CS 480 - Introduction of Artificial Intelligence
Written Assignment 1

Rutul Rajeshkumar Mehta
(A20476293)

Answer - 1

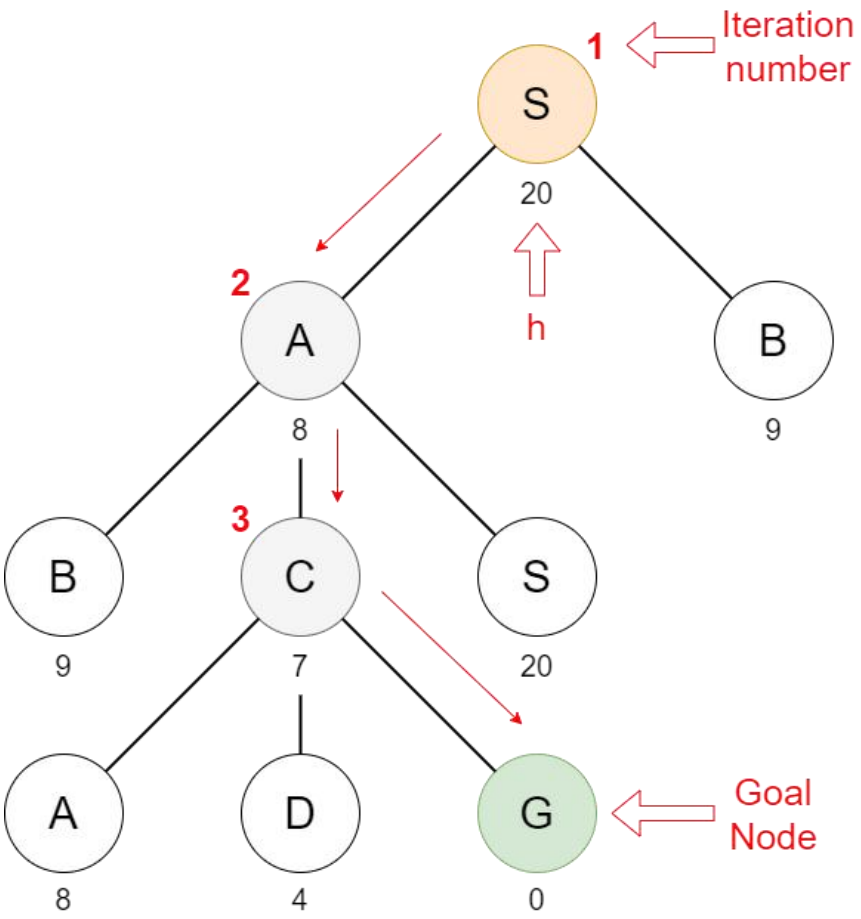

Figure:1 Greedy best-first search algorithm

❖ **Description:**

Greedy best-first search expand the node that is closest to the goal, this is likely to lead to a solution quickly. Thus, it evaluate nodes use just the heuristic function: f(n) = h(n)

In this search example, we are using two lists which are OPEN and CLOSED Lists. Following are the iteration for traversing the above example.

Iterations	**OPEN**	**CLOSED**
Iteration 1	A, B	S,
Iteration 2	B, B, C, S	S, A
Iteration 3	B, B, S, A, D, G	S, A, C

❖ **Final Solution path will be:**

S ---> A ---> C ---> G

❖ **Total path cost will be:**

= C(S, A) + C(A, C) + C(C, G)
= 7 + 7 + 10
= 24

Answer - 2

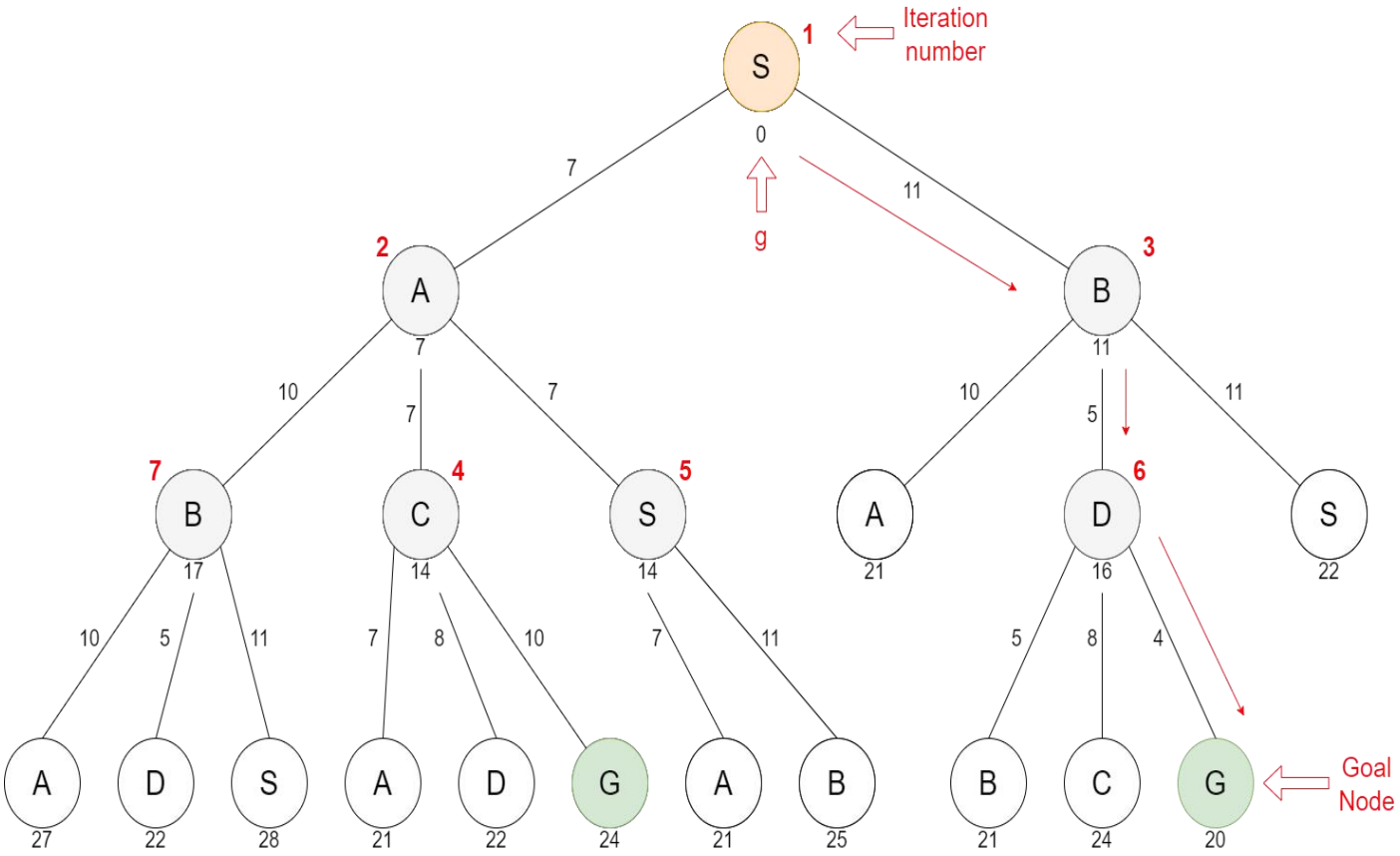

Figure:2 Uniform-cost tree search

❖ **Description:**

Uniform-cost search find a path to the goal node which has the lowest cumulative cost. Uniform-cost search expands nodes on the basis of their path costs g(n).
A uniform-cost search algorithm is implemented by the priority queue. It gives maximum priority to the lowest cumulative cost.

In this search example, we are using two lists which are OPEN and CLOSED Lists. Following are the iteration for traversing the above example

Iteration	OPEN	CLOSED
Iteration 1	A, B	S
Iteration 2	B, B, C, S	S, A
Iteration 3	B, C, S, A, D, S	S, A, B
Iteration 4	B, S, A, D, S, A, D, G	S, A, B, C
Iteration 5	B, A, D, S, A, D, G, A, B	S, A, B, C, S
Iteration 6	B, A, S, A, D, G, A, B, B, C, G	S, A, B, C, S, D
Iteration 7	A, S, A, D, G, A, B, B, C, G, A, D, S	S, A, B, C, S, D, B

❖ **Final Solution path will be:**

S ---> B ---> D ---> G

❖ **Total path cost will be:**

= C(S, B) + C(B, D), C(D, G)
= 11 + 5 + 4
= 20

Answer - 3

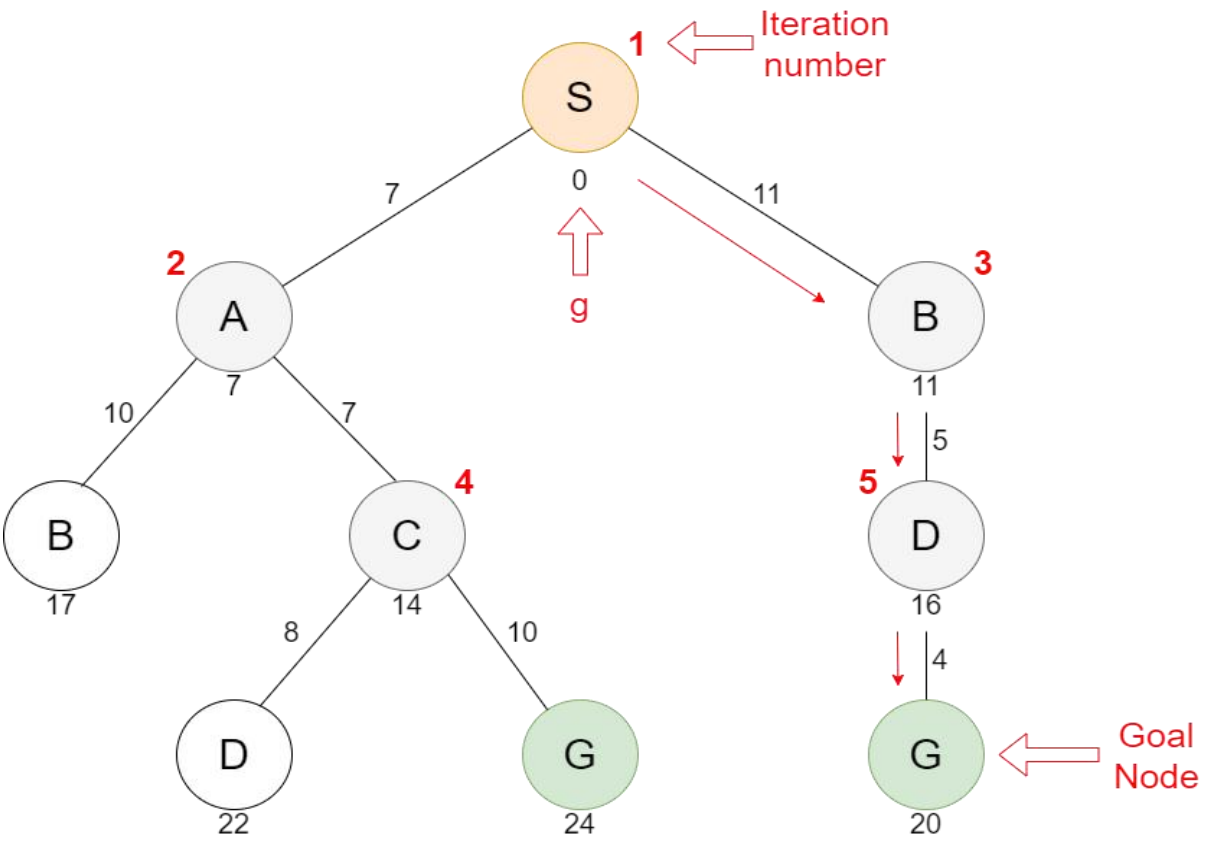

Figure: 3 Uniform-cost graph search

❖ **Description:**

Uniform-cost Graph search is very similar with the uniform-cost tree search algorithm. The only difference is graph remember already explored state and that explored states are never expanded further.
In this search example, we are using three lists which are OPEN, CLOSED and EXPLORED Lists. Following are the iteration for traversing the above example

Iterations	**OPEN**	**CLOSED**	**EXPLORED**
Iteration 1	A, B	S	S
Iteration 2	B, B, C	S, A	S, A
Iteration 3	B, C, D	S, A, B	S, A, B
Iteration 4	B, D, D, G	S, A, B, C	S, A, B, C
Iteration 5	B, D, G, G	S, A, B, C, D	S, A, B, C, D

❖ **Final Solution path will be:**

S ---> B ---> D ---> G

❖ **Total path cost will be:**

= C(S, B) + C(B, D), C(D, G)
= 11 + 5 + 4
= 20

Answer - 4

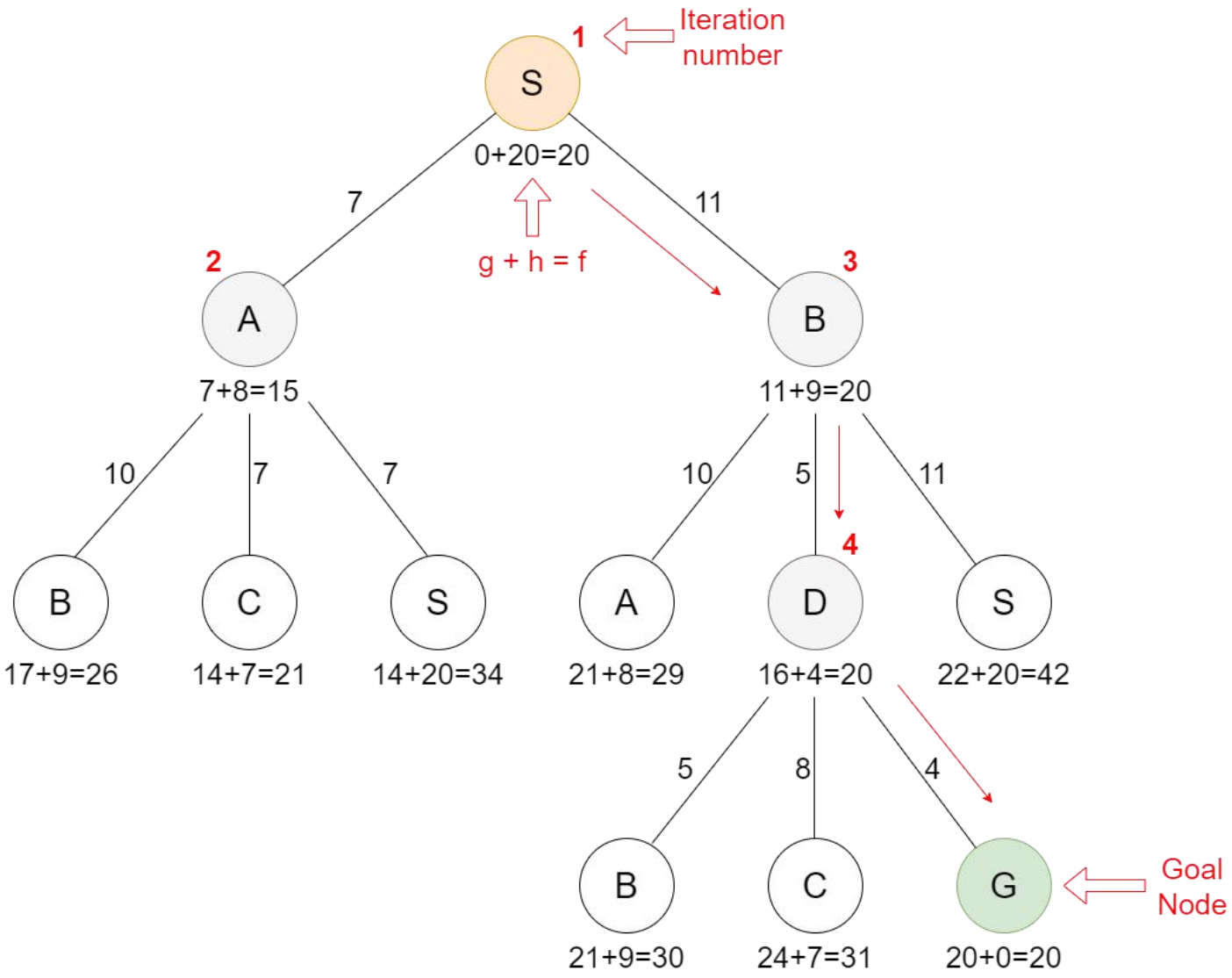

Figure:4 A* algorithm

❖ **Description:**

A* algorithm is a combination of UCS and greedy best-first search, by which it solve the problem efficiently.
f(n) = g(n) + h(n), where
f(n) = Estimated cost of the cheapest solution
g(n) = Path Cost to reach node n.
h(n) = Flying cost from starting node n to goal node

Iterations	(n --> n', f(n') = g(n') + h(n'))
Iteration 1	(S --> A, 15), (S --> B, 20)
Iteration 2	(A --> B, 26), (A --> C, 21), (A --> S, 34)
Iteration 3	(B --> A, 29), (B --> D, 20), (B --> S, 42)
Iteration 4	(D --> B, 30), (D --> C, 31), (D --> G, 20)

❖ **Final Solution path will be:**

S ---> B ---> D ---> G

❖ **Total path cost will be:**

= C(S, B) + C(B, D), C(D, G)
= 11 + 5 + 4
= 20

Answer - 5

h*(n) = Estimate cost from starting node 'n' to goal node 'G'
h(n) = Flying cost from starting node n to goal node.

h*(n) >= h(n) means h*(n) dominate h(n).

In this example, value of h*(n) remains same for UCS algorithm and A* algorithm.

h*(S) = 20	=	h(S) = 20	=>	h*(S) dominate h(S)
h*(A) = 17	>	h(A) = 8	=>	h*(A) dominate h(A)
h*(B) = 9	=	h(B) = 9	=>	h*(B) dominate h(B)
h*(C) = 10	>	h(C) = 7	=>	h*(C) dominate h(C)
h*(D) = 4	=	h(D) = 4	=>	h*(D) dominate h(D)
h*(G) = 0	=	h(G) = 0	=>	h*(G) dominate h(G)

Answer - 6

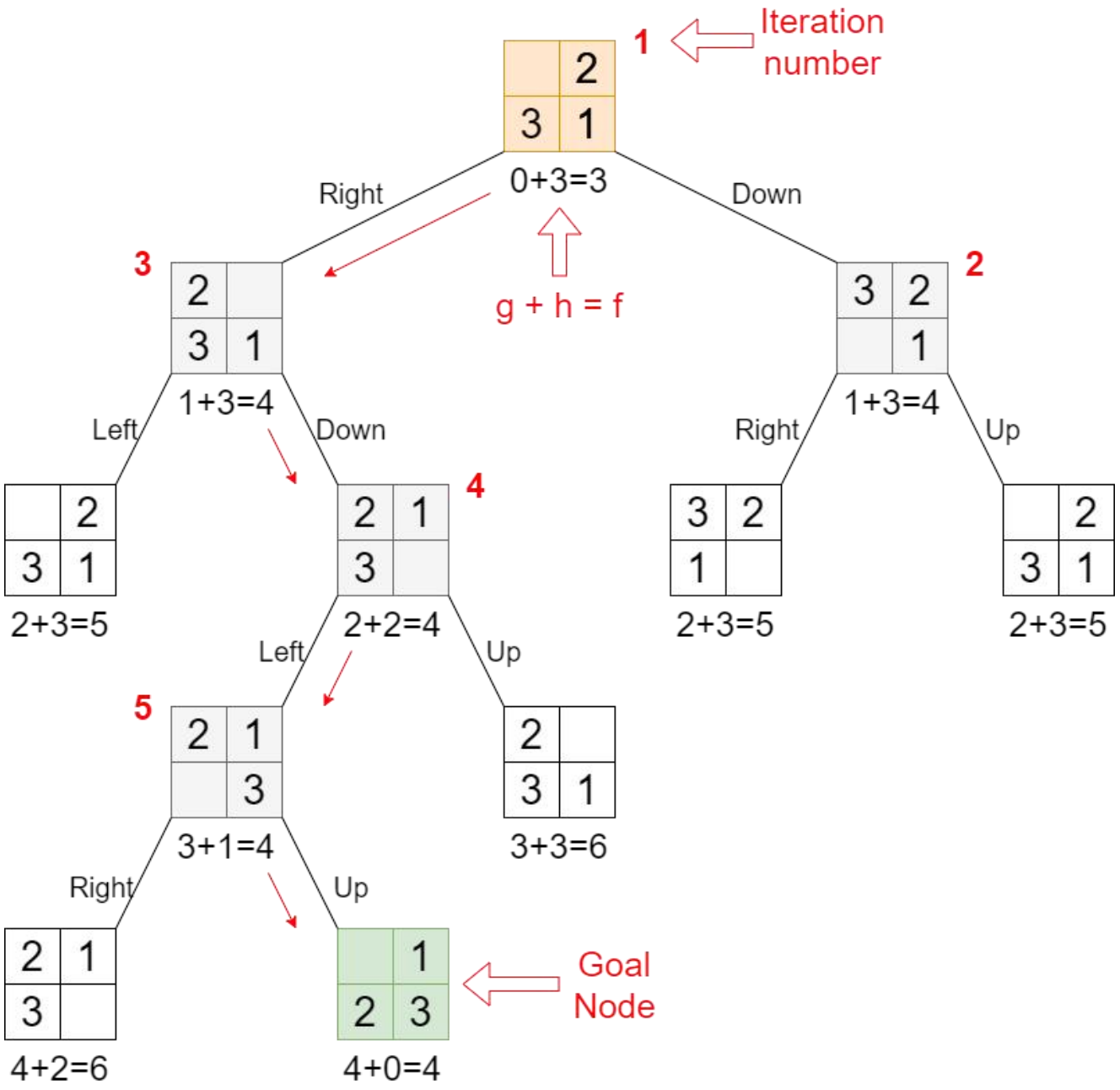

Figure: 5 3-puzzle problem using A* tree search algorithm.

❖ **Description:**

To reach from start node to goal node we can use left, right, up and down operations.
Each operation has 1 cost. g(n) = 1
For h(n) use the number of misplaced tile.
According to A* algorithm, find f(n) = g(n) + h(n) to find optimal solution.

❖ **Final Solution path will be:**

Represent solution path with in figure 5.

❖ **Total path cost will be:**

= Right + Down + Left + Up
= 1 + 1 + 1+ 1
= 4