程式作業 6

1. 演算法設計

   藉 vector 儲存各點間路徑，無向圖轉為有向圖、新增雙向的路徑、權重通通設為一，如此一來只要再對各點做 fordFulkerson 其輸出結果便是各點對其他點的 maxflow，取其中最小的值便可得到 minflow=圖形連接度，最後透過使用 bfs 以及先決定唯一起始點減少計算時間。

2. Pseudo code

3.

4. main()

5. 存輸入

6. int minD=INT_MAX //減少計算次數 先得到擁有最小 degree 的點 minV

7. int minV

8. for(int i=1;i<=v;i++) //減少計算次數 先得到擁有最小 degree 的點 minV

9. int Dcount=0;

10. for(int j=1;j<=v;j++)

11. if(rount[i][j]>0)

12. Dcount++

13. if(minD>Dcount)

14. minD=Dcount

15. minV=i

16. //進行 fordFulkerson 檢查 minV 對其他點的 minflow

17. for(int des=1; des<=v; des++)

18. ans = fordFulkerson(rount, minV, des, v, minD, minf);

19. return ans

20.

21. bool bfs(vector<vector<int> >rGraph, int s, int t, int parent[],int V)

22. bool visited[V];

23. queue<int> q;

24. q.push(s);

25. visited[s] = true;

26. parent[s] = -1;

27.

28. while (!q.empty())

29. u = q.front()

30. q.pop()

31. for (int v = 1; v <= V; v++)

32. if (!visited[v] && rGraph[u][v] > 0)

```
33.                 q.push(v);
34.                 parent[v] = u;
35.                 visited[v] = true;
36.                 if(v==t){
37.                     return true;
38.         return false;
39.
40.int fordFulkerson(vector<vector<int> > graph, int s, int t, int V, int mind,
   int minf)
41.     vector<vector<int> > rGraph=graph
42.     int parent[V+1]={0};
43.     int max_flow = 0;
44.
45.     while(bfs(rGraph, s, t, parent, V))
46.         int path_flow = 1;
47.         for (v = t; v != s; v = parent[v])
48.             u = parent[v];
49.             rGraph[u][v] -= path_flow;
50.             rGraph[v][u] += path_flow;
51.         max_flow += path_flow;
52.     if(max_flow < minf)
53.         minf = max_flow;
54.     return minf;
```

3. 時間複雜度分析

Bfs O(ve)

FordFulkerson O(ve)*O(v^2)

Main O(v^4(e))