Types of Search Implementations

Graph Search (Version 1)

```
frontier = { Node(initial_state)) }
reached = { initial_state : Node(initial_state) }

while (frontier not empty) :
    current = frontier.pop()

    if isGoal(current.state) :
        return current.getPath()

    for a in actions(current.state) :
        successor= Node(T(current.state, a), current)
        if successor.state not in reached :
            frontier.push(successor)
            reached.insert(successor.state : successor)

return failure
```

The main idea behind Graph Search Version 1 is that the algorithm ensures no revisits to the nodes. This may omit optimal path because the algorithm does not check for the cost before considering a node visited.

Graph Search (Version 2)

```
frontier = { Node(initial_state)) }
reached = { initial_state : Node(initial_state) }

while (frontier not empty) :
    current = frontier.pop()

    if isGoal(current.state) :
        return current.getPath()

    for a in actions(current.state) :
        successor= Node(T(current.state, a), current)
        if successor.state not in reached or
        successor.getCost() < reached[successor.state].getCost() :
            frontier.push(successor)
            reached.insert(successor.state : successor)

return failure
```

Compared to Version 1, the algorithm is more relaxed it also considers paths with lower cost than what was discovered before.

Graph Search (Version 3)

```
frontier = { Node(initial_state)) }
reached = { initial_state : Node(initial_state) }

while (frontier not empty) :
    current = frontier.pop()

    reached.insert(current.state : current)

    if isGoal(current.state) :
        return current.getPath()

    for a in actions(current.state) :
        successor= Node(T(current.state, a), current)
        if successor.state not in reached :
            frontier.push(successor)

return failure
```

In this version, the algorithm adds a node to the reached/visited set when it is popped.

Tree Search

No reached/visited set to maintain the revisits. Tree Search algorithms are free to visit all nodes, no matter how many number of times they do.