演算法程式作業四

110502518陳文獻

Edmonds-Karp

1. 使用BFS找到增廣路徑 (若沒找到表示已找到maxflow)
2. 算出增廣路徑的流
3. 更新最大流
4. 更新圖的權重
5. 重複1.

Edmonds-Karp pseudocode

```
V:= 點數
G:= [] dim = (V, V) #鄰接矩陣
funtion EK(s, t) # Edmonds-Karp
    max_flow = 0
    while True
        parent = bfs(s, t)
        if parent[V] == None: # 圖已斷開，找不到路徑
            break
        v = t, flow = INF # 計算增廣路徑的流
        while v != s
            u = parent[v]
            flow = min(flow, G[u][v])
            v = u
        max_flow += flow
        v = t
        while v != s # 以得到的流更新原圖
            u = parent[v]
            G[u][v] -= flow
            G[v][u] += flow
            v = u
    return max_flow
```

Edmonds-Karp pseudocode

```
funtion bfs(s, t) -> parent:= [] dim = (V) # bfs樹
    parent = [None, ..., None] dim = (V)
    visited = [0, ..., 0] dim = (V)
    Q = queue(s)
    while Q is not empty
        u = Q.pop()
        for v from 1 to V
            if not visited[v] and G[u][v] > 0
                parent[v] = u
                visited[v] = 1
                if v == t
                    return parent
                Q.append(v)
    return parent
```

maxpath

1. 使用類dijkstra(priority search)找到增廣路徑，同時可以得到流(若沒找到表示已找到maxflow)
2. 更新最大流
3. 更新圖的權重
4. 重複1.

maxpath priority search 設計方法

- 以流最大的點優先
- 若新的流大於該點原本的流舊更新權重(類似relax)

maxpath pseudocode

```
V:= 點數
G:= [] dim = (V, V) #鄰接矩陣
funtion MP(s, t) # max_path
    max_flow = 0
    while True
        flow, parent = priority_search(s, t)
        if parent[V] == None: # 圖已斷開，找不到路徑
            break
        max_flow += flow
        v = t
        while v != s # 以得到的流更新原圖
            u = parent[v]
            G[u][v] -= flow
            G[v][u] += flow
            v = u
    return max_flow
funtion priority_search(s, t) -> flow: scaler, parent:= [] dim = (V) # bfs樹
    parent = [None, ..., None] dim = (V)
    priorities = [0, ..., 0] dim = (V)
    Q = priority_queue(s)
    while Q is not empty
        u = Q.extract_max()
        for v from 1 to V
            f = min(priorities[u], G[u][v])
            if f > priorities[v]
                priorities[v] = f
                parent[v] = uv
                change_priority(v, f)

    return priorities[V], parent
```

複雜度分析

- n 點數; m 邊數; f 最大流
- Edmonds-Karp = n*m*(n*n) = n^3*m
- maxpath = 尋找增廣路徑次數*(n*n*lgn) 使用紅黑樹實作priority queue

設計測資比較兩種做法的執行時間

	V = 100 E = 1000 max_weight = 10e5	V = 1000 E = 200000 max_weight = 10e5	V = 300; E = 30000; max_w = 10000;
Edmonds-Karp	0.053	0.969	0.119
maxpath	0.046	3.342	0.393