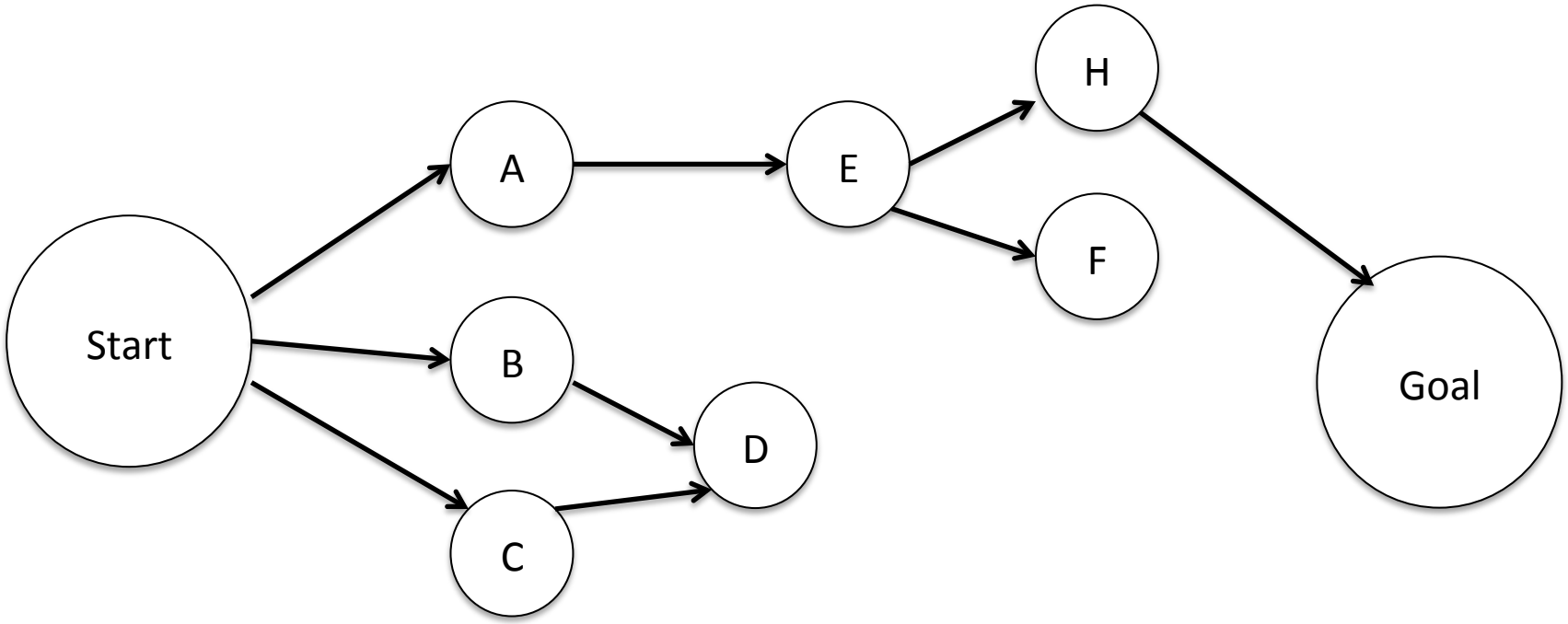
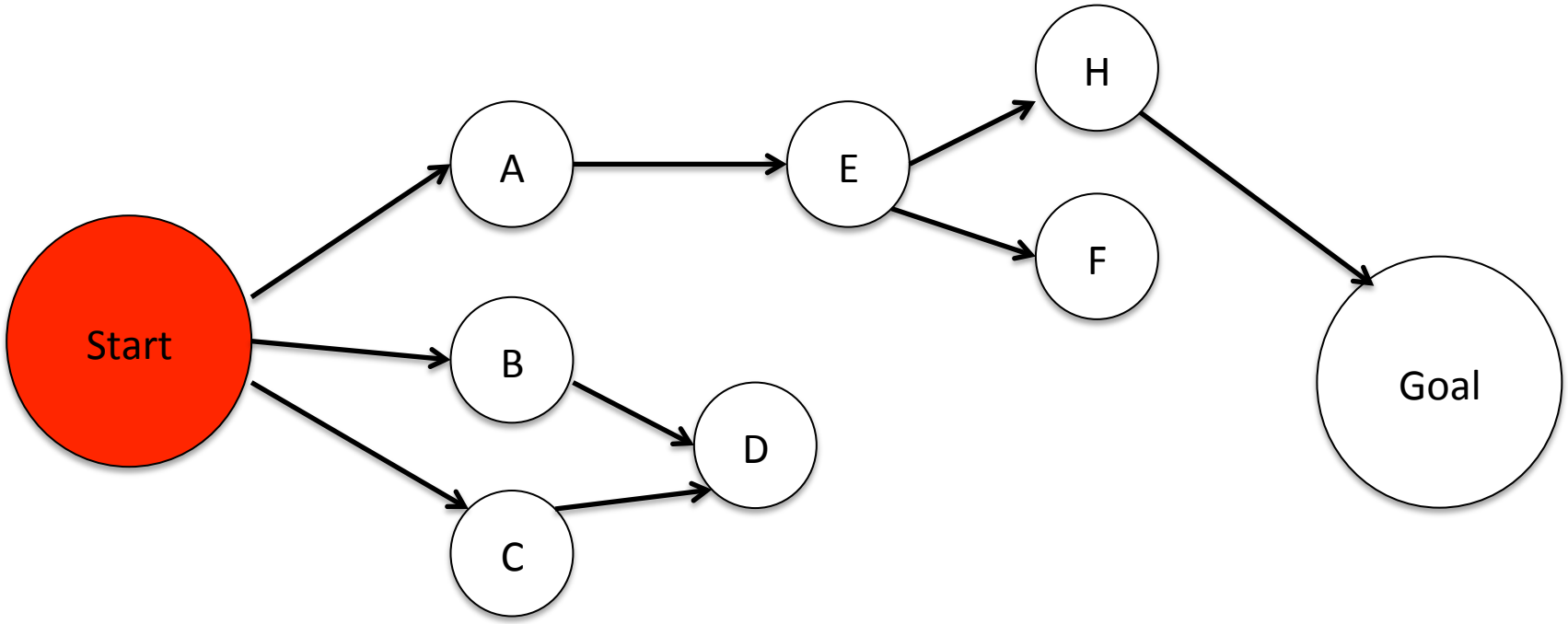
# Breadth-First Search Example

Todo list: Start



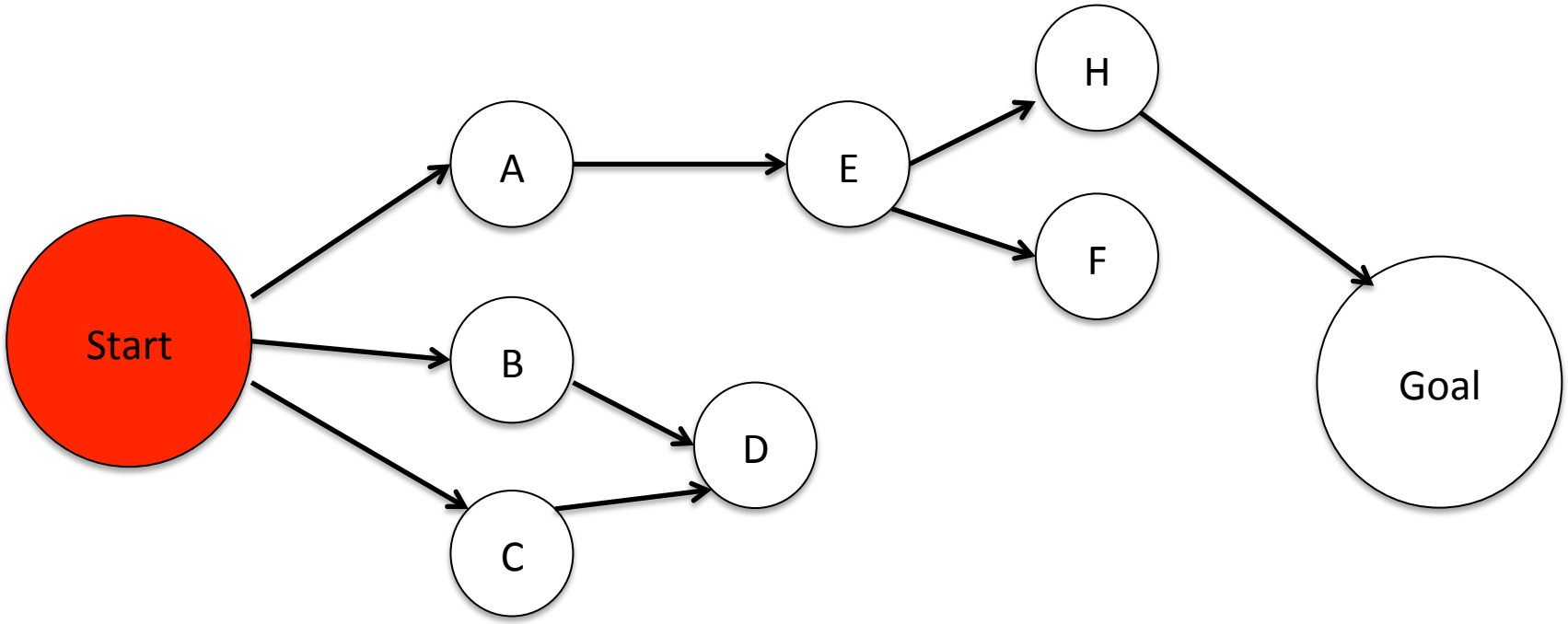
# Breadth-First Search Example

Todo list:



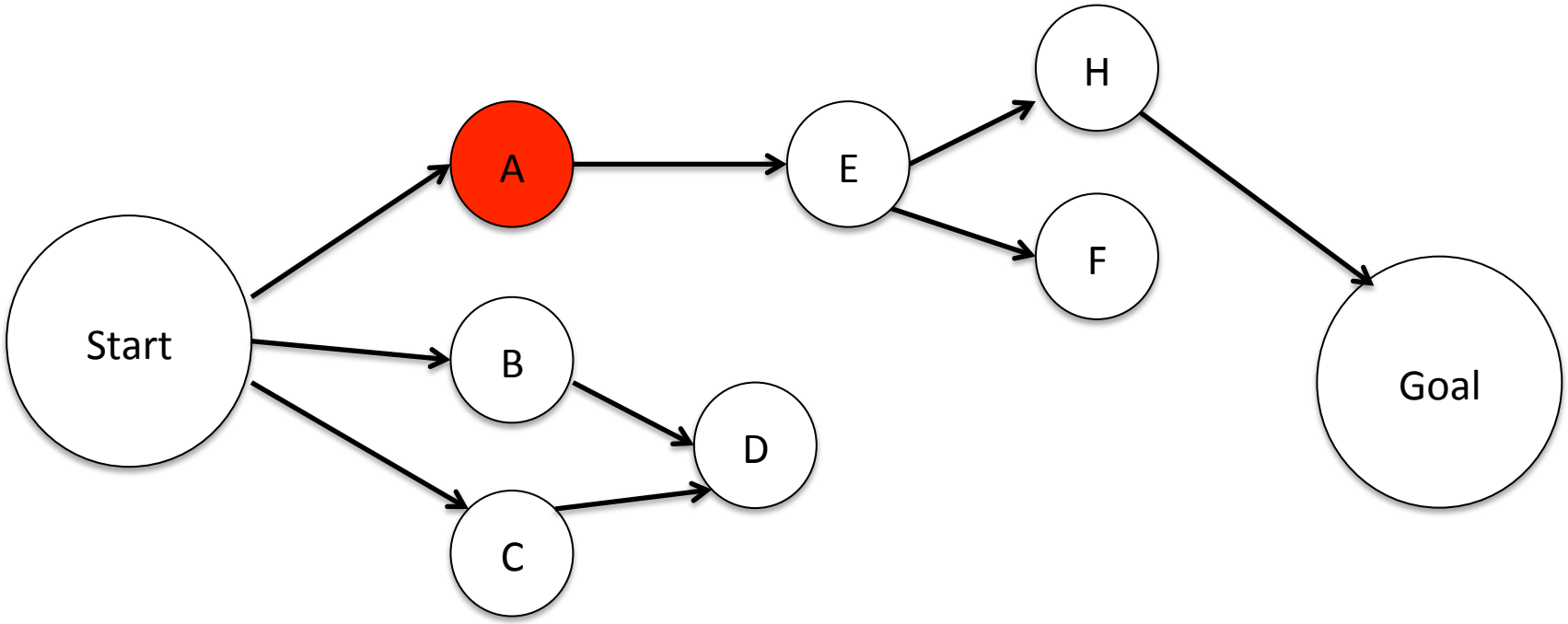
# Breadth-First Search Example

Todo list: A, B, C



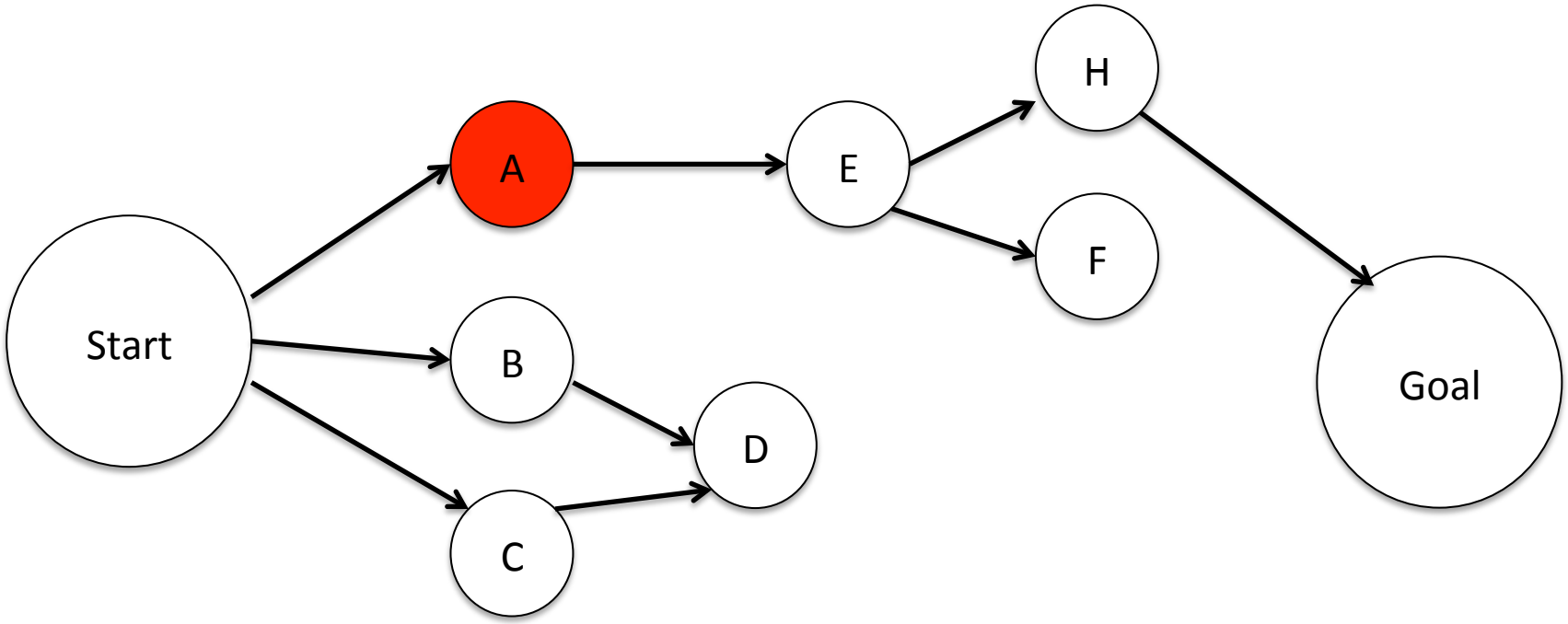
# Breadth-First Search Example

Todo list: B, C



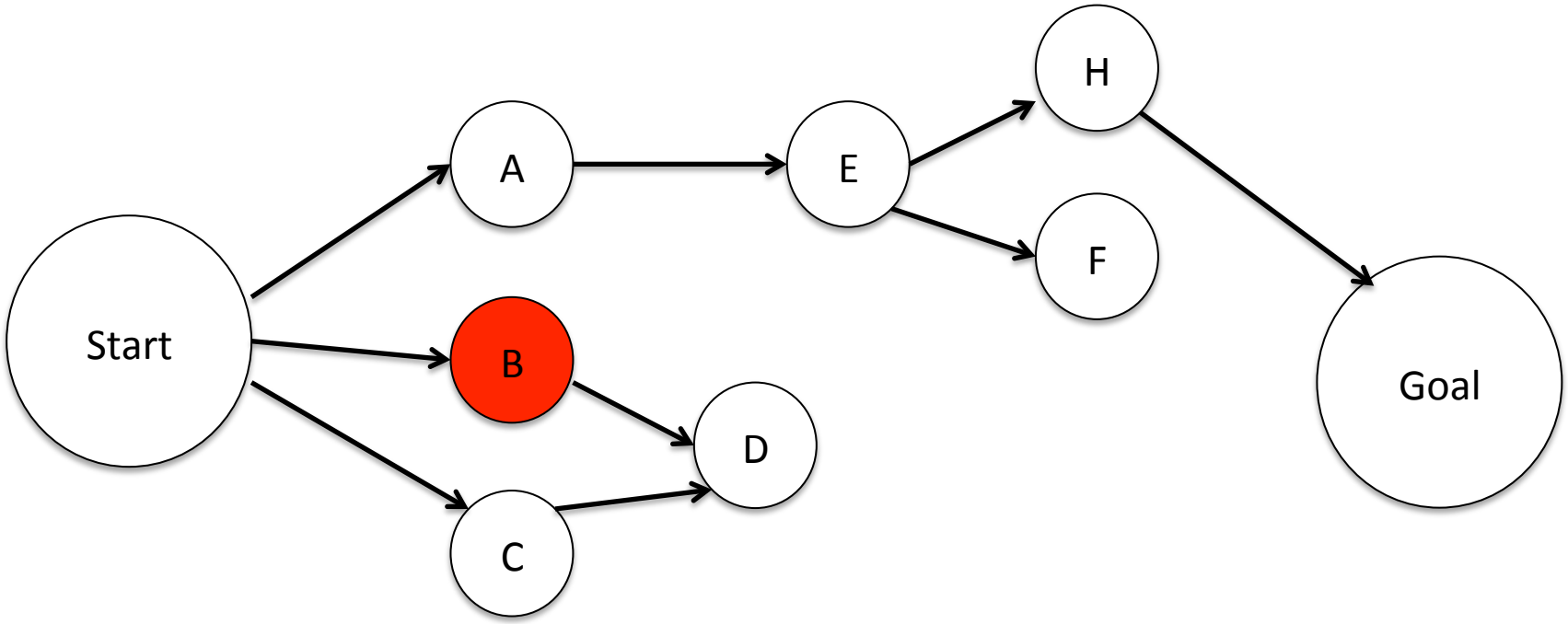
# Breadth-First Search Example

Todo list: B, C, E



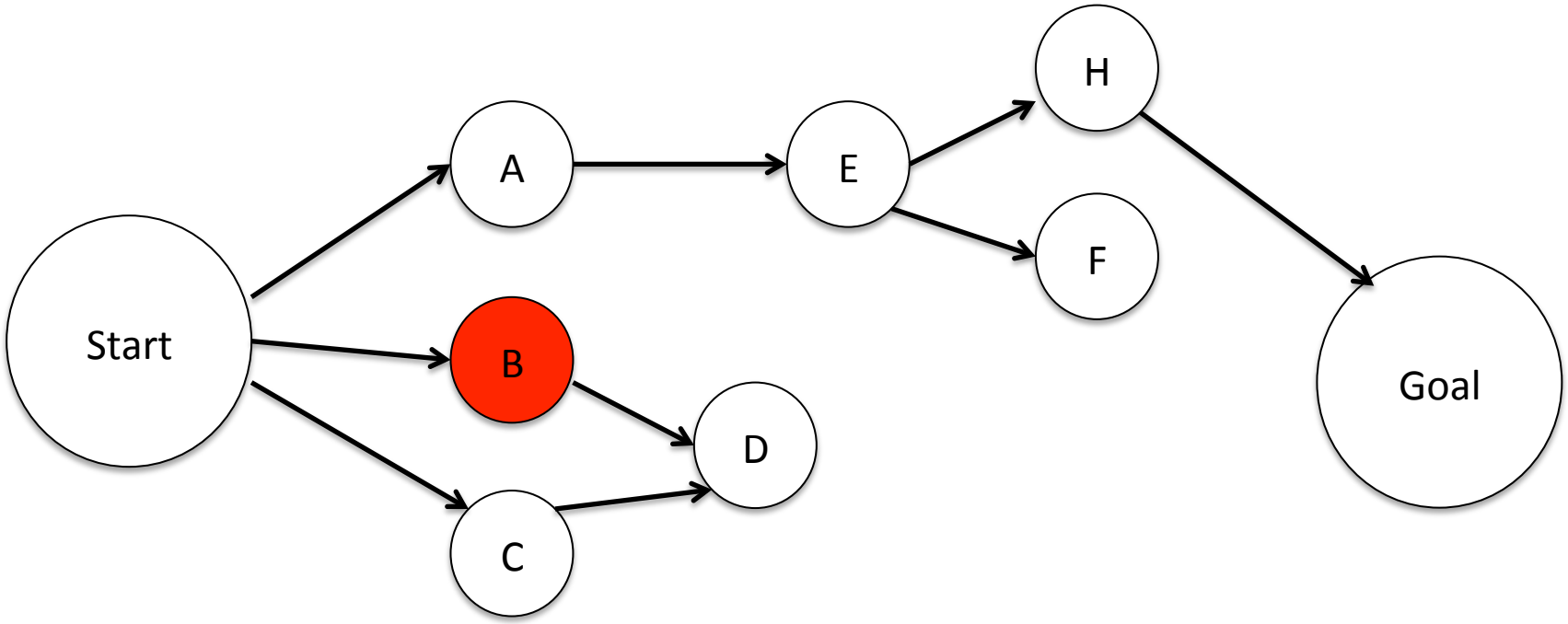
# Breadth-First Search Example

Todo list: C, E



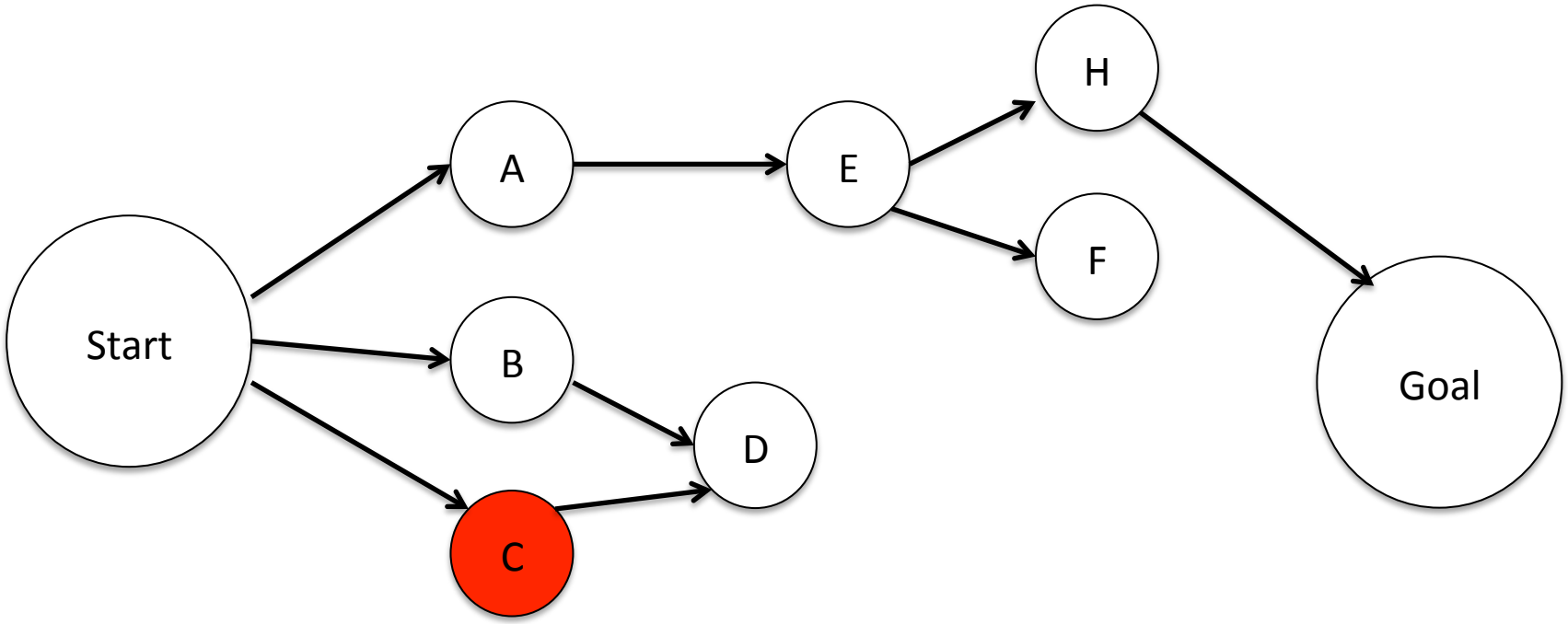
# Breadth-First Search Example

Todo list: C, E, D



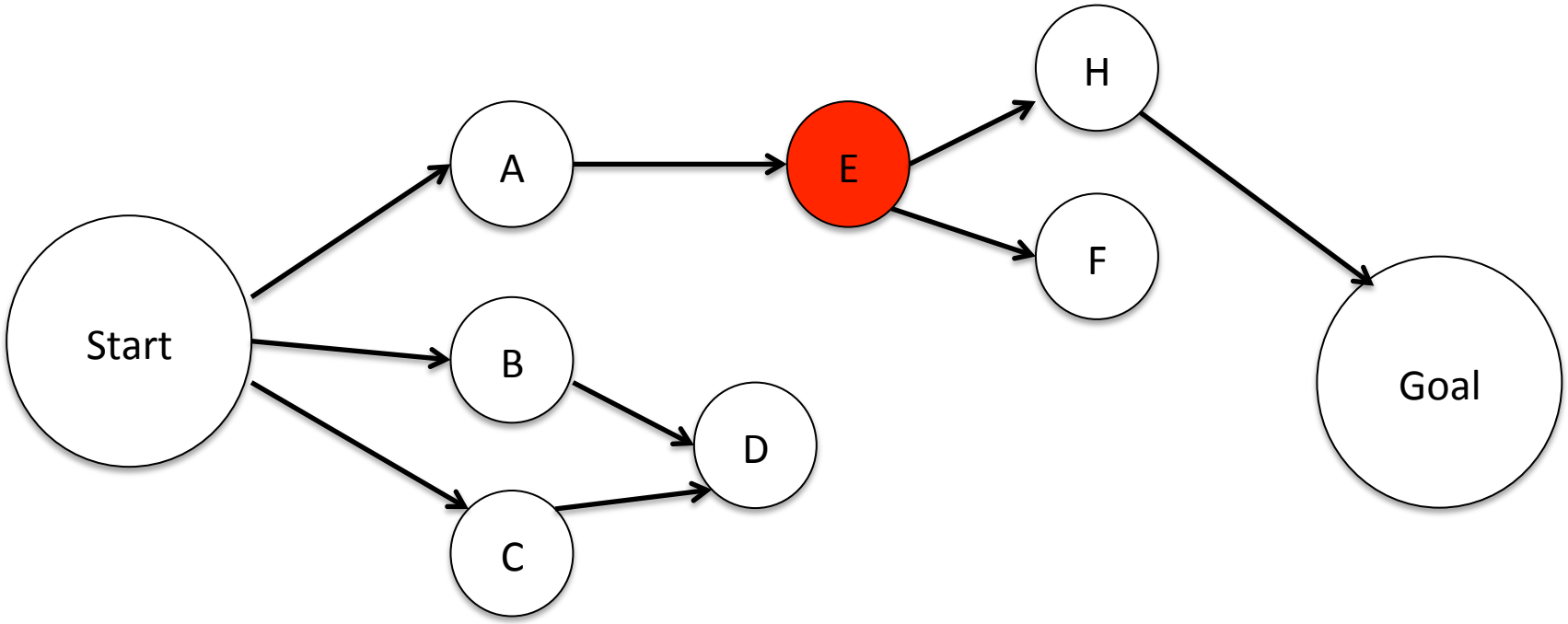
# Breadth-First Search Example

Todo list: E, D



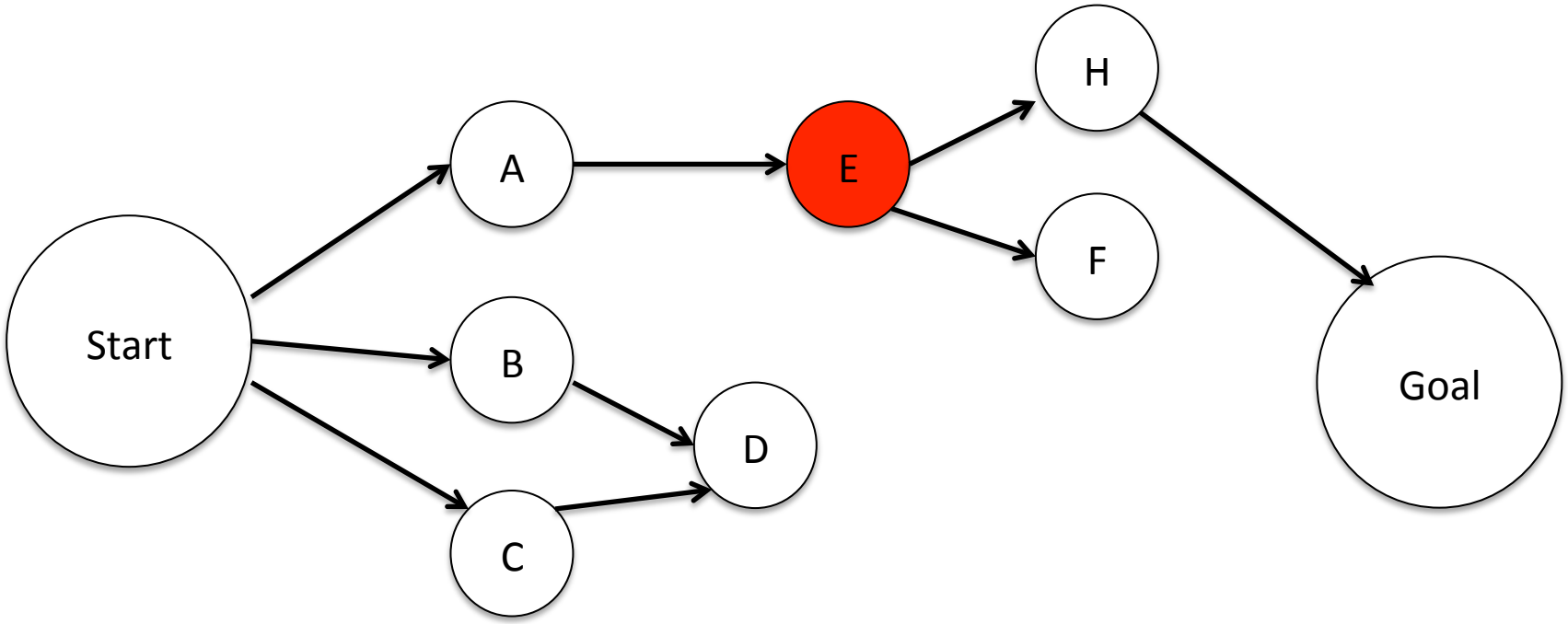
# Breadth-First Search Example

Todo list: D



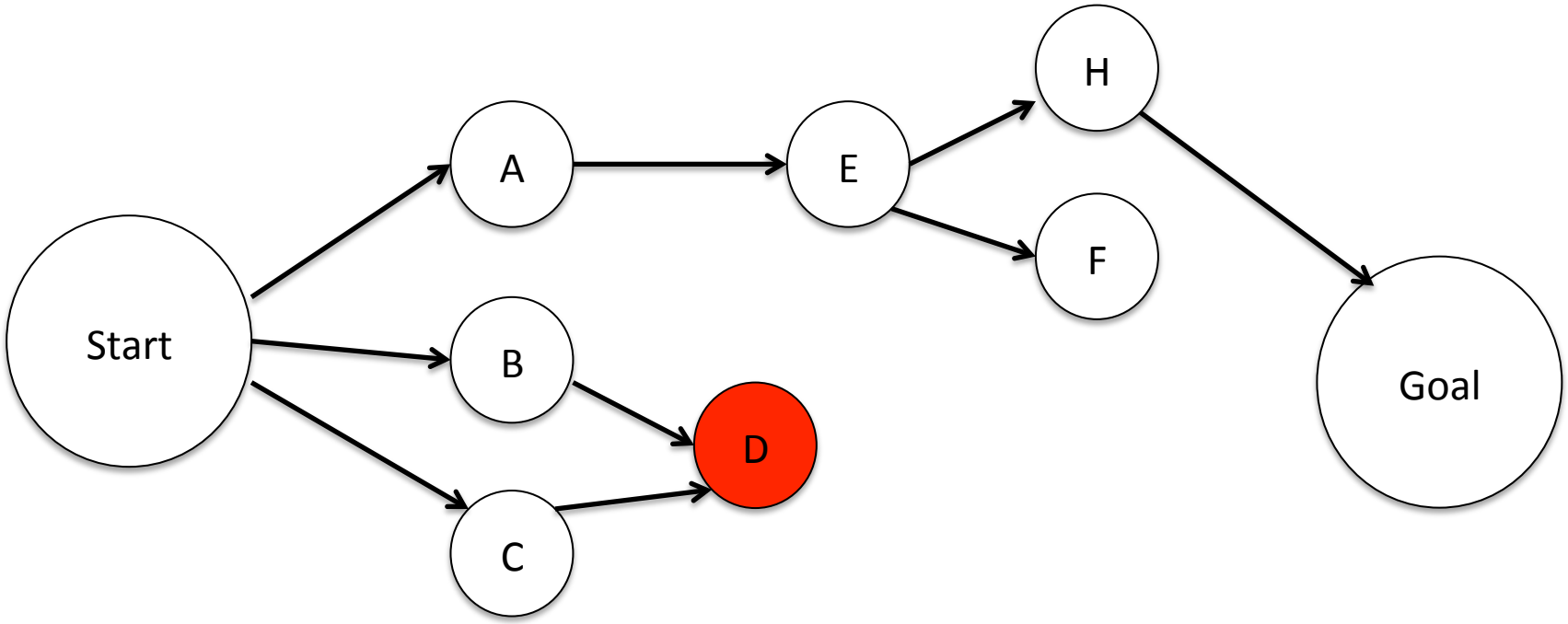
# Breadth-First Search Example

Todo list: D, H, F



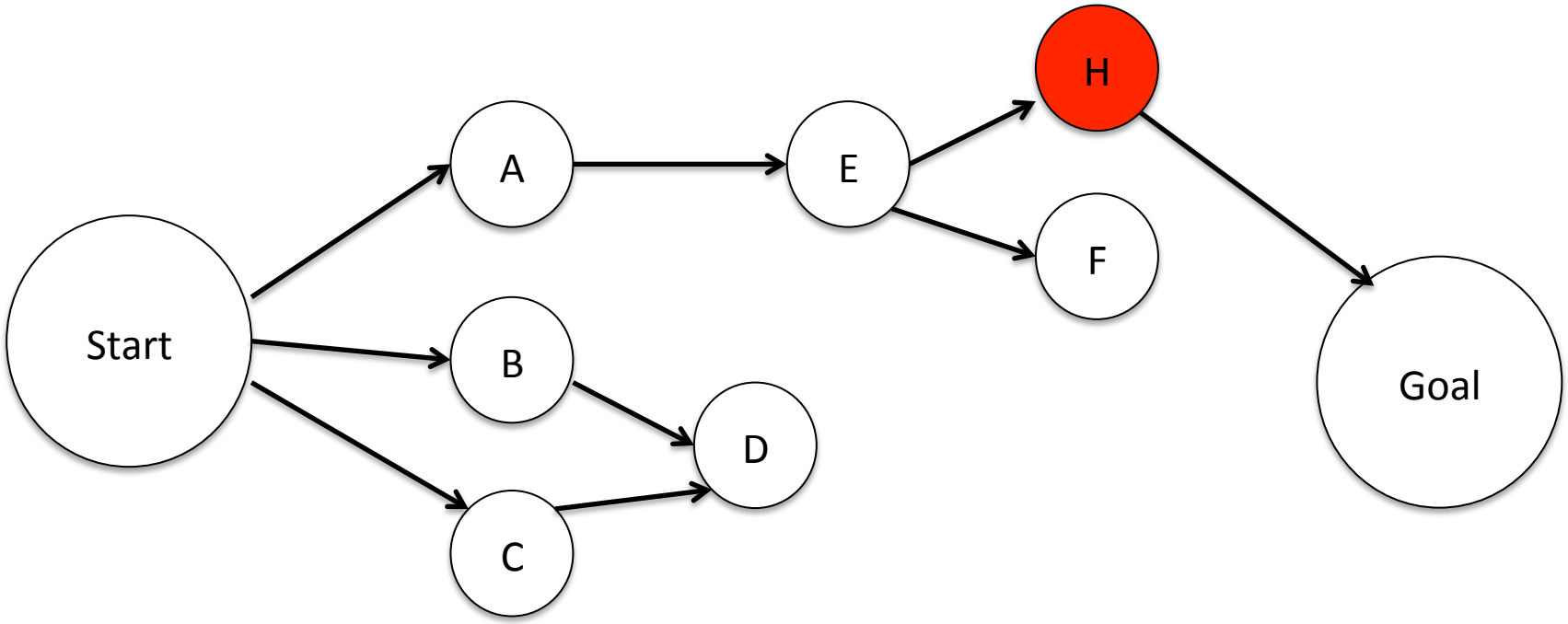
# Breadth-First Search Example

Todo list: H, F



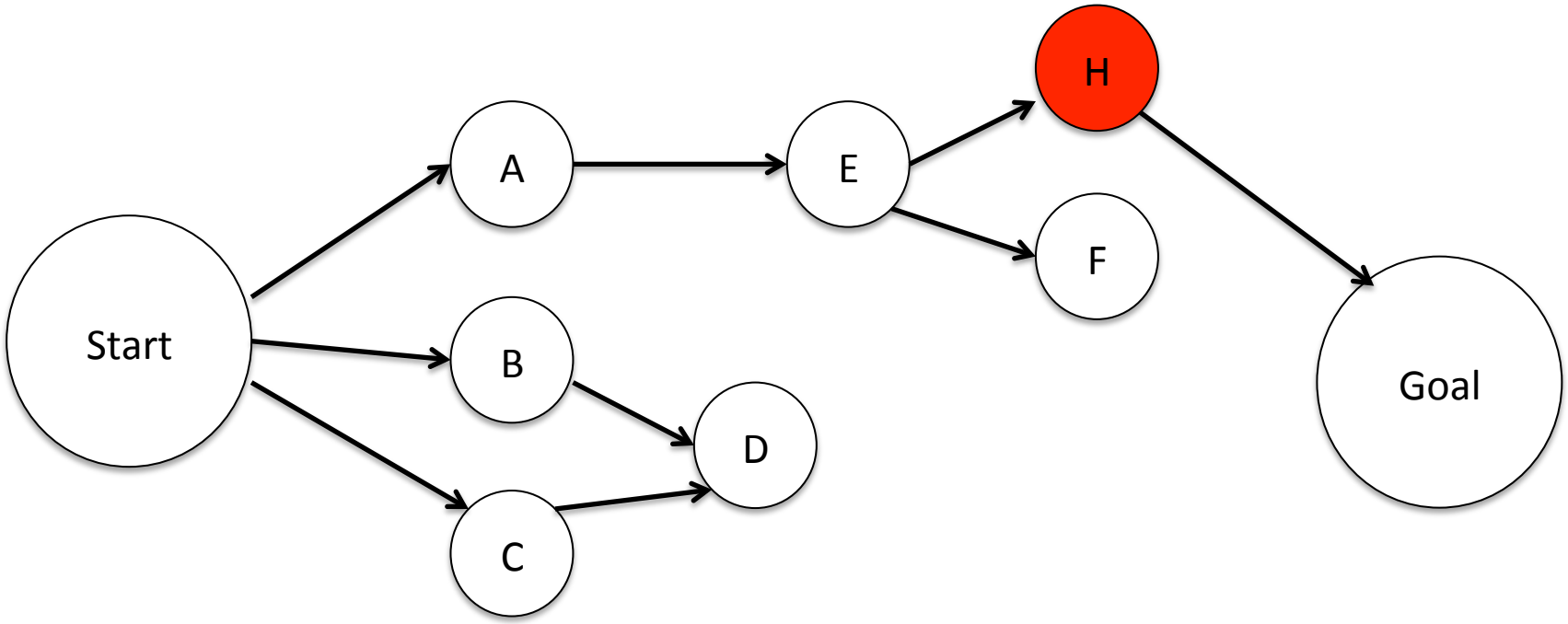
# Breadth-First Search Example

Todo list: F



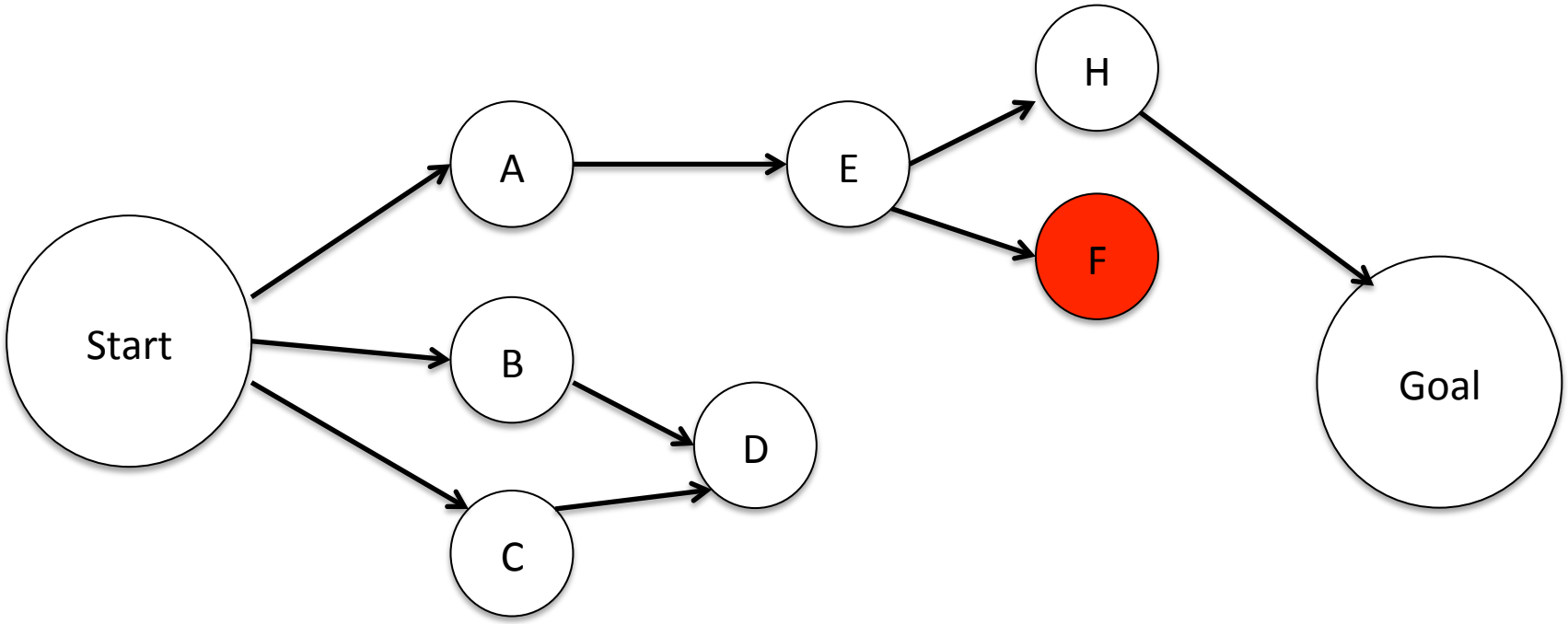
# Breadth-First Search Example

Todo list: F, Goal



# Breadth-First Search Example

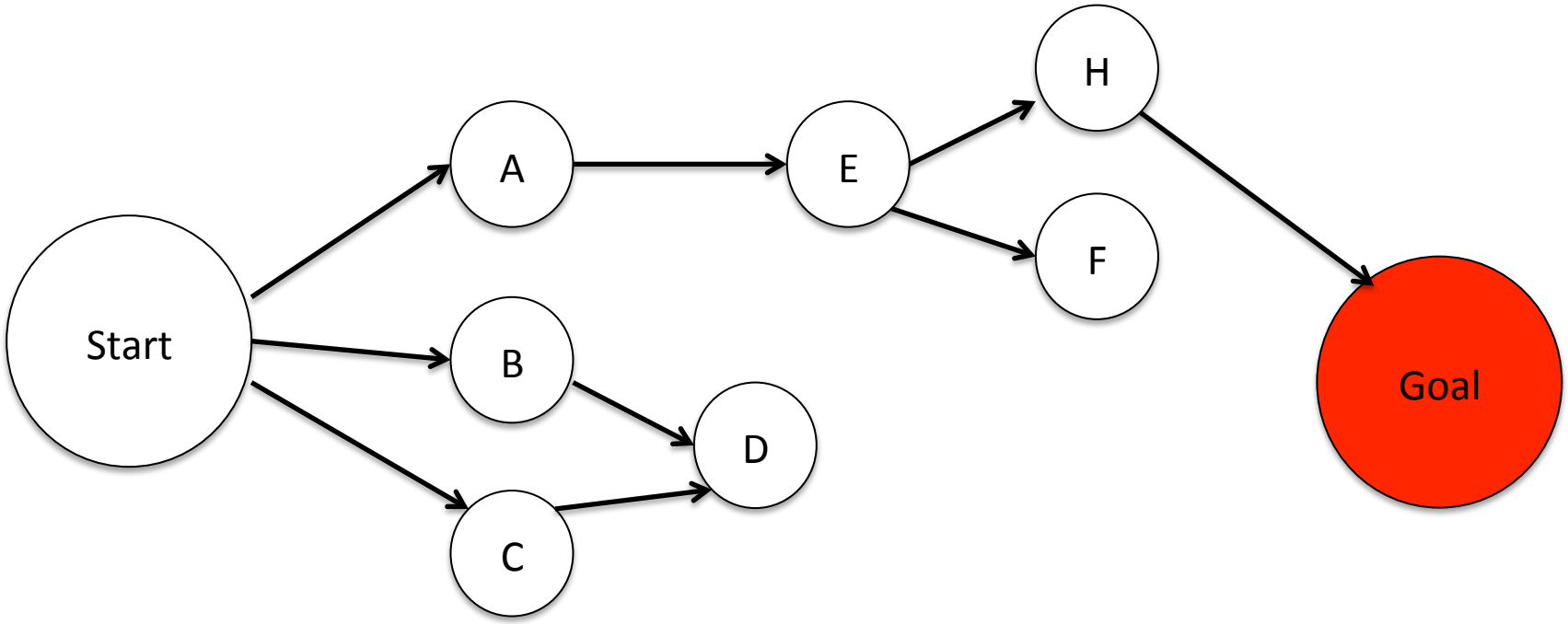
Todo list: Goal



# Breadth-First Search Example

Todo list:

Victory!!



# Breadth First Search

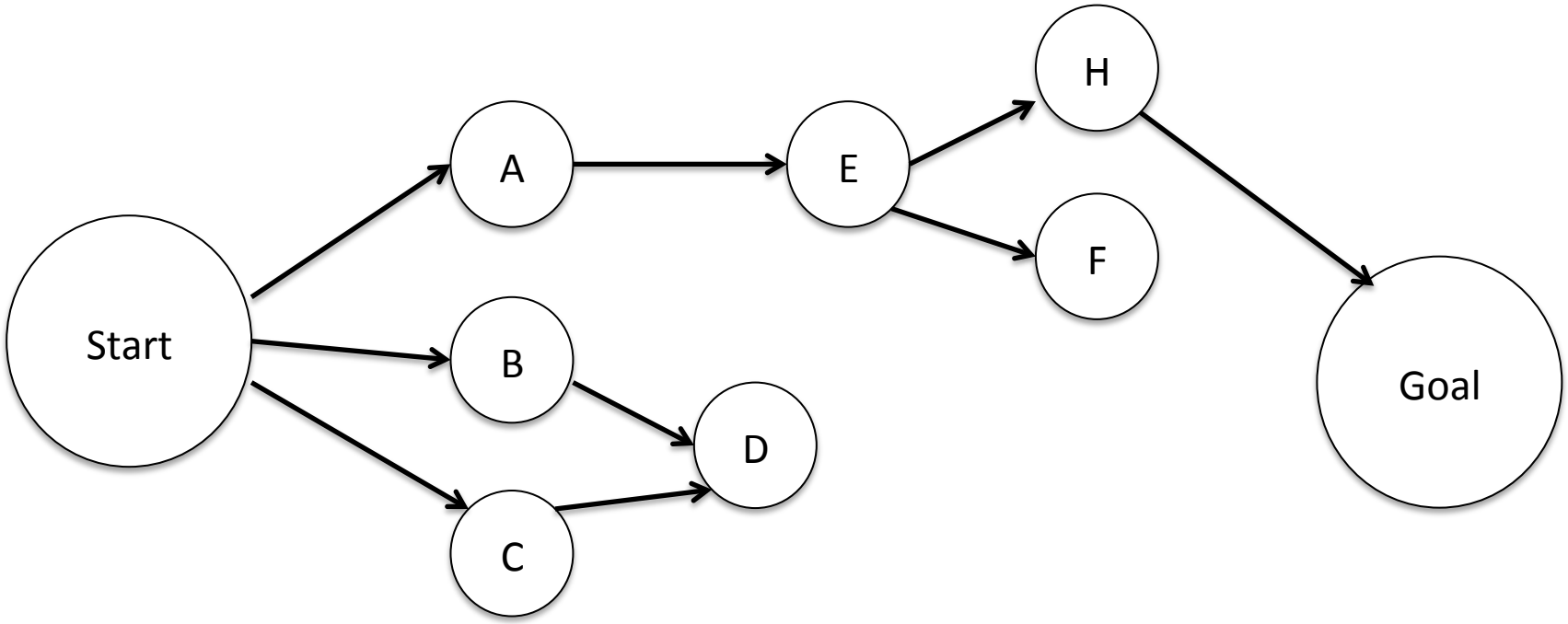
- Guaranteed to find shortest (in number of steps path to the goal)
- To recover the path just requires some additional bookkeeping
- How many operations does it take to complete?

# Depth-First Search

Same as breadth first search, but we structure our todo list differently to prioritize visiting children (i.e. connected by an arrow) of the node we are processing

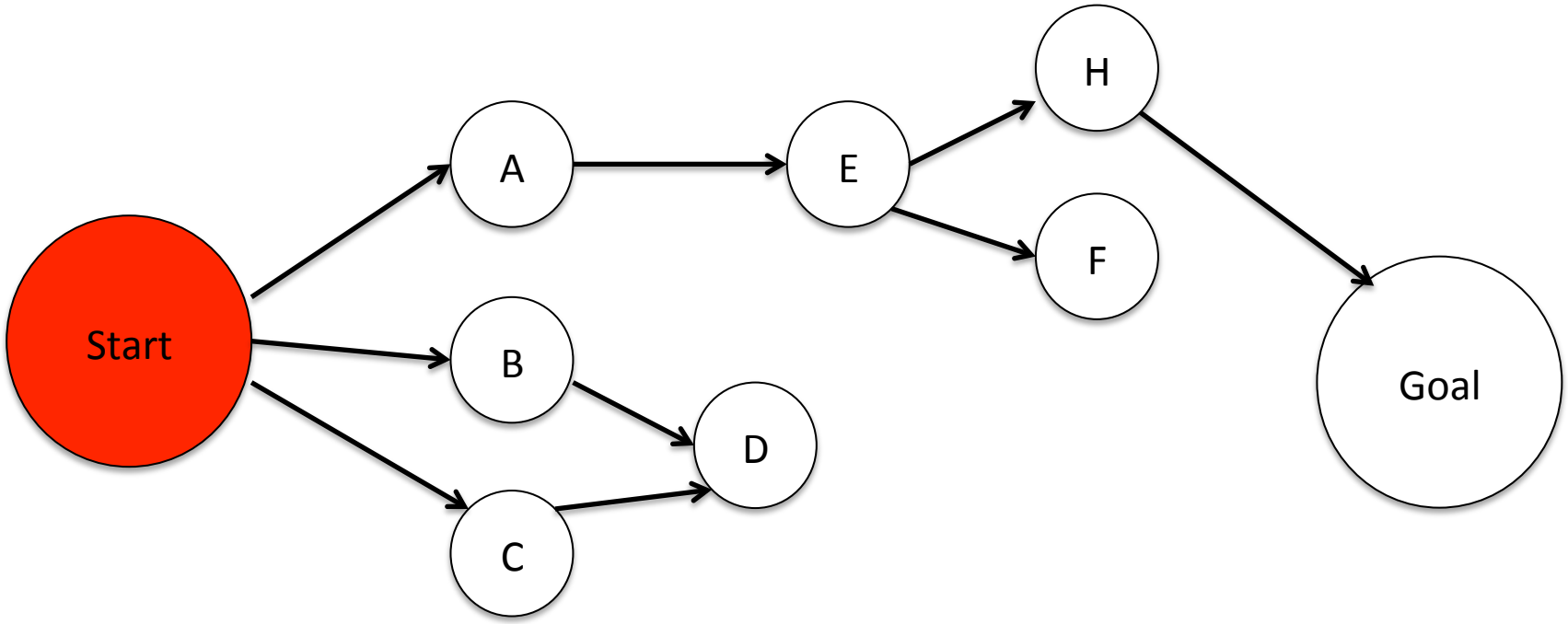
# Depth-First Search Example

Todo list: Start



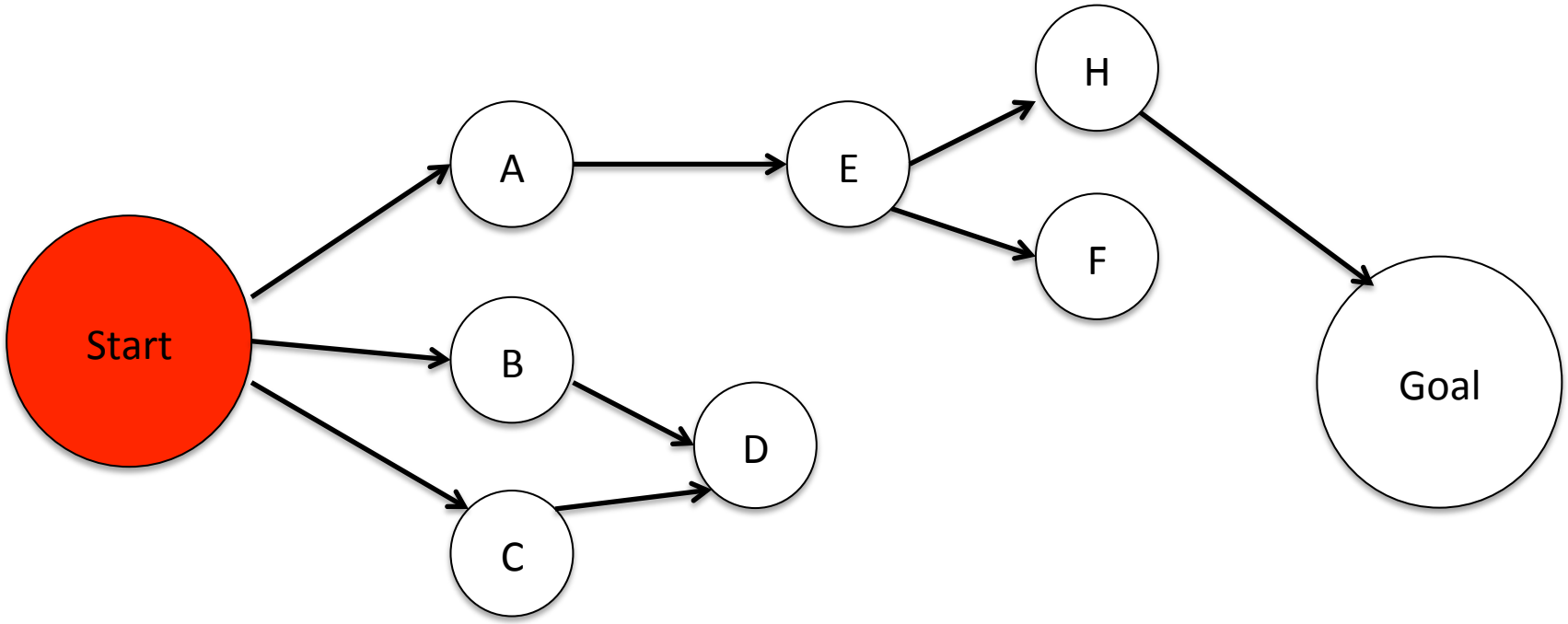
# Depth-First Search Example

Todo list:



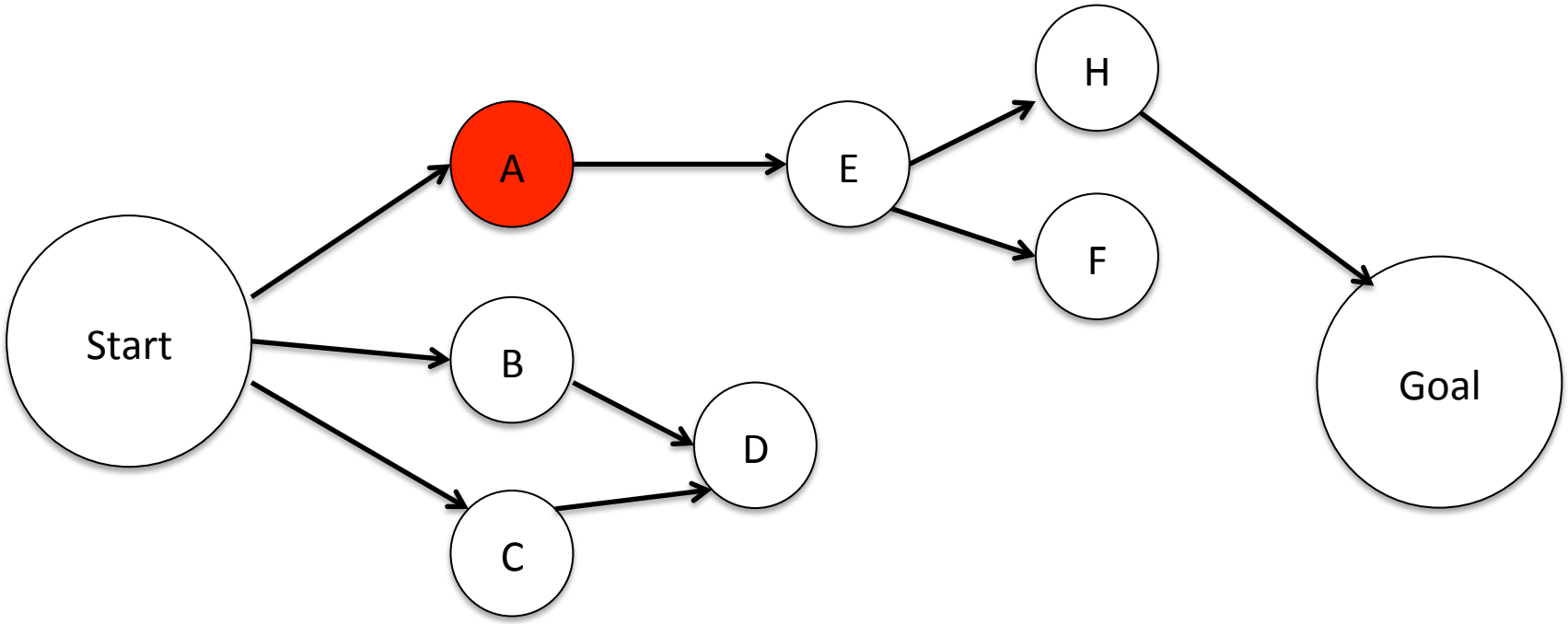
# Depth-First Search Example

Todo list: A, B, C



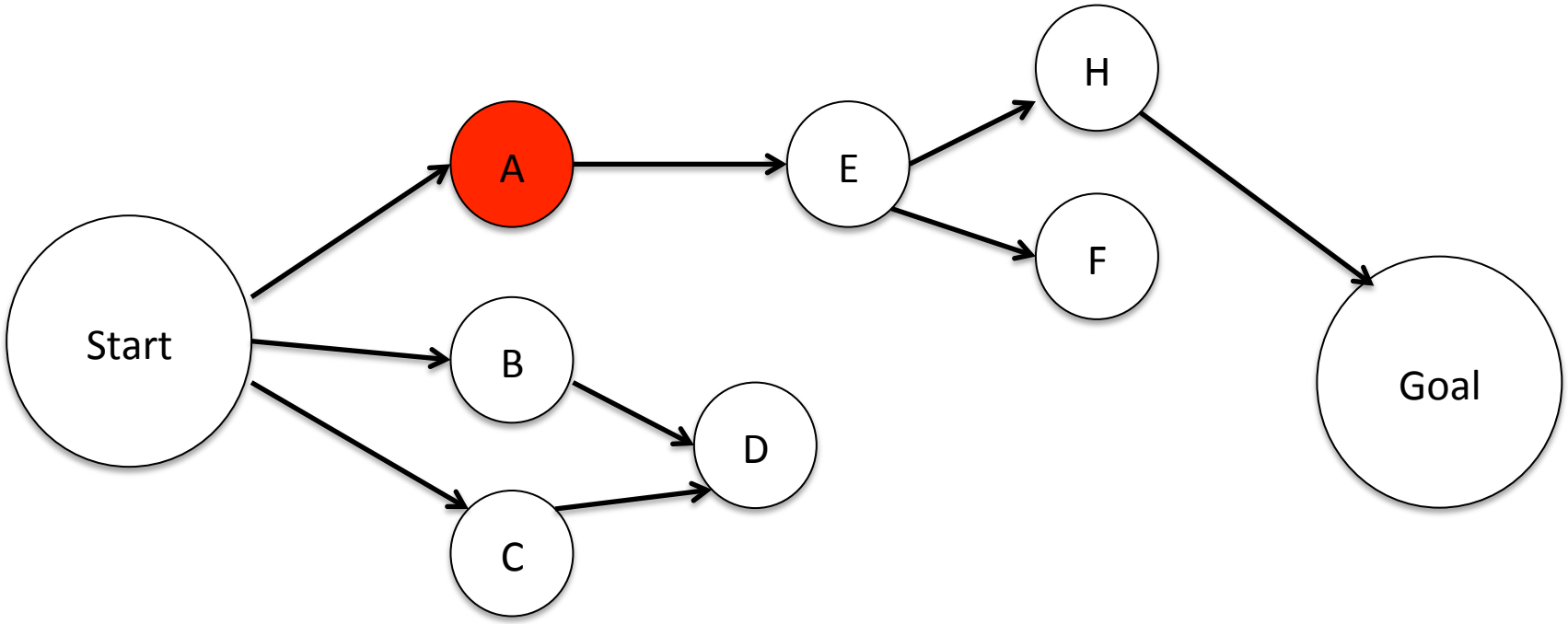
# Depth-First Search Example

Todo list: B, C



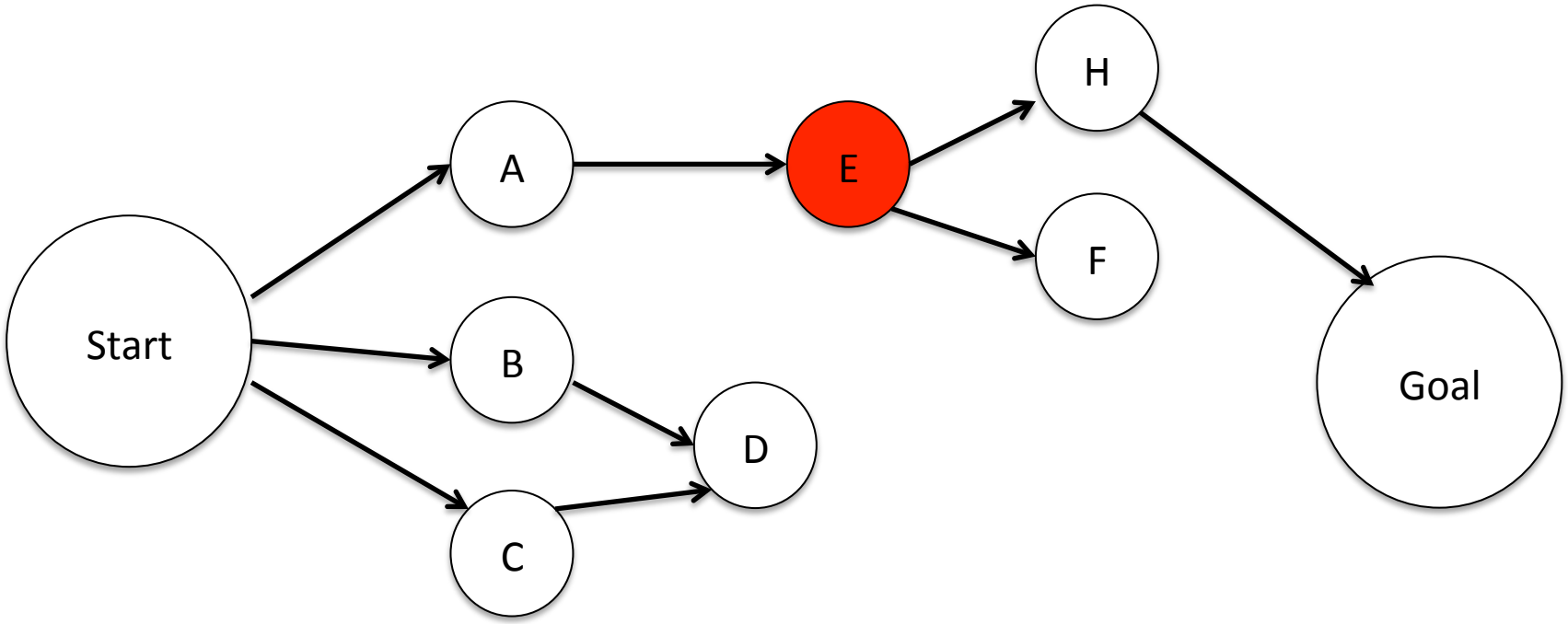
# Depth-First Search Example

Todo list: E, B, C



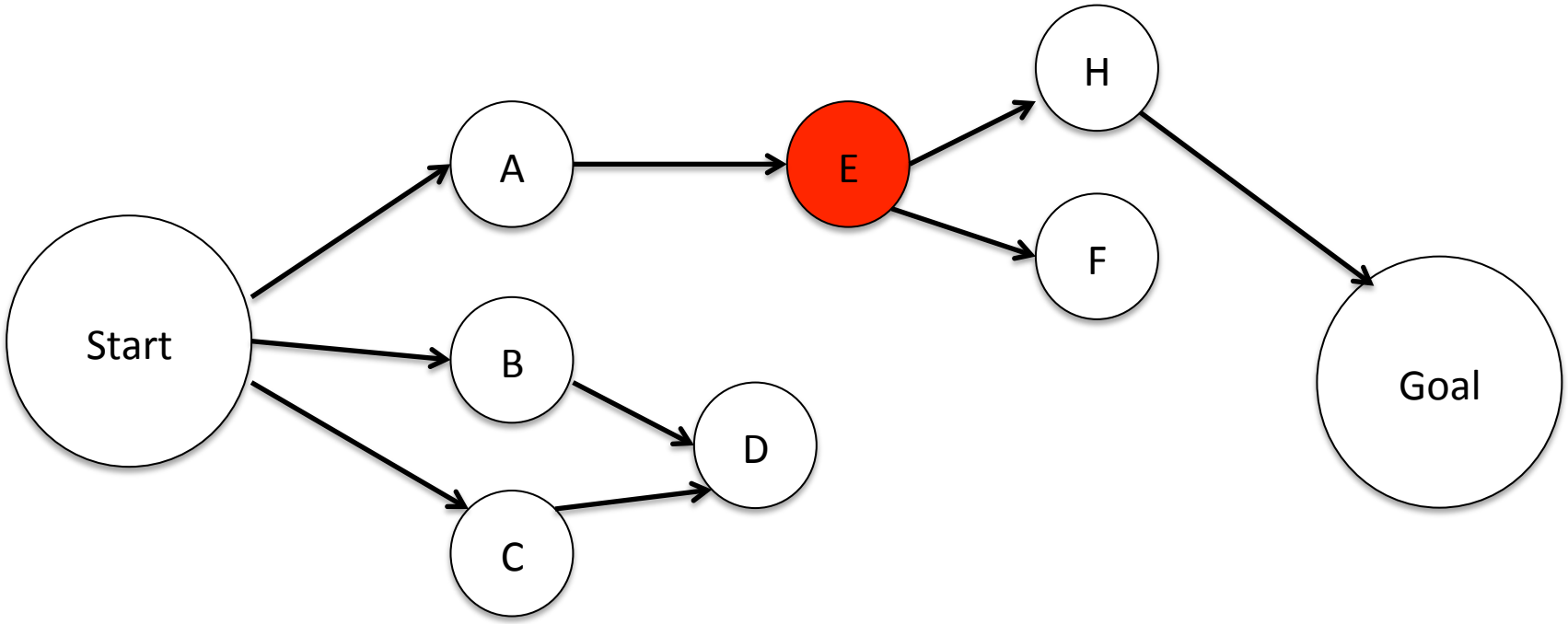
# Depth-First Search Example

Todo list: B, C



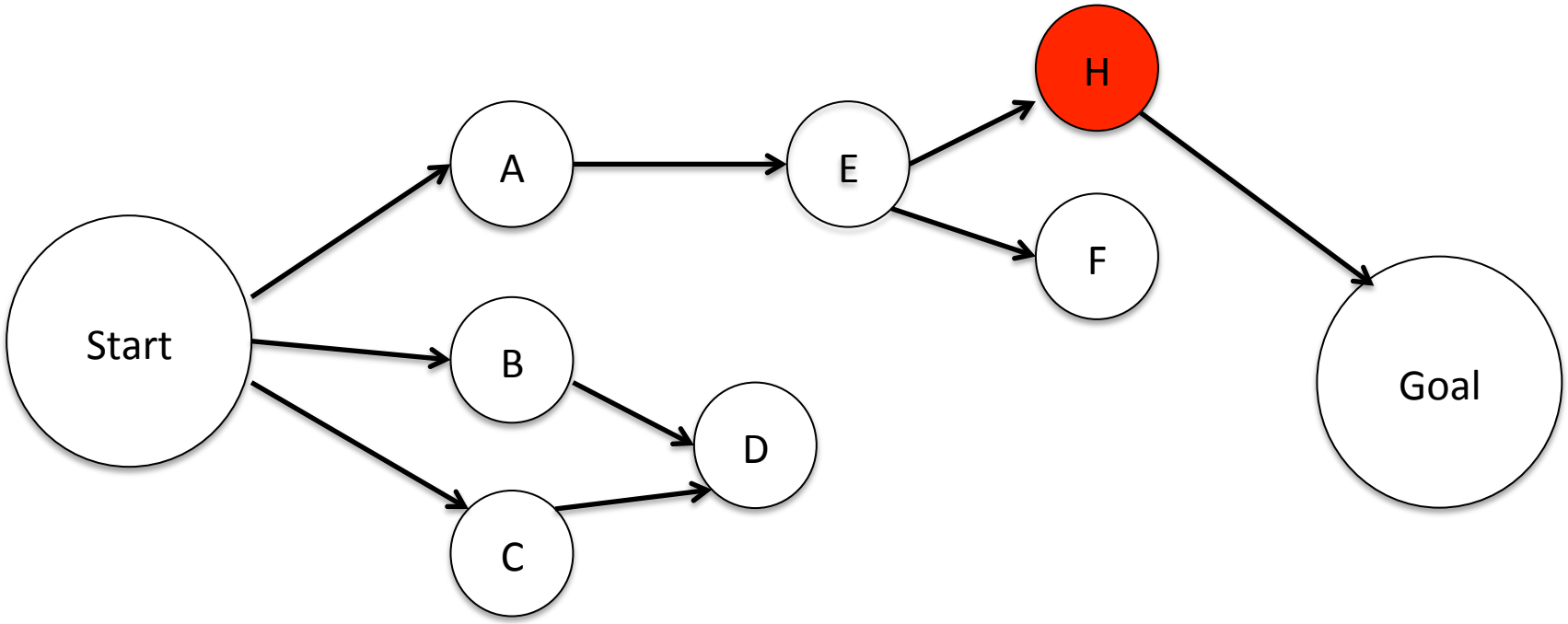
# Depth-First Search Example

Todo list: H, F, B, C



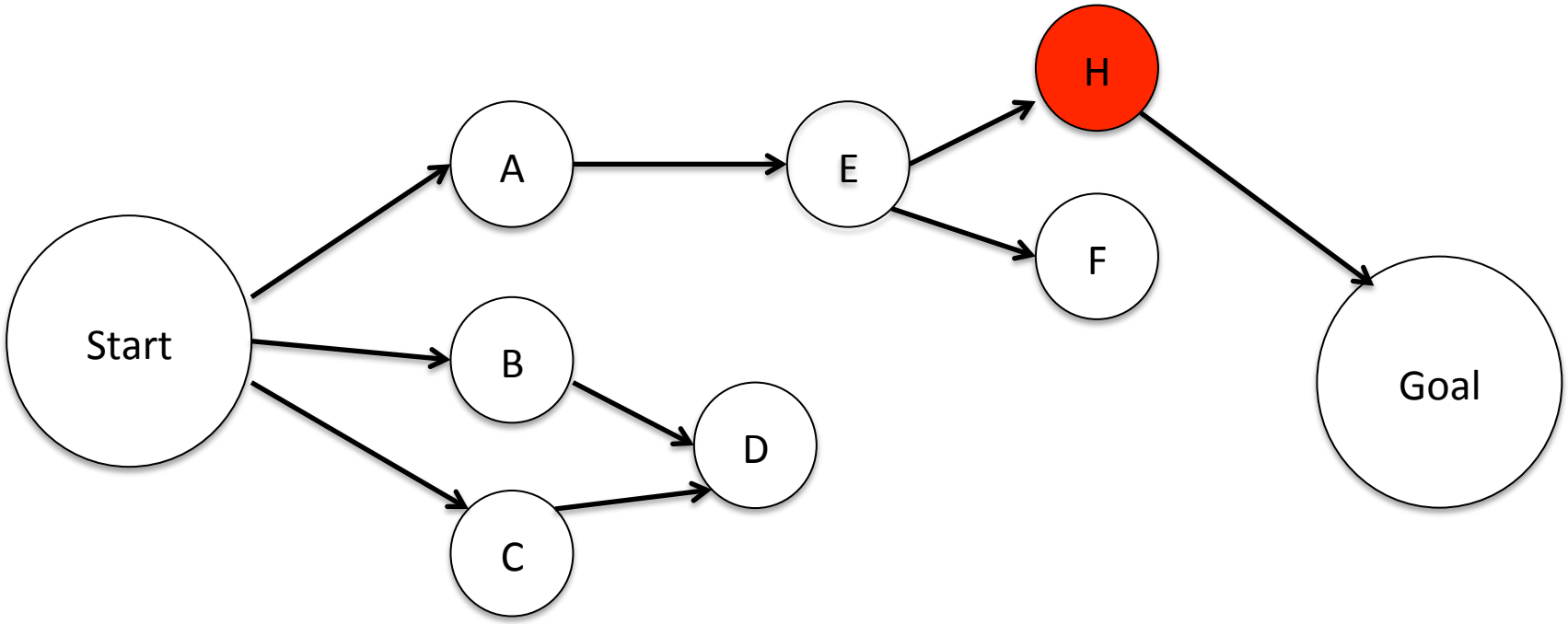
# Depth-First Search Example

Todo list: F, B, C



# Depth-First Search Example

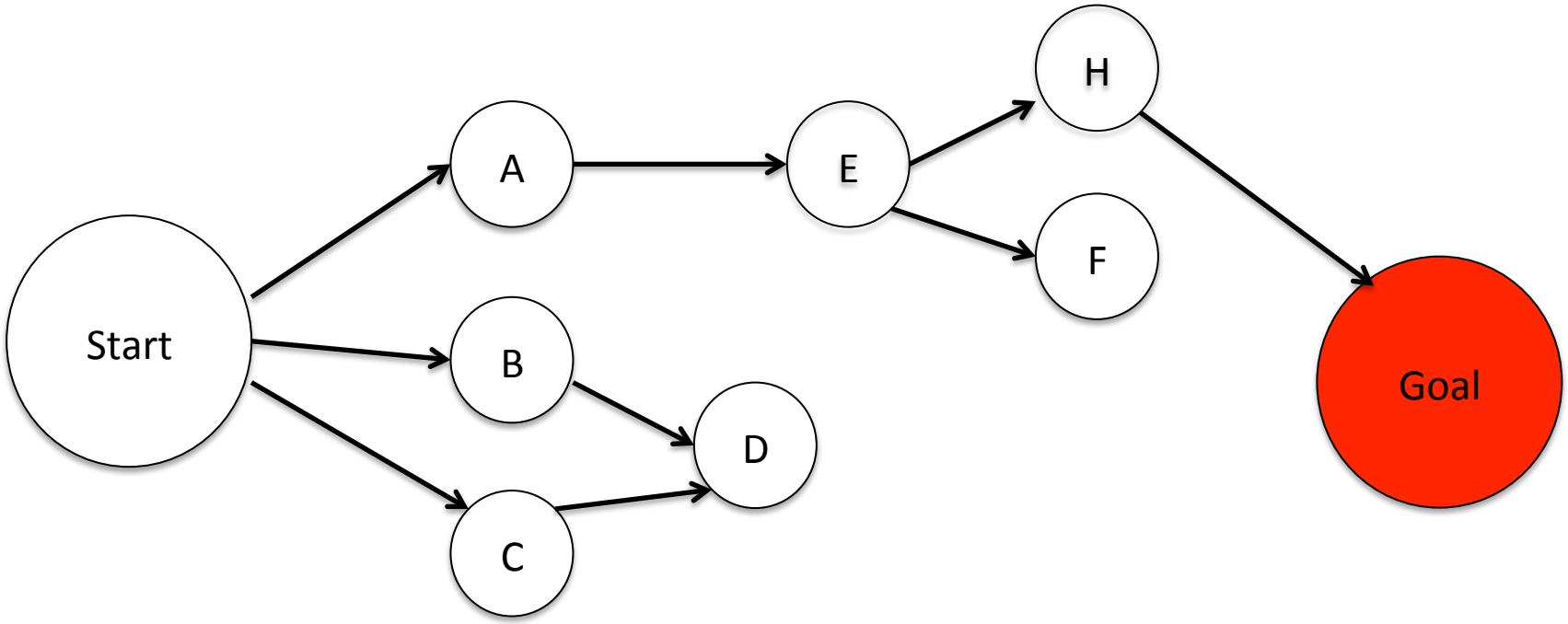
Todo list: Goal, F, B, C



# Depth-First Search Example

Todo list: F, B, C

Victory!!



# Similarities

- Both algorithms are very similar, except for the storage of the todo list
- Connects very nicely to data structures (BFS uses FIFO todo list, DFS uses LIFO todo list)
- FIFO = queue, LIFO = stack
- Next, we will define a more general formulation of graph search that will help us learn two more algorithms for path planning

# General Approach to Graph Search

```
to_visit = [(x_0, priority=1)]
dead_nodes = []

while not(to_visit.is_empty())
   (n, priority) = to_visit.get_highest_priority()
   if n = x_g then terminate
   for m in n.neighbors()
      if m not in dead_nodes
         to_visit.update_priority(m, calcpriority(m))
   to_visit.remove(n)
   dead_nodes.add(n)
```

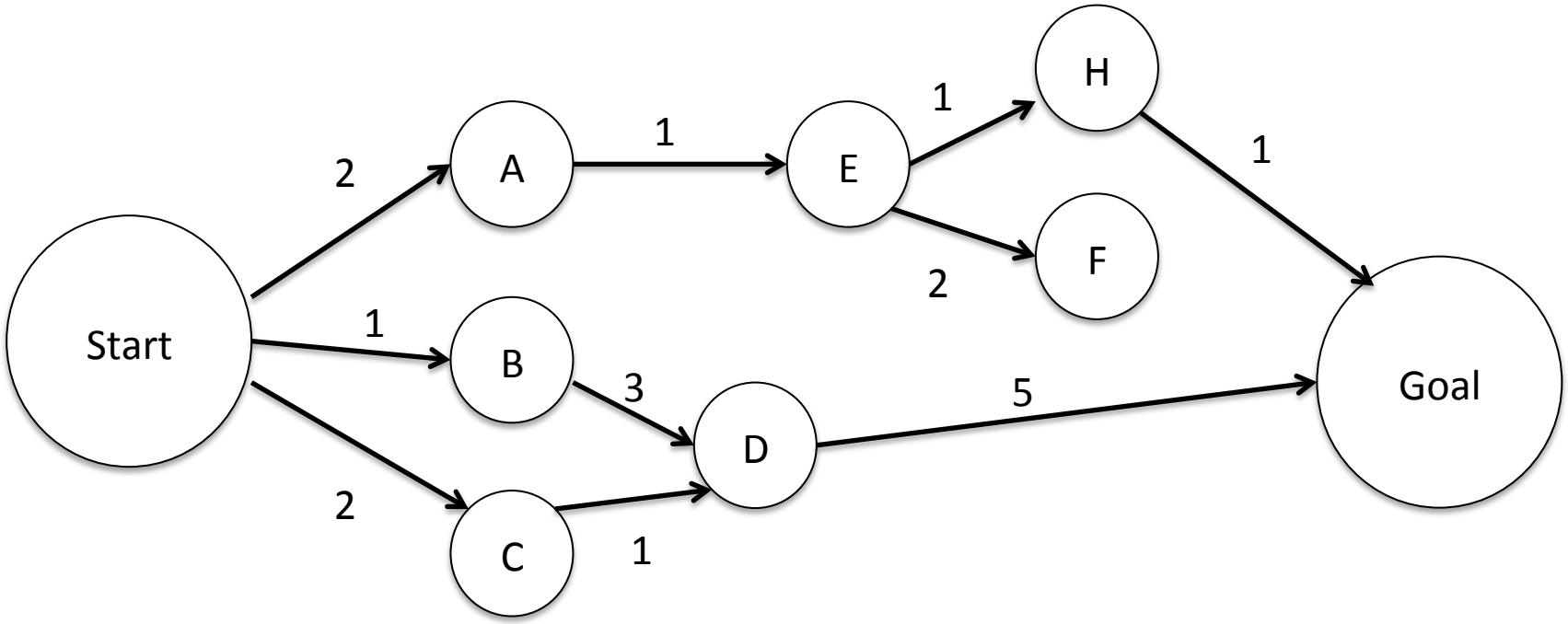
# Dijkstra's Algorithm

- Considers edge costs (not done in BFS or DFS)
- Guaranteed to find optimal path (minimum sum of costs)
- Key idea: store a tentative cost to each node in the todo list, update if possible (example will clarify)

# Dijkstra's Example

Todo list: (Start, 0)

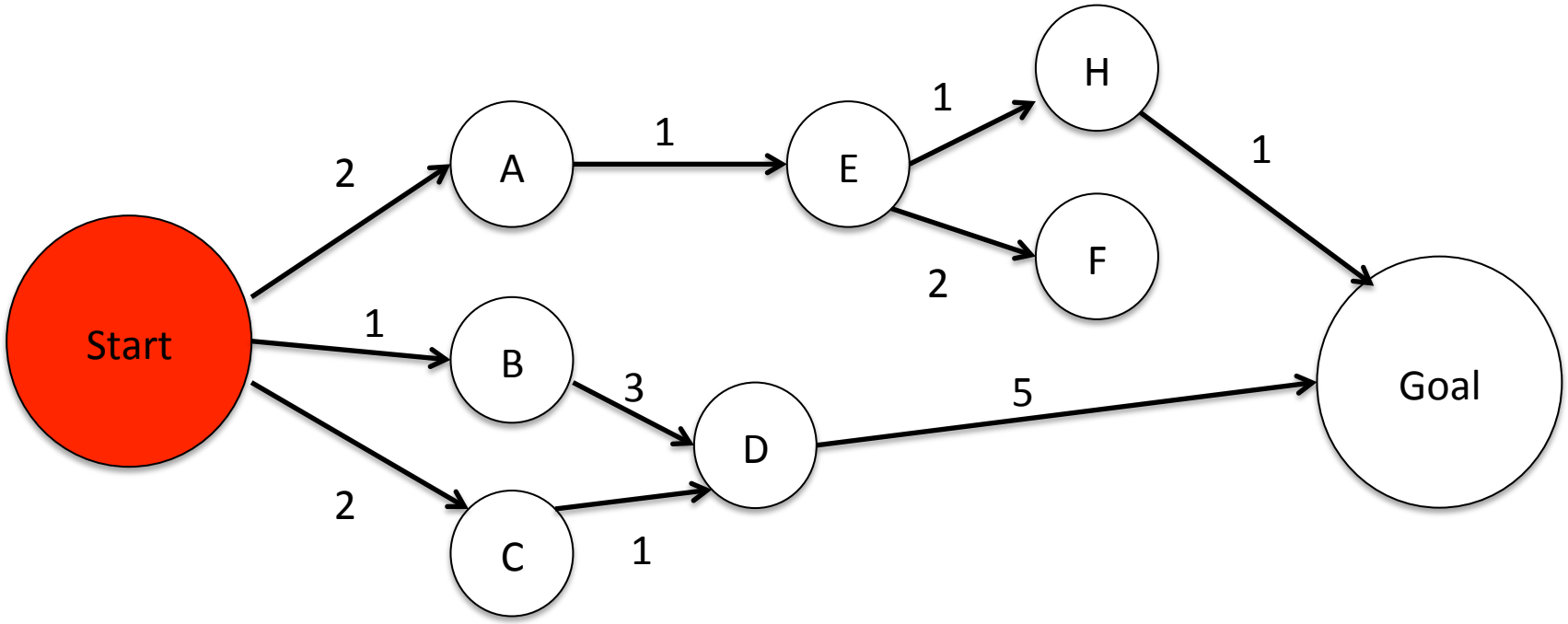
Dead nodes:



# Dijkstra's Example

Todo list:

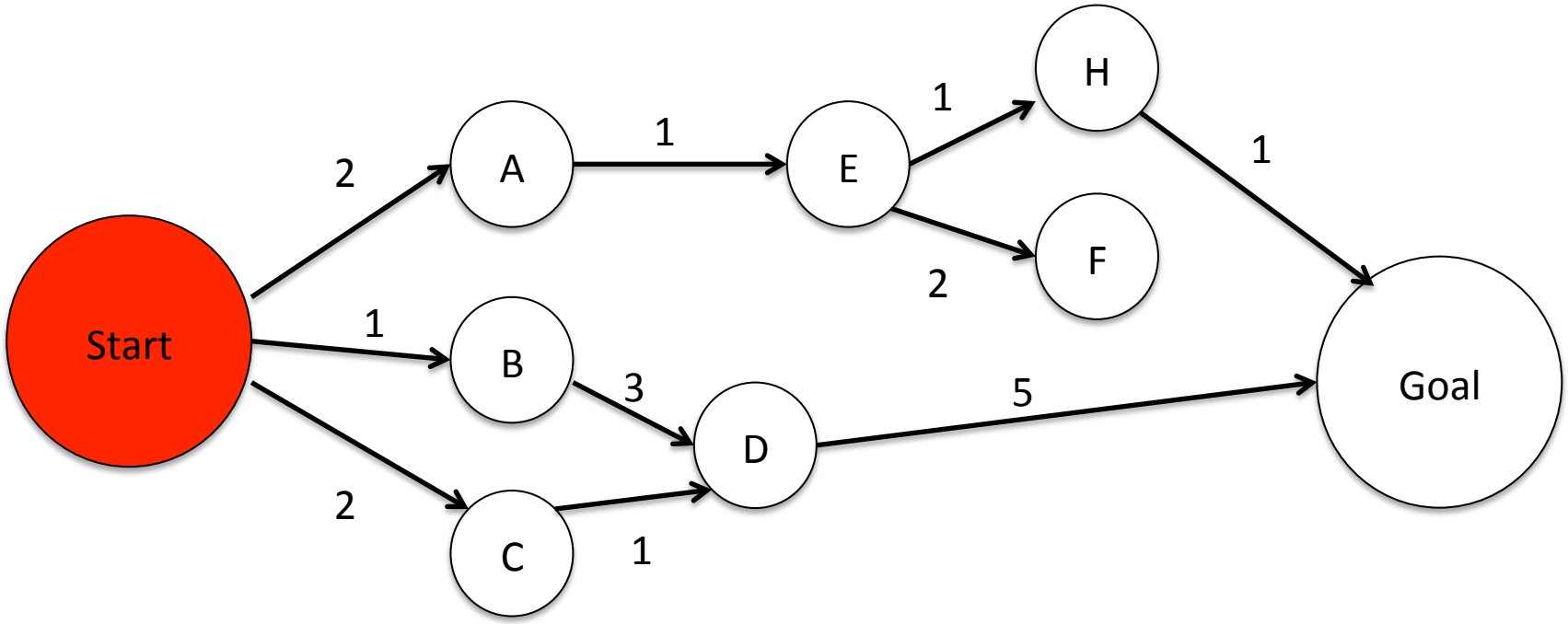
Dead nodes:



# Dijkstra's Example

Todo list: (B,-1), (A,-2), (C, -2)

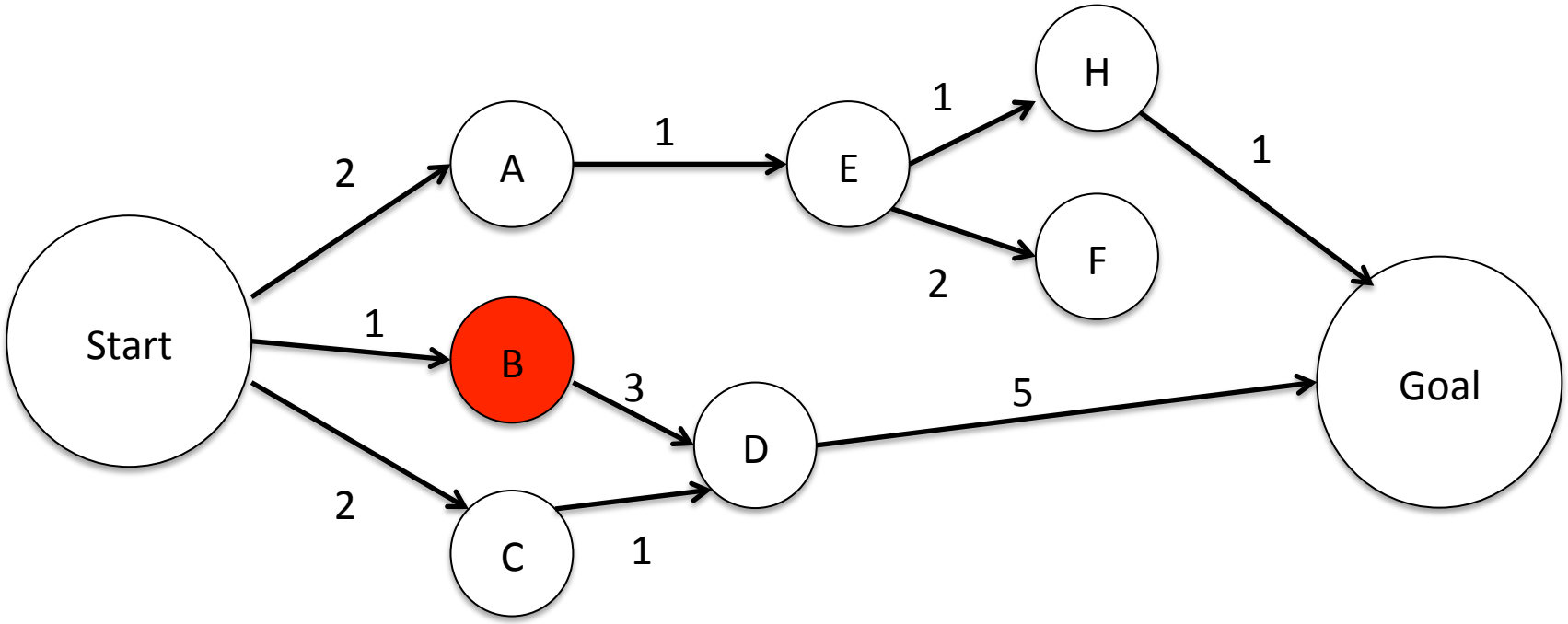
Dead nodes: (Start)



# Dijkstra's Example

Todo list: (A,-2), (C, -2)

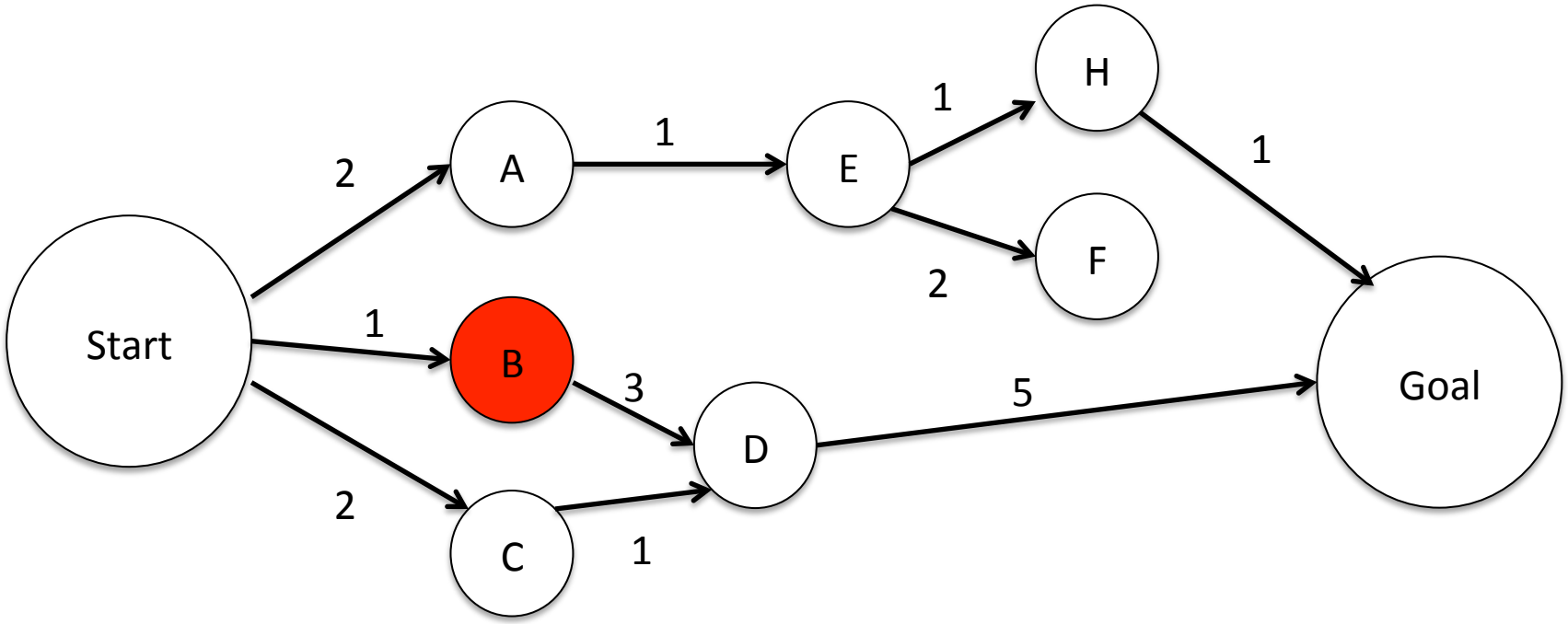
Dead nodes: (Start)



# Dijkstra's Example

Todo list: (A,-2), (C, -2), (D,-4)

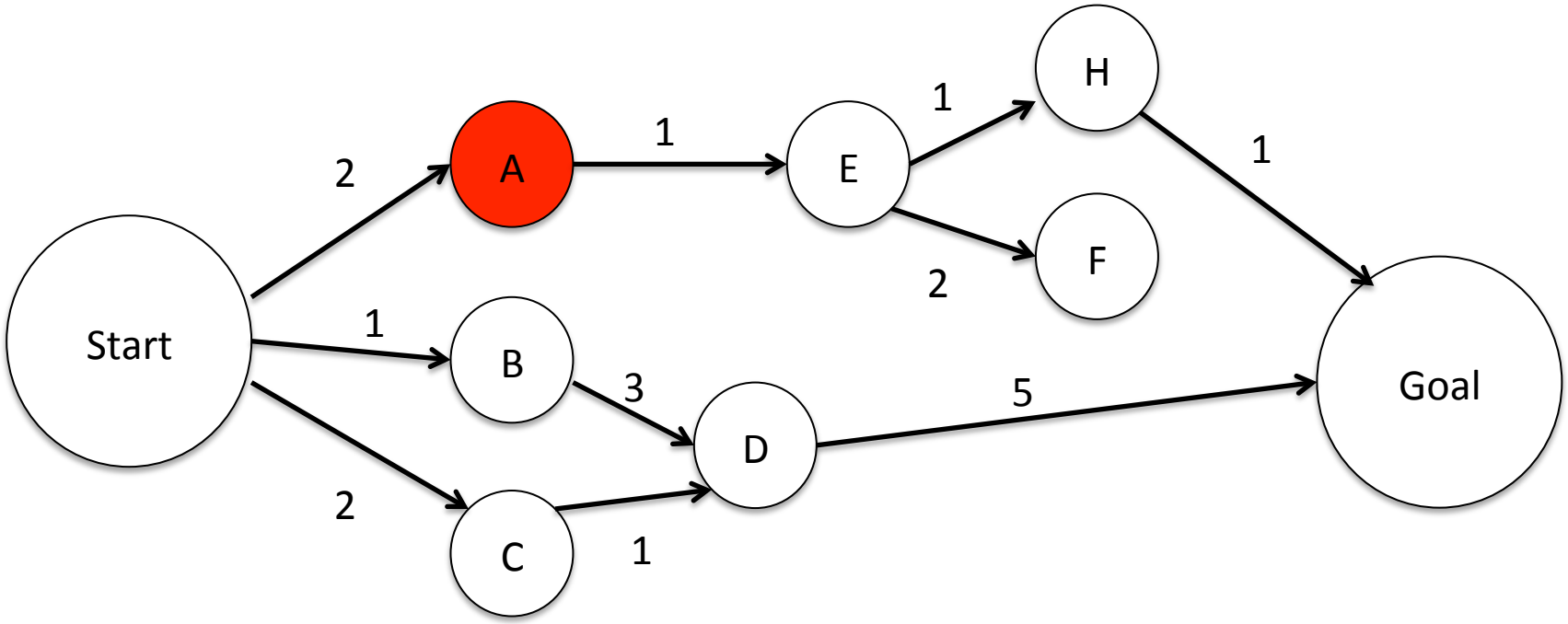
Dead nodes: (Start, B)



# Dijkstra's Example

Todo list: (C, -2), (D,-4)

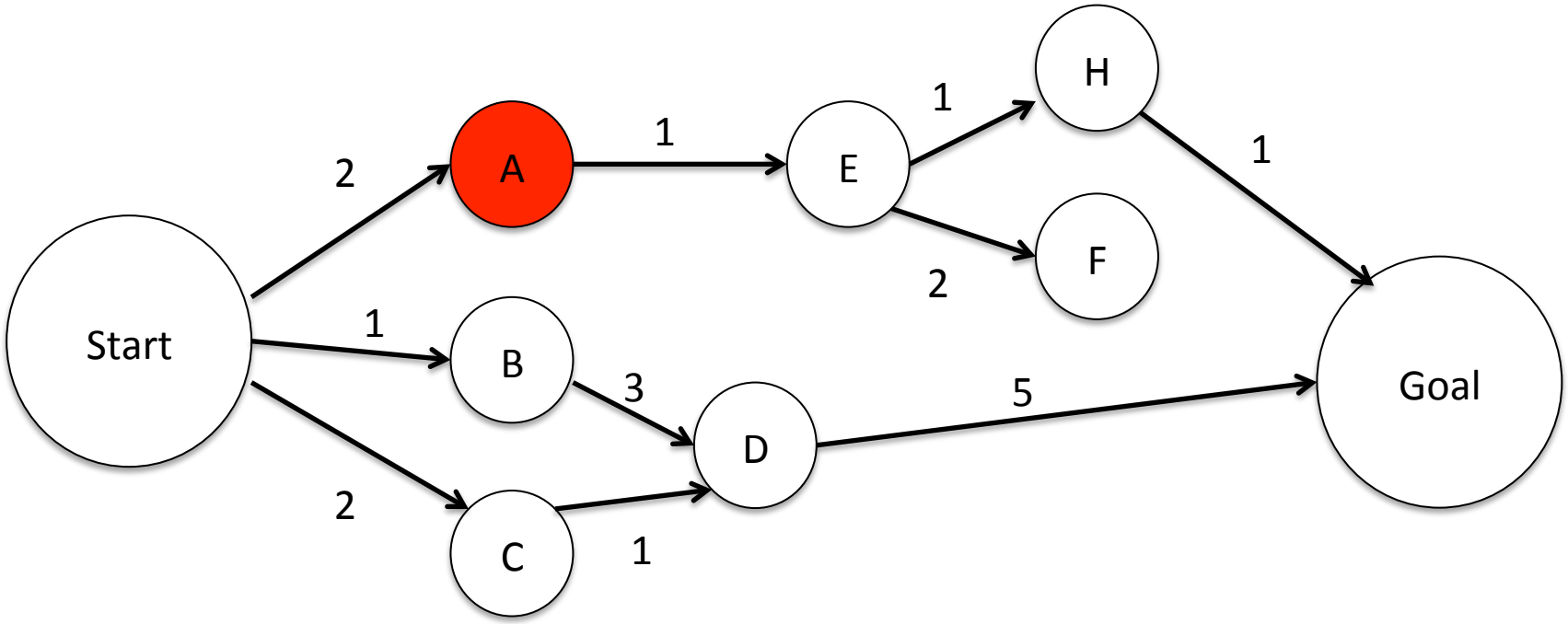
Dead nodes: (Start, B)



# Dijkstra's Example

Todo list: (C, -2), (E,-3), (D,-4)

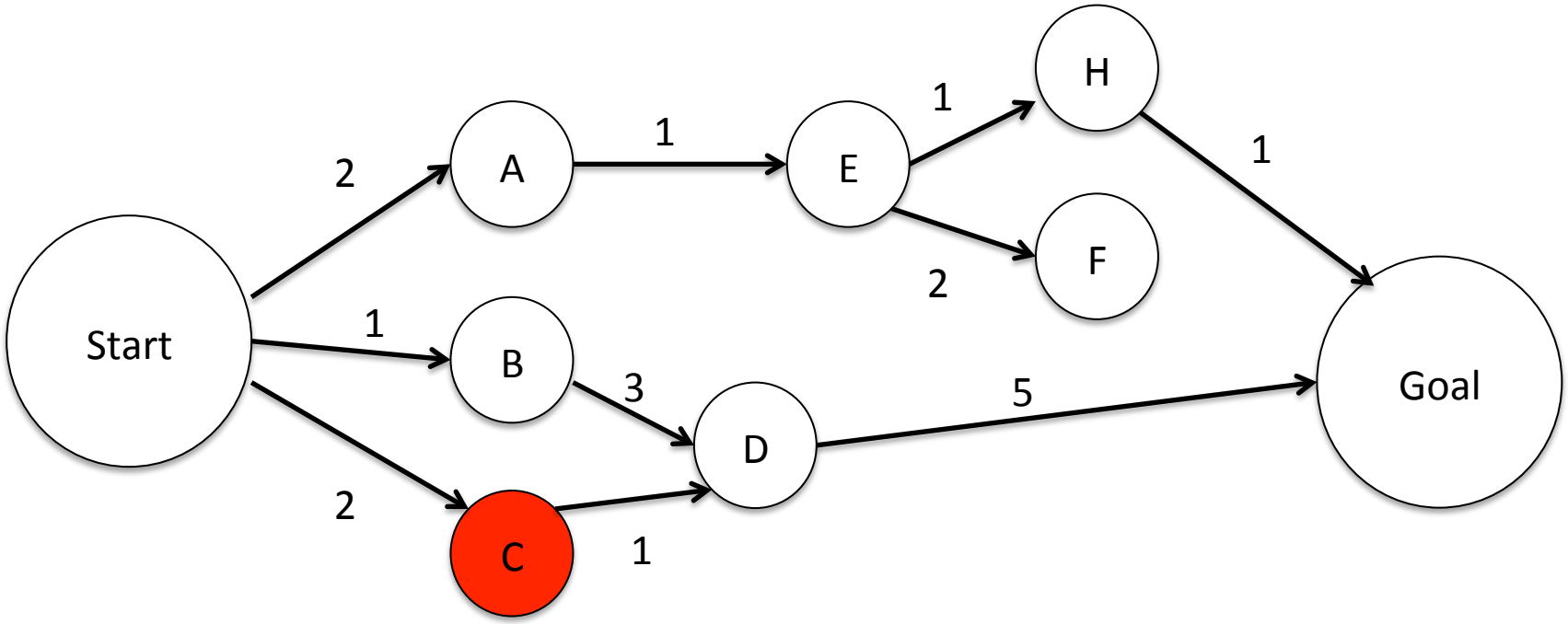
Dead nodes: (Start, B, A)



# Dijkstra's Example

Todo list: (E,-3), (D,-4)

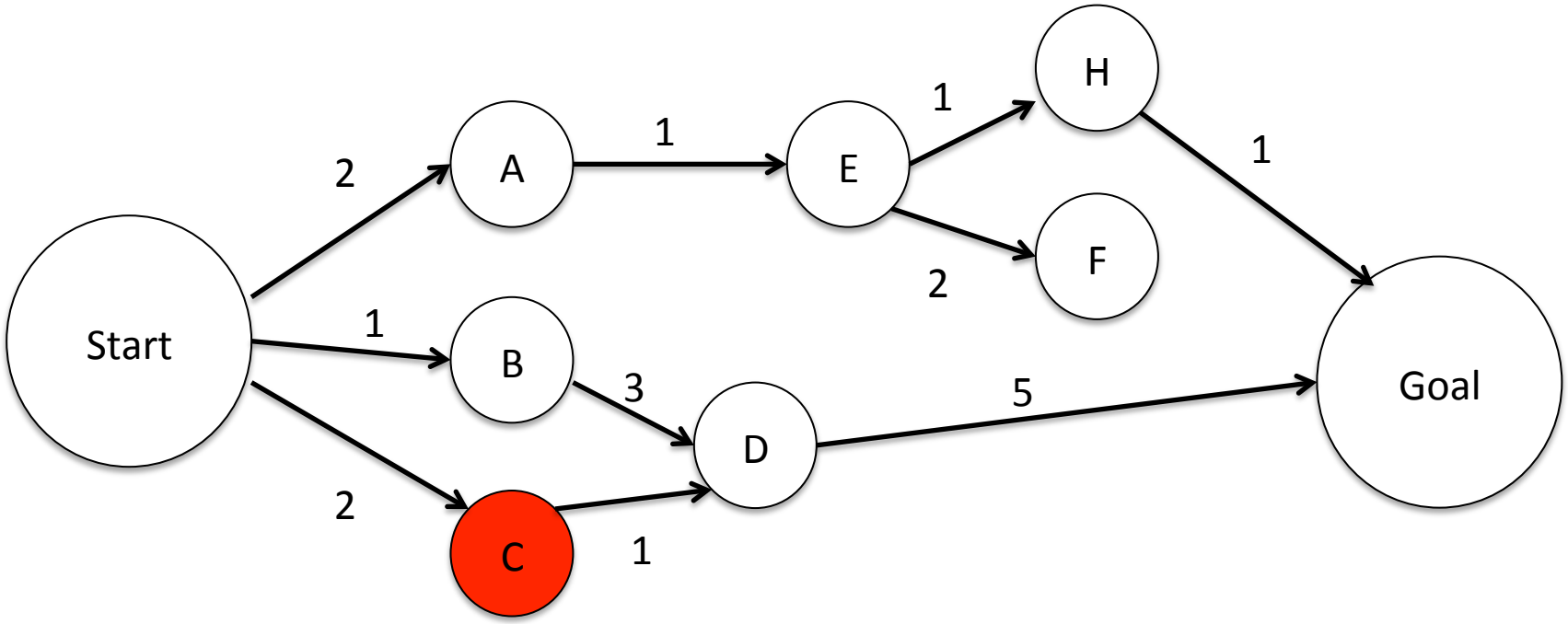
Dead nodes: (Start, B, A)



# Dijkstra's Example

Todo list: (E,-3), (D,-3)

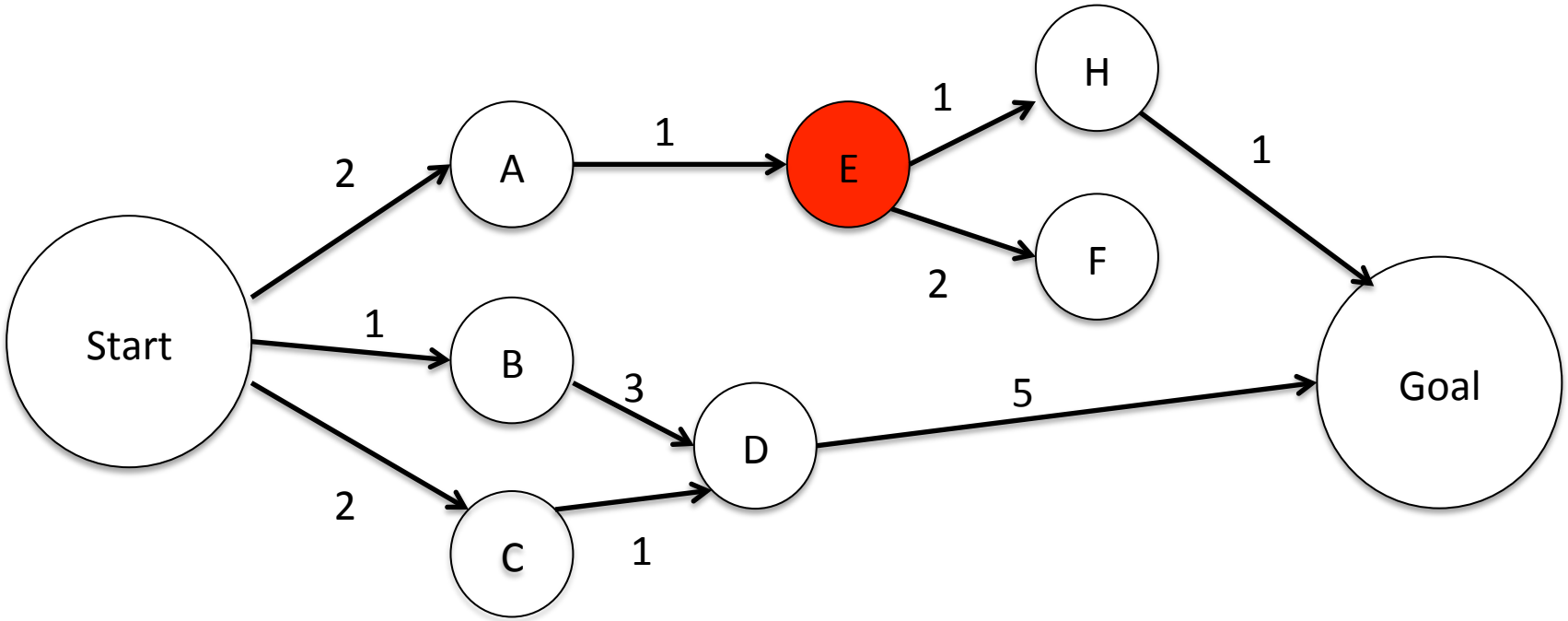
Dead nodes: (Start, B, A, C)



# Dijkstra's Example

Todo list: (D,-3)

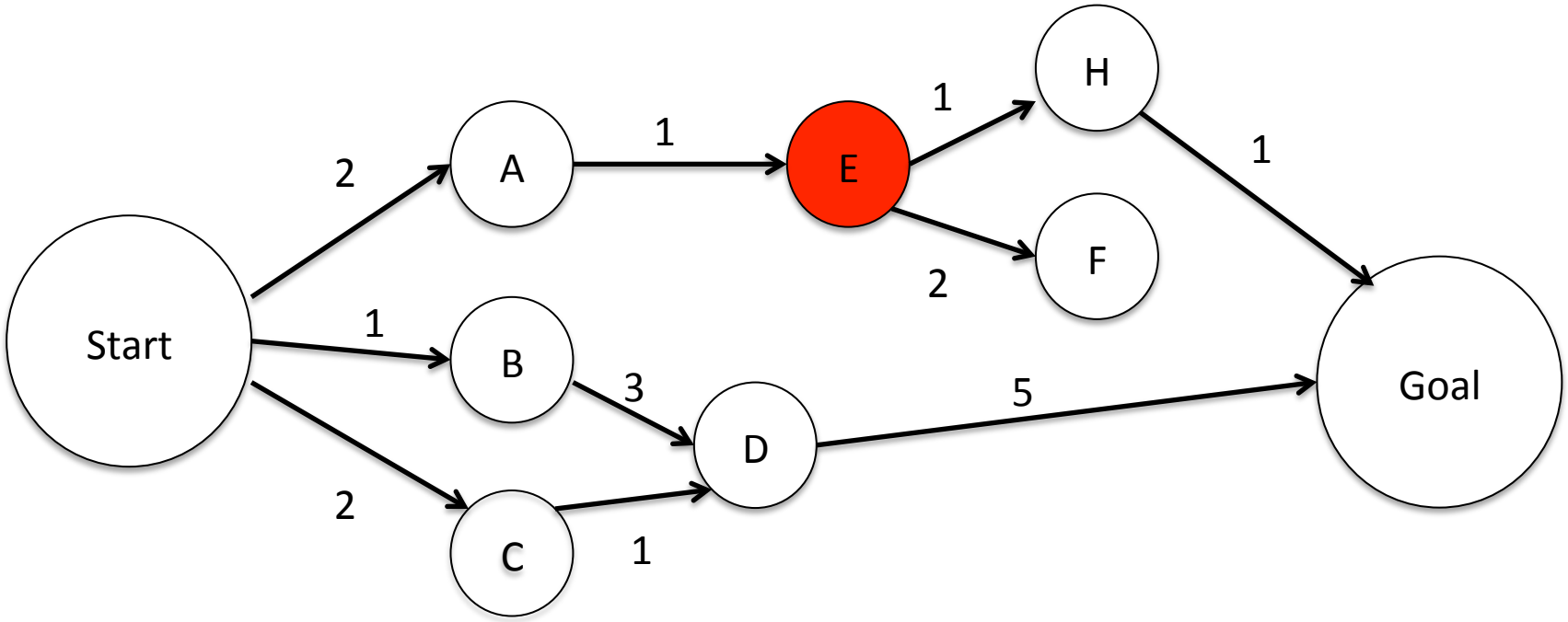
Dead nodes: (Start, B, A, C)



# Dijkstra's Example

Todo list: (D,-3), (H,-4), (F,-5)

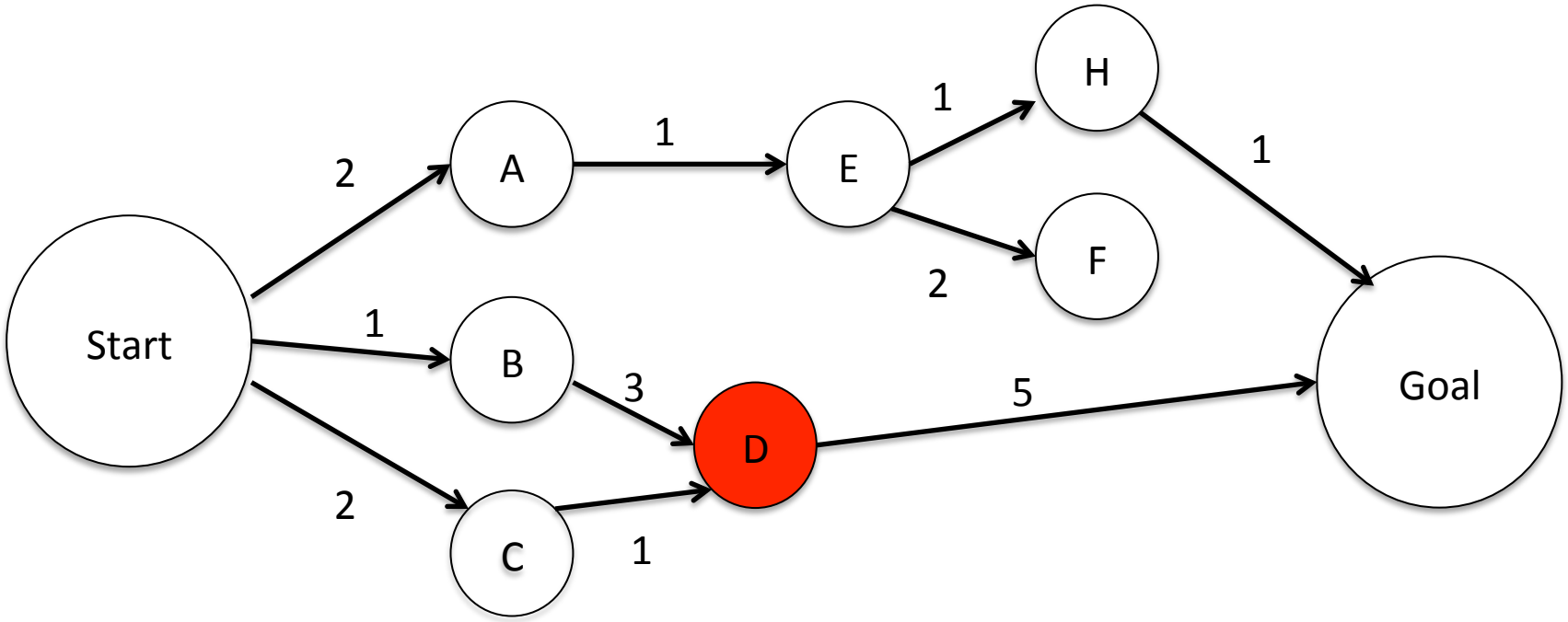
Dead nodes: (Start, B, A, C, E)



# Dijkstra's Example

Todo list: (H,-4), (F,-5)

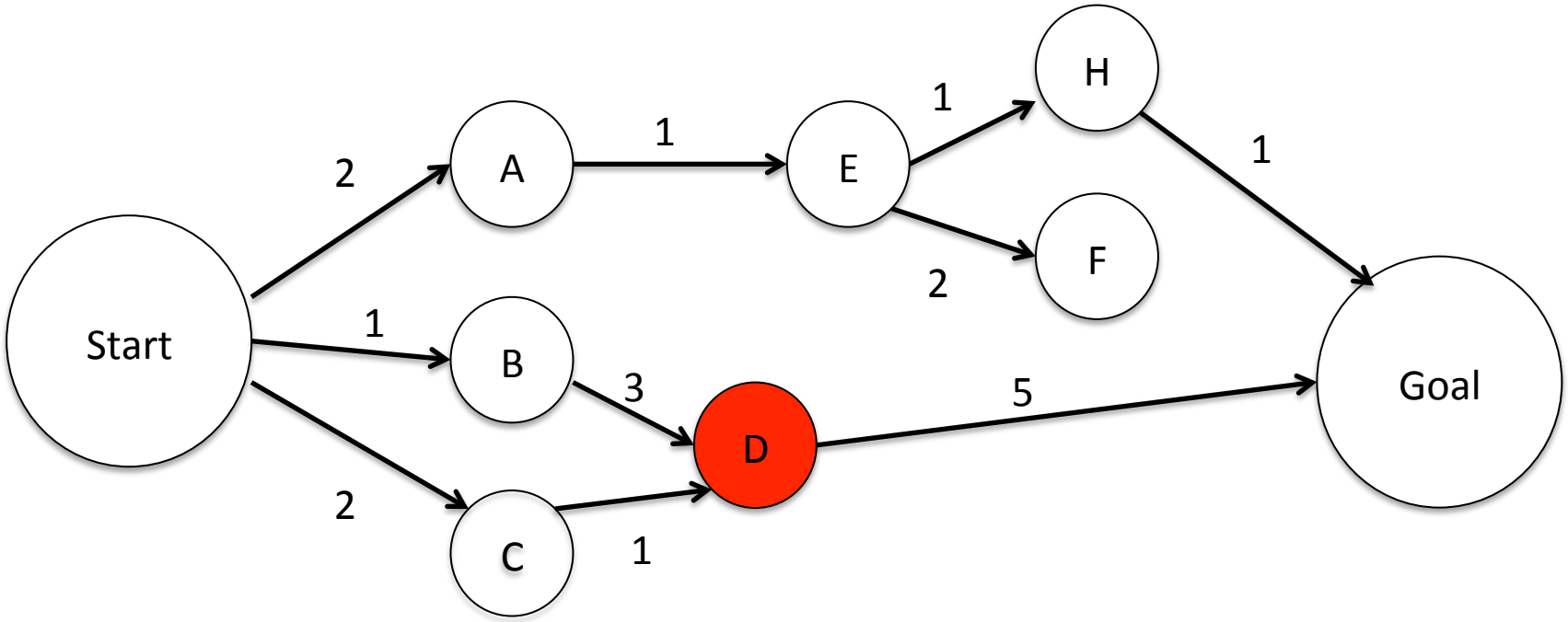
Dead nodes: (Start, B, A, C, E)



# Dijkstra's Example

Todo list: (H,-4), (F,-5), (Goal,-8)

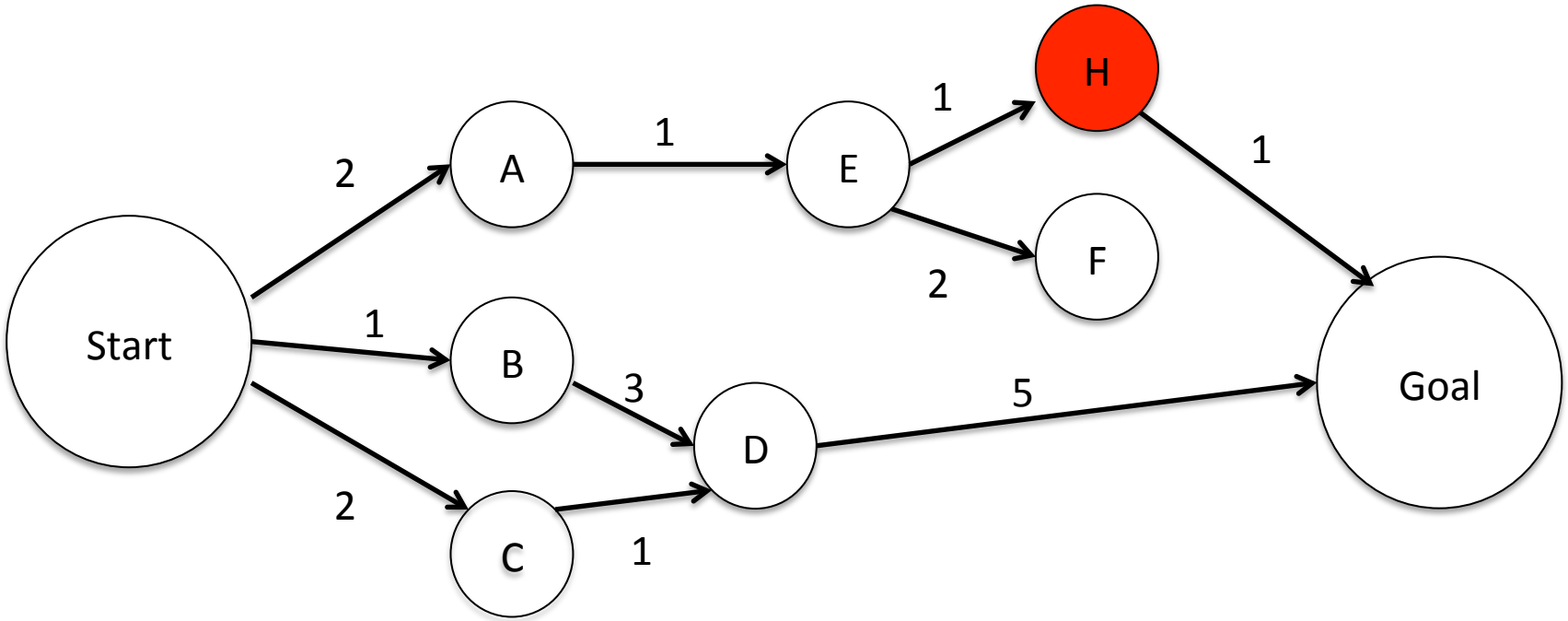
Dead nodes: (Start, B, A, C, E, D)



# Dijkstra's Example

Todo list: (F,-5), (Goal,-8)

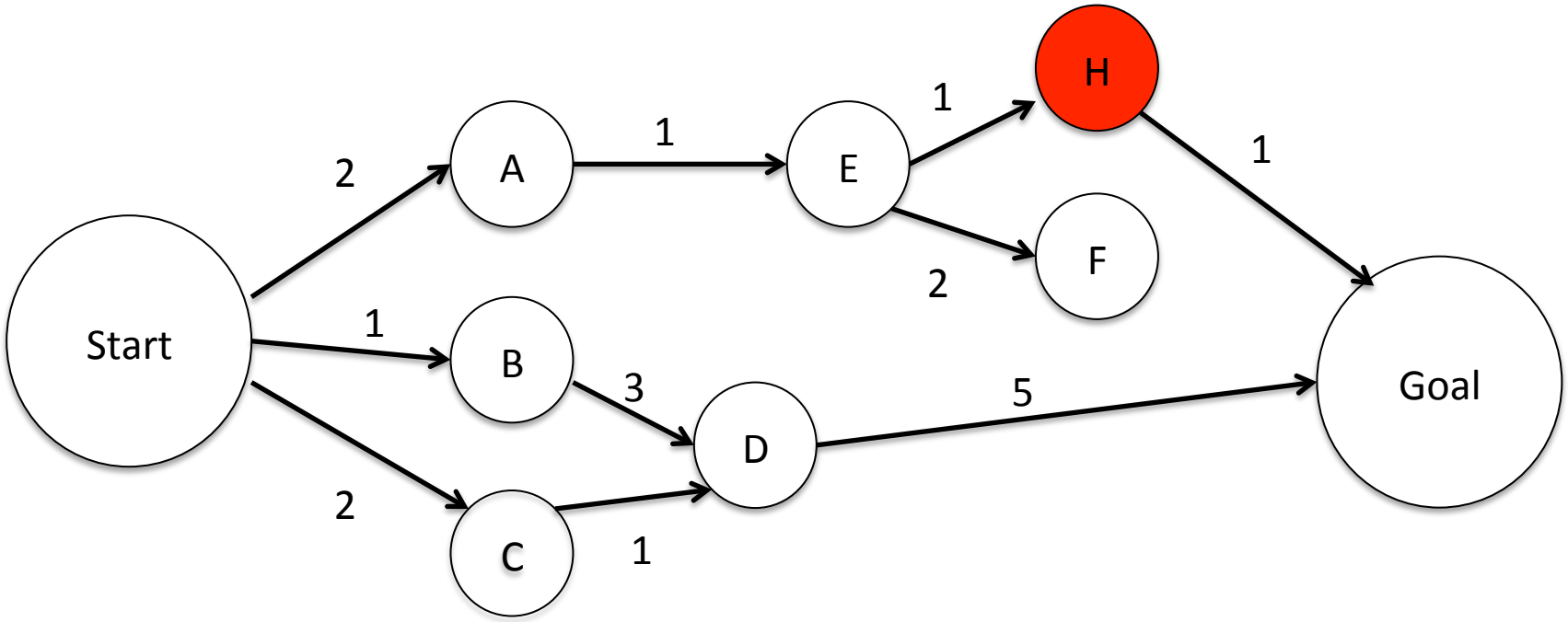
Dead nodes: (Start, B, A, C, E, D)



# Dijkstra's Example

Todo list: (F,-5), (Goal,-5)

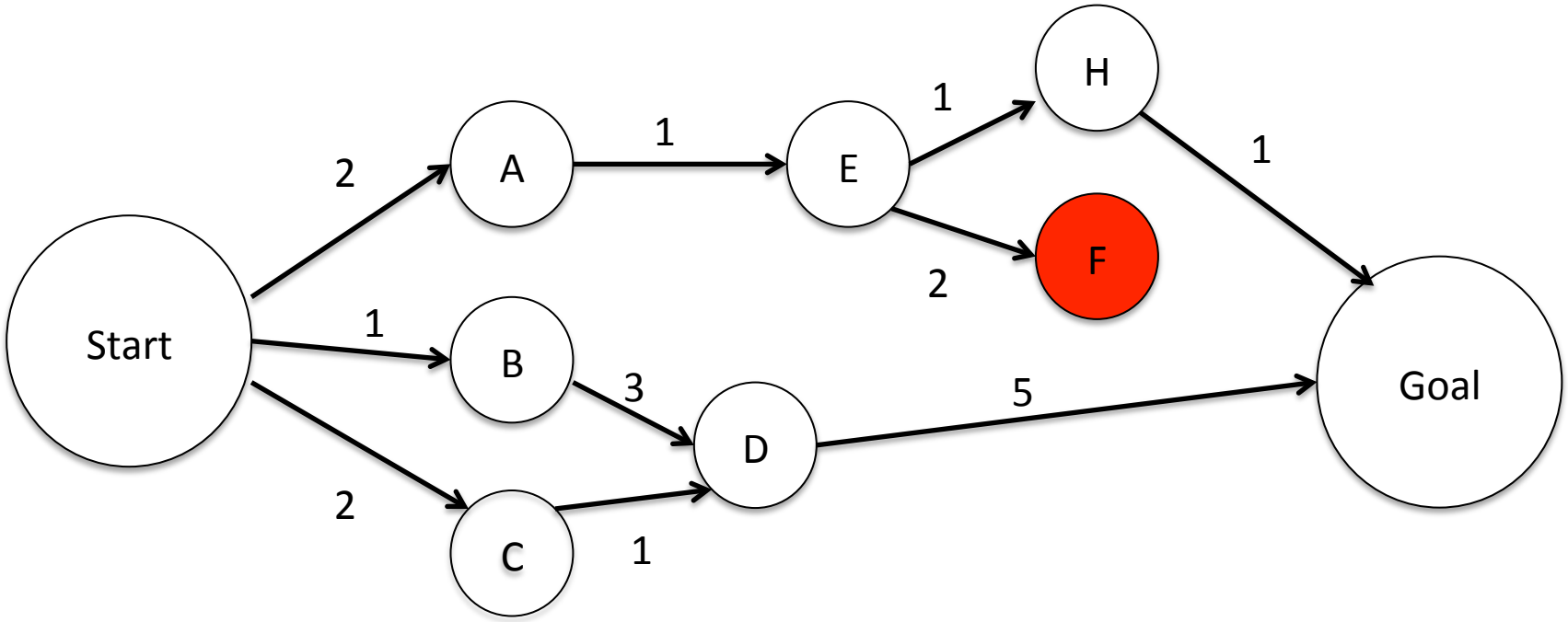
Dead nodes: (Start, B, A, C, E, D, H)



# Dijkstra's Example

Todo list: (Goal,-5)

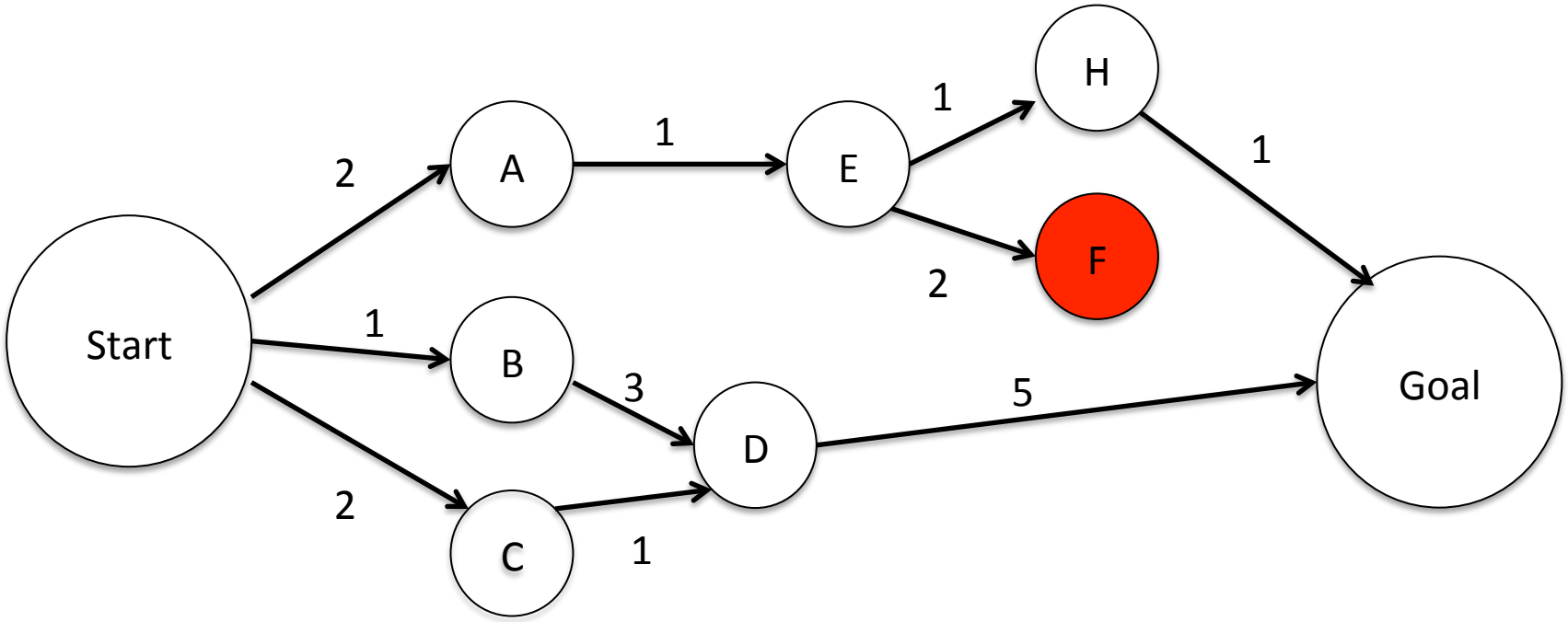
Dead nodes: (Start, B, A, C, E, D, H)



# Dijkstra's Example

Todo list: (Goal,-5)

Dead nodes: (Start, B, A, C, E, D, H, F)



# Dijkstra's Example

Todo list:

Dead nodes: (Start, B, A, C, E, D, H, F)

Victory!!