Iterative Deepening Search

very large search space: DFS, IDS or A*, not BFS, UCS
large branching factor: DFS or IDS
possibly infinite paths: not DFS

Tutorial 1 Searching for a Solution

CZ3005

a) Depth First Search. No heuristic function means A* and greedy search cannot be applied. DFS is faster than UCS and BFS as the solutions are dense due to a very large search space and large branching factor.

minimum number of states: optimal, so BFS, UCS, IDS

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.

b)Uniform Cost Search. Cycles can cause DFS, Depth limited search and iterative deepening search to get stuck and the paths have varying costs implying BFS is not effective. No heuristic functions so A* and greedy cannot be applied

(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.

(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.

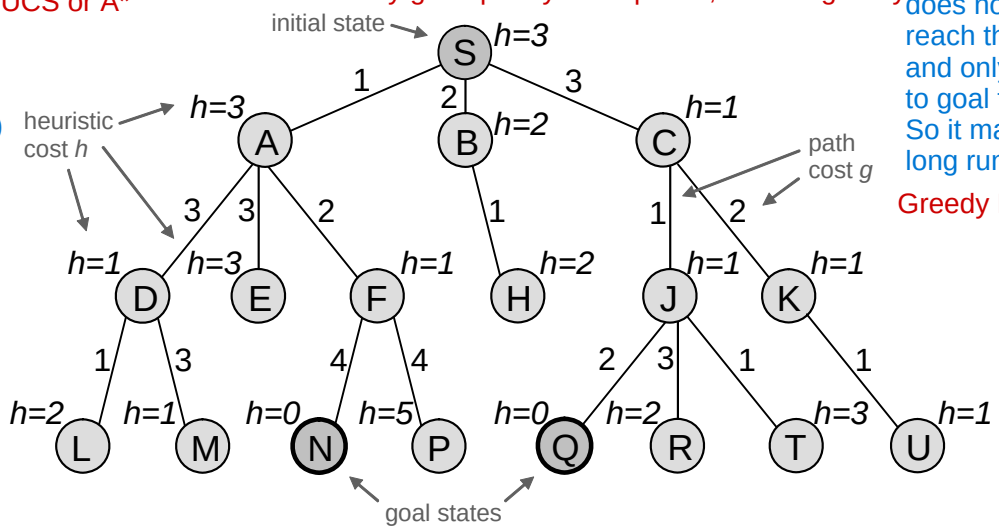
-fixed depth tree: DFS, others
-goals at the bottom: DFS, not BFS or IDS
-heuristic function: Greedy, not DFS
-find any goal quickly: not optimal, DFS or greedy

shortest path: optimal, so UCS or A*
varying costs: UCS or A*

c) A* search. Given a heuristic function, A* and greedy are considered. A* is chosen as greedy does not consider cost to reach the current state and only considers cost to goal from current state. So it may be slower in the long run

Greedy Best First Search

1.2 Consider the search problem defined by the annotated search tree below.



	h(n)	g(n)	total,f(n)
S	3	0	3
A	3	1	4
B	2	2	4
C	1	3	4
D	1	4	5
E	3	4	7
F	1	3	4
H	2	3	5
J	1	4	5
K	1	5	6
L	2	5	7
M	1	7	8
N	0	7	7
P	5	7	12
Q	0	6	6
R	2	7	9
T	3	5	8
U	1	6	7

(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.

1.2a)
S
ABC S
BCFDE A
CFDHE B
FDHJKE C
DHJKENP F
HJKENLMP D
JKENLMP H
KQENLMTRP J
QENLUMTRP K
ENLUMTRP Q

(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?

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 $w > 1$.

2

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

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

2 10 expanded nodes
10 generated nodes
18

1.2 b)The solution is optimal and complete. The search is effective but can be improved with a greater heuristic value range

nearly exhaustive search, ill-guided, poor heuristics, optimistic, misleading