于欣淼 YU Xinmiao 518021910792

VE477 lab6

Q1

```
def BFS(Bgraph, s, t, parent):
    vis = {}
    for num in range(Bgraph.v_num):
        vis[Bgraph.v_list[num]] = False

    queue = [s]

    vis[s] = True
    while queue:
        u = queue.pop(0)

        for key, val in Bgraph.graph[u].items():
            if not vis[key] and val > 0:
                queue.append(key)
                vis[key] = True
                parent[key] = u

    return True if vis[t] else False
```

Q2

```
# Returns tne maximum flow from s to t in the given graph
def EdmondsKarp(Bgraph, source, sink):
    parent = {}

    for num in range(Bgraph.v_num):
        parent[Bgraph.v_list[num]] = -1
    max_flow = 0
    while BFS(Bgraph, source, sink, parent):

        path_flow = float("Inf")
        s = sink
```

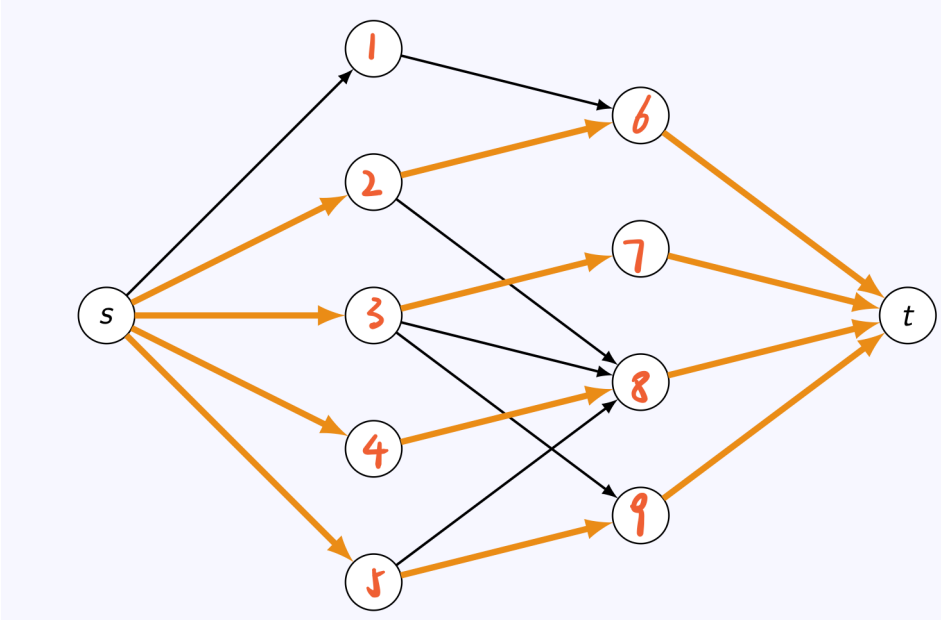
```
        while s != source:
            path_flow = min(path_flow, Bgraph.graph[parent[s]][s])
            s = parent[s]
        max_flow += path_flow

        v = sink
        while v != source:
            u = parent[v]
            new_val = Bgraph.graph[u][v] - path_flow
            Bgraph.graph[u][v] = new_val
            new_val = Bgraph.graph[u][v] + path_flow
            Bgraph.graph[v][u] = new_val
            v = parent[v]

    return max_flow
```

Q3

The graph used to demonstrate is the same as discussed in class.




```
from EdmondsKarp import EdmondsKarp, Graph


def Bipartite(graph, left, right):

    for l_v in left:
        graph.add_edge('s', l_v, 1)
    for r_v in right:
```

```
            graph.add_edge(r_v, 't', 1)
        return EdmondsKarp(graph, 's', 't')


# demonstrate the same graph in slides
left_ver = range(1, 6)
right_ver = range(6, 10)
g = Graph()
g.add_edge(1, 6, 4)
g.add_edge(2, 6, 4)
g.add_edge(2, 8, 3)
g.add_edge(3, 7, 2)
g.add_edge(3, 8, 7)
g.add_edge(3, 9, 10)
g.add_edge(4, 8, 1)
g.add_edge(5, 8, 2)
g.add_edge(5, 9, 3)

print(Bipartite(g, left_ver, right_ver))
```

gives the answer 4 as expected.

```
→  lab6 git:(master) ✗ python3 Bipartite.py
4
```