Problem Reduction Search-
(AO* Search)
Lecture-21

Hema Kashyap

AND/OR Search

- The Depth first search and Breadth first search given earlier for OR trees or graphs can be easily adopted by AND-OR graph. The main difference lies in the way termination conditions are determined, since all goals following an AND nodes must be realized; where as a single goal node following an OR node will do. So for this purpose we are using AO* algorithm.
- Like A* algorithm here we will use two arrays and one heuristic function.

OPEN: It contains the nodes that has been traversed but yet not been marked solvable or unsolvable.

CLOSE: It contains the nodes that have already been processed.

h(n):The distance from current node to goal node.

The AND/OR Graph Search Problem

- Problem Definition:
  - Given: [G , s , T] where
    - G: implicitly specified AND/OR graph
    - S: start node of the AND/OR graph
    - T: set of terminal nodes
    - H(n): heuristic function estimating the cost of solving the sub- problem at n
  - To find:
    - A minimum cost solution tree

Algorithm

Step 1: Place the starting node into OPEN.

Step 2: Compute the most promising solution tree say T0.

Step 3: Select a node n that is both on OPEN and a member of T0. Remove it from OPEN and place it

Step 4: If n is the terminal goal node then leveled n as solved and leveled all the ancestors of n as solved. If the starting node is marked as solved then success and exit.

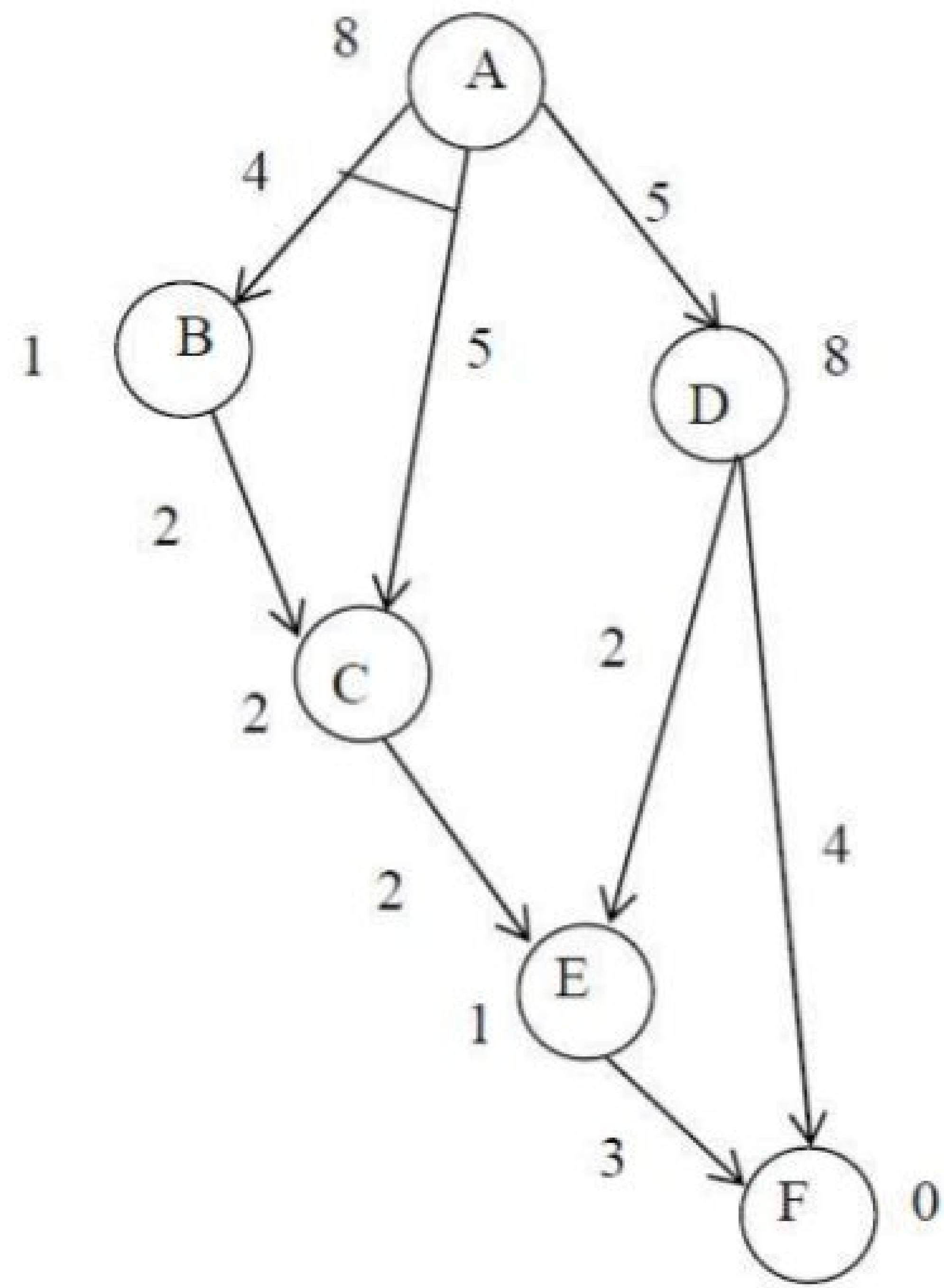
Step 5: If n is not a solvable node, then mark n as unsolvable. If starting node is marked as unsolvable, then return failure and exit.

Step 6: Expand n. Find all its successors and find their h (n) value, push them into OPEN.

Step 7: Return to Step 2.

Step 8: Exit. in CLOSE

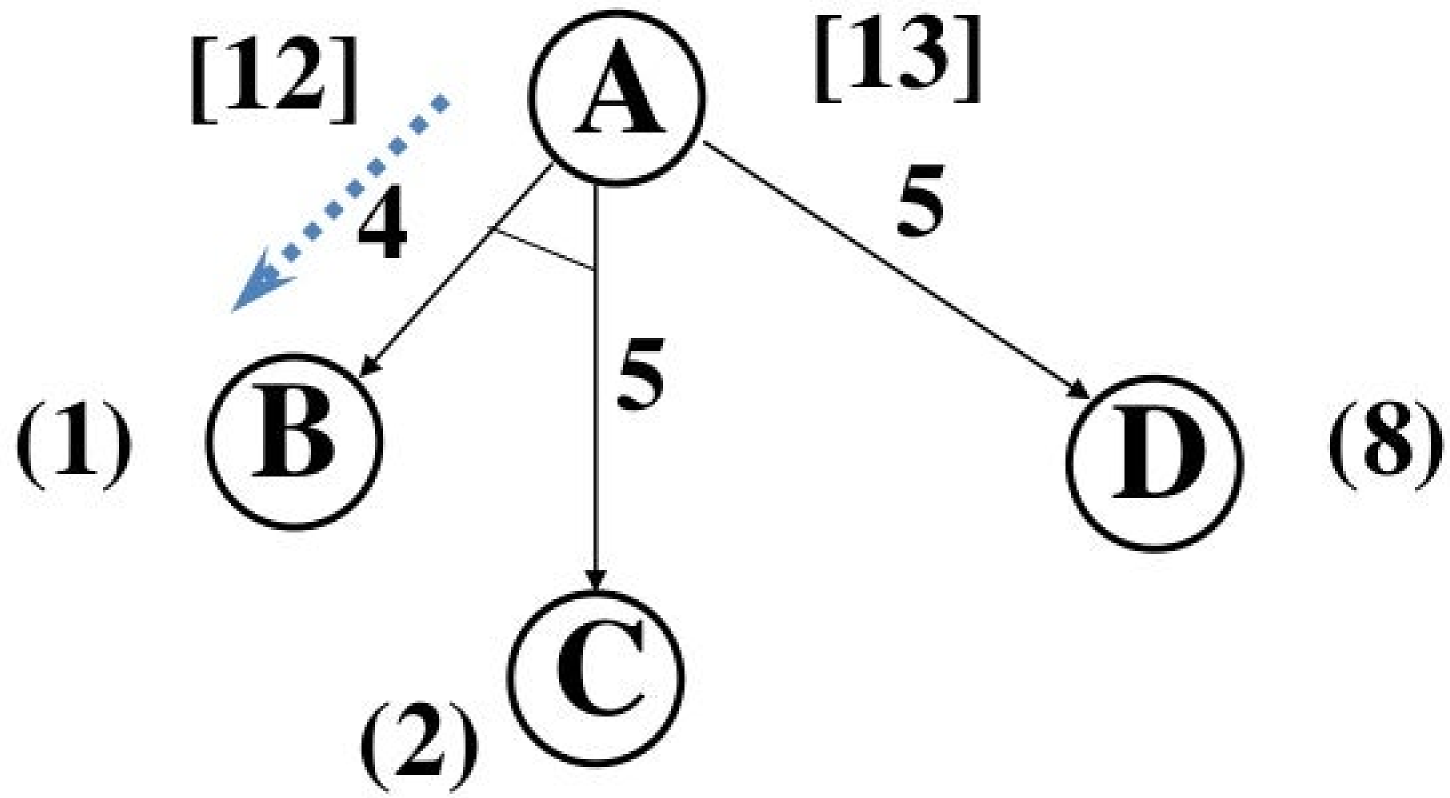
Example



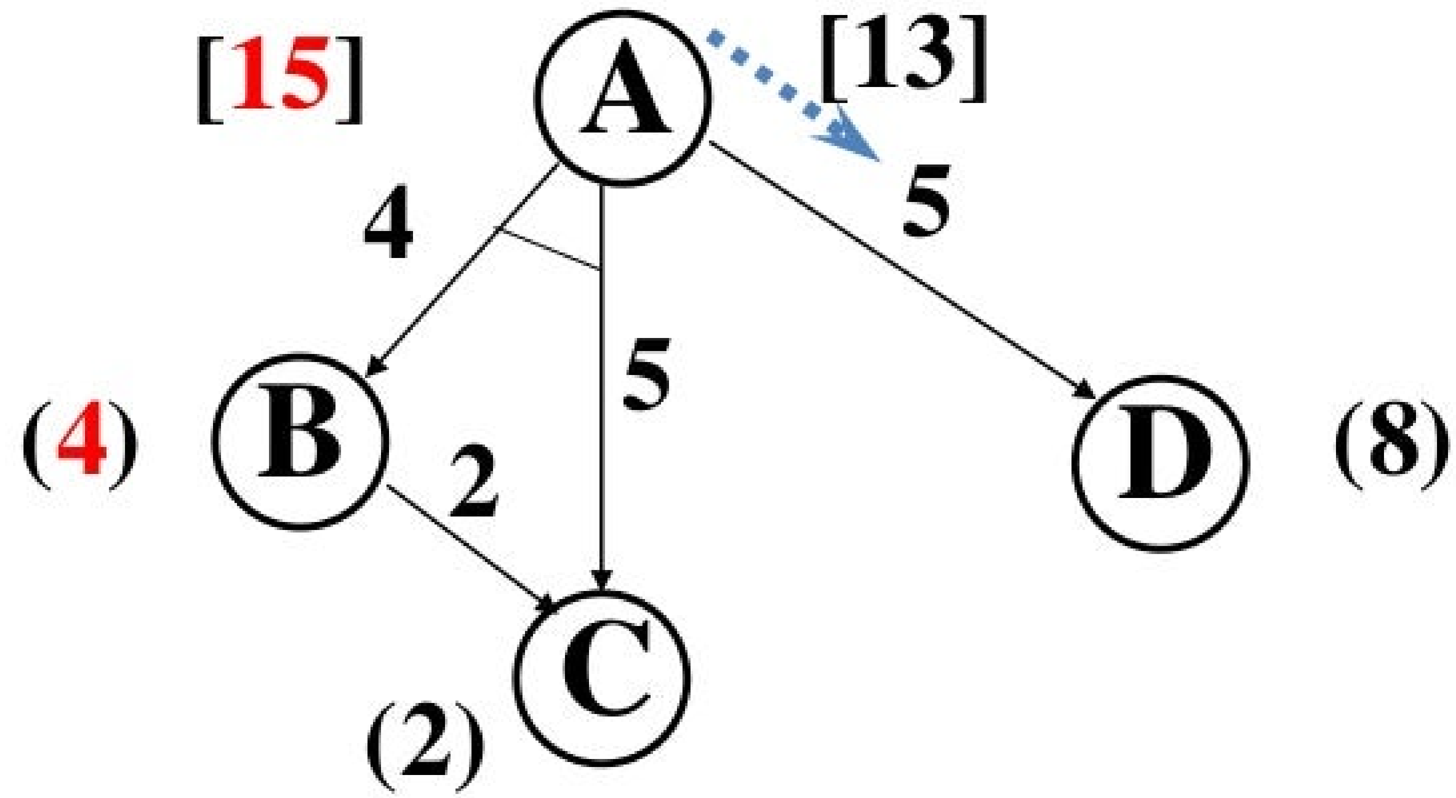
An Example

(8) A

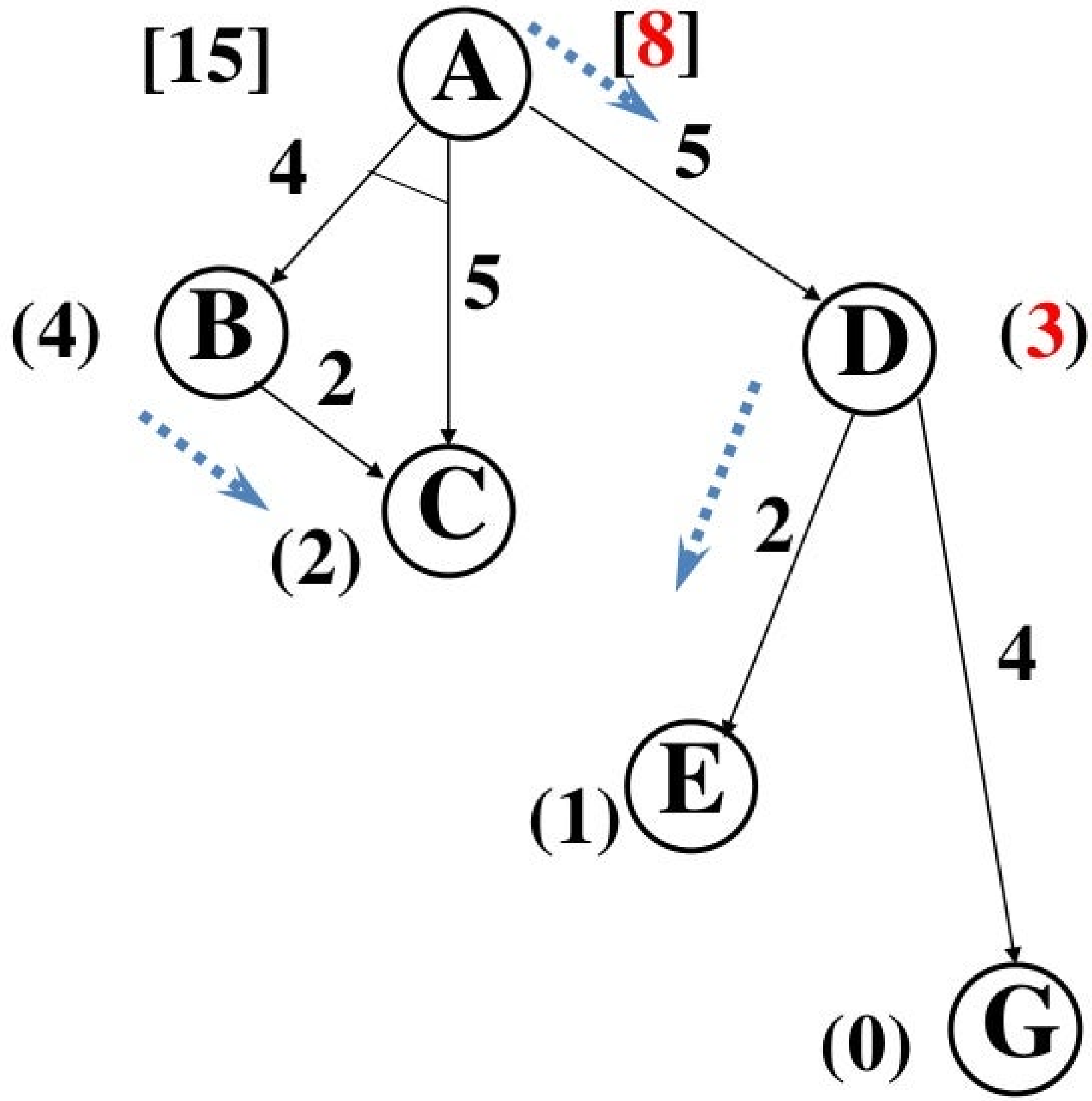
An Example



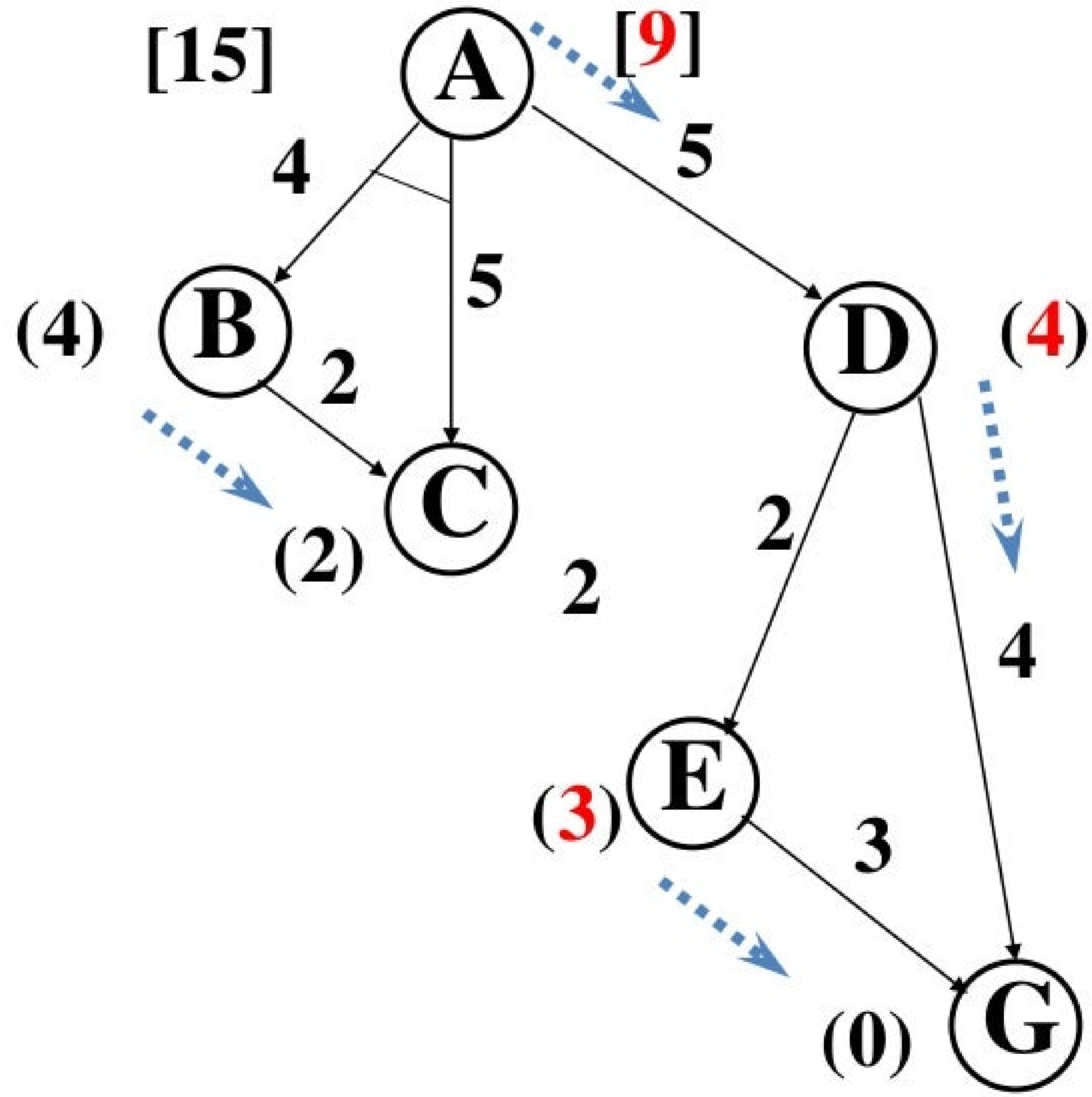
An Example



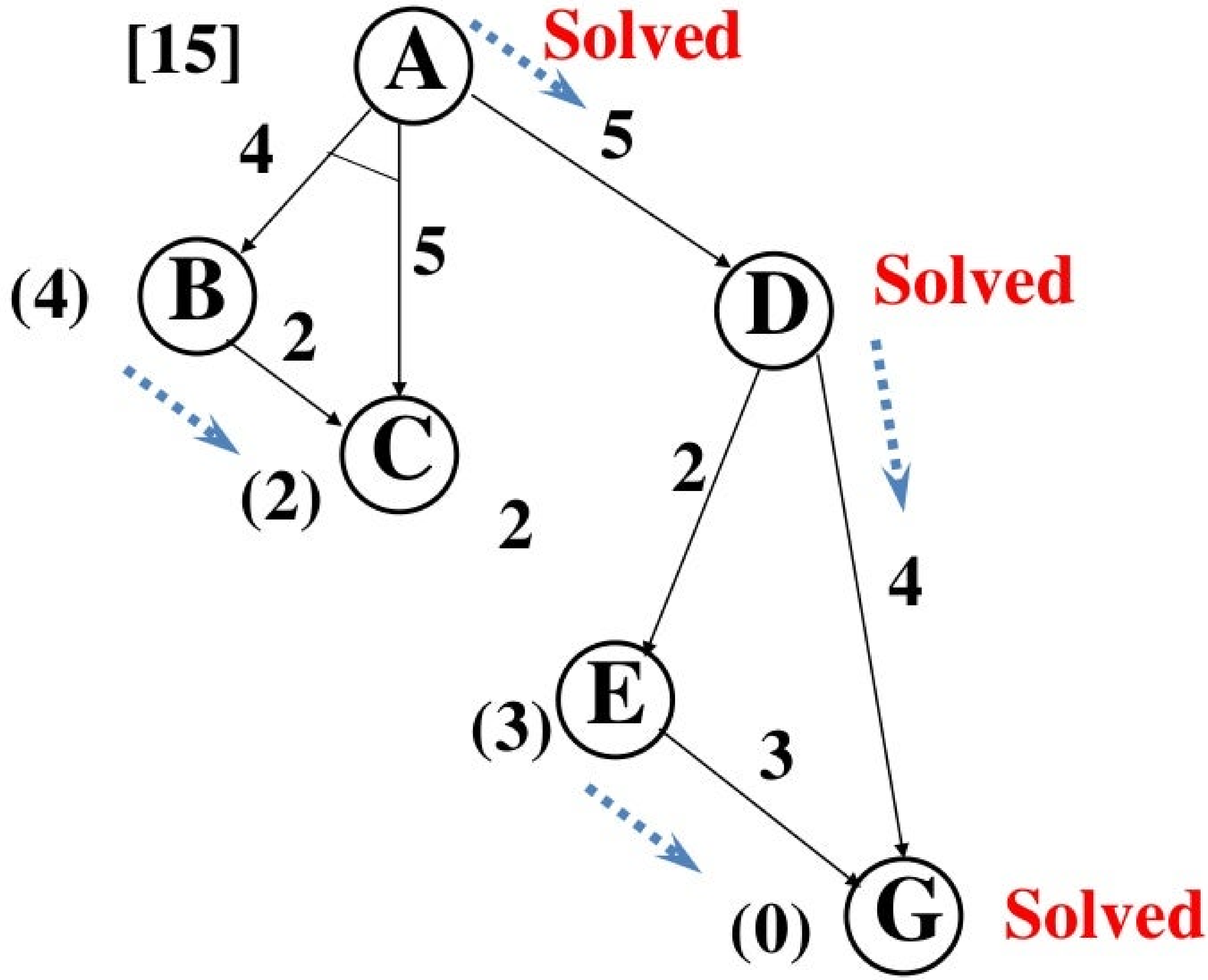
An Example



An Example



An Example



Another Example

Current State = S

f(A) = 3 + 5 = 8

f(B) = 2 + 4 = 6

Current State = B

f(S) = 2 + 8 = 10

f(A) = 4 + 5 = 9

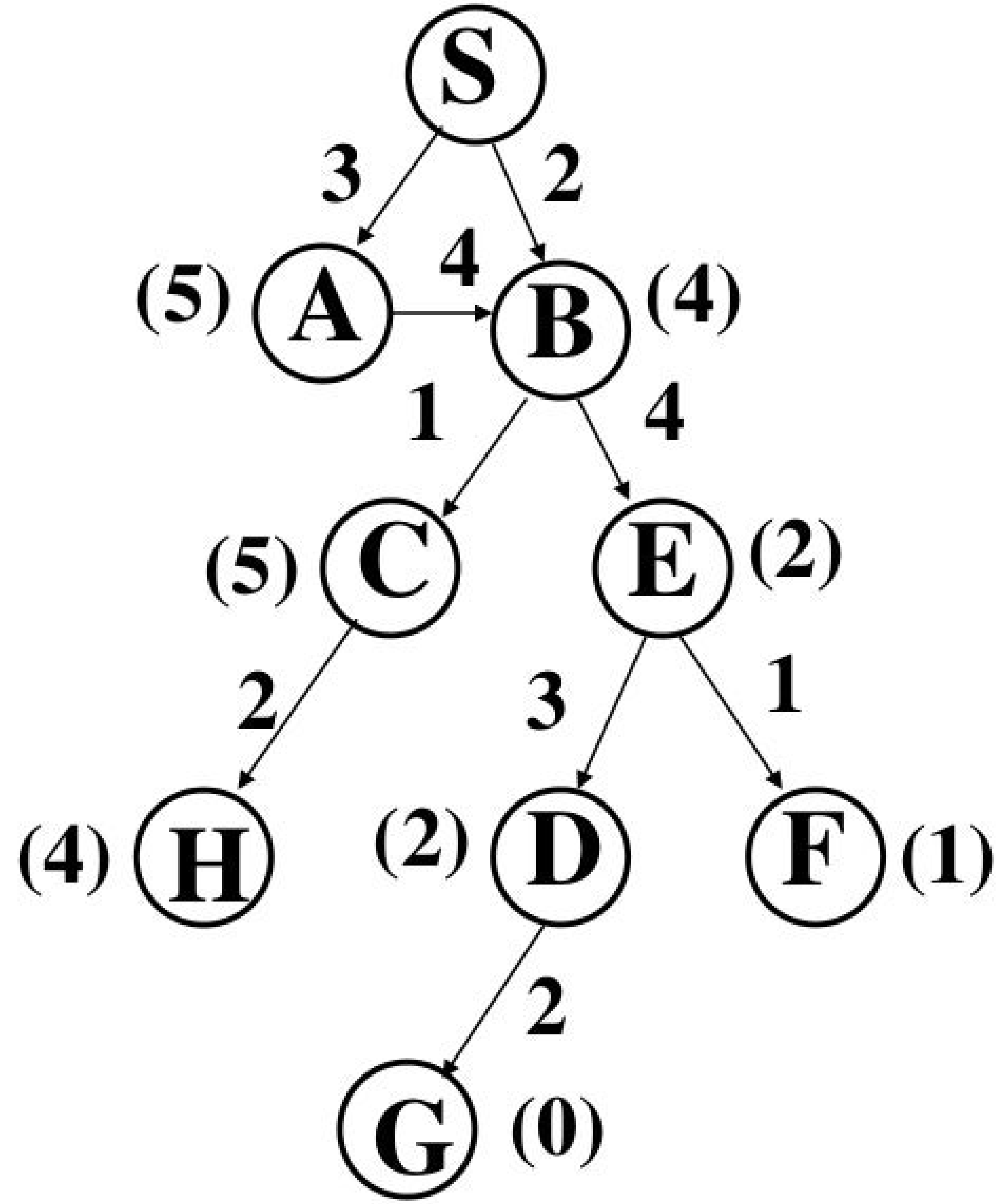
f(C) = 1 + 5 = 6

f(E) = 4 + 2 = 6

Current State = C

f(H) = 2 + 4 = 6

f(B) = 1 + 6 = 7



Another Example

Current State = H

f(C) = 2 + 7 = 9

Current State = C

f(B) = 1 + 6 = 7

f(H) = ∞

Current State = B

f(S) = 2 + 8 = 10

f(A) = 4 + 5 = 9

f(E) = 4 + 2 = 6

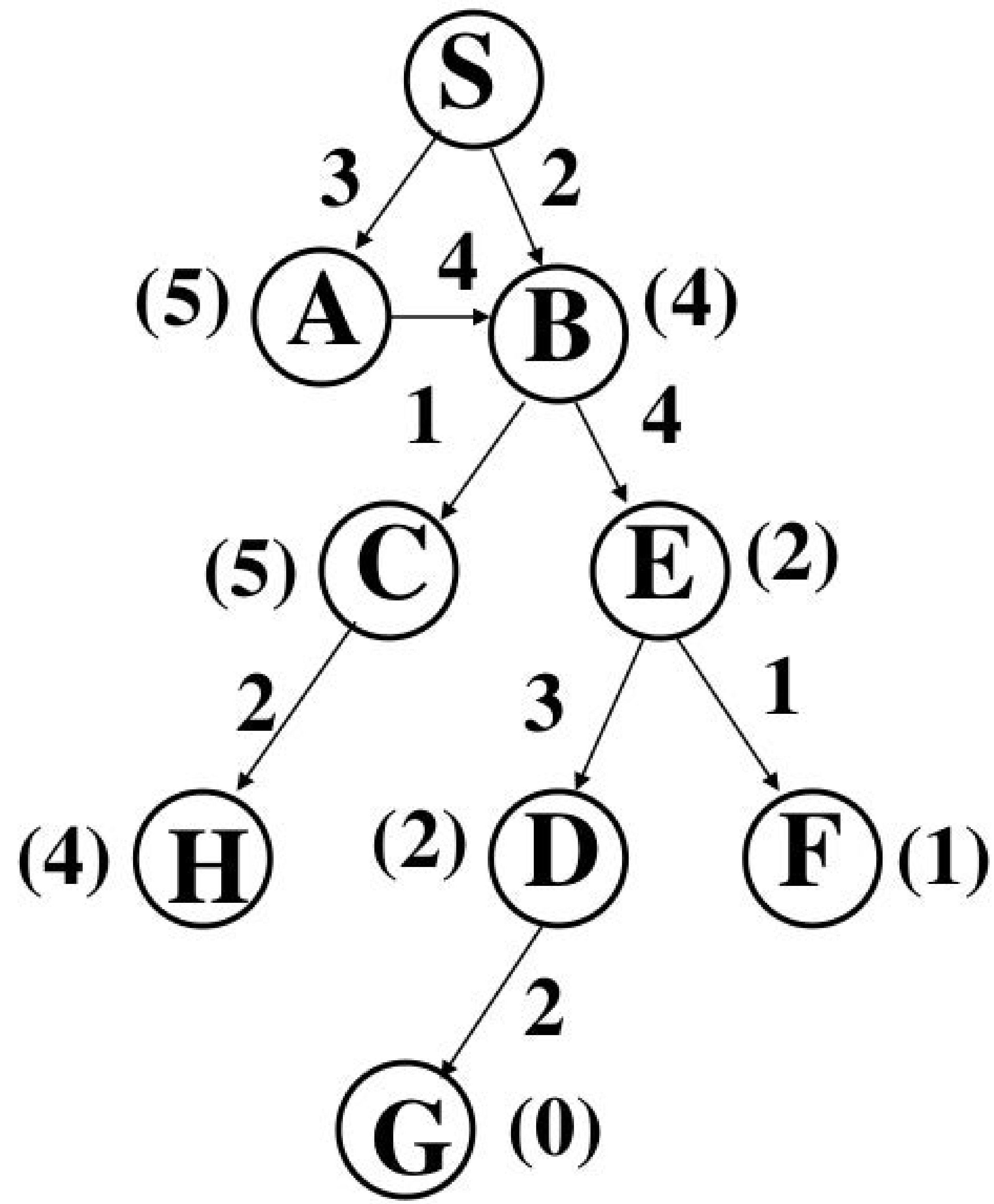
f(C) = ∞

Current State = E

f(B) = 4 + 9 = 13

f(D) = 3 + 2 = 5

f(F) = 1 + 1 = 2



Another Example

Current State = F

$f(E) = 1 + 5 = 6$

Current State = E

$f(D) = 3 + 2 = 5$

$f(B) = 4 + 9 = 13$

$f(F) = \infty$

Current State = D

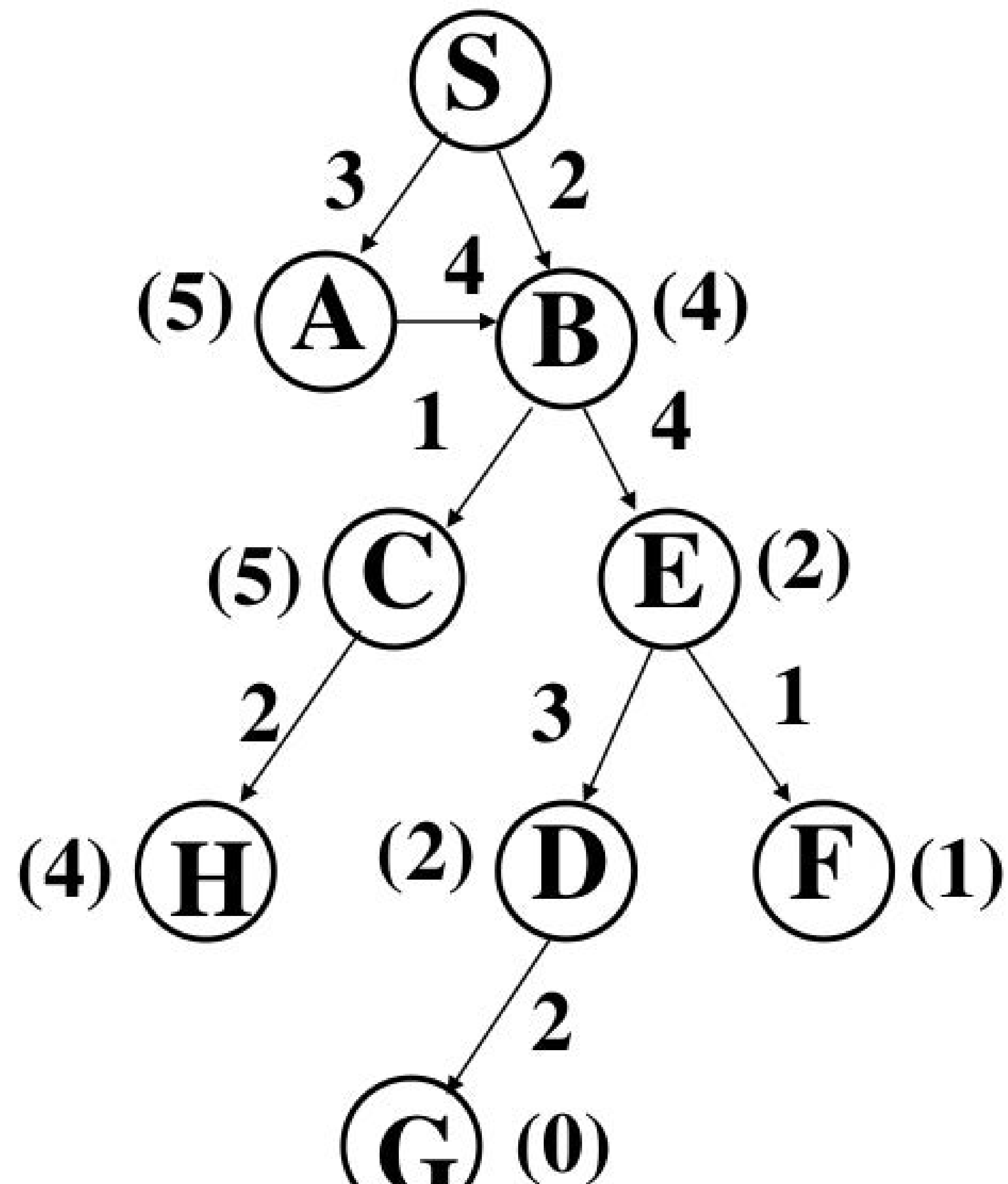
$f(G) = 2 + 0 = 2$

$f(E) = 3 + 13 = 16$

Visited Nodes =

S, B, C, H, C, B, E, F, E, D, G

Path = S, B, E, D, G



Advantages

- It is an optimal algorithm.
- If traverse according to the ordering of nodes.
- It can be used for both OR and AND graph.

Disadvantages

- Sometimes for unsolvable nodes, it can’t find the optimal path.
- Its complexity is than other algorithms.