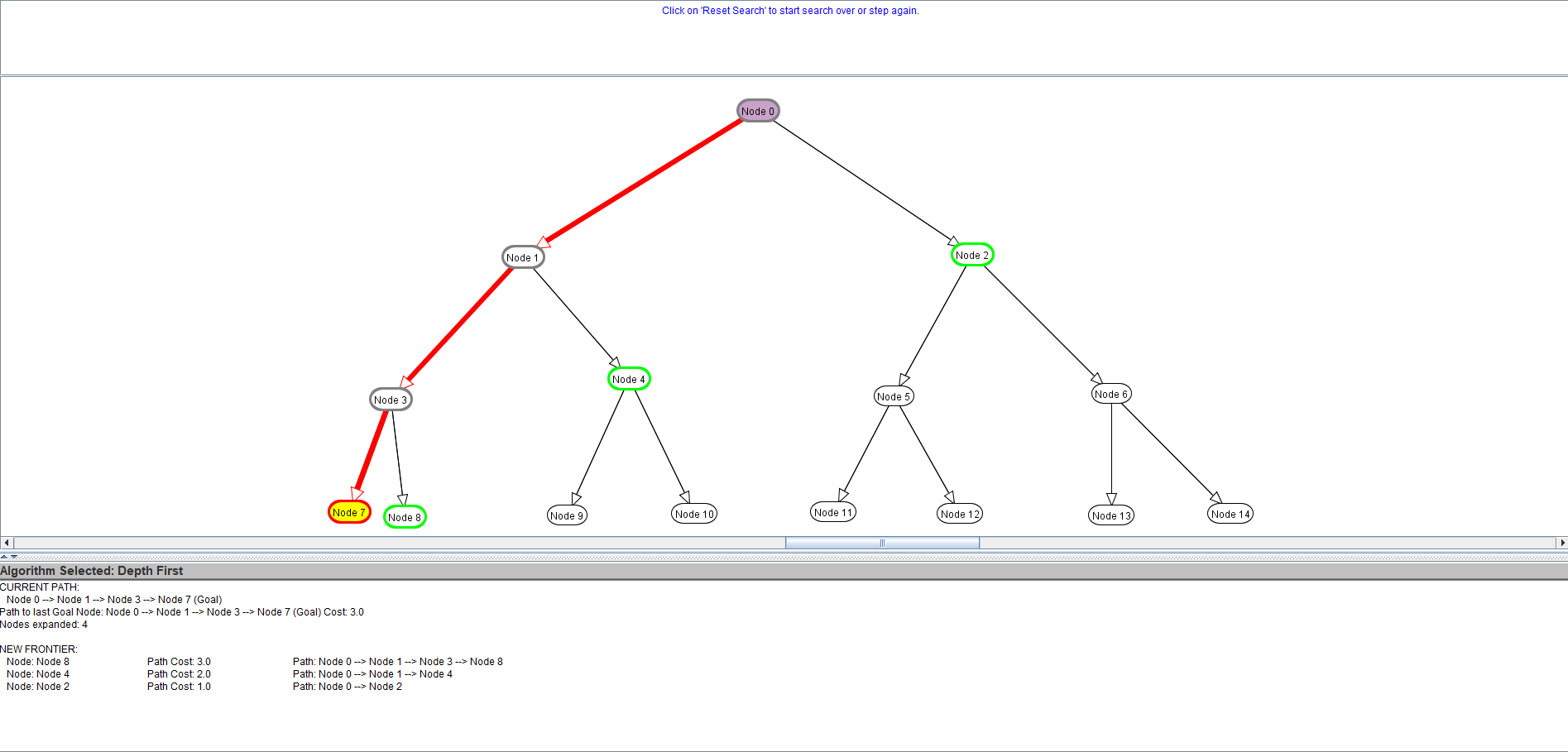
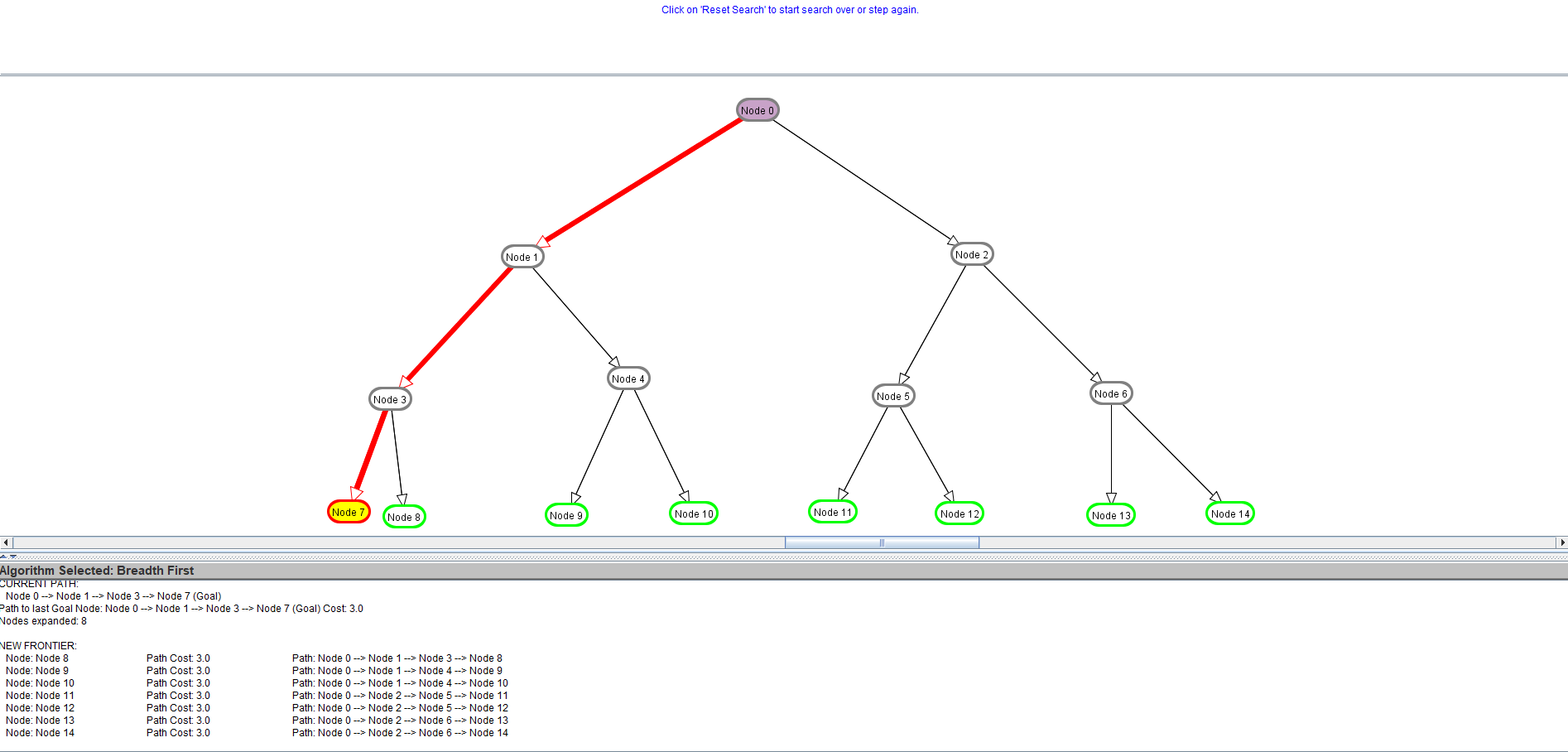
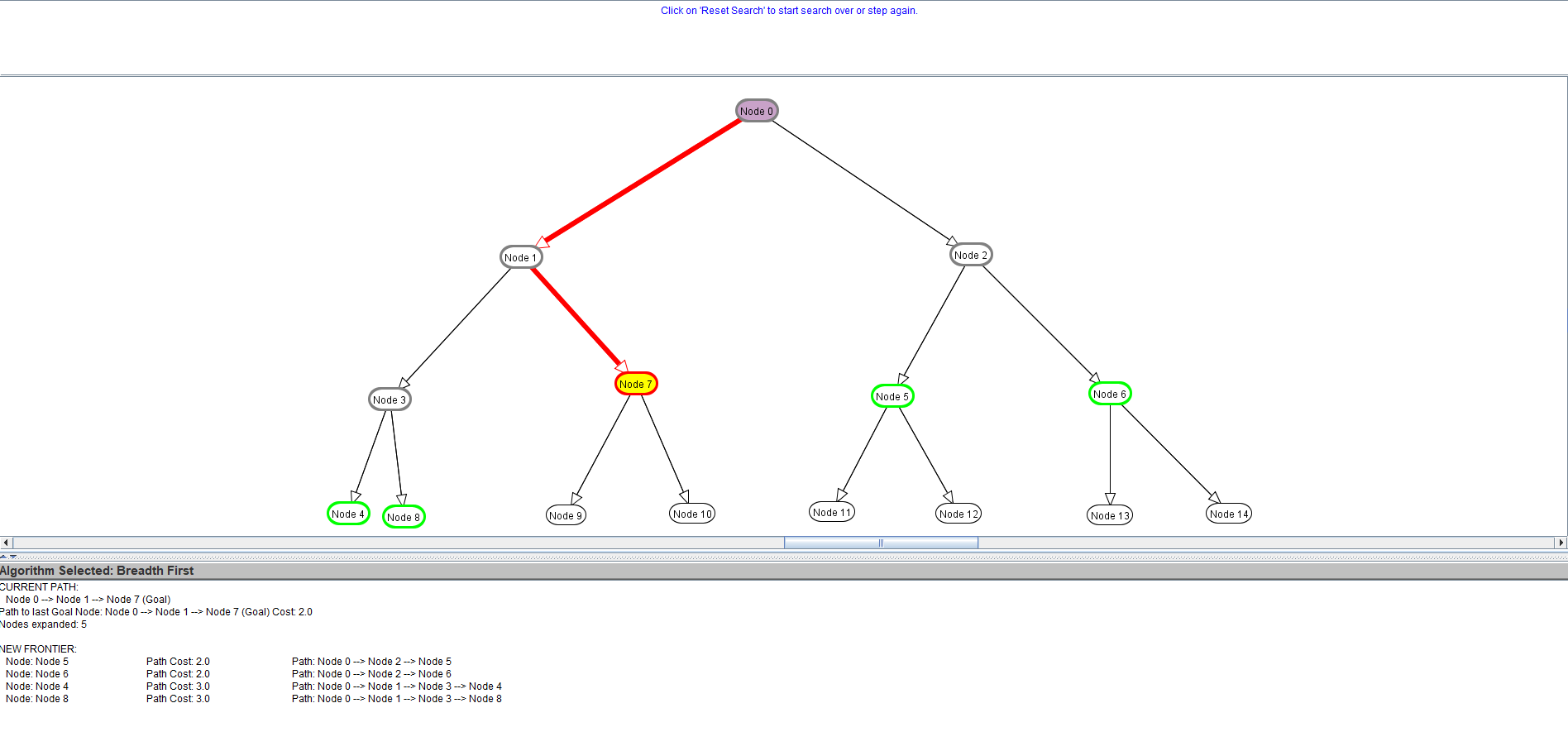
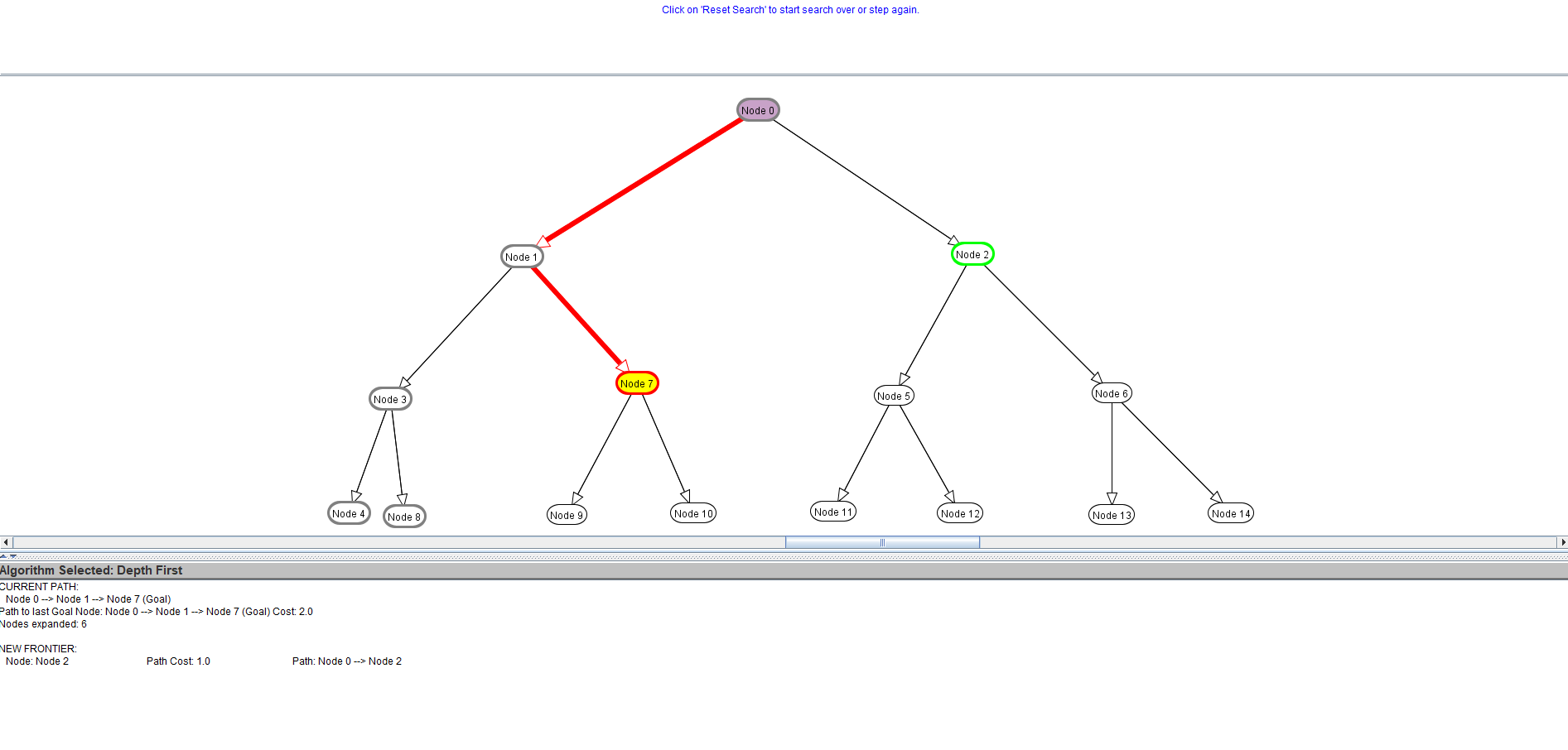
Q1a) Give a graph where Depth-First search (DFS) is much more efficient (expands fewer nodes) than breadth-first search (BFS)  
DFS (Node Expanded = 4): 

BFS (Node Expanded = 8):

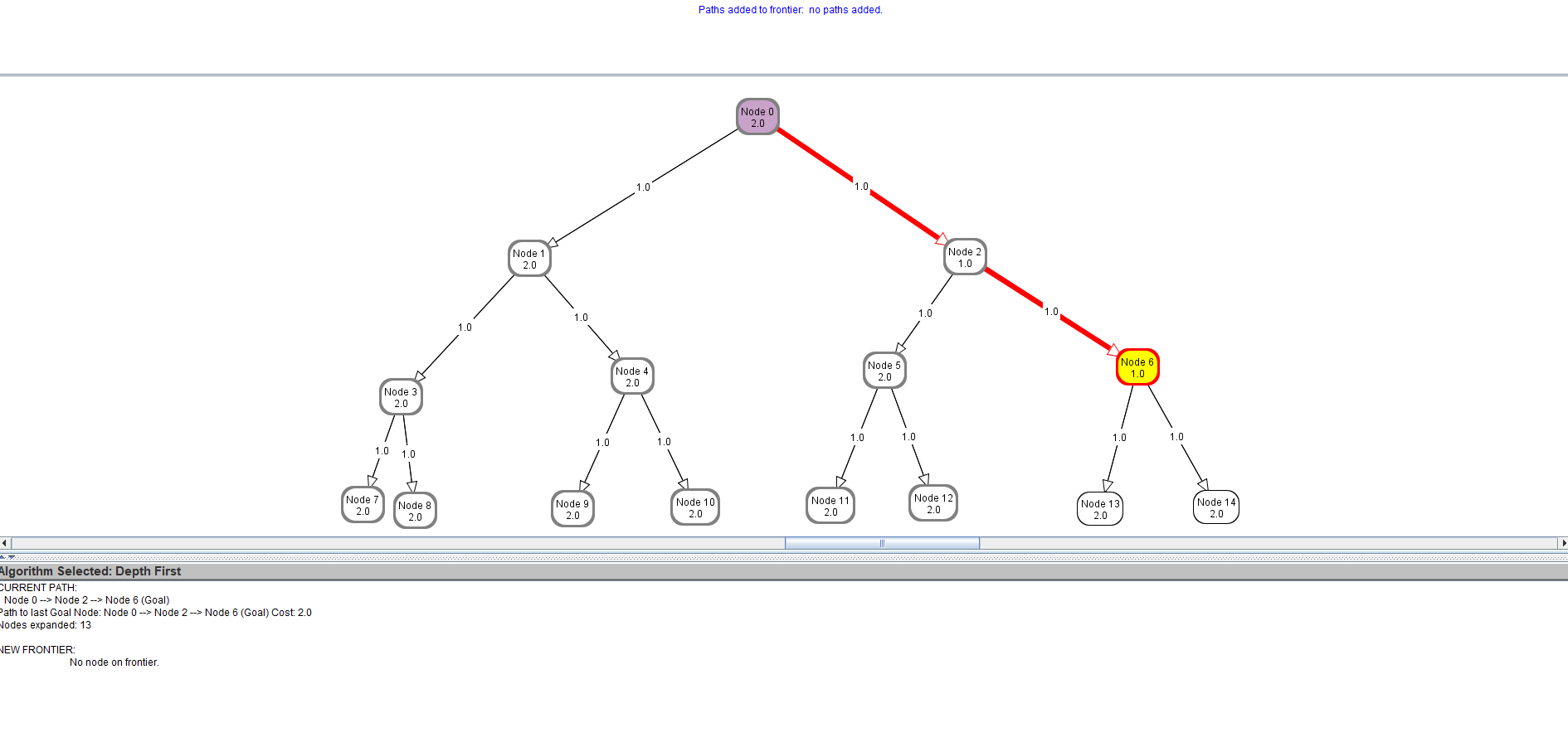


Q1b) Give a graph where BFS is much better than DFS.  
BFS (Node Expanded = 5):  


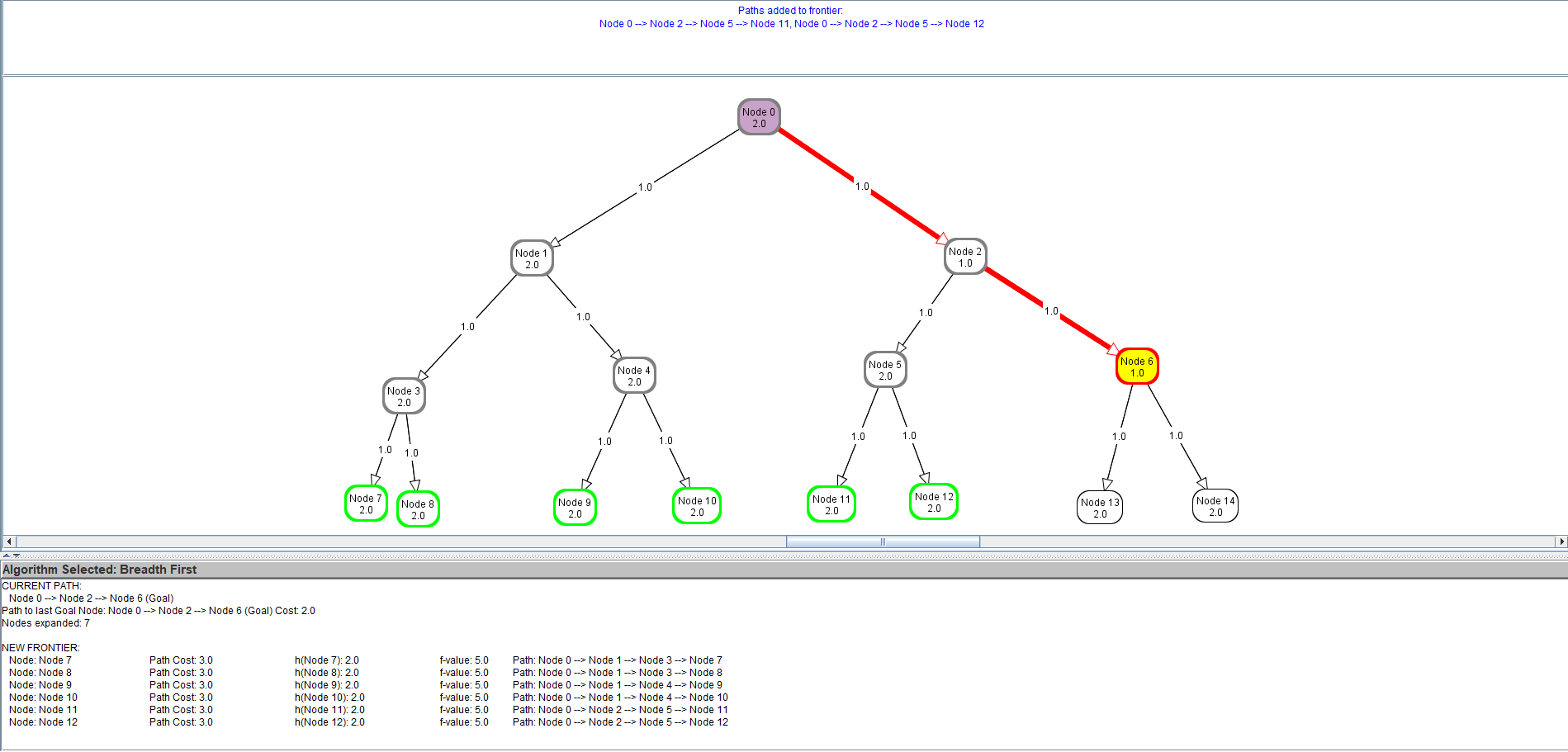
DFS (Node Expanded = 6)

Q1c) Give a graph where A\* search is more efficient than either DFS or BFS.

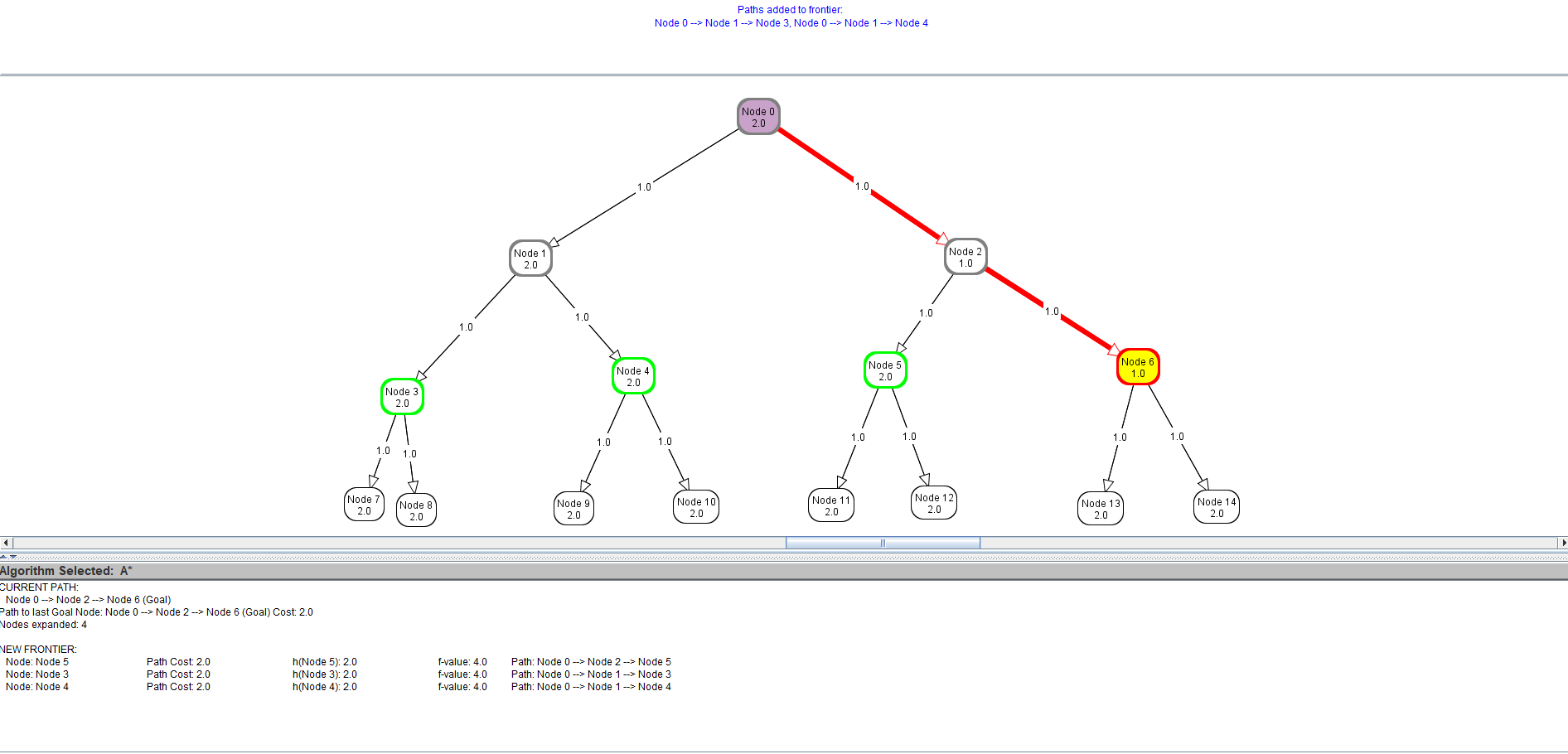
DFS (Node Expanded = 13)



BFS (Node Expanded = 7)

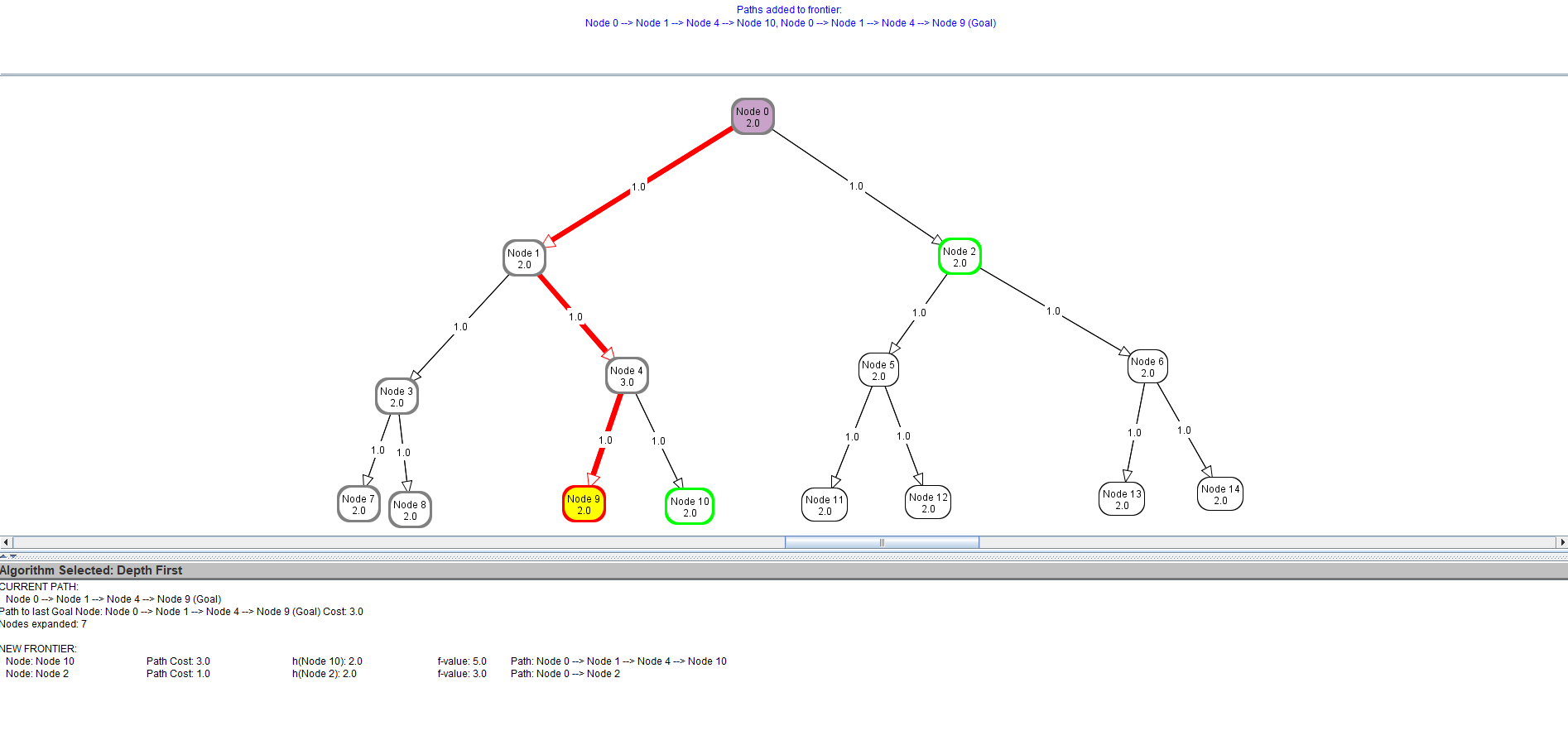


A\* Search (Node expanded = 4)

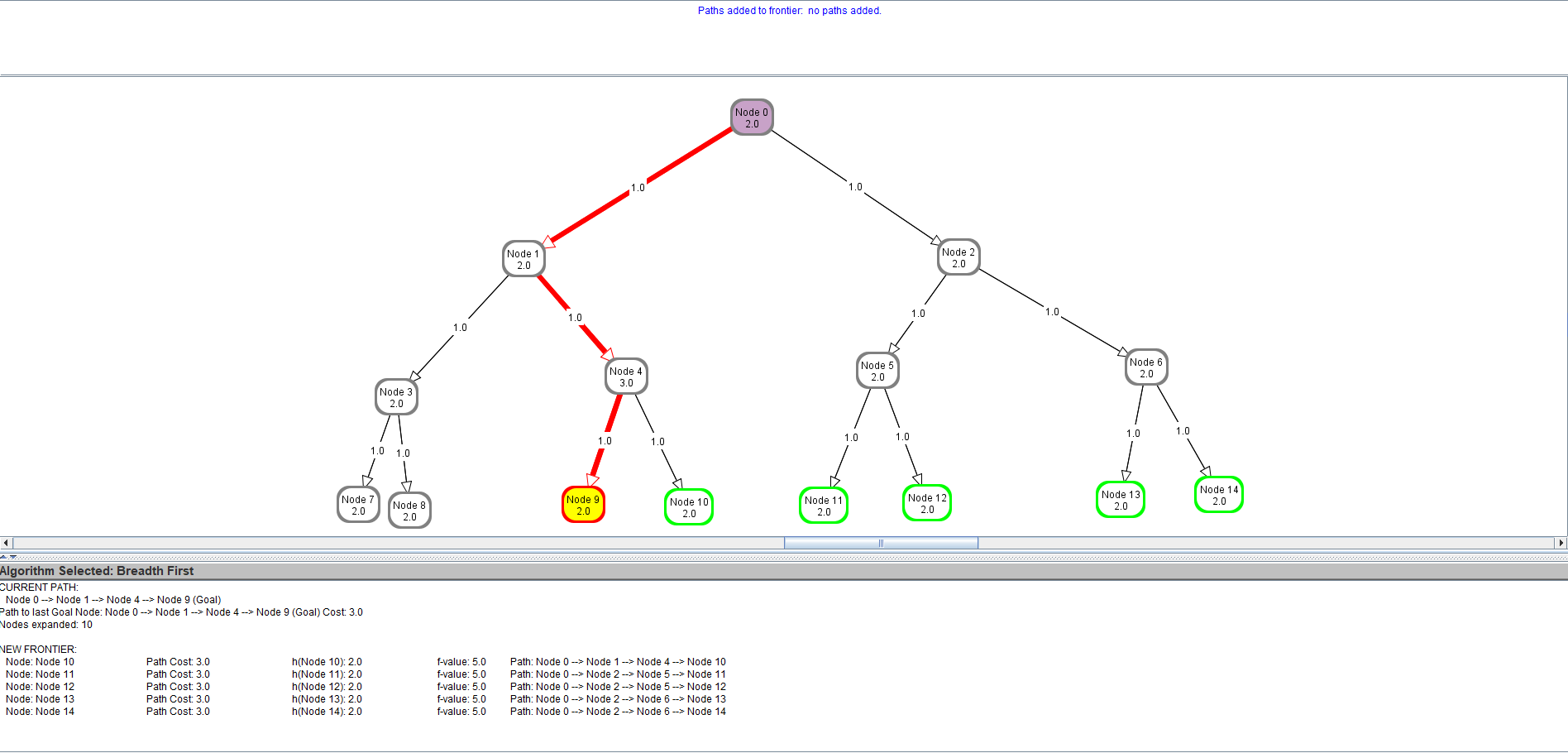


Q1d) Give a graph where DFS and BFS are both more efficient than A\* search.

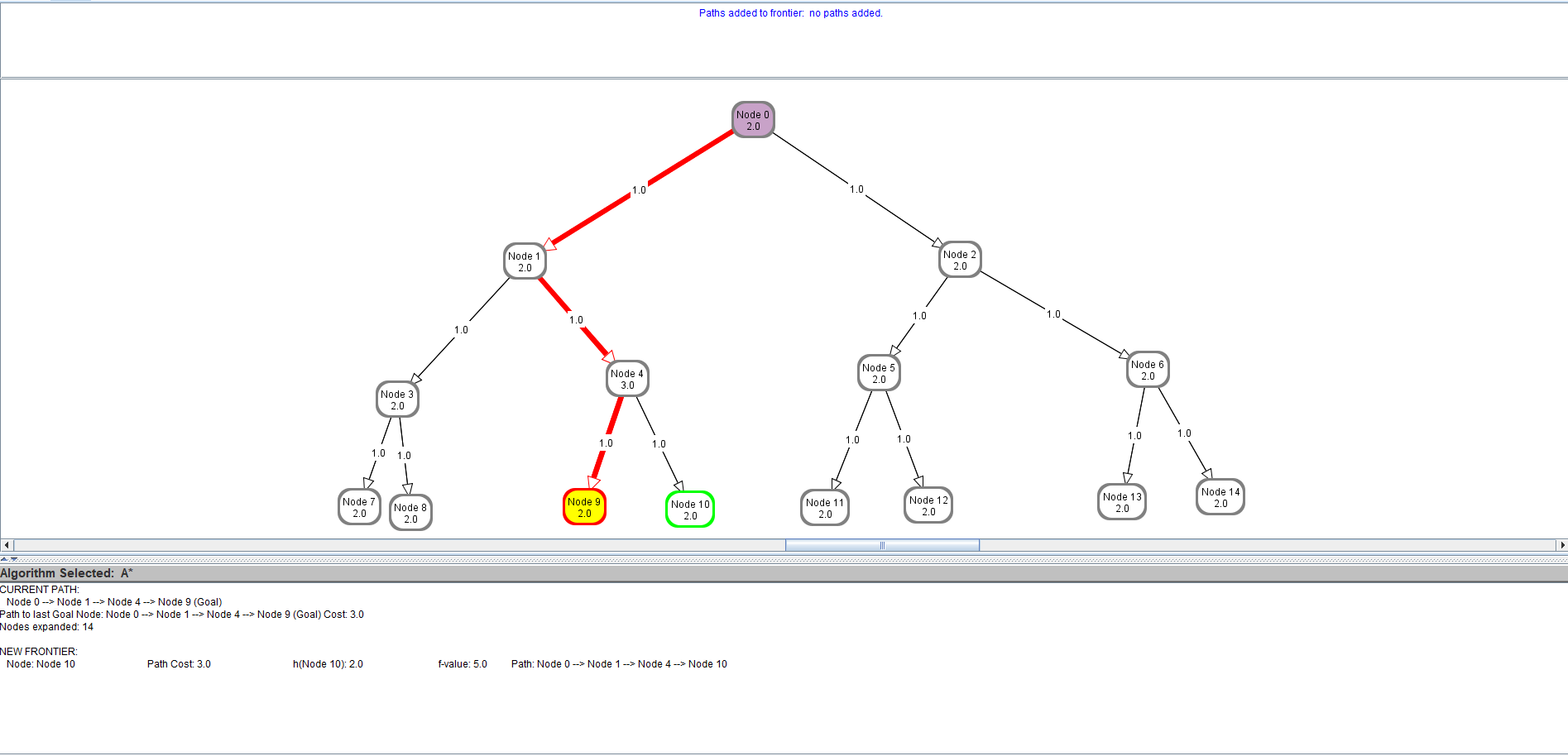
DFS (Node expanded = 7)



BFS (Node expanded = 10)



A\* Search (Node Expanded = 14)



Learning Points for Q1:  
For DFS and BFS search algorithm, only metric is used to find the goal node is just the number of nodes being expanded. When A\* search algorithm is added, Heuristic function h(n) is added into the metric to calculate to total cost function: f(n) = g(n) + h(n).  
  
In this observation, we assume the DFS algorithm will expand from the left part of binary of tree, although, we can configure the algorithm to expand from the right. From question 1a and 1b observation, DFS search is faster when the goal node is at the deepest level (left leaf node) of the binary tree. However, this is not an optimal solution if the goal node is on the right side of the binary tree. On the other hand, BFS is will expand lesser node when the goal is not on the deepest level of the binary.  
  
When A\* search is involved, we must set the heuristic level higher to be higher than other nodes for A\* search to less efficient than DFS and BFS. Another example would set heuristic level to lower than rest of the other nodes for A\* search to be more efficient than BFS and DFS.

Question 2:

What is the effect of reducing h(n) when h(n) is already an underestimate?  
When h(n) has the lowest value, the algorithm will take the same path first  
  
How does A\* perform when h(n) is the exact distance from n to a goal?  
It depends on the edge cost [g(n)] by calculating f(n) = g(n) + h(n), taking the path that has the lowest f(n).  
  
What happens if h(n) is not an underestimate? You can give an example to justify your answer.  
When h(n) is overestimated, the calculated f(n) of A\* will be the longest path.