1 a)

Goal Node reached! (Node 0 --> Node 1 --> Node 3 --> Node 7 (Goal))

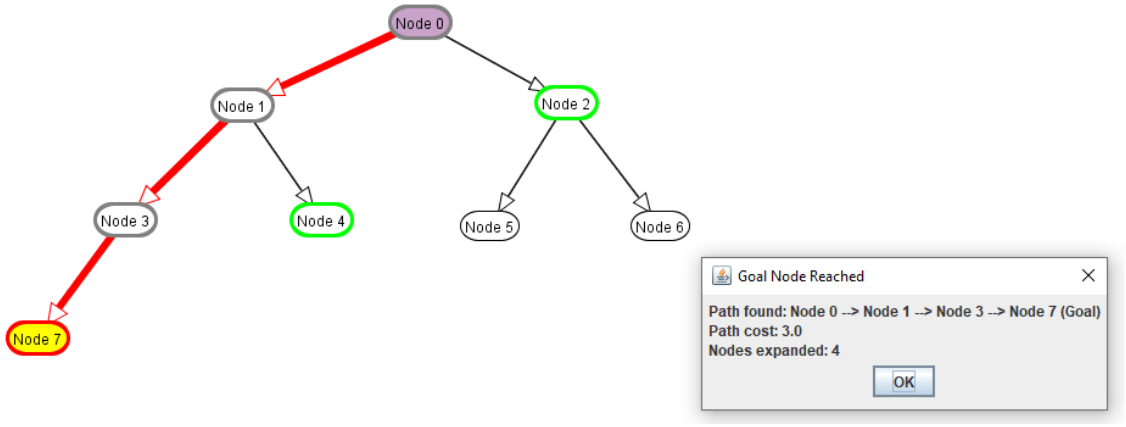


Figure 1.1: DFS

Goal Node reached! (Node 0 --> Node 1 --> Node 3 --> Node 7 (Goal))

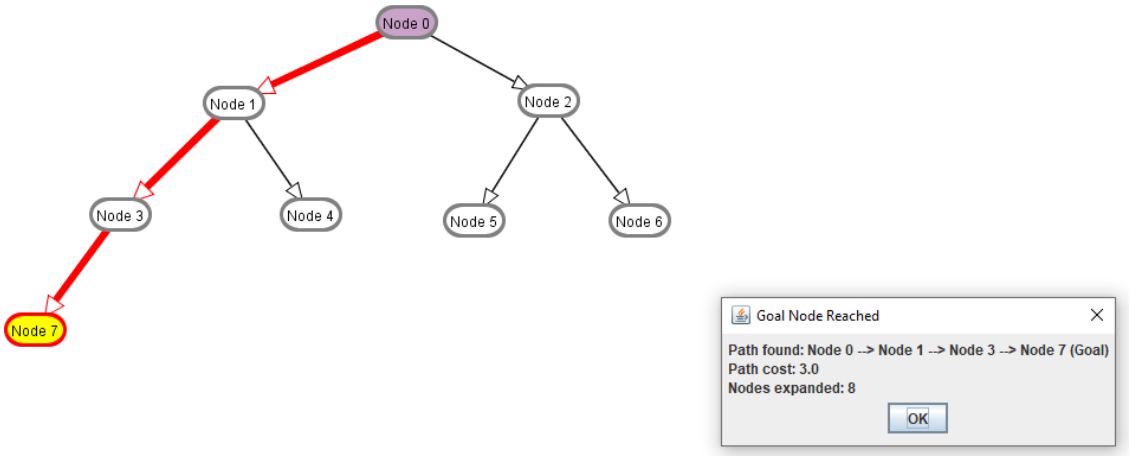


Figure 1.2: BFS

1 b)

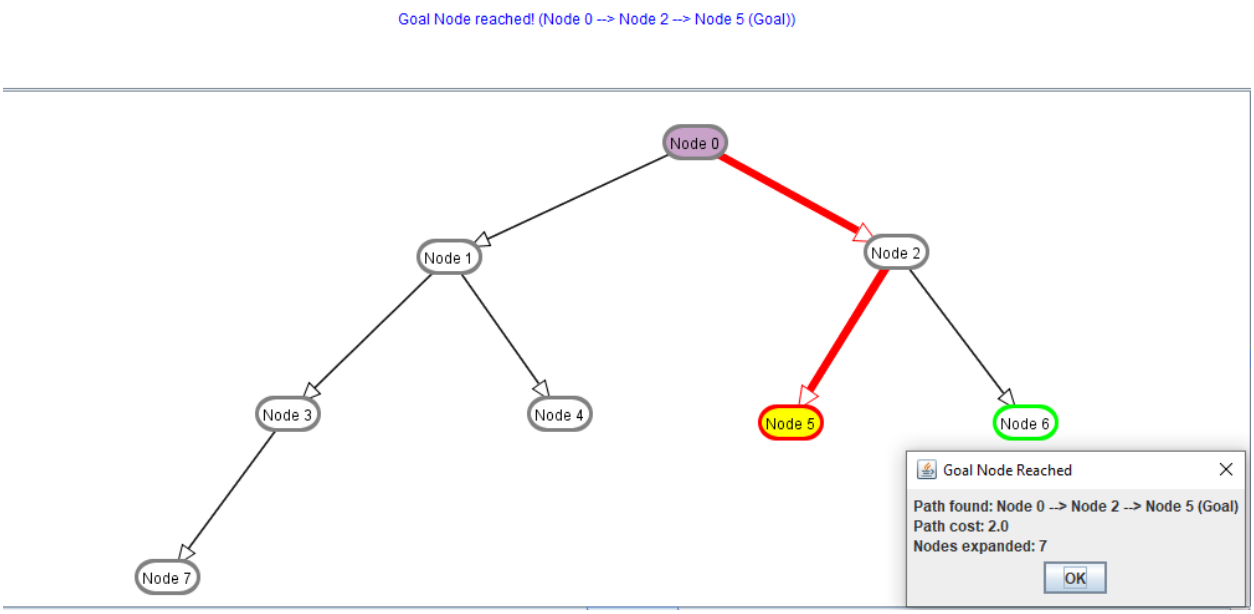


Figure 1.3: DFS

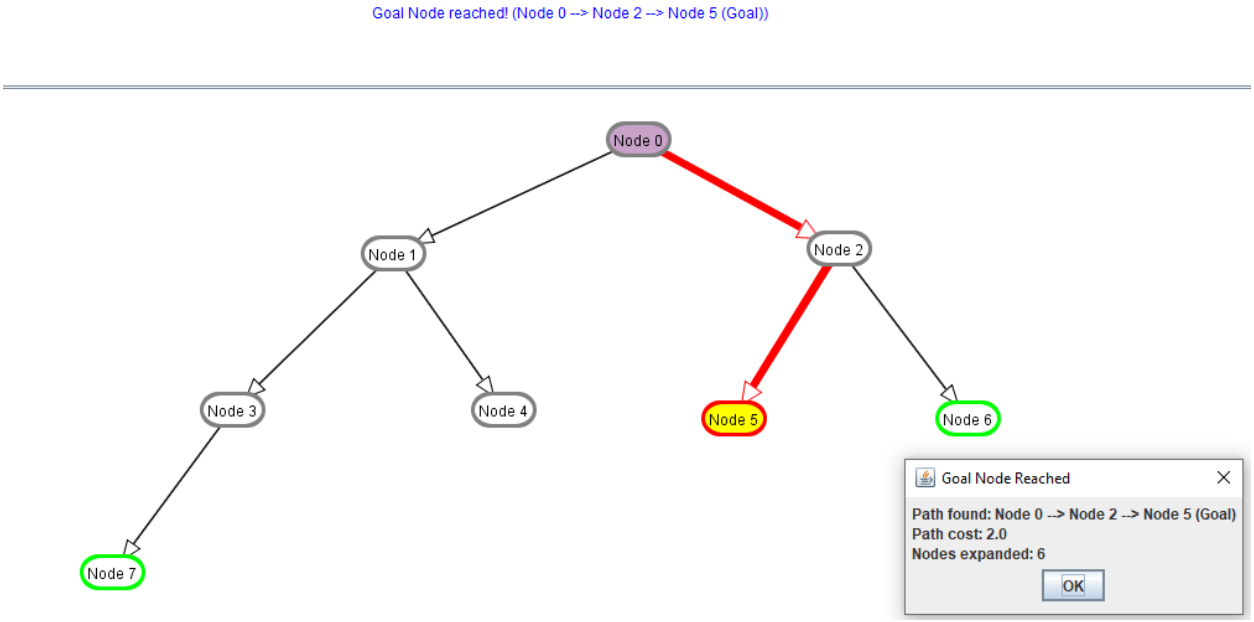


Figure 1.4: BFS

1 c) When there are heuristic values, estimated cost of reaching the goal.

Goal Node reached! (Node 0 --> Node 1 --> Node 3 --> Node 8 (Goal))

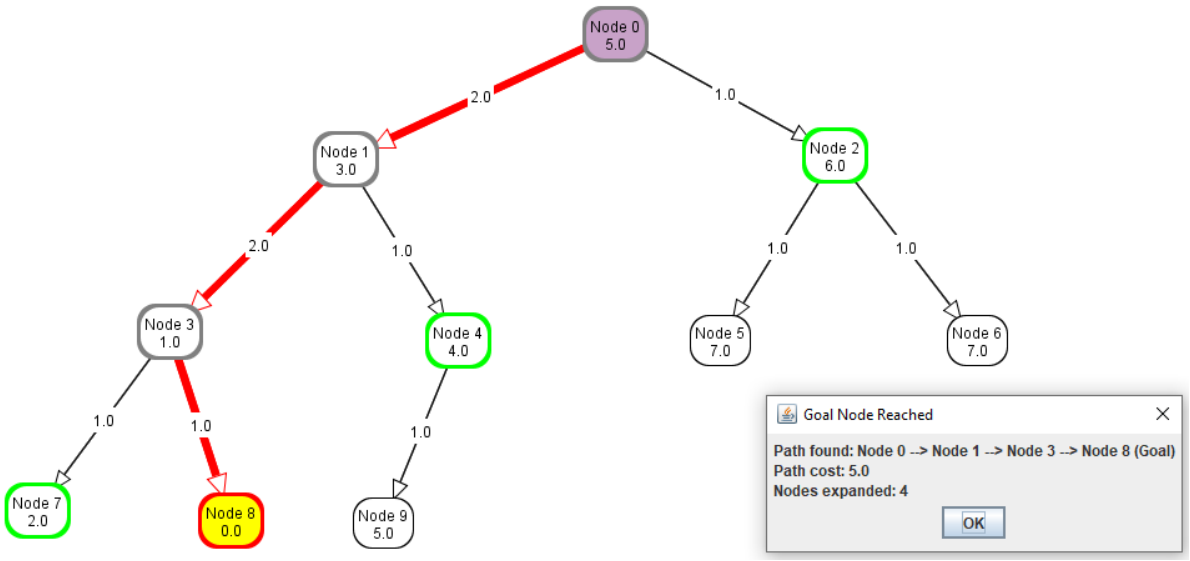


Figure 1.5: A*

Goal Node reached! (Node 0 --> Node 1 --> Node 3 --> Node 8 (Goal))

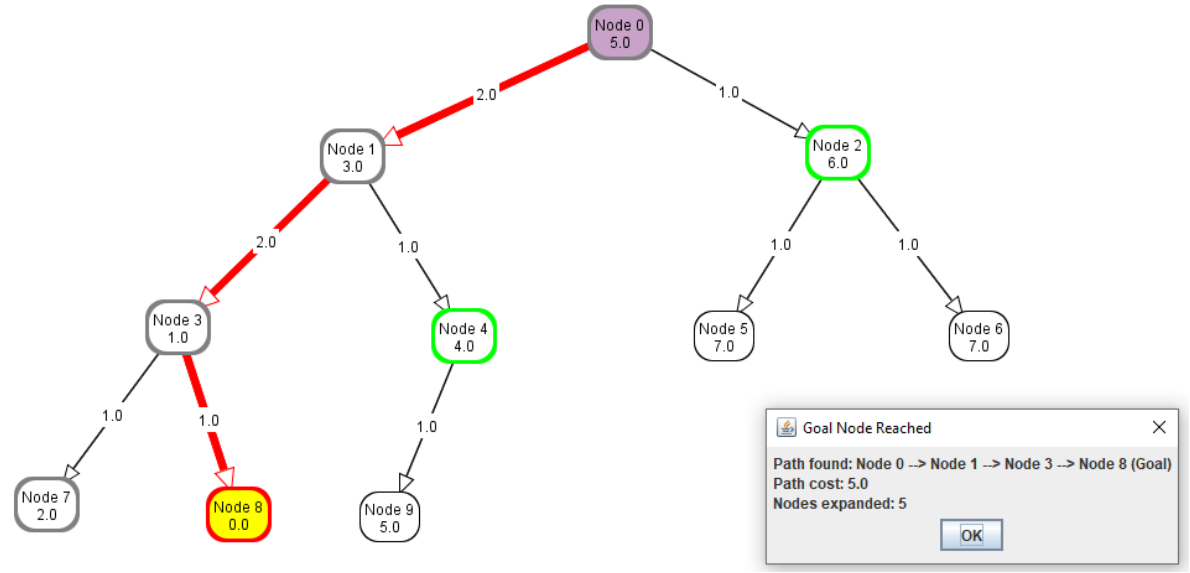


Figure 1.6: DFS

Goal Node reached! (Node 0 --> Node 1 --> Node 3 --> Node 8 (Goal))

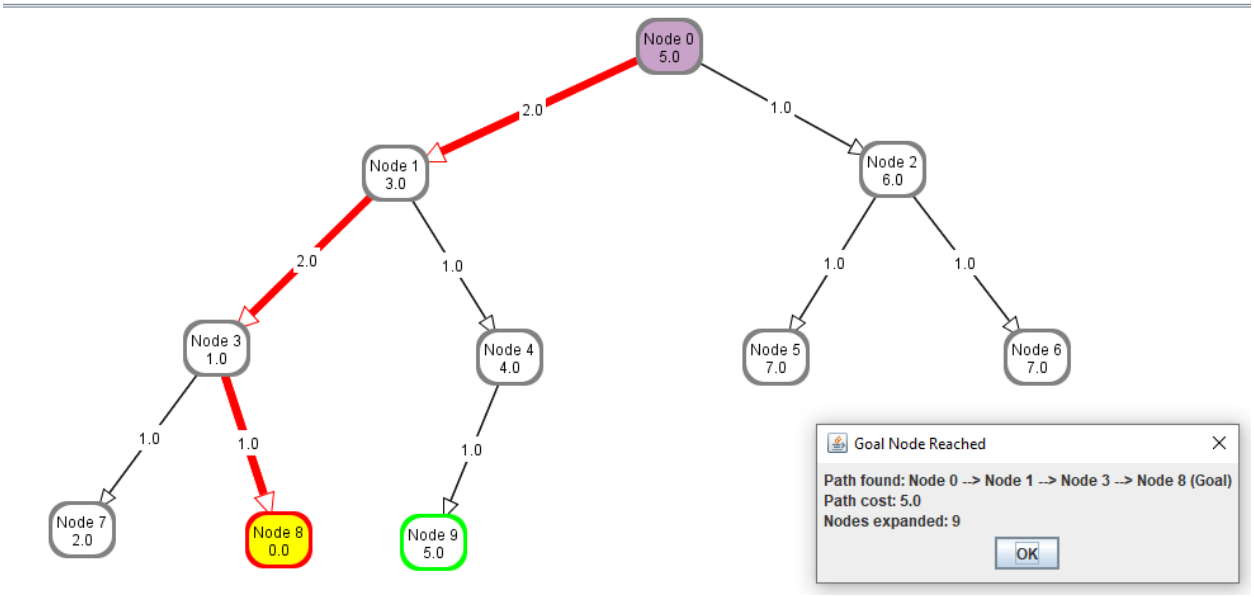


Figure 1.7: BFS

1 d) When the heuristic values are inconsistent, A* will sometimes be inefficient.

Goal Node reached! (Node 0 --> Node 1 --> Node 3 (Goal))

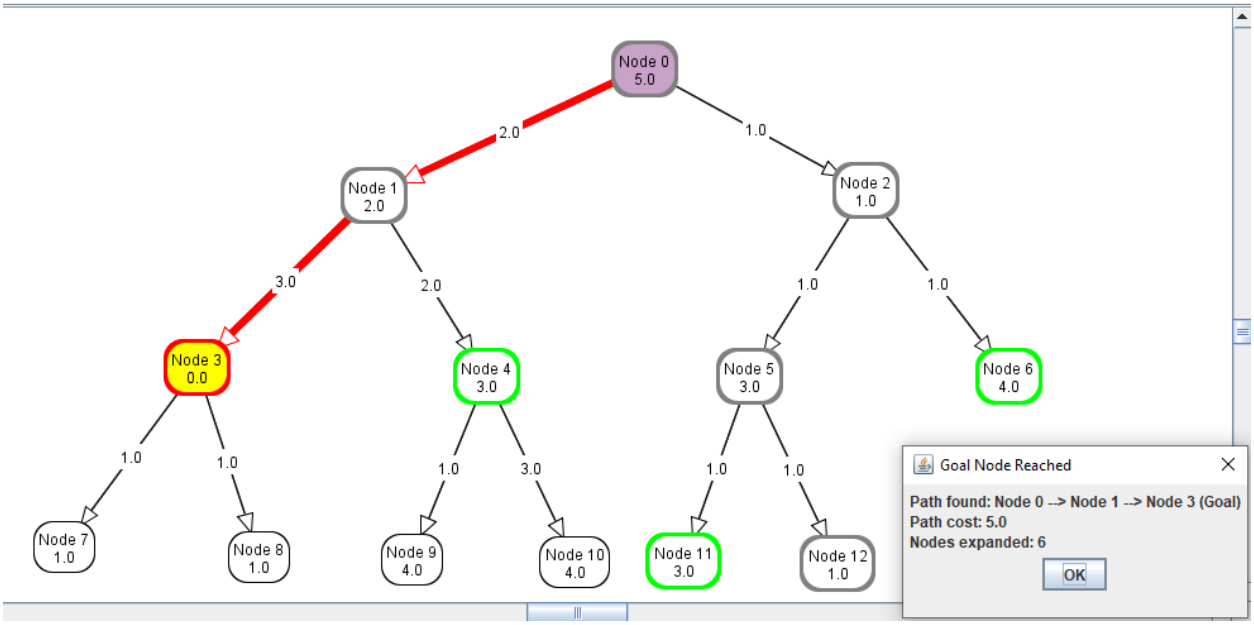


Figure 1.8: A*

Goal Node reached! (Node 0 --> Node 1 --> Node 3 (Goal))

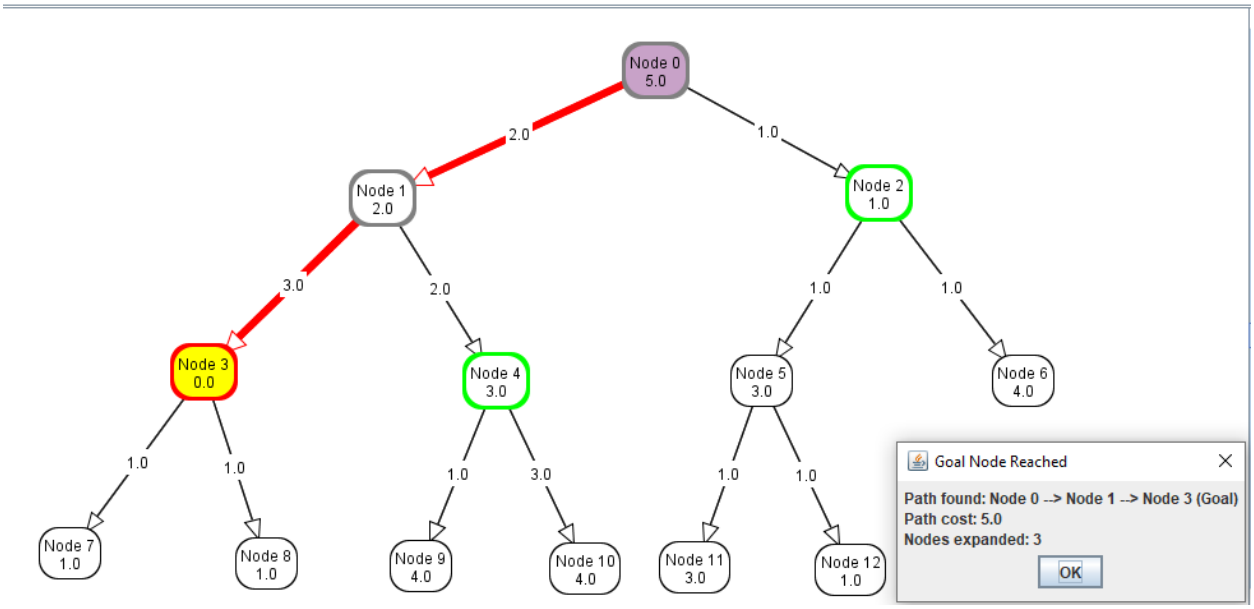


Figure 1.9: DFS

Goal Node reached! (Node 0 --> Node 1 --> Node 3 (Goal))

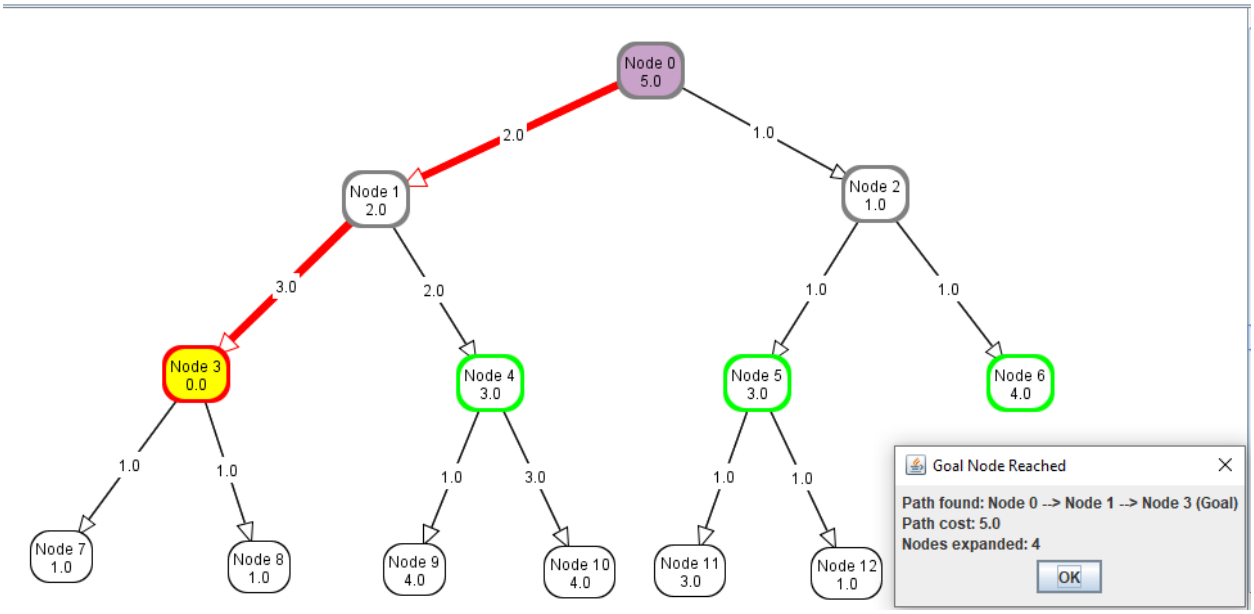


Figure 1.10: BFS

2 a) Time taken to find an optimal solution will sometimes be slower, but solution is always optimal.

If h(n) is an underestimate, A* will explore paths even if it is a non-optimal path, but the cost of exploration will start to add up and eventually be a non-optimal path. Then it will start to explore other paths until it reaches the goal. The solution will always be optimal.

Example: If we were to reduce h(n) by factors of its default value

Referring to Figure 2 below,

h(n) = 1x, Nodes Expanded = 5.

h(n) = 0.9x, Nodes Expanded = 5.

h(n) = 0.7x, Nodes Expanded = 5.

h(n) = 0.3x, Nodes Expanded = 7.

h(n) = 0.1x, Nodes Expanded = 7.

We can conclude that as the value of h(n) decreases, time taken will increase, however solution is always optimal.

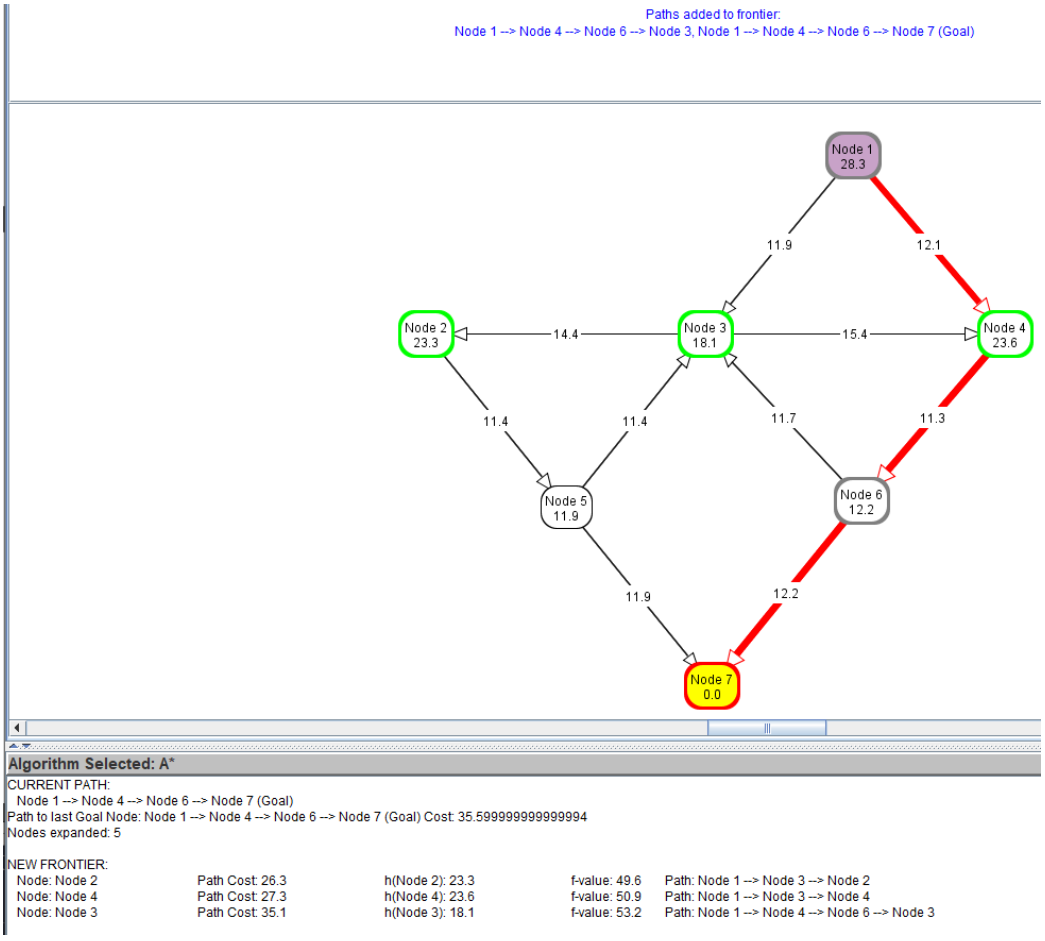


Figure 2.1: Auto Generated A* (Default h(n) values)

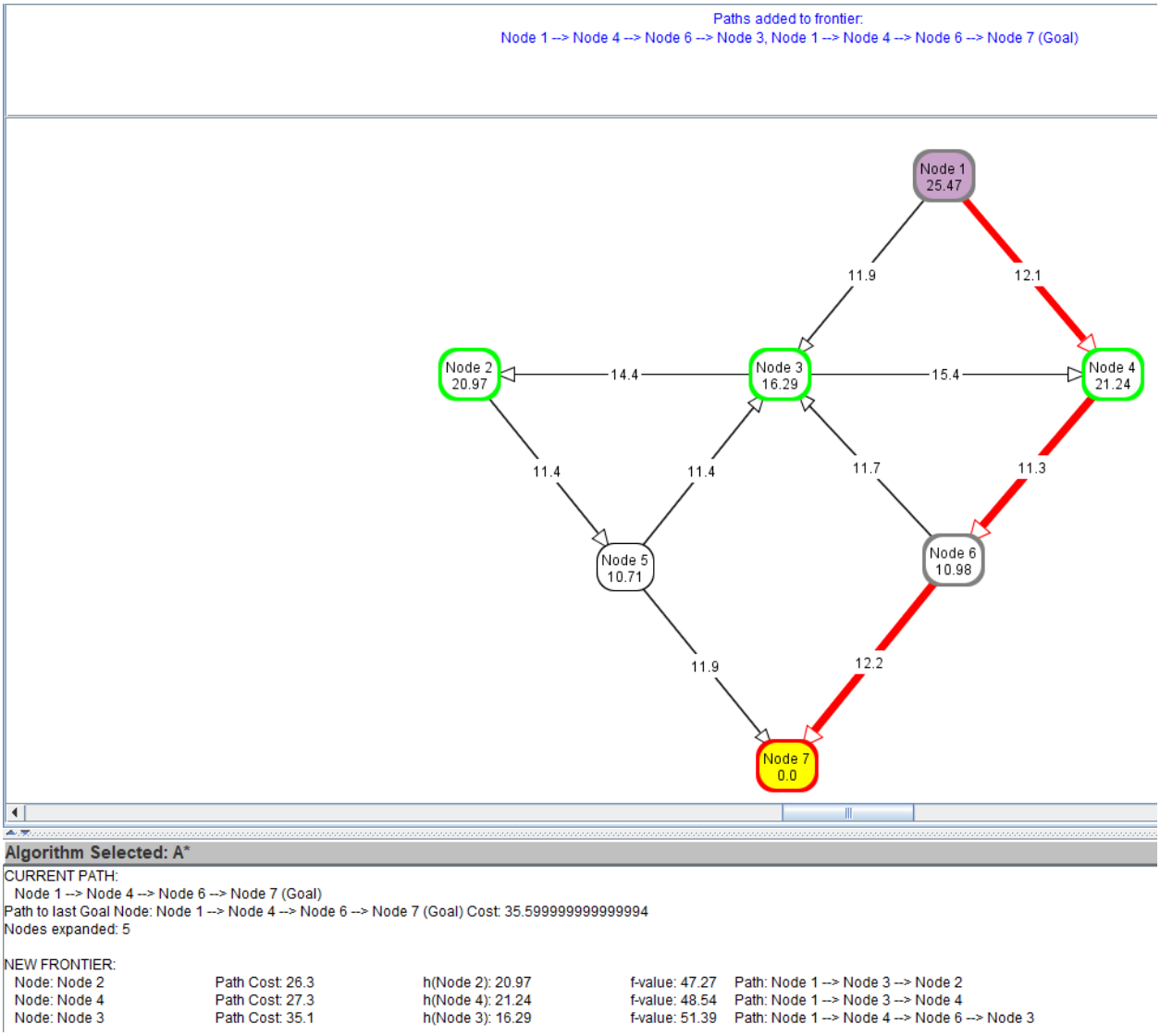


Figure 2.2: Auto Generated A* (h(n) = 0.9*h(n))

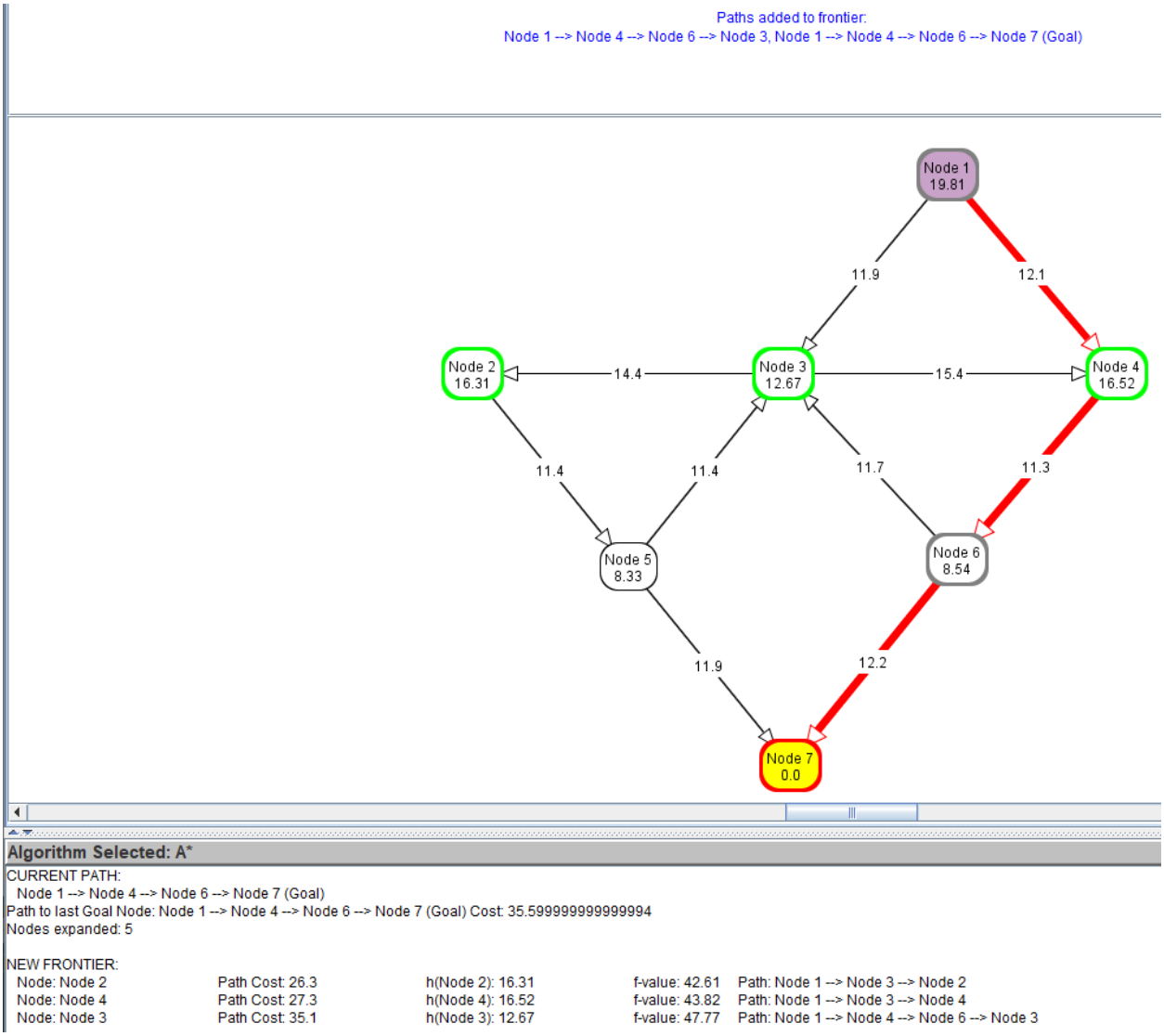


Figure 2.3: Auto Generated A* (h(n) = 0.7*h(n))

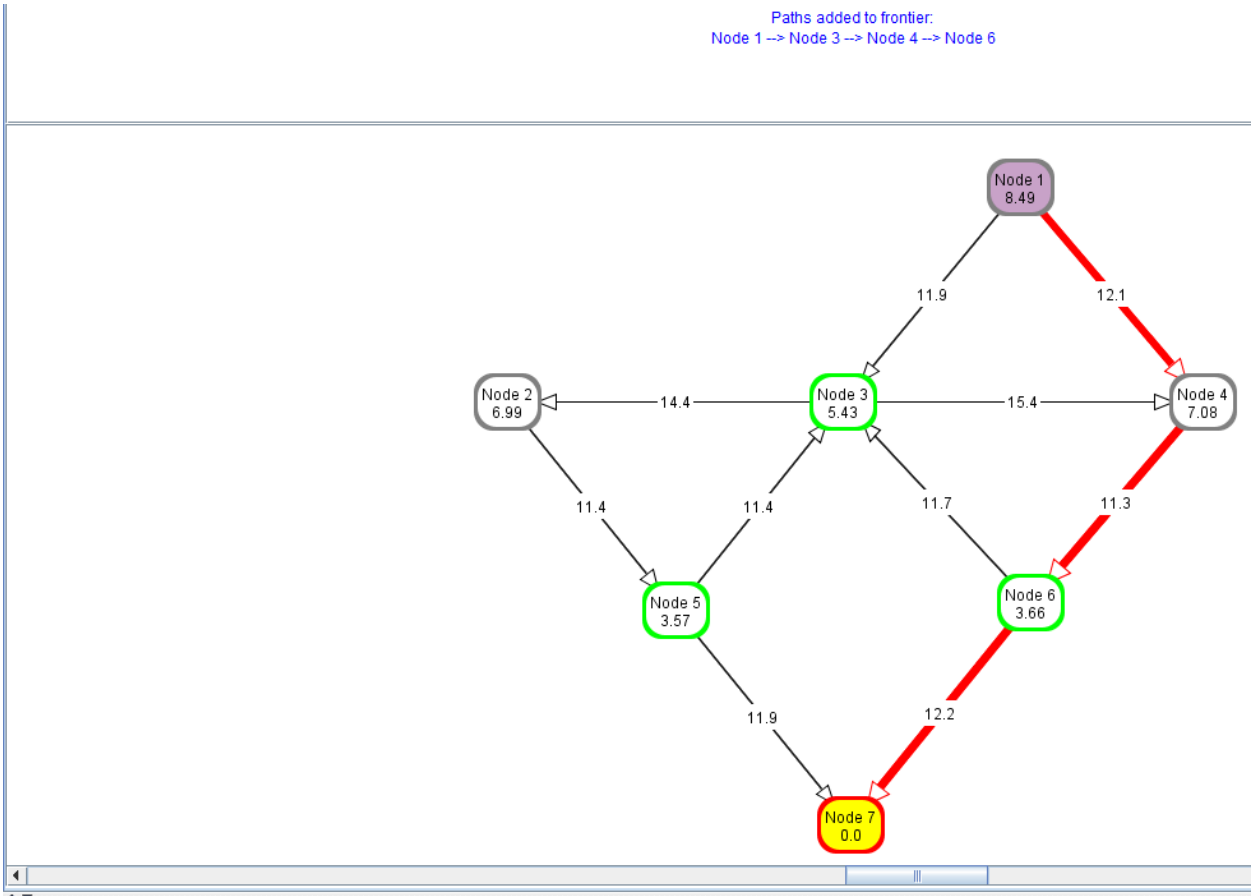
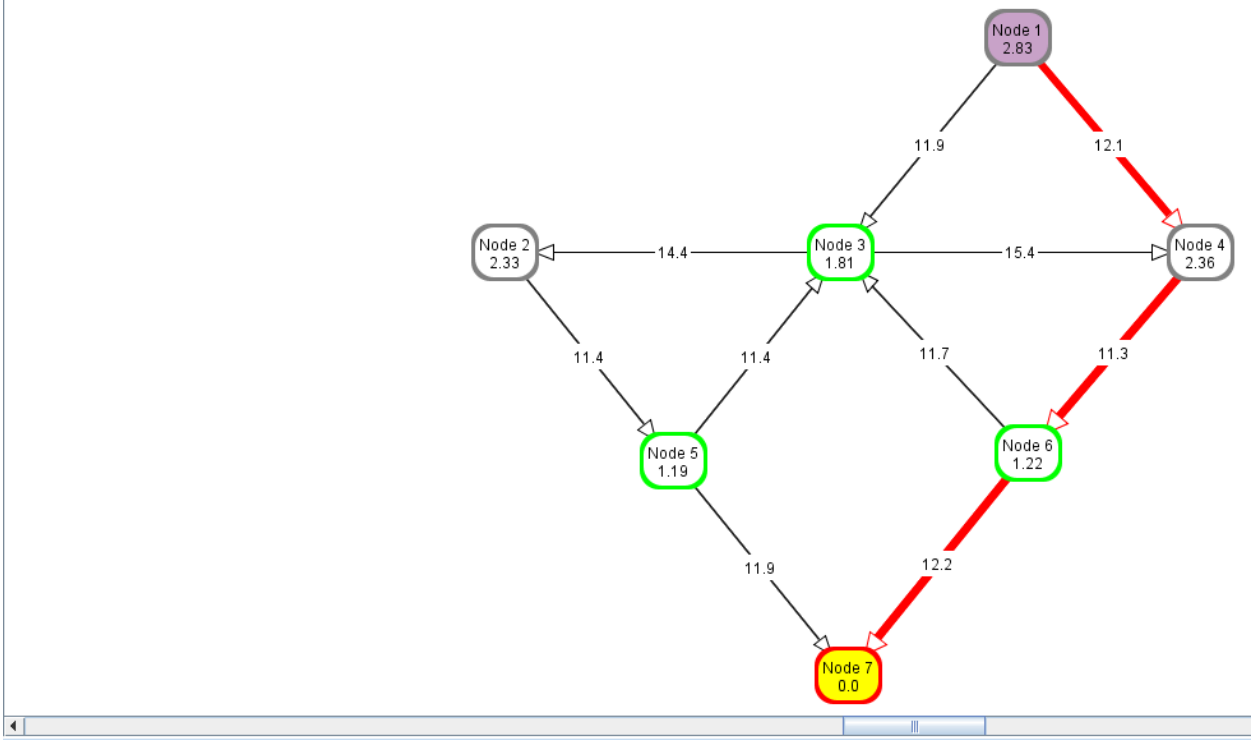


Figure 2.4: Auto Generated A* (h(n) = 0.3*h(n))

Paths added to frontier:
Node 1 --> Node 3 --> Node 4 --> Node 6



Algorithm Selected: A*

CURRENT PATH:
Node 1 --> Node 4 --> Node 6 --> Node 7 (Goal)
Path to last Goal Node: Node 1 --> Node 4 --> Node 6 --> Node 7 (Goal) Cost: 35.599999999999994
Nodes expanded: 7

NEW FRONTIER:

Node: Node 3	Path Cost: 35.1	h(Node 3): 1.81	f-value: 36.91	Path: Node 1 --> Node 4 --> Node 6 --> Node 3
Node: Node 5	Path Cost: 37.7	h(Node 5): 1.19	f-value: 38.89	Path: Node 1 --> Node 3 --> Node 2 --> Node 5
Node: Node 6	Path Cost: 38.6	h(Node 6): 1.22	f-value: 39.82	Path: Node 1 --> Node 3 --> Node 4 --> Node 6

Figure 2.5: Auto Generated A* (h(n) = 0.1*h(n))

2 b) If h(n) is the exact distance, A* will calculate the estimated cost and it will reach the goal with the optimal path.

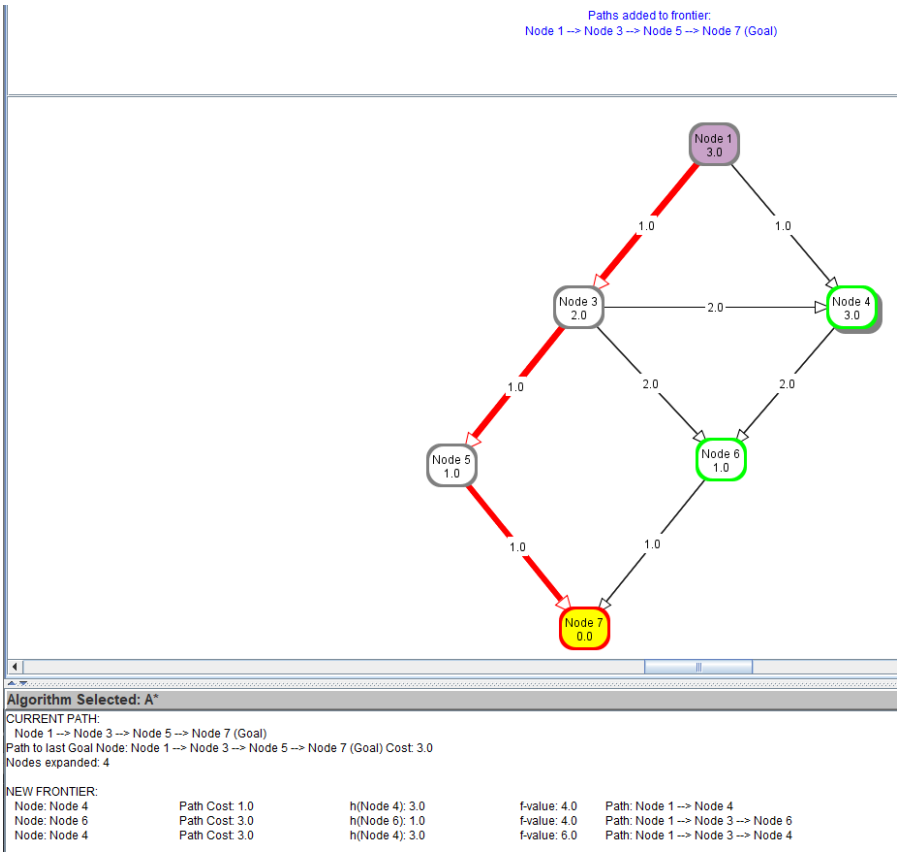


Figure 2.6: A* with exact h(n)

2 c) If h(n) overestimates, A* will explore paths even if it is a non-optimal path, and it will not explore an optimal path with overestimated h(n), because its value is worser. The solution will not be optimal.

Example:

We can see that the optimal path is from N1>N3>N5>N7, with a cost of 6. However, since the h(n) values of N3 and N5 are overestimated, A* will not go down that optimal path, and will go on another path to reach the goal, where the solution is not optimal.

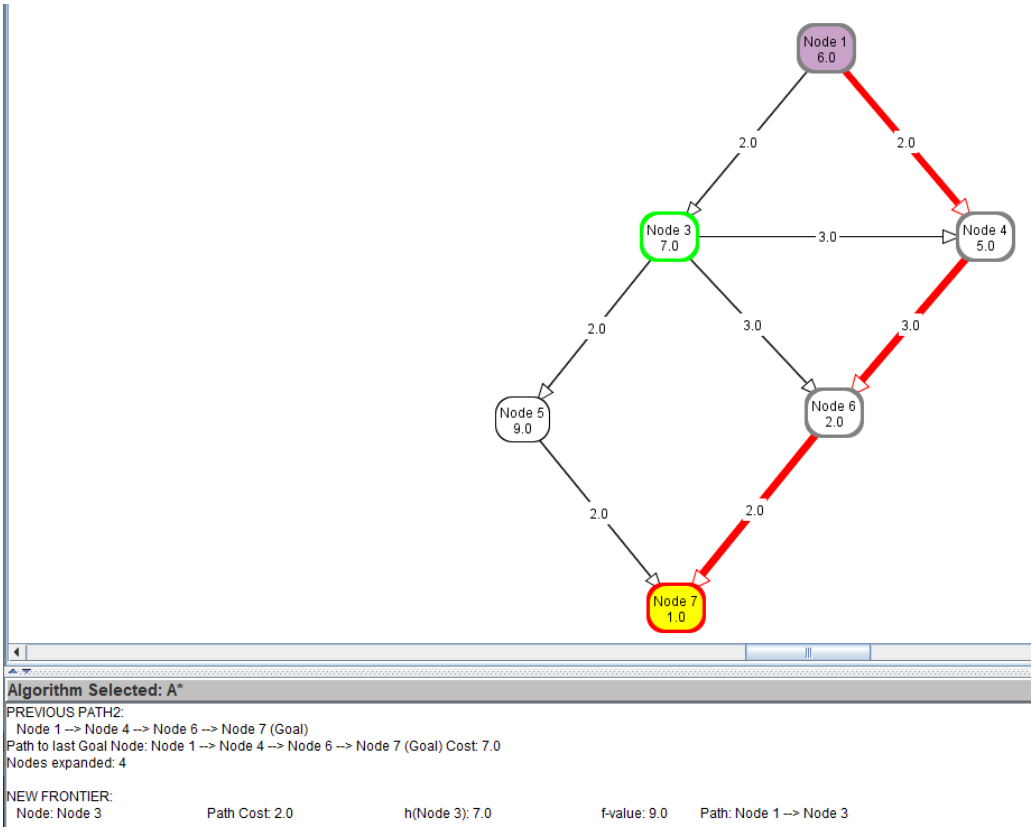


Figure 2.7: A* with overestimated h(n)