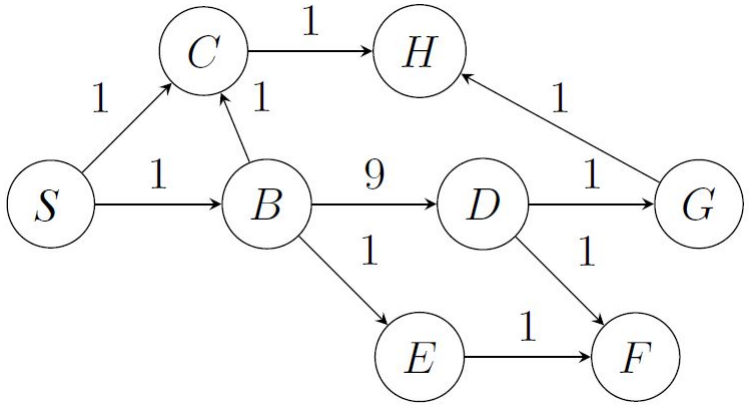
node, n	h(n)
S	8
B	7
C	3
D	1
E	4
F	6
G	0
H	100



Note: Normally there shouldn't be paths pointing out from the goal node. You may choose to ignore it in this problem.

Heuristic Search: Exercise

Consider the following graph, with initial state **S** and goal **G**, and the heuristic function h.



Fill in the form using **greedy best-first search.** Indicate the f value in parenthesis after the node label, e.g. A(8).

Assume the algorithm **does not re-visit each node.**

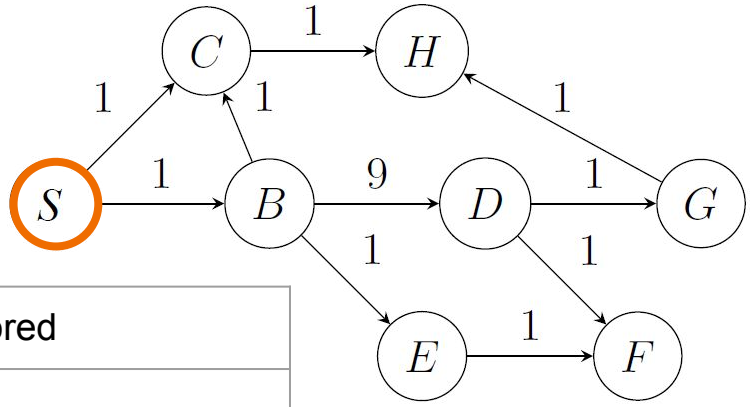
Current Node	Frontier	Explored
S(8)	C(3), B(7)	S(8)

node, n	h(n)
S	8
B	7
C	3
D	1
E	4
F	6
G	0
H	100

Heuristic Search: Exercise

Greedy best-first search

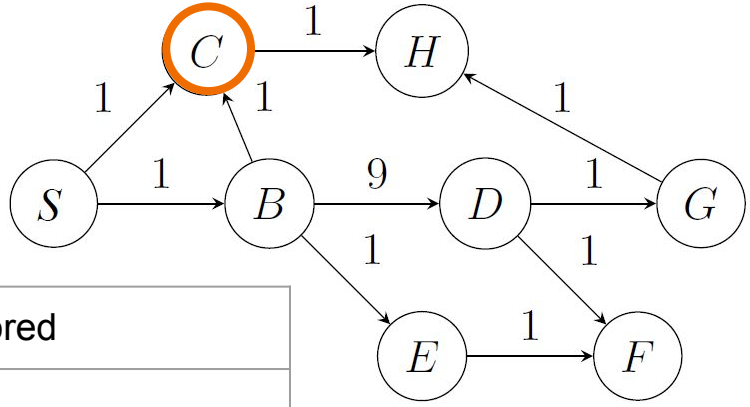
Recall: f(n) = h(n)



node, n	h(n)
S	8
B	7
C	3
D	1
E	4
F	6
G	0
H	100

Current Node	Frontier	Explored
S(8)	C(3), B(7)	S(8)

Heuristic Search: Exercise

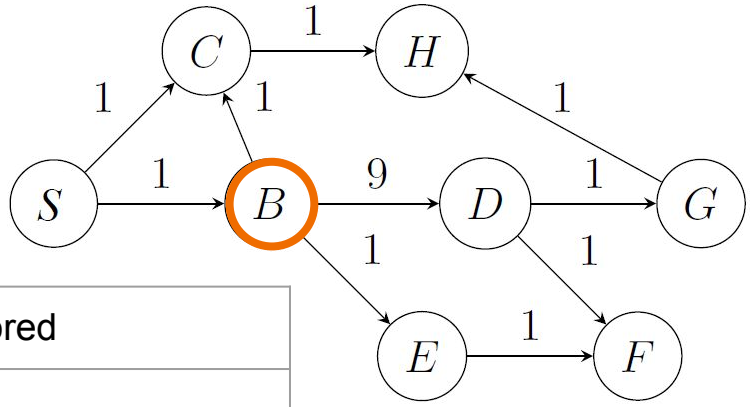


Greedy best-first search

Current Node	Frontier	Explored
S(8)	C(3), B(7)	S
C(3)	B(7), H(100)	S, C

node, n	h(n)
S	8
B	7
C	3
D	1
E	4
F	6
G	0
H	100

Heuristic Search: Exercise

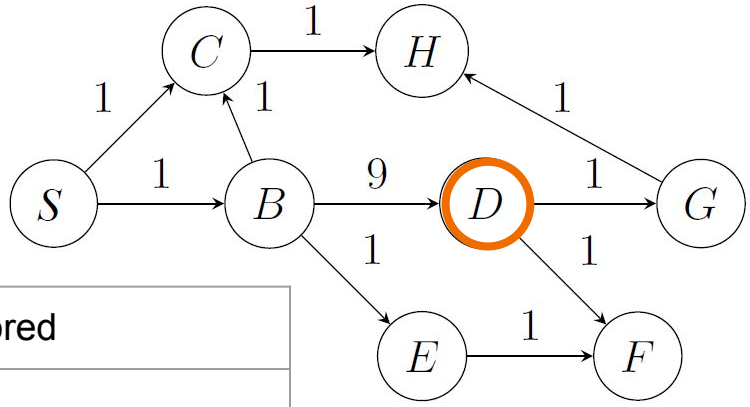


Greedy best-first search

Current Node	Frontier	Explored
S(8)	C(3), B(7)	S
C(3)	B(7), H(100)	S, C
B(7)	H(100), D(1), E(4)	S, C, B

node, n	h(n)
S	8
B	7
C	3
D	1
E	4
F	6
G	0
H	100

Heuristic Search: Exercise

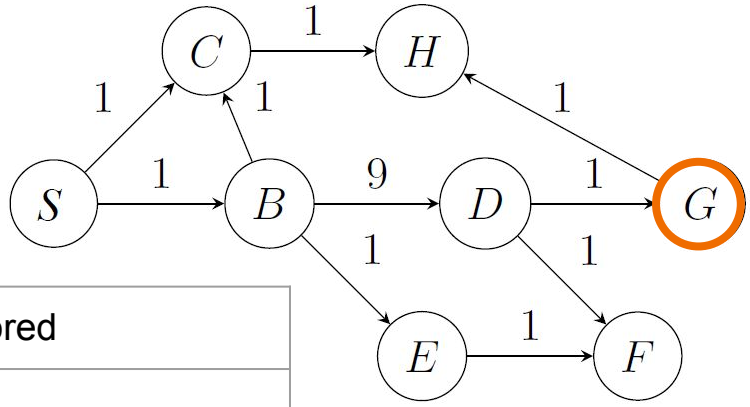


Greedy best-first search

Current Node	Frontier	Explored
S(8)	C(3), B(7)	S
C(3)	B(7), H(100)	S, C
B(7)	H(100), D(1), E(4)	S, C, B
D(1)	H(100), E(4), G(0), F(6)	S, C, B, D

node, n	h(n)
S	8
B	7
C	3
D	1
E	4
F	6
G	0
H	100

Heuristic Search: Exercise



Greedy best-first search

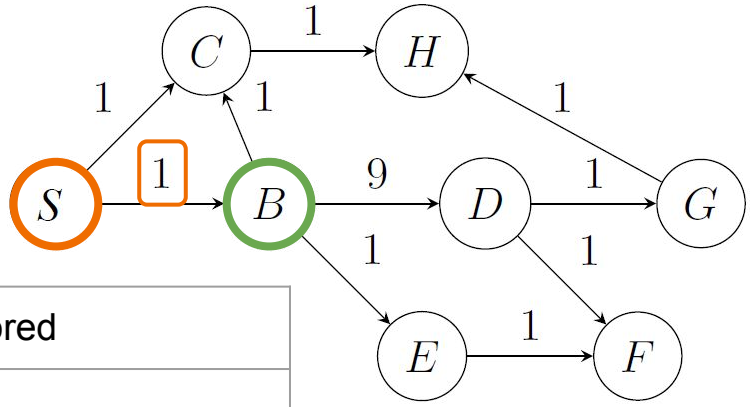
Current Node	Frontier	Explored
S(8)	C(3), B(7)	S
C(3)	B(7), H(100)	S, C
B(7)	H(100), D(1), E(4)	S, C, B
D(1)	H(100), E(4), G(0), F(6)	S, C, B, D
G(0)		

node, n	h(n)
S	8
B	7
C	3
D	1
E	4
F	6
G	0
H	100

Heuristic Search: Exercise

A* search

Recall: f(n) = g(n) + h(n)



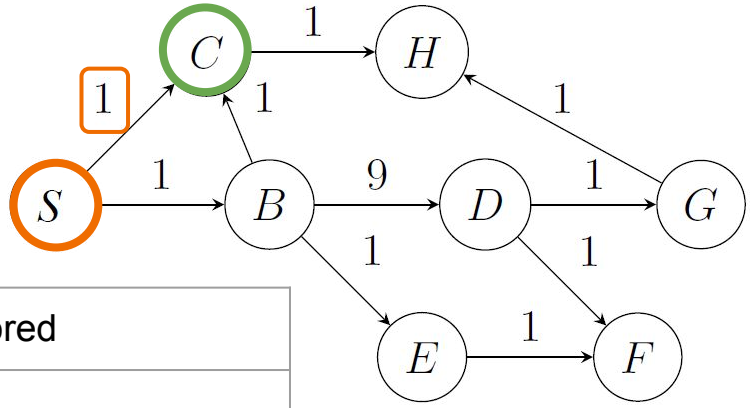
node, n	h(n)
S	8
B	7
C	3
D	1
E	4
F	6
G	0
H	100

Current Node	Frontier	Explored
S(8)	B(8)	

Heuristic Search: Exercise

A* search

Recall: f(n) = g(n) + h(n)



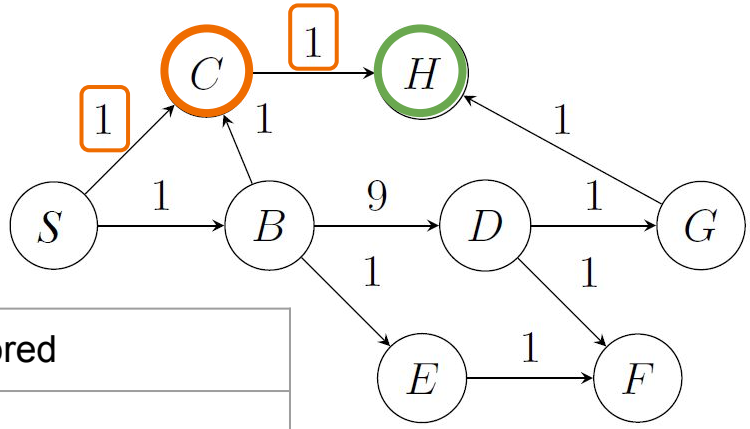
node, n	h(n)
S	8
B	7
C	3
D	1
E	4
F	6
G	0
H	100

Current Node	Frontier	Explored
S(8)	B(8), C(4)	S

Heuristic Search: Exercise

A* search

Recall: f(n) = g(n) + h(n)



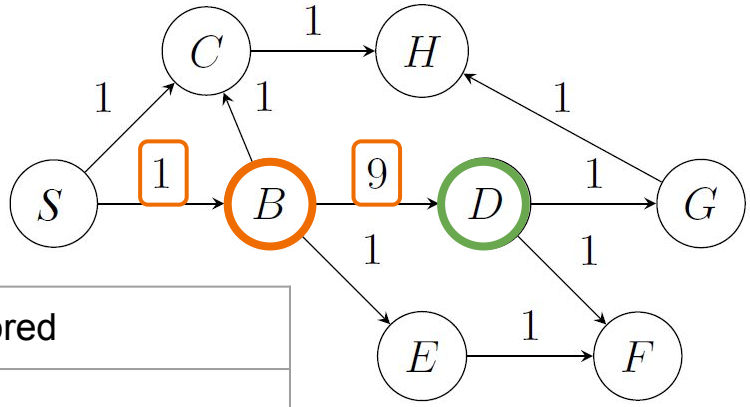
Current Node	Frontier	Explored
S(8)	B(8), C(4)	S
C(4)	B(8), H(102)	S, C

node, n	h(n)
S	8
B	7
C	3
D	1
E	4
F	6
G	0
H	100

Heuristic Search: Exercise

A* search

Recall: f(n) = g(n) + h(n)



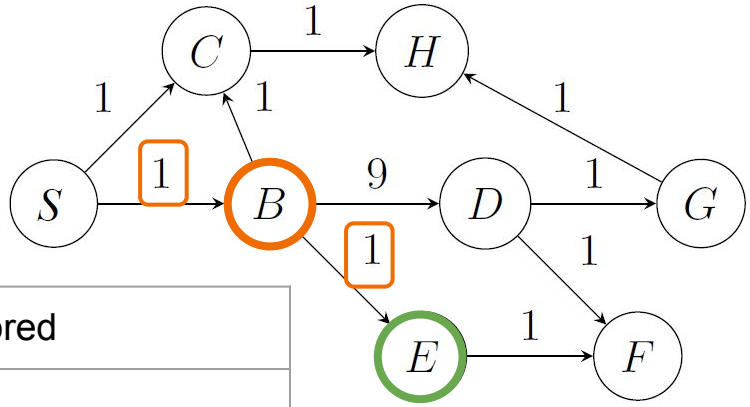
node, n	h(n)
S	8
B	7
C	3
D	1
E	4
F	6
G	0
H	100

Current Node	Frontier	Explored
S(8)	B(8), C(4)	S
C(4)	B(8), H(102)	S, C
B(8)	H(102), D(11)	

Heuristic Search: Exercise

A* search

Recall: f(n) = g(n) + h(n)



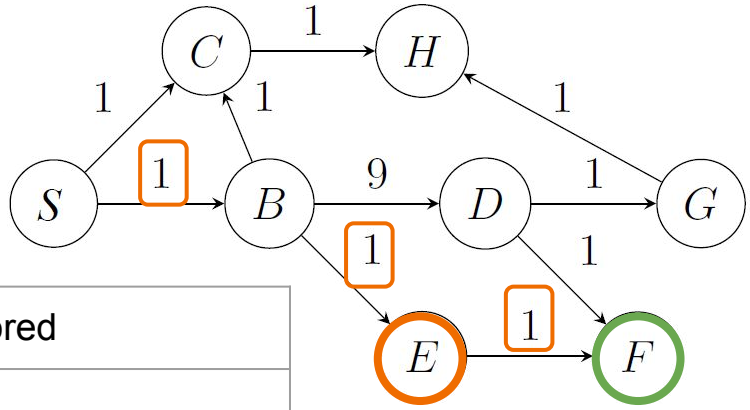
Current Node	Frontier	Explored
S(8)	B(8), C(4)	S
C(4)	B(8), H(102)	S, C
B(8)	H(102), D(11), E(6)	S, C, B

node, n	h(n)
S	8
B	7
C	3
D	1
E	4
F	6
G	0
H	100

Heuristic Search: Exercise

A* search

Recall: f(n) = g(n) + h(n)



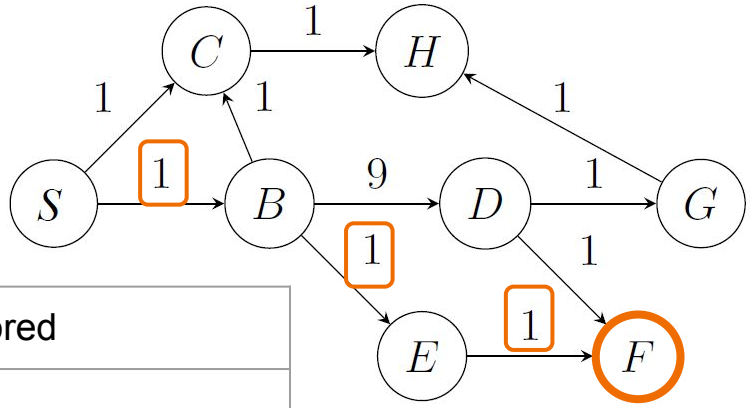
Current Node	Frontier	Explored
S(8)	B(8), C(4)	S
C(4)	B(8), H(102)	S, C
B(8)	H(102), D(11), E(6)	S, C, B
E(6)	H(102), D(11), F(9)	S, C, B, E

node, n	h(n)
S	8
B	7
C	3
D	1
E	4
F	6
G	0
H	100

Heuristic Search: Exercise

A* search

Recall: f(n) = g(n) + h(n)



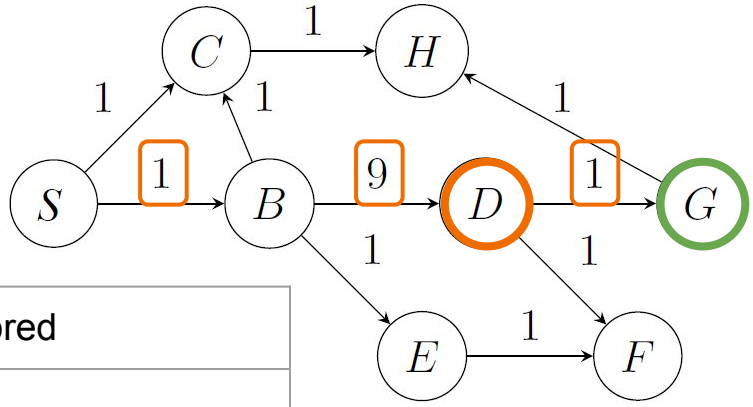
Current Node	Frontier	Explored
S(8)	B(8), C(4)	S
C(4)	B(8), H(102)	S, C
B(8)	H(102), D(11), E(6)	S, C, B
E(6)	H(102), D(11), F(9)	S, C, B, E
F(9)	H(102), D(11)	S, C, B, E, F

node, n	h(n)
S	8
B	7
C	3
D	1
E	4
F	6
G	0
H	100

Heuristic Search: Exercise

A* search

Recall: f(n) = g(n) + h(n)



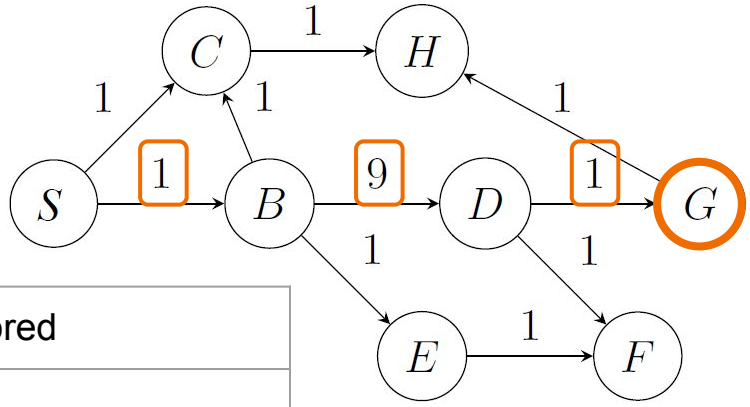
Current Node	Frontier	Explored
S(8)	B(8), C(4)	S
C(4)	B(8), H(102)	S, C
B(8)	H(102), D(11), E(6)	S, C, B
E(6)	H(102), D(11), F(9)	S, C, B, E
F(9)	H(102), D(11)	S, C, B, E, F
D(11)	H(102), G(11)	S, C, B, E, F, D

node, n	h(n)
S	8
B	7
C	3
D	1
E	4
F	6
G	0
H	100

Heuristic Search: Exercise

A* search

Recall: f(n) = g(n) + h(n)

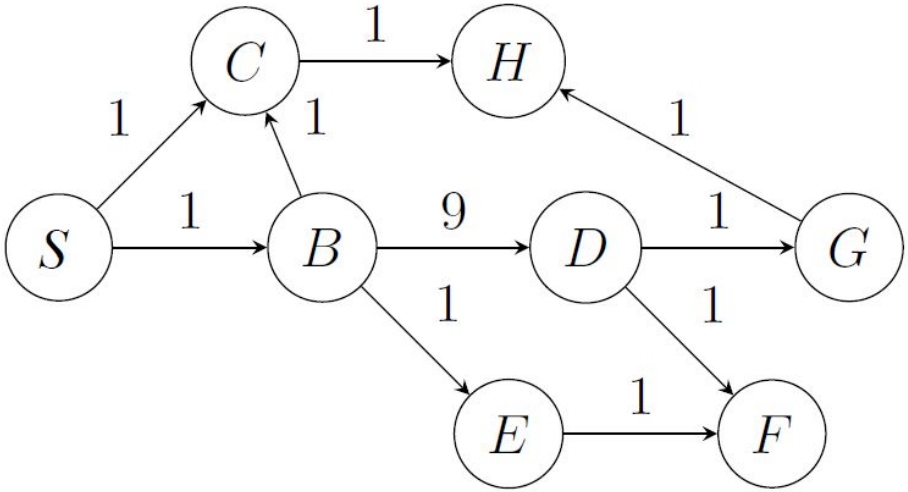


Current Node	Frontier	Explored
S(8)	B(8), C(4)	S
C(4)	B(8), H(102)	S, C
B(8)	H(102), D(11), E(6)	S, C, B
E(6)	H(102), D(11), F(9)	S, C, B, E
F(9)	H(102), D(11)	S, C, B, E, F
D(11)	H(102), G(11)	S, C, B, E, F, D
G(11)		

node, n	h(n)
S	8
B	7
C	3
D	1
E	4
F	6
G	0
H	100

Heuristic Search: Exercise

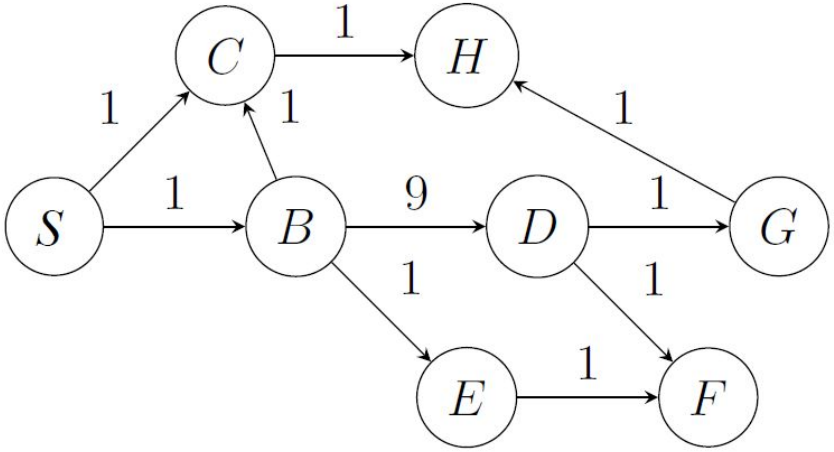
Is this heuristic admissible? Explain why or why not.



node, n	h(n)
S	8
B	7
C	3
D	1
E	4
F	6
G	0
H	100

Heuristic Search: Exercise

Is this heuristic admissible? Explain why or why not.



node, n	h(n)
S	8
B	7
C	3
D	1
E	4
F	6
G	0
H	100

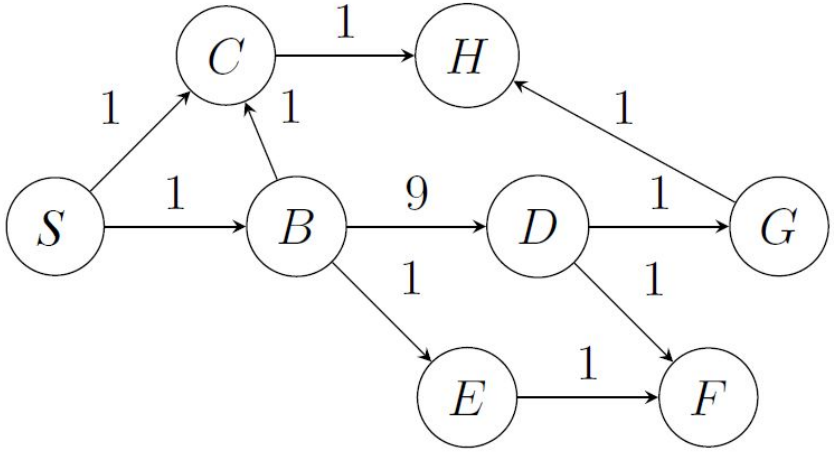
Recall: An admissible heuristic is one that **never overestimates** the cost to reach the goal, which is:

h(n) ≤ h*(n), for all n, where h*(n) is the true cost to reach the goal state from n.

Heuristic Search: Exercise

Is this heuristic admissible? Explain why or why not.

n	h(n)	h*(n)
S	8	11
B	7	10
C	3	∞
D	1	1
E	4	∞
F	6	∞
G	0	0
H	100	∞



node, n	h(n)
S	8
B	7
C	3
D	1
E	4
F	6
G	0
H	100

Yes, it is admissible.

Gradient Descent (Ascent): Review

- An iterative method to find the max/min* values

  *: Local max/min; sensitive to starting point
- Intuition: At each step, walk towards the steepest descent/ascent direction.
- Used widely in practice

Steps:

1. Pick a starting point
2. Repeat
   a. Calculate the gradient of the function at current point
   b. Move a step towards the direction of the gradient to reduce cost function $J(\theta)$
3. Stop when $J(\theta)$ is small enough

Local Search: Exercise

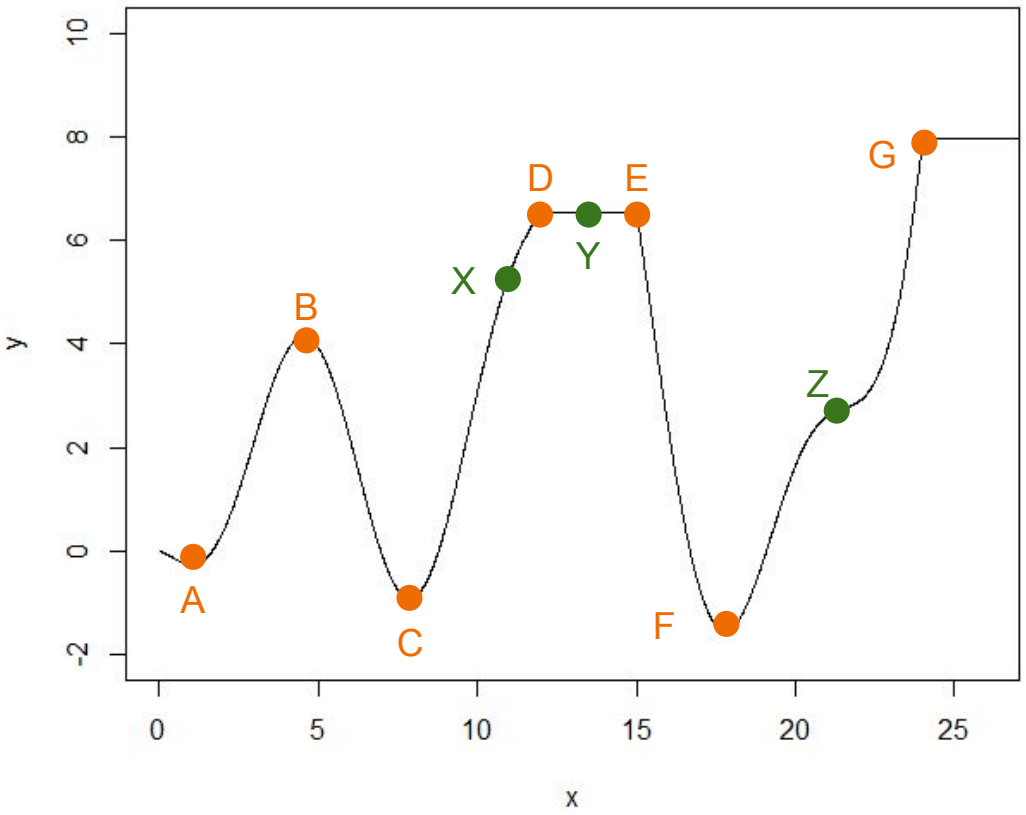
Use gradient descent/ascent method to find optimum, which point will be returned?

Starting at X to find a minimum returns ____

Starting at X to find a maximum returns ____

Starting at Y to find a minimum returns ____

Starting at Z to find a maximum returns ____



Local Search: Exercise

Use gradient descent/ascent method to find optimum, which point will be returned?

Starting at X to find a minimum returns C

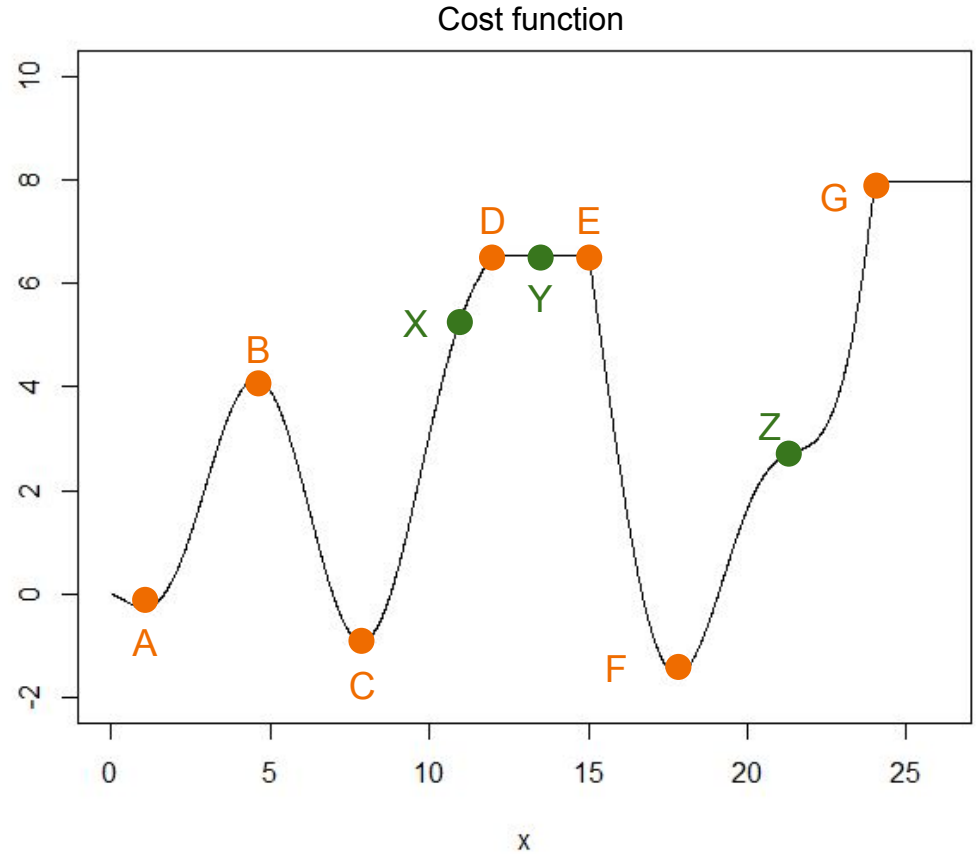
Starting at X to find a maximum returns D

Starting at Y to find a minimum returns Y

Starting at Z to find a maximum returns G

In standard gradient descent method, the stopping condition is the gradient of the cost function being small. So if we start at Y, the gradient of the cost function = 0. It will just return Y.

You could design more clever methods to deal with the plateau.



Example: Gradient Descent in Linear Regression

Recall: Linear Regression $\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots$ (See lecture note **30 June - Lecture 1**)

Assume we have n 1-dimensional datacases $x^{(1)}, x^{(2)}, \ldots, x^{(n)}$ with target value $y^{(1)}, y^{(2)}, \ldots, y^{(n)}$. (Note here $(\ )^{(i)}$ means the i-th datacase, not exponent).

Our prediction is $h_\theta(x^{(i)}) = \theta_0 + \theta_1 x_1^{(i)}$

Let the cost function $J(\theta)$ be the sum square error (SSE)

$$J(\theta) = \frac{1}{2} \sum_{i=1}^{n} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$$

This example illustrates how gradient descent can be used in machine learning problems involving finding optimum values. In fact, gradient descent is a practical approach to solving many complex machine learning problems.

It is ok if you are not familiar with the math.

Our goal is to find the optimal parameter θ^* that minimizing the cost function $J(\theta)$:

$$\theta^* = \arg\min_\theta J(\theta)$$

Example: Gradient Descent in Linear Regression

At each iteration, update the parameter $\theta_i \leftarrow \theta_i - \alpha \frac{\partial J(\theta)}{\partial \theta_i}$, where α is the step size.

$$
\begin{aligned}
\frac{\partial J(\theta)}{\partial \theta_j} &= \frac{\partial}{\partial \theta_j} \frac{1}{2} \sum_{i=1}^{n} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2 \\
&= \sum_{i=1}^{n} \left(h_\theta(x^{(i)}) - y^{(i)} \right) \frac{\partial}{\partial \theta_j} \left(h_\theta(x^{(i)}) - y^{(i)} \right) \\
&= \sum_{i=1}^{n} \left(h_\theta(x^{(i)}) - y^{(i)} \right) \frac{\partial}{\partial \theta_j} \left(\theta_0 + \theta_1 x^{(i)} - y^{(i)} \right)
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial J(\theta)}{\partial \theta_0} &= \sum_{i=1}^{n} \left(h_\theta(x^{(i)}) - y^{(i)} \right) \\
\frac{\partial J(\theta)}{\partial \theta_1} &= \sum_{i=1}^{n} \left(h_\theta(x^{(i)}) - y^{(i)} \right) x^{(i)}
\end{aligned}
$$

Simulated Annealing: Illustration

https://www.youtube.com/embed/KQYfaitQn7g