Solution

Most suitable search algorithm:

Search Methods	Time	Space	Optimal	Complete	Remarks
BFS – Breadth-first search	b^d	b^d	Yes	Yes	Uninformed (w/o heuristic function)
DFS – Depth-first search	b^m	b^*m	No	No	Uninformed (w/o heuristic function)
IDS – Iterative deepening search	b^d	b^*d	Yes	Yes	Uninformed (w/o heuristic function)
UCS – Uniform cost search	b^d	b^d	Yes	Yes	Uninformed (w/o heuristic function, w/ varying cost)
Greedy Search	b^m	b^m	No	No	Informed (w/ heuristic function)
A^* - f(n) = g(n)+h(n)	b^d	b^d	Yes (if h is admissible)	Yes	Informed (w/ heuristic function)

b: branching factor of the search tree
d: depth of the least-cost solution
m: maximum depth of the state space

Solution

Most suitable search algorithm:

"very large search space" ➔ Not BFS (possibly: A*, DFS, IDS)
"no heuristic function" ➔ Not A* (possibly: DFS, IDS)
"possibly infinite paths" ➔ Not DFS (possibly: IDS)

Search Methods	Time	Space	Optimal	Complete
~~BFS~~	~~b^d~~	~~b^d~~	~~Yes~~	~~Yes~~
~~DFS~~	~~b^m~~	~~$b*m$~~	~~No~~	~~No~~
IDS	b^d	$b*d$	Yes	Yes
~~UCS~~	~~b^d~~	~~b^d~~	~~Yes~~	~~Yes~~
~~Greedy Search~~	~~b^m~~	~~b^m~~	~~No~~	~~No~~
~~A*~~	~~b^d~~	~~b^d~~	~~Yes~~	~~Yes~~

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.

Solution

Most suitable search algorithm:

"lots of cycles" → Not DFS (possibly: A*, BFS, IDS, UCS)
"no heuristic function" → Not A* (possibly: BFS, IDS, UCS)
"varying costs" → Not BFS, IDS (possibly: UCS)
"shortest path" → optimal, UCS (Dijkstra Algorithm)

Search Methods	Time	Space	Optimal	Complete
~~BFS~~	~~b^d~~	~~b^d~~	~~Yes~~	~~Yes~~
~~DFS~~	~~b^m~~	~~$b*m$~~	~~No~~	~~No~~
~~IDS~~	~~b^d~~	~~$b*d$~~	~~Yes~~	~~Yes~~
UCS	b^d	b^d	Yes	Yes
~~Greedy Search~~	~~b^m~~	~~b^m~~	~~No~~	~~No~~
~~A*~~	~~b^d~~	~~b^d~~	~~Yes~~	~~Yes~~

(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.

Solution

Most suitable search algorithm:

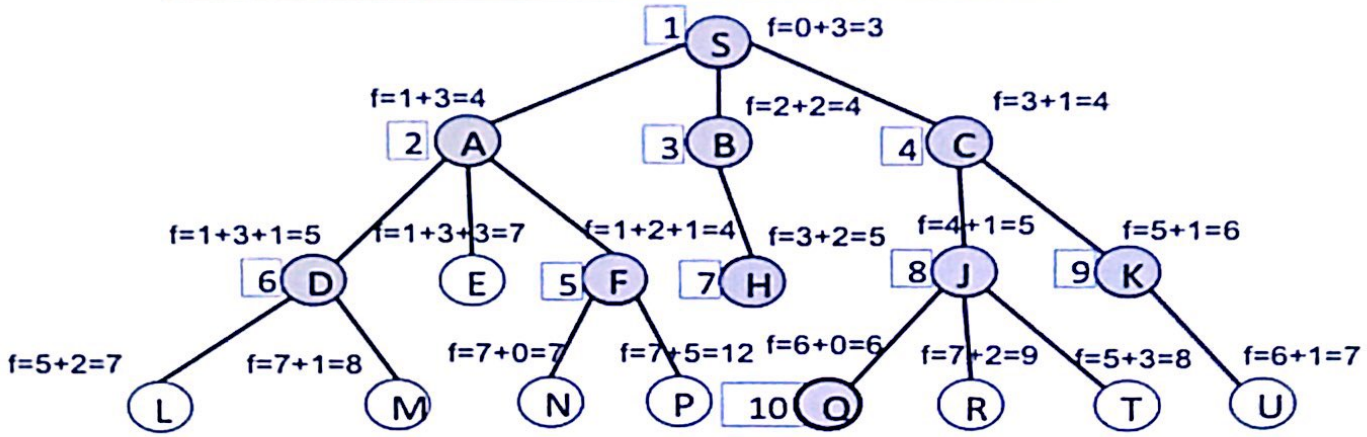
"fixed depth tree" ➔ DFS (possibly: All)
"goals at the bottom" ➔ Not BFS, IDS (possibly: DFS, A*, Greedy)
"heuristic function" ➔ Not DFS (possibly: A*, Greedy Best First)
"find any goal quickly" ➔ not optimal, Greedy Best First

Search Methods	Time	Space	Optimal	Complete
~~BFS~~	~~b^d~~	~~b^d~~	~~Yes~~	~~Yes~~
~~DFS~~	~~b^m~~	~~$b*m$~~	~~No~~	~~No~~
~~IDS~~	~~b^d~~	~~$b*d$~~	~~Yes~~	~~Yes~~
~~UCS~~	~~b^d~~	~~b^d~~	~~Yes~~	~~Yes~~
Greedy Search	b^m	b^m	No	No
~~A*~~	~~b^d~~	~~b^d~~	~~Yes~~	~~Yes~~

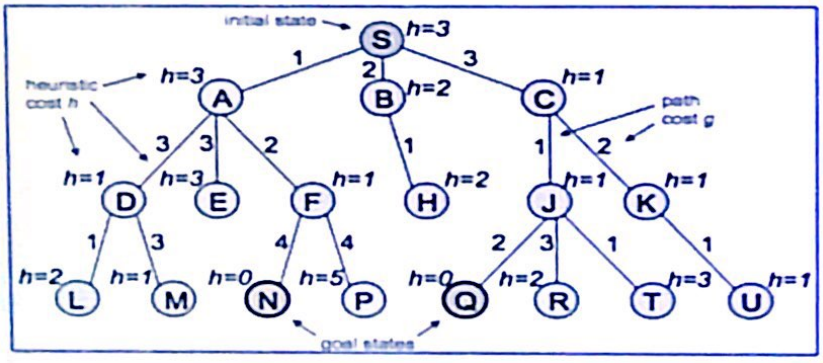
(c) Our search space is a tree of fixed depth and all the goals are at the bottom of the tree. We have a heuristic function and we want to find any goal as quickly as possible.

Solution

A* search, solution and performance:



3.2 (a)



FIFO (queue sorted by f values):

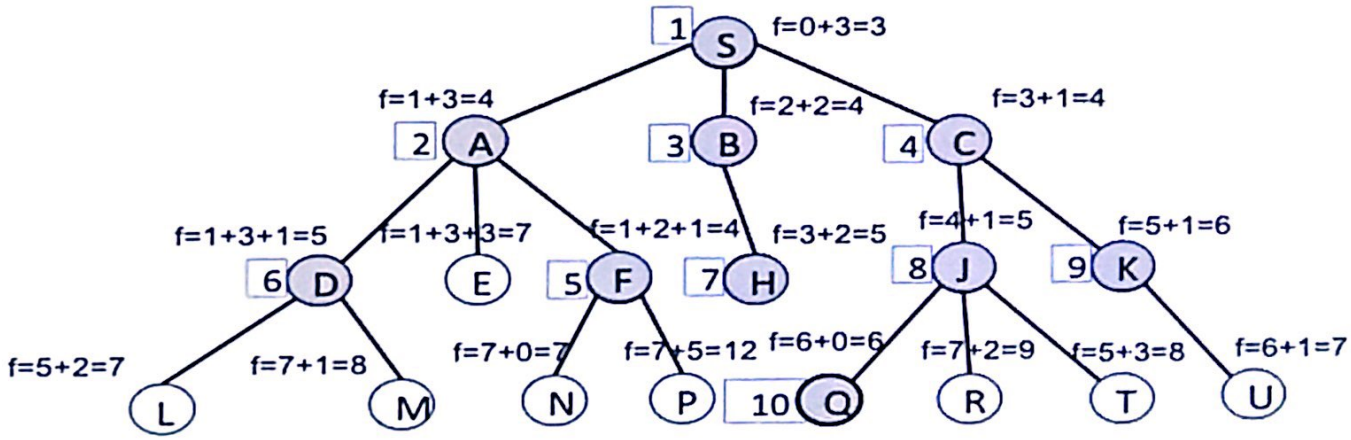
1. S (0+3=3)
2. A (1+3=4), B (2+2=4) , C (3+1=4)
3. B, C, F (3+1=4), D (4+1=5), E (4+3=7)
4. C, F, D, H (3+2=5), E
5. F, D, H, J (4+1=5), K (5+1=6), E
6. D, H, J, K, E, N (7+0=7), P (7+5=12)
7. H, J, K, E, N, L (5+2=7), M (7+1=8), P
8. J,K,E,N,L,M,P
9. K, Q (6+0=6), E, N, L, M, T (5+3=8), R (7+2=9),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

Solution

A* search, solution and performance:

3.2 (b)

- Number of nodes generated:
  18
- Number of nodes expanded:

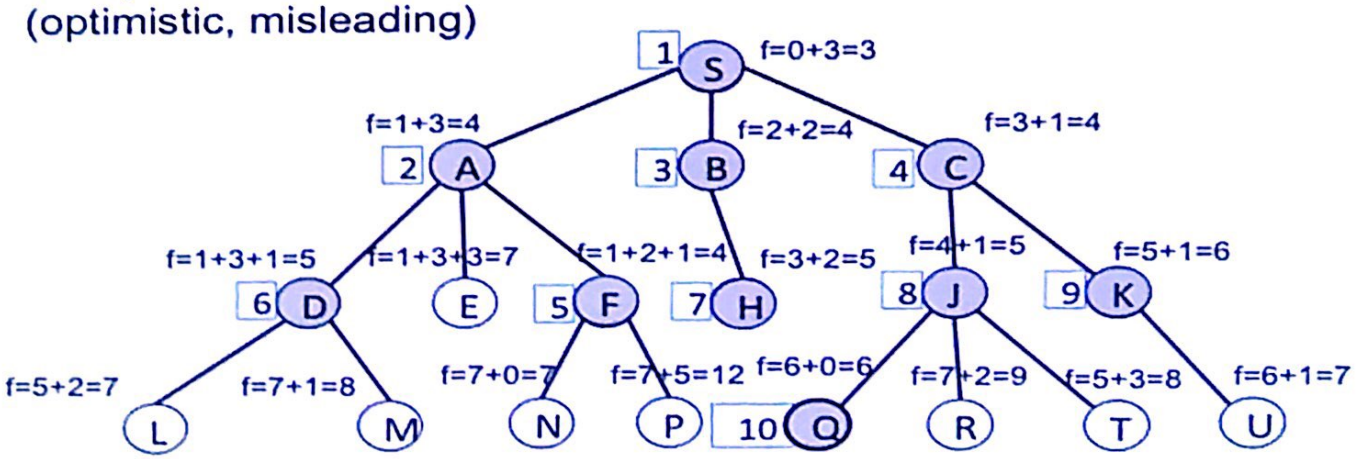


Solution

A* search, solution and performance:

3.2 (b)

- Number of nodes generated:
  18
- Number of nodes expanded:
  10
- nearly exhaustive search !➔ill-guided by the poor heuristics (optimistic, misleading)

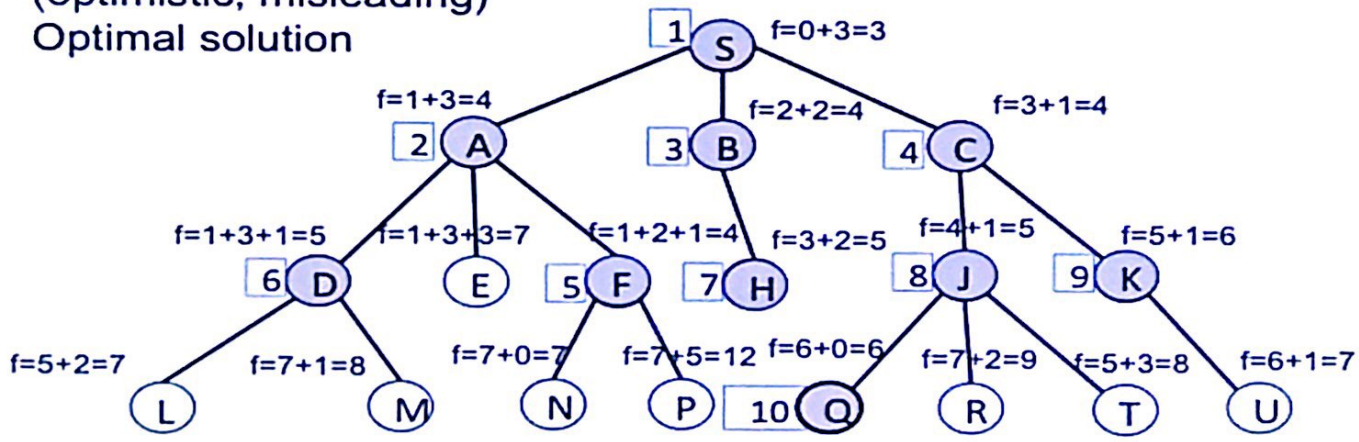


Solution

A* search, solution and performance:

3.2 (b)

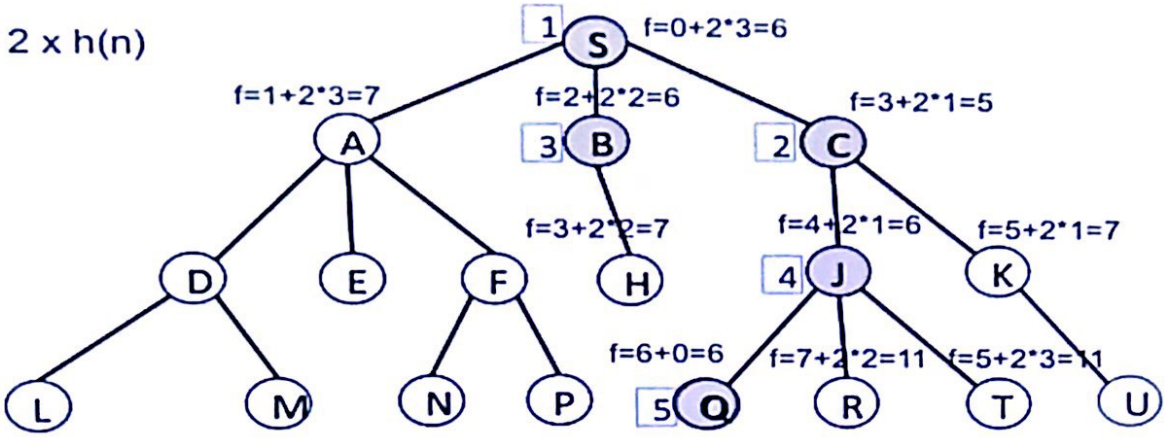
- Number of nodes generated:
  18
- Number of nodes expanded:
  10
- nearly exhaustive search !➔ill-guided by the poor heuristics (optimistic, misleading)
- Optimal solution



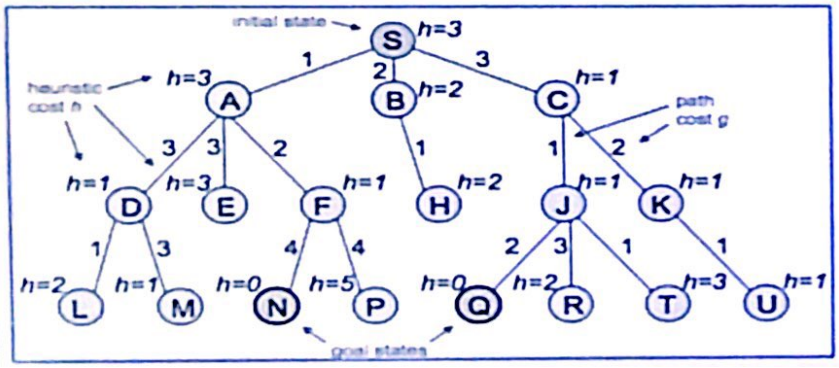
Solution

Weighted A search, solution and performance:*

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



3.3 (a)



FIFO (queue sorted by f values):

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)

Solution

Weighted A* search, solution and performance:

3.3 (b) $f_w(n) = g(n) + 2 \times h(n)$

- Number of nodes generated:
  10
- Number of nodes expanded:
  5
- half(!) of the search tree compared to the heuristics in 3.2 ➔ well-guided search with much improved heuristics

w-A* in summary:

- pros: faster, complete
- cons: not optimal (no guarantee)

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