CHAPTER

06

圖論演算法進階

生活中的電力網路、自來水網路、交通運輸網都有一個共同點：在網路傳輸中有方向和容量。假設有向帶權圖 $G=(V,E)$，$V=\{s,v_1,v_2,v_3,\cdots,t\}$，其中，節點 s 為源點，節點 t 為匯點。邊的方向表示流向，邊的權值表示該邊允許通過的最大流量 cap（cap≥0），即邊的容量。若在有向帶權圖中有一條邊 (u,v)，則必然不存在反向邊 (v,u)。這樣的有向帶權圖稱為「網路」。

網路流即網路中的流，是定義在網路邊集上的一個非負函式集，flow= {flow(u,v)}，flow(u,v) 表示經過邊 (u,v) 的流量。滿足以下 3 個性質的網路流稱為「可行流」。

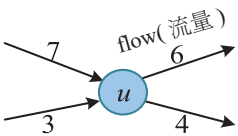
（1）容量約束：經過任意一條邊 (u,v) 的流量都不能超過其最大容量，即 flow(u,v)≤cap(u,v)。

（2）反對稱性：假設從節點 u 到節點 v 的流量是 flow(u,v)，從節點 v 到節點 u 的流量是 flow(v,u)，則滿足 flow(u,v)=−flow(v,u)。

（3）流量守恆：除源點和匯點外，所有內部節點的流入量都等於其流出量，即

$$\sum_{(x,u)\in E} \text{flow}(x,u) = \sum_{(u,y)\in E} \text{flow}(u,y)$$

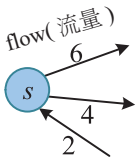
例如，若節點 u 的流入量之和是 10，則其流出量之和也是 10。



源點主要流出流量，但也可能流入流量。例如，工廠為源點，倉庫為匯點，部分產品在出廠後因檢測不合格需要返廠，則對源點來說，這些產品就是流入量。源點的淨輸出 f= 流出量之和 – 流入量之和。

$$f = \sum_{(s,x)\in E} \text{flow}(s,x) - \sum_{(y,s)\in E} \text{flow}(y,s)$$

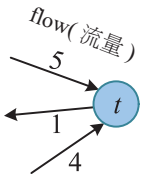
例如，若源點的流出量之和是 10，流入量之和是 2，則其淨輸出是 8。



匯點主要流入流量，但也可能流出流量。例如，部分產品在入庫後因檢測不合格需要回廠，則對匯點來說，這些產品就是流出量。匯點的淨輸入 f= 流入量之和 – 流出量之和。

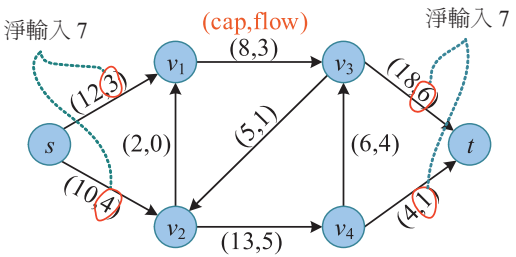
$$f = \sum_{(x,t)\in E} \text{flow}(x,t) - \sum_{(t,y)\in E} \text{flow}(t,y)$$

例如，若匯點的流入量之和是 9，流出量之和是 1，則其淨輸入是 8。



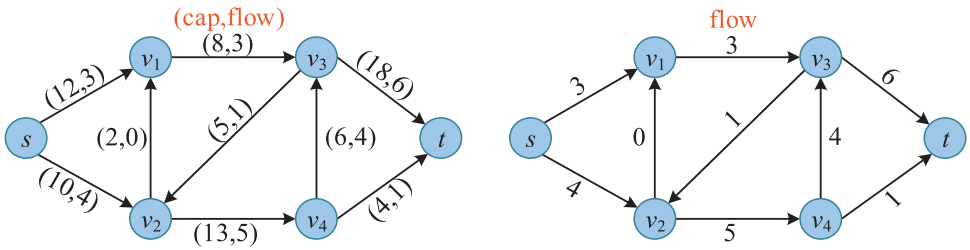
注意

對於任意一個可行流 flow，其淨輸出都等於其淨輸入，滿足流量守恆定律。



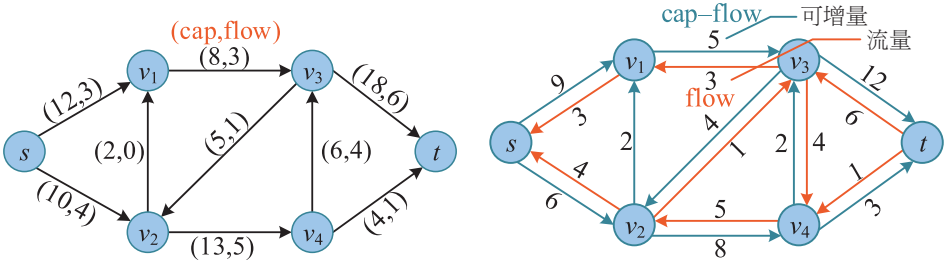
網路最大流指在滿足容量約束和流量守恆定律的前提下，淨輸出最大的網路流。求解網路最大流的基本概念是在網路中找擴充路徑，沿著擴充路徑擴充流量（增加流量），直到不存在擴充路徑時為止。

實流網路是只包含實際流量的網路。網路 G 及可行流 flow 對應的實流網路如下圖所示。

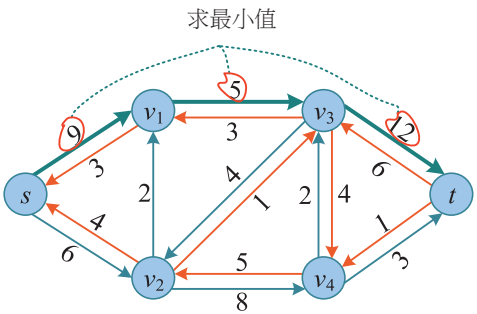


殘餘網路 G^* 與網路 G 中的節點相同，網路 G 中的每條邊都對應殘餘網路 G^* 中的一條邊或兩條邊。在殘餘網路 G^* 中，與網路 G 中的邊對應的同向邊的權值是可增量（還可增加多少流量），反向邊的權值是實際流量。

在殘餘網路中不顯示流量為 0 的邊。網路 G 及可行流 flow 對應的殘餘網路 G^* 如下圖所示。



擴充路徑是殘餘網路中從源點到匯點的一條簡單路徑。增廣量指擴充路徑上每條邊可增量的最小值。例如，一個網路流如下圖所示，s-v_1-v_3-t 就是一條擴充路徑，增廣量為 5。



增廣路定理：設 flow 是網路 G 的一個可行流，若不存在從源點到匯點的擴充路徑，則 flow 是 G 的一個最大流。

增廣路演算法的基本概念：首先在殘餘網路中搜尋擴充路徑，然後在實流網路中沿著擴充路徑擴充流量，在殘餘網路中沿著擴充路徑減少流量，重複以上步驟，直到不存在擴充路徑時為止。此時，實流網路中的可行流就是最大流。找擴充路徑的演算法不同，時間複雜度相差很大。

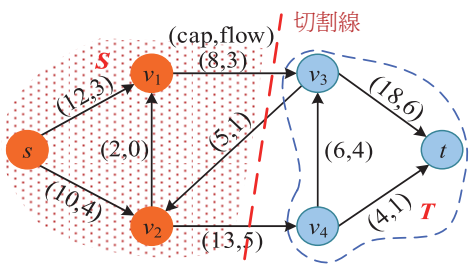
6.1 EK 演算法

EK（Edmonds-Karp）演算法是以廣度優先搜尋為基礎的最短擴充路徑演算法。用佇列 q 儲存已經存取且尚未檢查的節點，存取標記陣列 vis[] 標記節點是否已經存取，前導子陣列 pre[] 記錄擴充路徑上節點的前導子。pre[v]=u 表示擴充路徑上節點 v 的前導子是節點 u。

```
            }
            for(int j=0;j<cnt[i];j++){
                if(dfs(match[i][j])){
                    match[i][j]=u;
                    return 1;
                }
            }
        }
    }
    return 0;
}
```

6.5 最大流最小割

最大流最小割定理：任何網路中最大流的流量都等於最小割的容量。割指對網路中節點的劃分，它把網路中的所有節點都劃分成 S 和 T 兩個集合，源點 $s \in S$，匯點 $t \in T$，記為 CUT(S,T)，就像透過一條切割線把網路中的節點切割成 S 和 T 兩部分，S={s,v_1,v_2}，T={v_3,v_4,t}，如下圖所示。



割的淨流量 $f(S,T)$：指在切割線切中的邊中，從 S 到 T 的邊的流量減去從 T 到 S 的邊的流量。上圖中從 S 到 T 的邊 v_1-v_3 和邊 v_2-v_4 的流量分別為 3、5，從 T 到 S 的邊 v_3-v_2 的流量為 1。割的淨流量 $f(S,T)$=3+5−1=7。

割的容量 $c(S,T)$：指在切割線切中的邊中，從 S 到 T 的邊的容量之和。最小割指容量最小的割。上圖中從 S 到 T 的邊 v_1-v_3 和邊 v_2-v_4 的流量分別為 8、13。割的容量 $c(S,T)$=8+13=21。

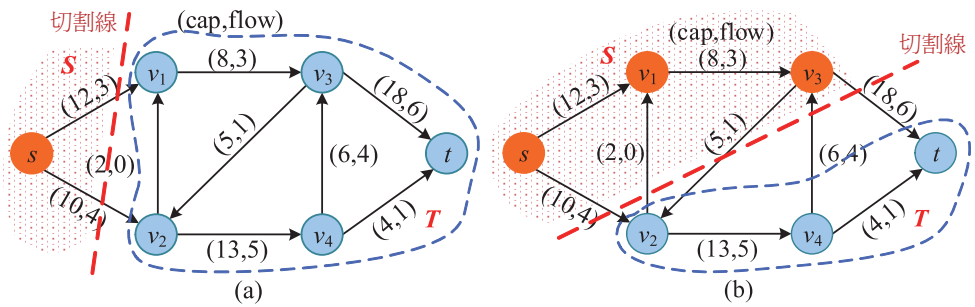
注意

在計算割的容量時不計算從 T 到 S 的邊的容量。

引理 1：若 f 是網路 G 中的一個流，CUT(S,T) 是網路 G 中的任意一個割，則 f 的值等於割的淨流量 $f(S,T)$。

$$f(S,T)=|f|$$

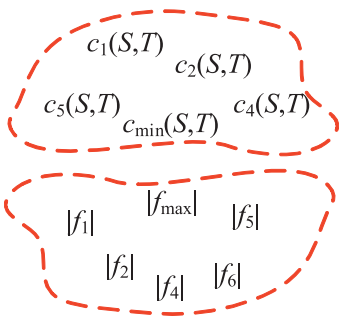
如下圖所示，圖 (a) 中割的淨流量 $f(S,T)$=3+4=7，圖 (b) 中割的淨流量 $f(S,T)$=4+1+6−4−0=7。畫出任意一個割，會發現所有割的淨流量 $f(S,T)$ 都等於 f 的值。



推論 1：若 f 是網路 G 中的一個流，CUT(S,T) 是網路 G 中的任意一個割，則 f 的值不超過割的容量 $c(S,T)$。

$$|f|\leqslant c(S,T)$$

所有流的流量都小於或等於割的容量，把流的流量和割的容量用圖表示出來，如下圖所示。



最大流最小割定理：若 f 是網路 G 中的最大流，CUT(S,T) 是 G 中的最小割，則最大流 f 的值等於最小割的容量 $c(S,T)$。

$$|f_{\max}| = c_{\min}(S,T)$$

若在很多問題上都需要求解最小割，則只需求解最大流。

訓練 1 最小邊割集

題目描述（HDU3251）：國王決定獎勵給你一些城市，除了可以從首都到達的城市，你可以任意選擇屬於自己的城市！為了避免選到可以從首都到達的城市，必須毀壞一些道路，每條道路都是單向的。有些城市是留給國王的，即使無法從首都到達這些城市，也不可以選擇。首都的編號為 1。你的最終收益等於你所選擇的城市的總價值減去毀壞道路的成本。

輸入：第 1 行為測試案例的數量 T（T≤20）。每個測試案例的第 2 行都以 3 個整數 n、m、f（1≤f<n≤1,000，1≤m<100,000）開頭，分別表示城市數量、道路數量和可以選擇的城市數量，城市編號為 1 ～ n，道路編號為 1 ～ m。接下來的 m 行，每行都為 3 個整數 u、v、w，表示從城市 u 到城市 v 的道路，成本為 w。在接下來的 f 行中，每行都為 2 個整數 u 和 w，表示可選城市 u 的價值為 w。

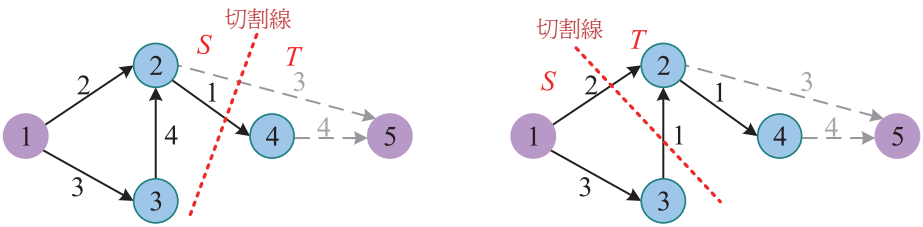
輸出：對於每個測試案例，第 1 行都輸出測試案例的編號和最大收益，第 2 行都輸出毀壞的道路數量 e，後面緊跟 e 個整數，表示已毀壞道路的編號，與它們的輸入順序相同。若有多種解決方案，則選擇其中任意一種都可以。

輸入範例	輸出範例
2 4 4 2 1 2 2 1 3 3 3 2 4 2 4 1 2 3 4 4 4 4 2 1 2 2 1 3 3 3 2 1 2 4 1 2 3 4 4	Case 1: 3 1 4 Case 2: 4 2 1 3

題解：在本題中可以將城市看作節點，將道路看作邊，每個節點、每條邊都有權值。在選擇一部分節點後必須毀壞一部分邊，保證從節點 1（首都）不可以到達這些節點，毀壞的邊就是將選擇的節點和節點 1 斷開的邊割集。收益 = 選擇的節點的權值之和 – 毀壞的邊的權值之和，要求輸出毀壞哪些邊。

建立網路：將節點 1 作為源點 s，增加匯點 $t=n+1$，將每條道路都連一條邊（單向道路），從可選城市到匯點連一條邊，容量為該城市的價值。可選城市的總價值減去最小割就是獲得的最大收益。若可選城市沒被選擇，則最小割的切割線一定會切中該城市與匯點 t 之間的邊，在將總價值減去最小割時會將該城市的價值去掉。若可選城市獲選，則最小割的切割線一定沒切中該城市所在節點與節點 t 之間的邊，在將總價值減去最小割時會保留該城市的價值。

透過輸入範例 1 建立的網路如下面左圖所示，求解出最大流（最小割）為 4，可選城市 2 和 4 的總價值為 7，獲得的最大收益為 7−4=3。因為城市 2 沒被選擇，所以在減去最小割時將城市 2 所在節點的權值 3 去掉了，只需毀壞一條邊（2-4），選擇城市 4 即可獲得最大收益。透過輸入範例 2 建立的網路如下面右圖所示，需要毀壞兩條邊（1-2、3-2），選擇城市 2 和 4，獲得的最大收益為 7−3=4。



在求解最小割（最大流）後檢查道路的邊，若一條邊的起點屬於集合 S，終點屬於集合 T，則這條邊就是要毀壞的邊。若用 Dinic 演算法求解最大流，則可以直接根據最後一次分層進行判斷，層次為真的節點屬於集合 S，其他節點屬於集合 T。若用 EK 或 ISAP 演算法求解最大流，則需要從源點出發，沿著 cap>flow 的邊進行深度優先搜尋，標記已經存取的節點，源點和已經存取的節點屬於集合 S，其餘節點和匯點屬於集合 T。範例程式碼請至本書第 v 頁的網址下載。

訓練 2 最小點割集

題目描述（HDU3491）：有 n（$2\le n\le 100$）個城市，由 m（$m\le 10{,}000$）條雙向道路連接各個城市。一群竊賊計畫盜竊 H 市的博物館。員警知道了這個計畫，計畫抓捕竊賊。竊賊目前在城市裡，員警想在從 S 市到 H 市的道路上抓捕他們。員警已