```
adjacency_list = {
    'WA': ['NT', 'SA'],
    'NT': ['WA', 'SA', 'Q'],
    'SA': ['WA', 'NT', 'Q', 'NSW', 'V'],
    'Q': ['NT', 'SA', 'NSW'],
    'NSW': ['Q', 'SA', 'V'],
    'V': ['SA', 'NSW'],
    'T': []
}


colors = ['Red', 'Green', 'Blue']


def is_valid(state, color, color_assignment):
    for neighbor in adjacency_list[state]:
        if color_assignment.get(neighbor) == color:
            return False
    return True


def color_map(state_list, color_assignment):

    if len(color_assignment) == len(state_list):
        return True


    state = state_list[len(color_assignment)]


    for color in colors:
        if is_valid(state, color, color_assignment):

            color_assignment[state] = color


            if color_map(state_list, color_assignment):
                return True


            del color_assignment[state]

    return False


def solve_map_coloring():
    state_list = list(adjacency_list.keys())
    color_assignment = {}

    if color_map(state_list, color_assignment):
        return color_assignment
    else:
        return "No solution"


solution = solve_map_coloring()


print("Coloring of the Australian map:")
for state, color in solution.items():
    print(f"{state}: {color}")
```

```
Coloring of the Australian map:
WA: Red
NT: Green
SA: Blue
Q: Red
NSW: Green
V: Red
T: Red
```

Start coding or generate with AI.