UNIVERSITY PARTNER

Academic Year	Module	Assessment Number	Assessment Type
S20	Introductory Data Structures and Algorithms (DipIT02)	A1	Assignment Submission

[Assignment Submission]

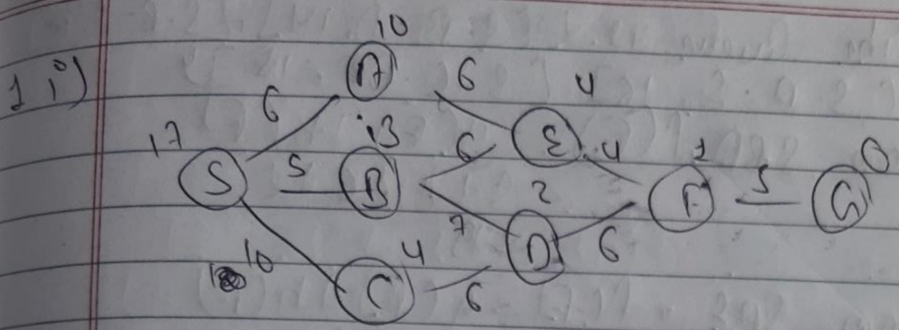
Student Id : [NP03A190299]
Student Name : [Yogesh Shrestha]
Section : [DC8]
Module Leader : [Mr. Prakash Gautam]

Submitted on : 06-03-2020

Tutorial - 11

Page No.
Date : / /

1 i)



Sol^n:

For initial stage:

$f(S) = g(S) + h(S)$

$= 0 + 14$

$= 17$

$\therefore$ The Que $= \{[S, 17]\}$.

From S, $f(A) = g(A) + h(A)$

$= 6 + 10 = 16$

$f(B) = g(B) + h(B)$

$= 5 + 13 = 18$

$f(C) = g(C) + h(C)$

$= 10 + 4 = 14$

$\therefore$ The Queue is $= \{[S-C, 14], [S-A, 16],$
$[S-B, 18]\}$

From SC, $f(D) = g(D) + h(D)$

$= 10 + 6 + 2 = 18$

From SB, $f(D) = g(D) + h(D)$

$= 5 + 7 + 2 = 14$

$f(E) = g(E) + h(E)$

$= 6 + 6 + 4 = 16$

Page No.
Date: / /

∴ The Queue is {[S-B-D, 14], [S-B-E, 15], [S-A-E, 16], [S-C-D, 18]}

From SBD, f(F) = g(F) + h(F)
= 5+7+6+1
= 19

From SAE, f(F) = g(F) + h(F)
= 6+4+6+1
= 17

From SBE f(F) = g(F) + h(F)
= 5+4+6+1 = 16

From SCD, f(F) = g(F) + h(F)
= 10+6+6+1 = 23

From SCD, f(F) = g(F) + h(F)
= 10+6+6+1 = 23

∴ The Queue is {[S-B-E-F, 16], [S-A-E-F, 17], [S-B-D-F, 19], [S-C-D-F, 23]}

From SAEF, f(G) = g(G) + h(G)
= 6+6+4+3+0 = 19

From SBEF, f(G) = g(G) + h(G)
= 6+5+4+3+0 = 18

From SBDF, f(G) = 7+5+6+3+0 = 21

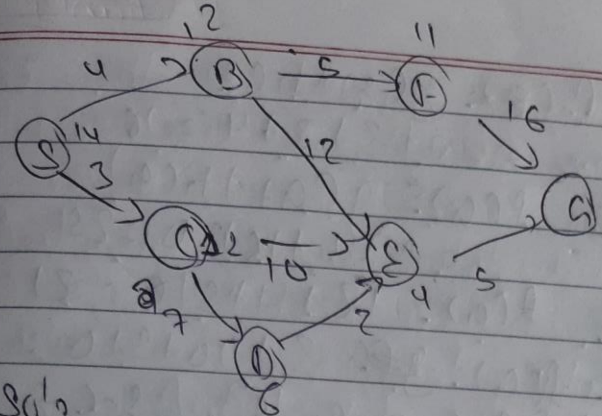
From SCDF f(G) = 10+6+6+3+0 = 25

∴ Queue is {[S-B-E-F-G, 18], [S-A-E-F-G, 19], [S-B-D-F-G, 21], [S-C-D-F-G, 25]}

Final path is S-B-E-F-G with cost of 18

Page No.
Date : / /

ii)



Sol'n

from initial

$f(S) = g(S) + h(S)$ ∴ Queue is {[S, 14]}

$= 0 + 14$

$= 14$

from S, $f(B) = g(B) + h(B)$

$= 4 + 12 = 16$

$f(C) = g(C) + h(C)$

$= 3 + 11 = 14$

∴ Queue is {[S-C, 14], [S-B, 16]}

from SB, $f(F) = g(F) + h(F)$

$= 4 + 5 + 11 = 20$

$f(E) = g(E) + h(E)$

$= 4 + 12 + 4 = 20$

from SC $f(E) = g(E) + h(E)$

$= 10 + 3 + 4 = 17$

$f(D) = g(D) + h(D)$

$= 7 + 3 + 6 = 16$

∴ The Queue = {[S-C-D, 16], [S-C-E, 17],
[S-B-E, 20], [S-B-F, 20]}

Page No.
Date : / /

from SC, f(E) = g(E) + h(E)
= 7 + 3 + 2 + 4 = 16

from SBF f(G) = g(G) + h(G)
= 5 + 4 + 16 + 0 = 25

from SBE f(G) = 12 + 4 + 5 + 0 = 21

from SCE f(G) = ~~g(G)~~ 3 + 10 + 5 + 0 = 18

∴ Queue is {[S-C-d-e, 16], [S-C-E-G, 18], [S-B-E-G, 21], [S-B-F-G, 25]}

from SCDE f(G) = 7 + 3 + 2 + 5 + 0 = 17

∴ Queue {[S-C-D-E-G, 17], [S-C-E-G, F, 18], [S-B-E-G, 21], [S-B-F-G, 25]}

∴ The final path is S-C-d-e-G which we lost of 17 #

2 → Heuristic is a ~~teq~~ techique that designed to solve problem's more quickly. It can be achieved by trading optimality completeness, accuracy or precision for speed. It is used to solve NP-complete problems.

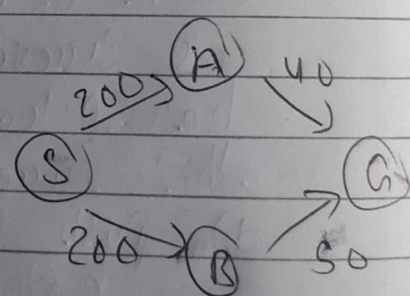
a) Initialize the open list
b) Initialize the closed list put the starting node on the open list.
c) While the open list is not empty
   i) find the node with the least f on the open list, call it "q".

Page No.
Date : / /

ii.) ~~pop~~ Pop q off the open list.
iii) generate q's 8 successors and set their parents to q
iv) for each successor
a) if successor is the goal, stop search
successor. g = q.g + distance between successor and q.
Successor. h = distance from goal to successor
Successor. f = successor. g + successor. h
b) if a node with the same position as successor is in the open list which has a lower f than successor, skip this successor.
c) if a node with the same position as successor is in the closed list which has a lower f than successor, skip this successor.
d) push q on the closed list.
end (while loop)

3 ⇒ - An admissible heuristic is the function which never overestimates the cost of ~~reaching the goal, from a~~ nod. Or the minimum cost path from a node to the goal node. So, a heuristic is a specific to a particular state space and also to a particular goal state in that state space. It will lead to search paths that turn out to be more costly than the optimal path. If $h(n)$ and $h^*(n)$ be estimated and actual heuristics respectively then for admissible heuristics
$h(n) \leq h^*(n)$

Overestimation:
if $h(n) \geq h^*(n)$



We have,
$h^*(A) = 40$
$h^*(B) = 50$

Now, let's take $h(A) = 80$ and
$h(B) = 70$

From A* Algorithm
$f(A) = g(A) + h(A)$
$= 200 + 80 = 280$
$f(B) = g(B) + h(B)$
$= 200 + 70 = 270$

Page No.
Date : / /

from SB, f(G) = g(G) + h(G)
which = 200 + 50 + 0 = 250
∴ ~~SASB~~ is not the optimal solution.

Underestimation
if $h(n) \le h*(n)$
we have,
$h*(A) = 40$
$h*(B) = 50$
Again, $h(A) = 30$
$h(B) = 20$
from the A* algorithm
$f(A) = g(A) + h(A)$
$= 200 + 30 + 0 = 230$
$f(B) = g(B) + h(B)$
$= 200 + 20 + 0 = 220$
Now, from SB, $f(G) = g(G) + h(G)$
~~from SA, f(G) =~~ $= 200 + 50 + 0 = 250$
from SA, $f(G) = g(G) + h(G)$
$= 200 + 40 + 0 = 240$
which is the optimal solution.
∴ To make the graph admissible we need to choose heuristic function $h(A) \le h*(A)$ & $h(B) \le h*(B)$

Page No.
Date : / /

4 -> A consident ~~is~~ heuristic is a function which never overstimates the cost of reaching the goal line converse however is not always true. This is proved by induction on m, By assumption. ~~h(Nm)~~ h(Nm) ≤ h* (Nm) where h* (n) denotes the cost of the shortest path from n.

The benifits of consistent heuristic are:

i) It never Overestimates the cost of reaching the goal.

ii) It helps to donates the cost of the shortest path from n to the goal.

iii) The length of the best from node to the goal.

Page No.
Date : / /

5.> The advantages of A* algorithm.
i) It is complete and optimal.
ii) It is the best ~~one~~ one from other technique
iii) It is used to solve very complex problems
iv) It is optimally efficient i.e. there is no other optimal algorithm ~~go~~ guranteed to expand fewer nodes than A*.

The disadvantages of A* algorithm.
i) ~~This~~ It is algorithm ~~is~~ contain with complexity problems
ii) The speed execution of A* search is highly dependent on the accuracy of the heuristic algorithm that is used to compute h(n).
iii) The algorithm is complete if the brancing factor is finite and every action has fixed cost.