Undirected Search

Basic Search Algorithm

```
procedure Search(G,S,goal)
    Inputs
        G: graph with nodes N and arcs A
        s: start node
        goal: Boolean function of states
    Output
        path from a member of S to a node for which goal is true
        or ⊥ if there are no solution paths
    Local
        Frontier: set of paths

    Frontier ← {⟨s⟩}
    while (Frontier ≠ {})
        select and remove ⟨n_0,...,n_k⟩ from Frontier
        if ( goal(n_k)) then
            return ⟨n_0,...,n_k⟩
        Frontier ← Frontier ∪ {⟨n_0,...,n_k, s⟩: ⟨n_k,n⟩ ∈ A}
    return ⊥
```

(From Artificial Intelligence: Foundations of Computational Agents 2nd ed., Poole and Mackworth, 2017.)

Multiple Path Pruning

```
procedure Search(G,S,goal)
    Inputs
        G: graph with nodes N and arcs A
        s: start node
        goal: Boolean function of states
    Output
        path from a member of S to a node for which goal is true
        or ⊥ if there are no solution paths
    Local
        Frontier: set of paths
        Explored: set of nodes that have been expanded

    Frontier ← {⟨s⟩: s ∈ S}
    Explored ← {}
    while (Frontier ≠ {})
        select and remove ⟨n_0,...,n_k⟩ from Frontier
        Explored ← Explored ∪ {n_k}
        if ( goal(s_k)) then
            return ⟨n_0,...,n_k⟩
        Frontier ← Frontier ∪ {⟨n_0,...,n_k, n⟩: ⟨n_k,n⟩ ∈ A ∧
                                                 n ∉ Frontier ∧
                                                 n ∉ Explored}

    return ⊥
```

Abuse of notation

Multiple Path Pruning

Depth First Search

```
procedure DepthFirstSearch(G,S,goal)
    Inputs
        G: graph with nodes N and arcs A
        s: start node
        goal: Boolean function of states
    Output
        path from a member of S to a node for which goal is true
        or ⊥ if there are no solution paths
    Local
        Frontier: a stack of paths
        Explored: set of nodes that have been expanded

    Frontier ← Empty Stack
    Frontier.push(⟨s⟩)
    Explored ← {}
    while (Frontier is not empty)
        Pop ⟨n_0,...,n_k⟩ from Frontier
        Explored ← Explored ∪ {n_k}
        if ( goal(n_k)) then
            return ⟨n_0,...,n_k⟩
        For all {⟨n_k,n⟩ : ⟨n_k,n⟩ ∈ A  ∧ n ∉ Frontier ∧ n ∉ Explored}
             Frontier.push(⟨n_0,...,n_k, n⟩)
    return ⊥
```

For Breadth-First Search, Just replace the Stack with a Queue.