NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

CZ3005 Tutorial 1

Yu Han

Nanyang Assistant Professor

han.yu@ntu.edu.sg
N4-02c-109

Question 1.1

Explain which *search algorithm* is most appropriate in the following situations:

(a) We have a very large search space with a large branching factor and with possibly infinite paths.
We have no heuristic function.
We want to find a path to the goal with minimum number of states.

Criterion	Breadth-first	Uniform-Cost	Depth-First	Depth-Limited	Iterative Deepening	Bidirectional (if applicable)
Time	b^d	b^d	b^m	b^l	b^d	$b^{d/2}$
Space	b^d	b^d	bm	bl	bd	$b^{d/2}$
Optimal	Yes	Yes	No	No	Yes	Yes
Complete	Yes	Yes	No	Yes, if $l \geq d$	Yes	Yes

Question 1.1

Explain which *search algorithm* is most appropriate in the following situations:

(a) We have a very large search space with a large branching factor:

DFS (√), DLS (√), IDS (√), BFS (X), UCS (X), Informed Search (√)

and with possibly infinite paths:

DFS (X), DLS (√), IDS (√), BFS (X), UCS (X), Informed Search (√)

We have no heuristic function:

DFS (X), DLS (√), IDS (√), BFS (X), UCS (X), Informed Search (X)

We want to find a path to the goal with minimum number of states (i.e. implying optimality):

DFS (X), DLS (X), IDS (√), BFS (X), UCS (X), Informed Search (X)

Iterative Deepening Search is the most suitable

Question 1.1

Explain which *search algorithm* is most appropriate in the following situations:

(b) We have a state space with lots of cycles and links of varying costs.
We have no heuristic function.
We want to find the shortest path.

Question 1.1

Explain which *search algorithm* is most appropriate in the following situations:

(b) We have a state space with lots of cycles:

DFS (X), DLS (√), IDS (√), BFS (√), UCS (√), Informed Search (√)

and links of varying costs:

DFS (X), DLS (X), IDS (X), BFS (X), UCS (√), Informed Search (√)

We have no heuristic function:

DFS (X), DLS (X), IDS (X), BFS (X), UCS (√), Informed Search (X)

We want to find the shortest path (i.e. implying optimality):

DFS (X), DLS (X), IDS (X), BFS (X), UCS (√), Informed Search (X)

Uniform Cost Search is the most suitable

Question 1.1

Explain which *search algorithm* is most appropriate in the following situations:

(c) Our search space is a tree of fixed depth and
all the goals are at the bottom of the tree.
We have a heuristic function and
we want to find any goal as quickly as possible.

Question 1.1

Explain which *search algorithm* is most appropriate in the following situations:

(c) Our search space is a tree of fixed depth and

DFS (√), DLS (√), IDS (√), BFS (√), UCS (√), Informed Search (√)

all the goals are at the bottom of the tree.

DFS (√), DLS (√), IDS (X), BFS (X), UCS (√), Informed Search (√)

We have a heuristic function and

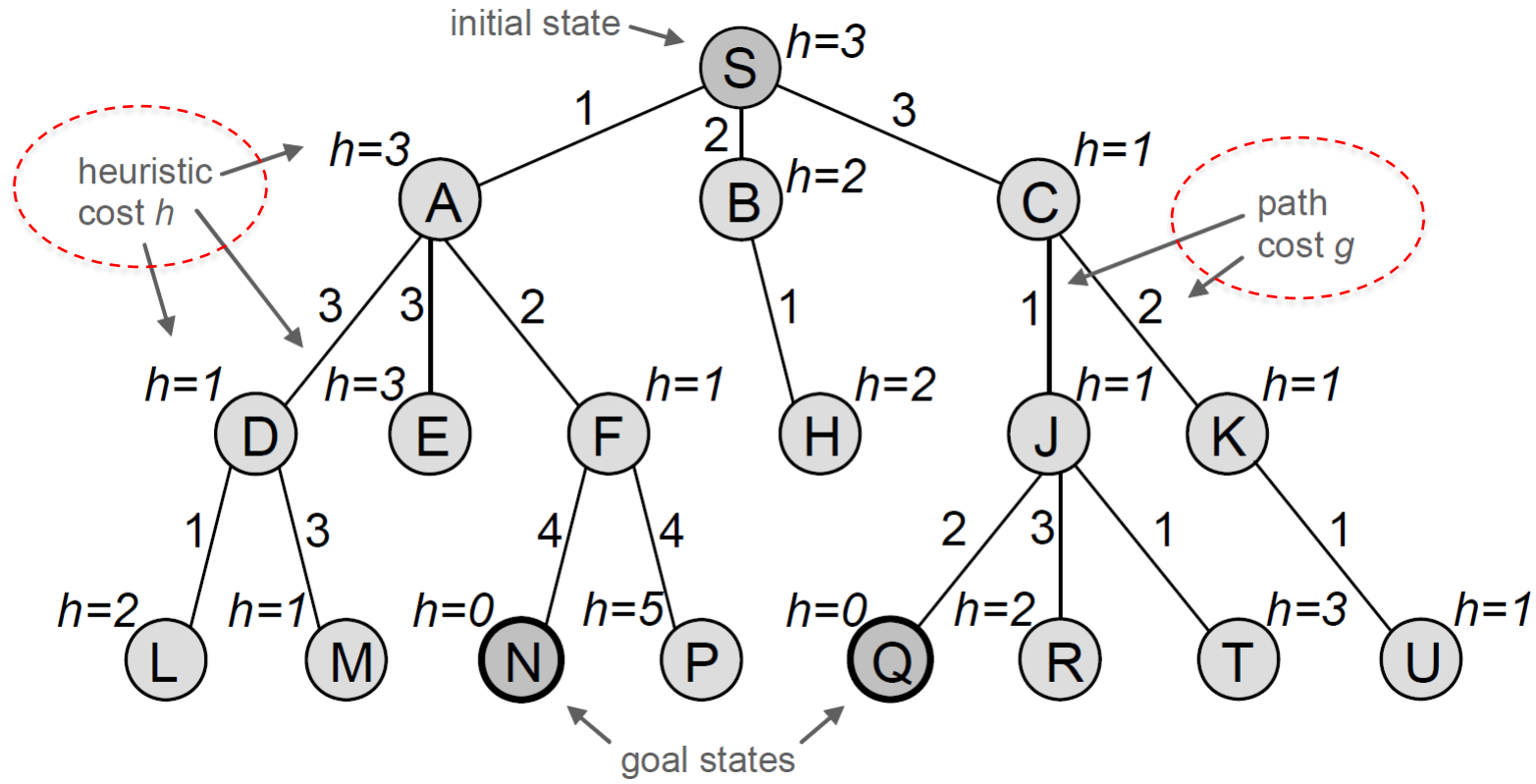
DFS (X), DLS (X), IDS (X), BFS (X), UCS (X), Informed Search (√)

we want to find any goal as quickly as possible (i.e. need not to be optimal).

Greedy Best First Search is the most suitable

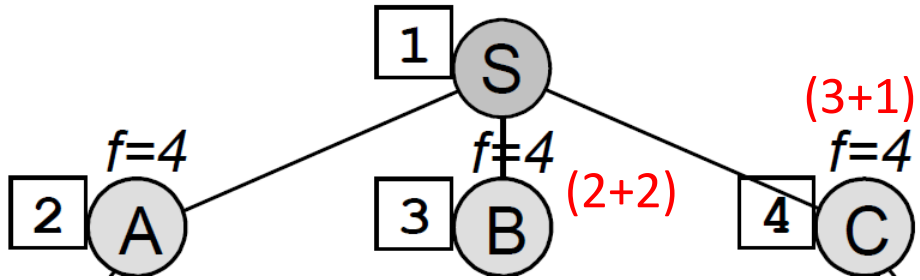
Question 1.2

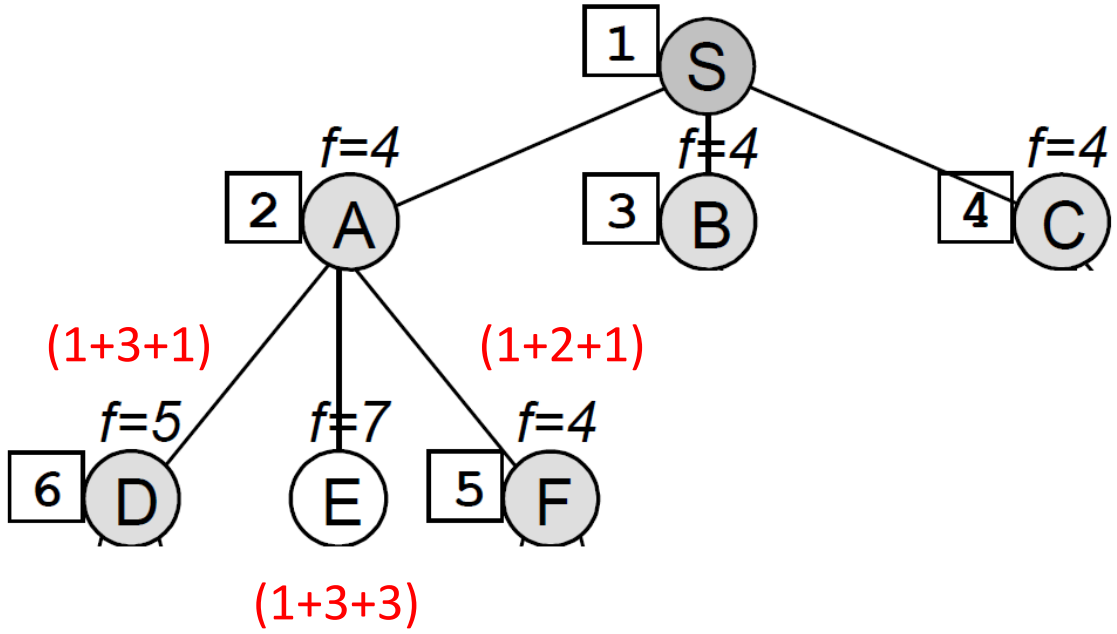
Consider the search problem defined by the annotated search tree below (f = h + g).



Question 1.2

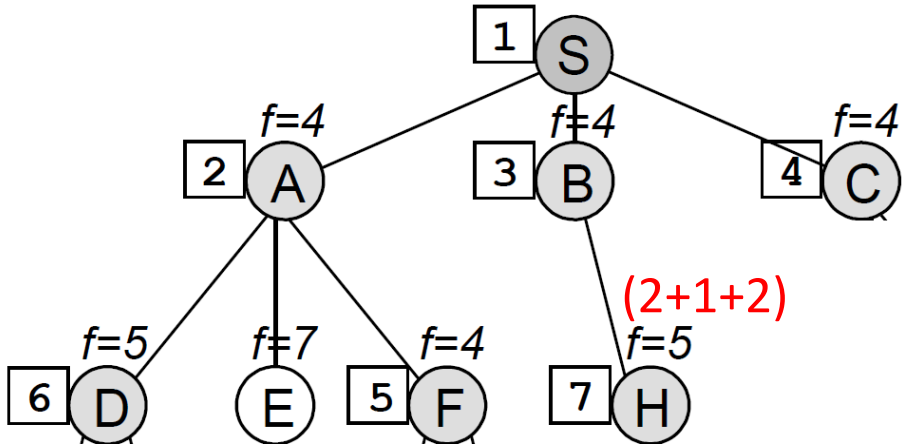
(a) Apply the standard *A* search algorithm.* Draw all generated nodes, write their f-costs, and number expanded nodes in order of expansion. Assume that the children of a node are processed in alphabetical order, and that nodes of equal priority are extracted from the search queue in FIFO order.

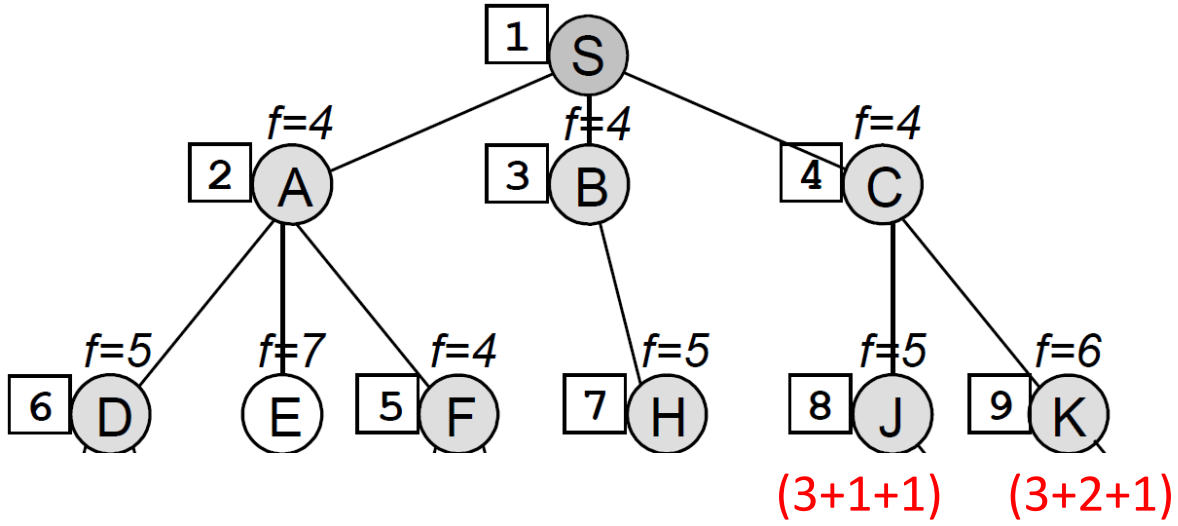




Question 1.2

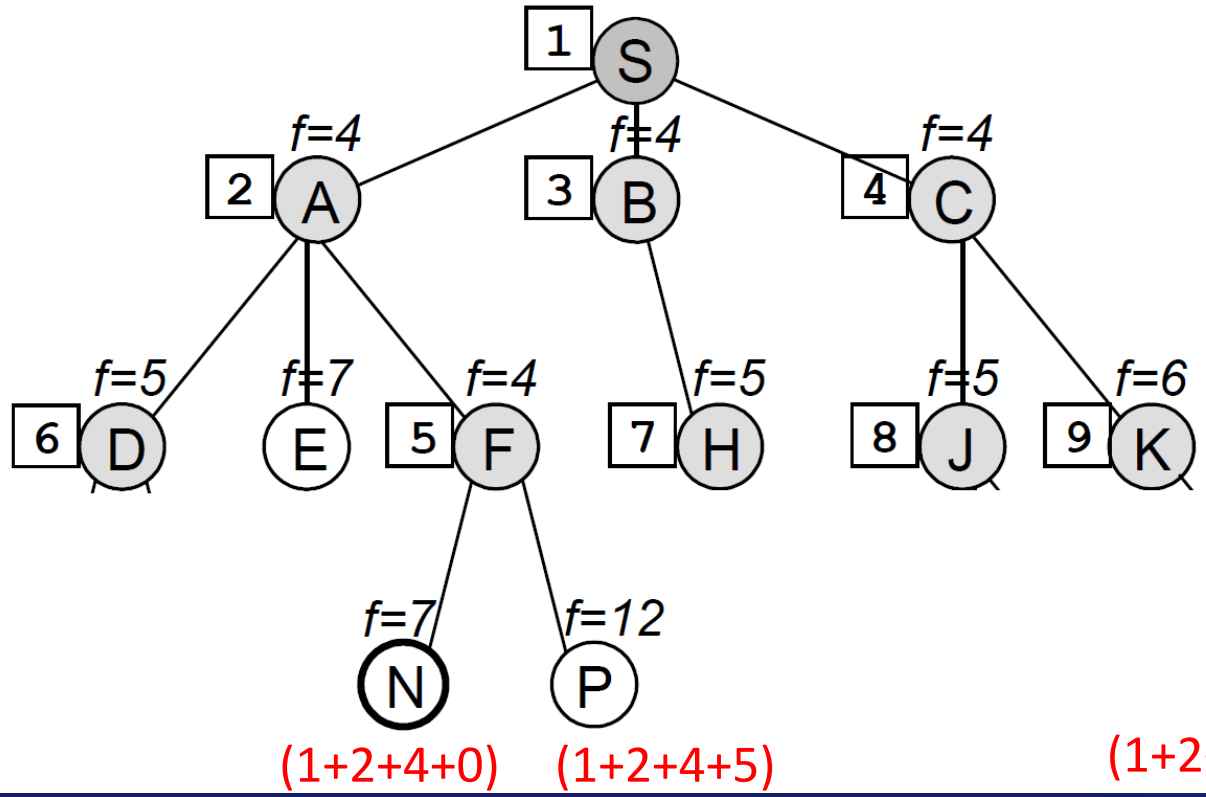
(a) Apply the standard *A* search algorithm.* Draw all generated nodes, write their f-costs, and number expanded nodes in order of expansion. Assume that the children of a node are processed in alphabetical order, and that nodes of equal priority are extracted from the search queue in FIFO order.

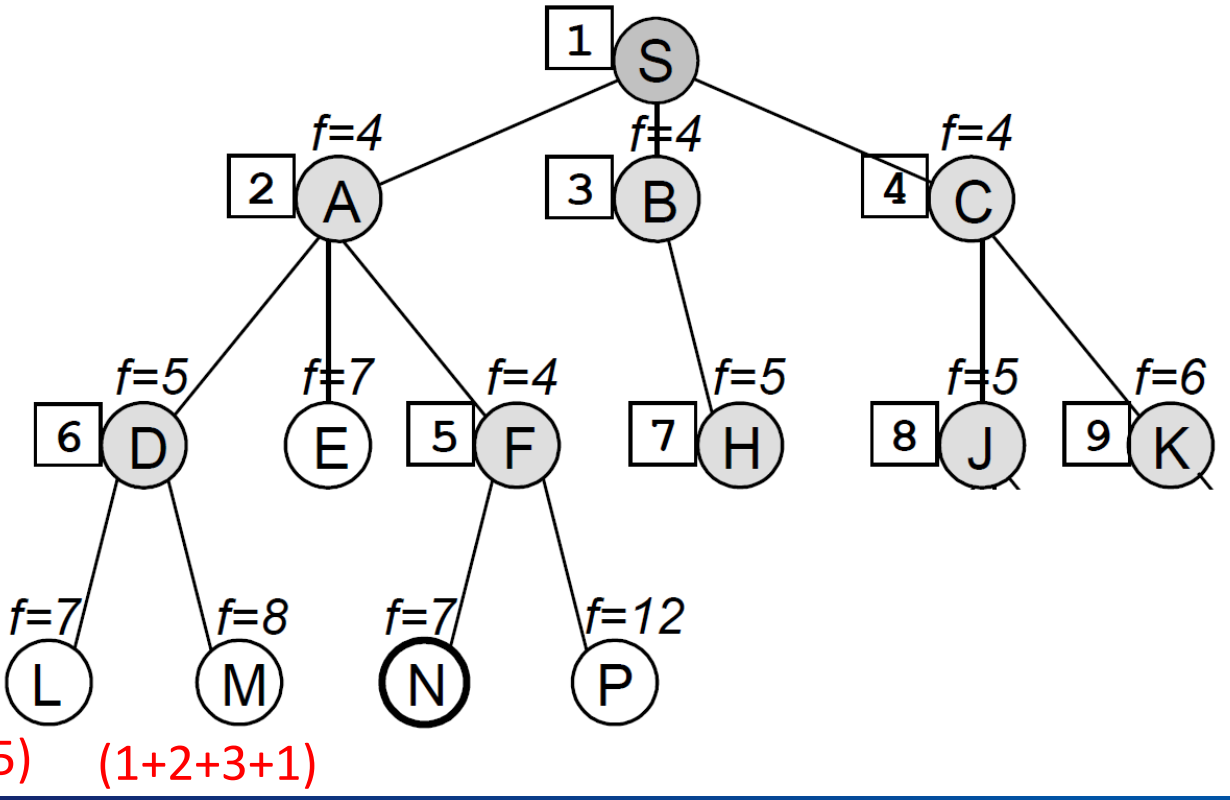




Question 1.2

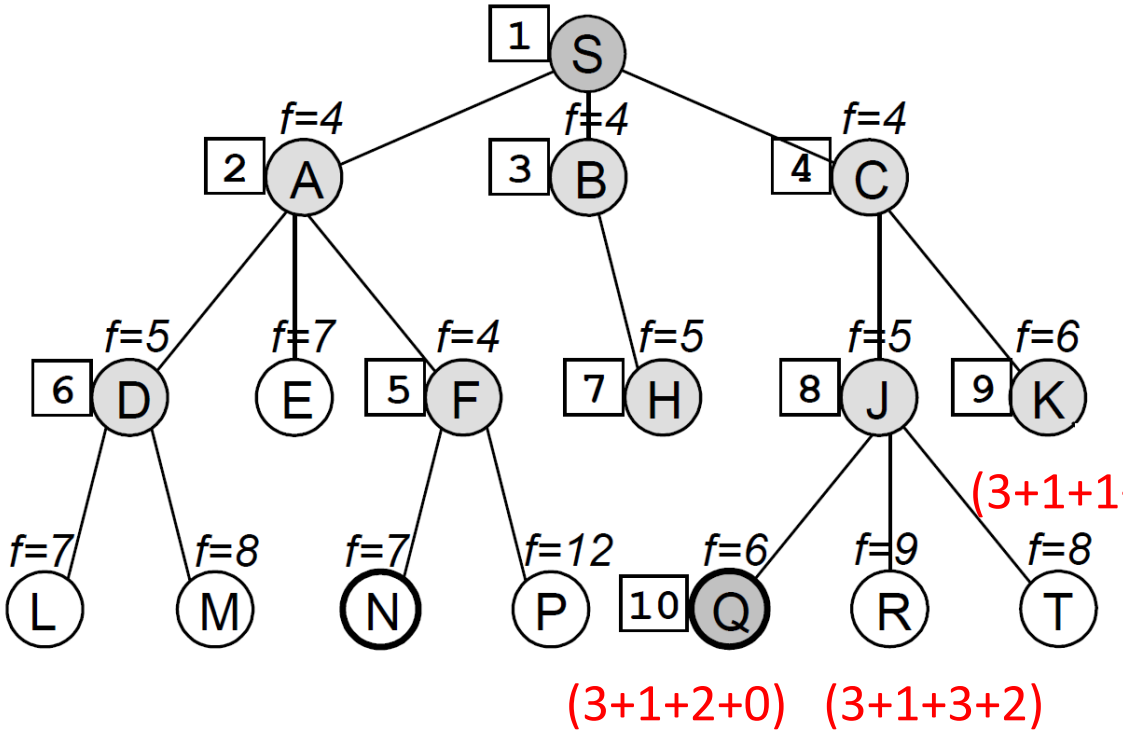
(a) Apply the standard *A* search algorithm.* Draw all generated nodes, write their f-costs, and number expanded nodes in order of expansion. Assume that the children of a node are processed in alphabetical order, and that nodes of equal priority are extracted from the search queue in FIFO order.

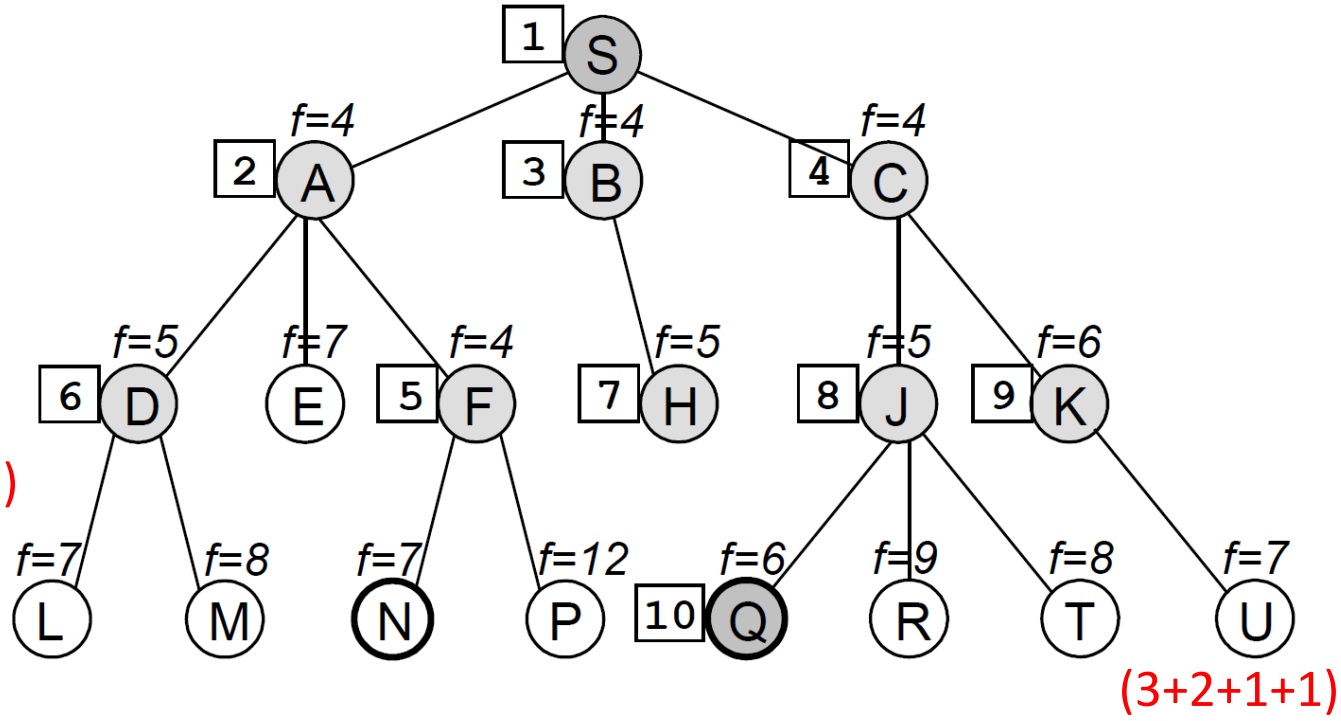




Question 1.2

(a) Apply the standard *A* search algorithm.* Draw all generated nodes, write their f-costs, and number expanded nodes in order of expansion. Assume that the children of a node are processed in alphabetical order, and that nodes of equal priority are extracted from the search queue in FIFO order.





Question 1.2

(a) Apply the standard *A* search algorithm.* Draw all generated nodes, write their f-costs, and number expanded nodes in order of expansion. Assume that the children of a node are processed in alphabetical order, and that nodes of equal priority are extracted from the search queue in FIFO order.

		queue
1.	**S** (0+3=3)	
2.	**A** (1+3=4), **B** (2+2=4) , **C** (3+1=4)	
3.	**B**, **C**, **F** (3+1=4), **D** (4+1=5), **E** (4+3=7)	
4.	**C**, **F**, **D**, **H** (3+2=5), **E**	
5.	**F**, **D**, **H**, **J** (4+1=5), **K** (5+1=6), **E**	
6.	**D**, **H**, **J**, **K**, **E**, **N** (7+0=7), **P** (7+5=12)	
7.	**H**, **J**, **K**, **E**, **N**, **L** (5+2=7), **M** (7+1=8), **P**	
8.	**J**, **K**, **E**, **N**, **L**, **M**, **P**	
9.	**K**, **Q** (6+0=6), **E**, **N**, **L**, **M**, **T** (5+3=8), **R** (7+2=9), **P**	
10.	**Q**, **E**, **N**, **L**, **U** (6+1=7), **M**, **T**, **R**, **P**	

Question 1.2

(b) State how many nodes were generated and how many were expanded. Comment on the solution obtained and the *effectiveness* of the search. What do you think of the *heuristic function h* employed?

All nodes

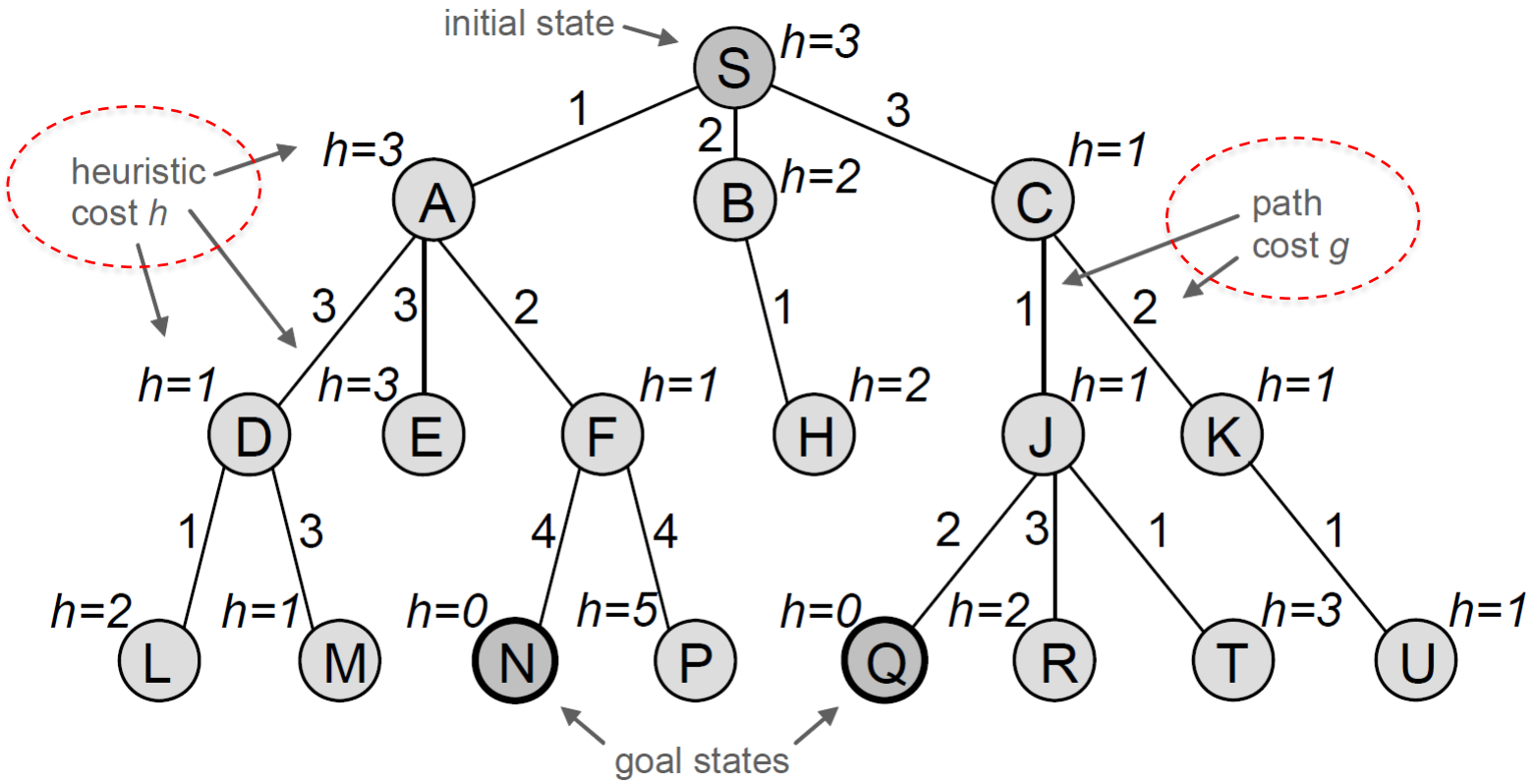
nodes generated: 18 — *nearly exhaustive search* (!)

nodes expanded: 10 — ill-guided → poor heuristics

optimal solution

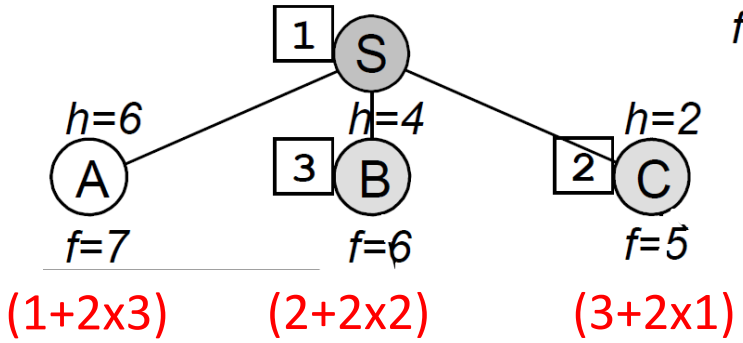
Question 1.3

The *w-A** search algorithm is a *weighted* variant of *A** that places more emphasis on the heuristic function by using the *f*-cost $f_w(n) = g(n) + w \times h(n)$, for any $w > 1$.

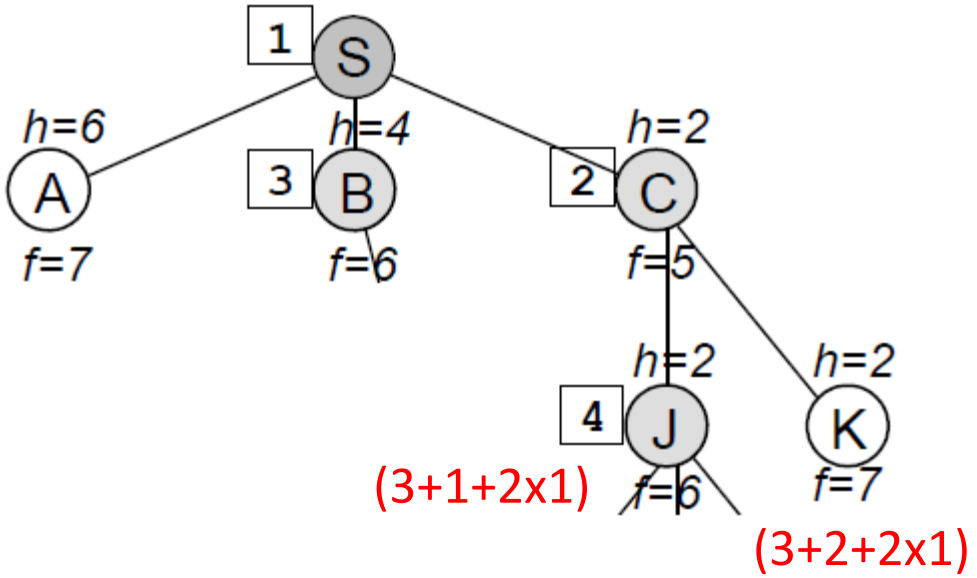


Question 1.3

(a) Similarly to question 1.2a, apply the *w-A** search algorithm for $w = 2$.

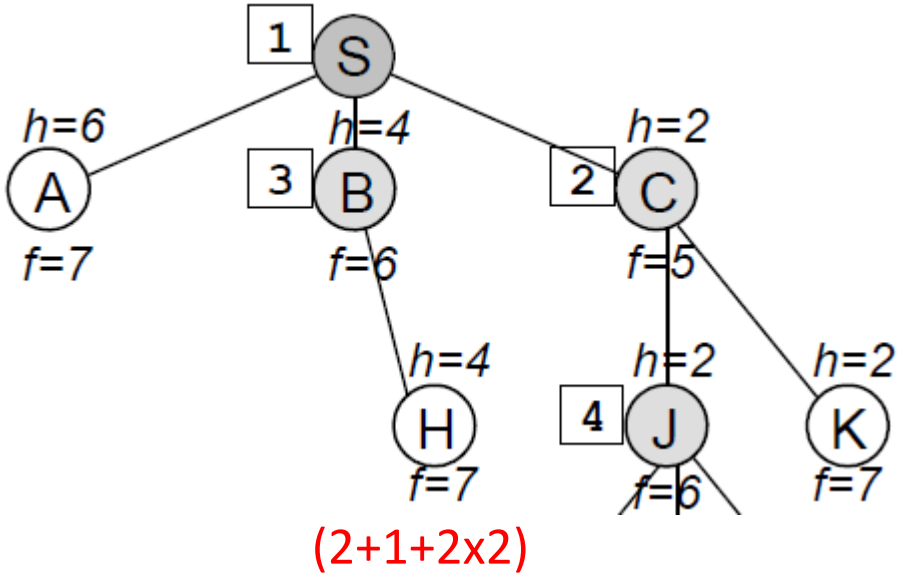


$$f_W(n) = g(n) + 2 \times h(n)$$

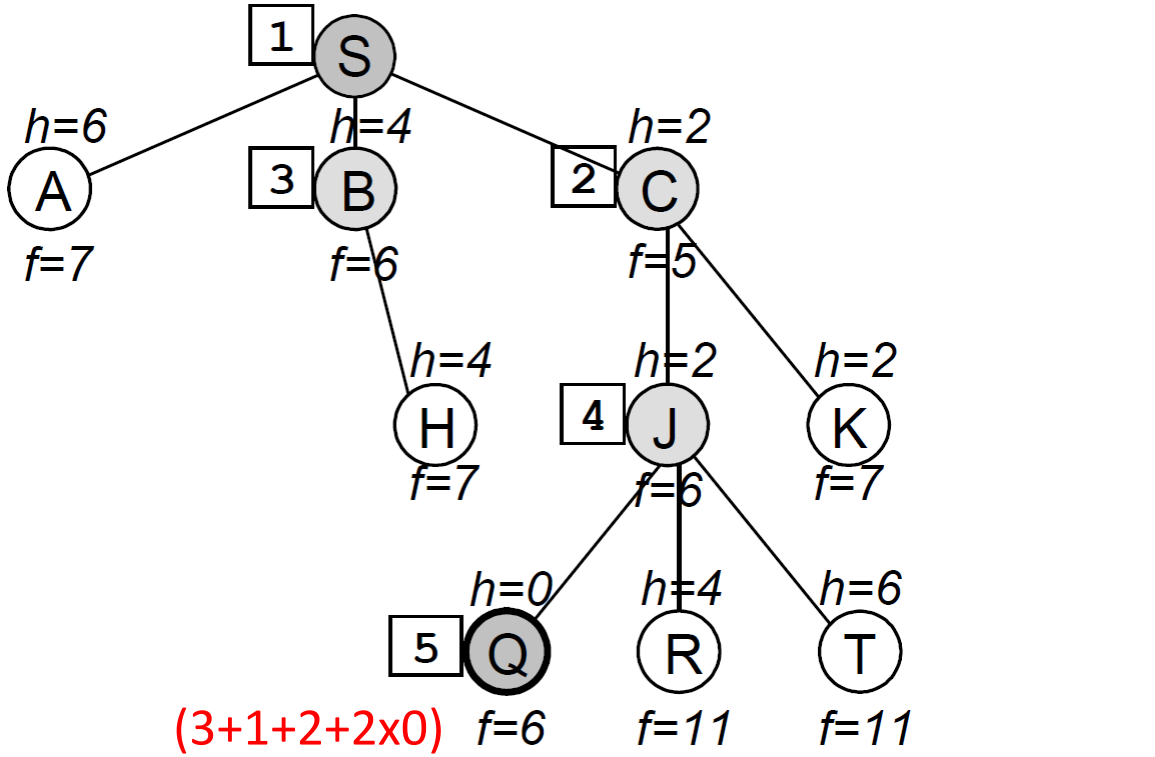


Question 1.3

(a) Similarly to question 1.2a, apply the *w-A** search algorithm for $w = 2$.



(2+1+2x2)



(3+1+2+2x0)

(3+1+3+2x2)

(3+1+1+2x3)

Question 1.3

(a) Similarly to question 1.2a, apply the *w-A** search algorithm for $w = 2$.

queue

1. **S** (0+6=6)
2. **C** (3+2=5), **B** (2+4=6), **A** (1+6=7)
3. **B**, **J** (4+2=6), **A**, **K** (5+2=7)
4. **J**, **A**, **K**, **H** (3+4=7)
5. **Q** (6+0=6), **A**, **K**, **H**, **R** (7+4=11), **T** (5+6=11)

Question 1.3

(b) Similarly to question 1.2b, comment on the *performance* and usefulness of the *w-A** search algorithm – in this case and in general.

nodes generated: 10
nodes expanded: 5

half(!) *well-guided search* → much improved heuristics

w-A* – pros: faster, complete

cons: not optimal (no guarantee)

increase *w*? faster yet, less and less optimal (still better than greedy search!)

Completeness: If at least one solution exists then the algorithm is guaranteed find a solution in a finite amount of time.

Thank you!