26-2

(a)

我們可以把問題換個方式想，原本的圖形每點都是一條長度 0 的 path，在不違反下面三個原則下從原本的圖型中選 edge 一條一條加下去：

1.每個 node 最多 out degree=1

2.每個 node 最多 in degree=1

3.選出來的 edge 不能讓圖形產生 cycle

則每多選一條 edge 就能讓 path 總數減少 1(類似 MST 的想法，每條 edge 加下去就會讓兩個集合合成一個，而且因爲遵守 3，因此每次都會減少，不會做白工)，而 path 上每點的特性是 in degree≤1 且 out degree≤1，每次的動作可以看成是把兩條 path 接起來，所以問題就變成，在不違反上面三個前提下，究竟最多能選出多少條這樣的 edge?

由於這題已經說圖形沒有 cycle，所以我們就不需要考慮上面 3 的發生，只需要考慮如何在不違反 1 和 2 的前提下選出最多條 edge，使得 path 數最少，稍微觀察上面 1 和 2 的要求以及 edge 的特性，我們知道一條 edge 加下去會讓某個 node in degree -1，同時也讓某個 node out degree -1，而每個 node 最多只能當 in 去和某個 node 的 out 搭配一次，且也只會 in 搭配 out，不會 in 搭配 in 或是 out 搭配 out，從這樣的觀察，我們可以發現這問題其實就是 max bipartite match，所有 node 分爲 in 和 out 兩邊，中間的連接關係就依照原本圖形的 edge 來連，最後加上起點 s 指向所有屬於 in 的 node，加上終點 t 讓所有 out 的 node 都指向 t，所有 edge 的流量都設定爲 1，然後跑課本的 max bipartite match algorithm，找出最大的流量，得到最大流量下選到的 edge 配對，假設找出來的流量是 f，那麼我們可以知道此圖形最少需要花 $|V|-f$ 個 path 去 cover 所有點，如果要知道這些 path 長怎麼樣，把那些選到的 edge 配對一條一條加回去原本圖形即可

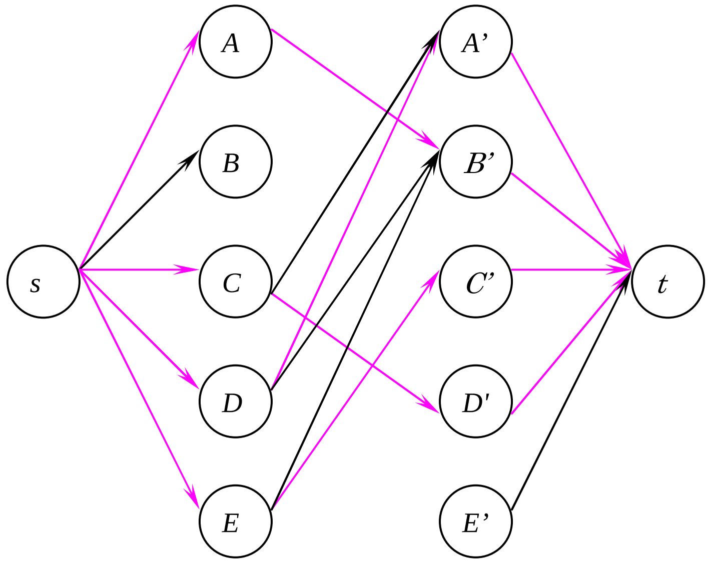
時間分析：

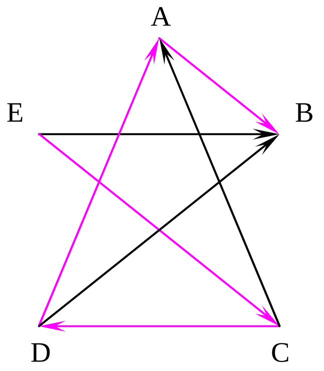
1.產生左側代表 in node	$O(V)$
2.產生右側代表 out node	$O(V)$
3.產生中間 in 指向 out 的 edge	$O(E)$
4.產生起點 s 和終點 t 並建立好 edge	$O(V)$
5.套用課本的 max bipartite match algorithm	$O(Ef)=O(EV)$
6.單純回答最少需要幾條 path	$O(1)$

7.要把 path 建構出來的話，找一條最後被選到的 edge(只要從 s 沿著最後殘留下來的 max flow 圖形走三步到 t)$O(1)$，全部可能選到 $O(V)$條，紀錄下有選 $O(1)$，總共 $O(V)$

8.要印出每條 path，反正印每條 path 時間不可能超過 $O(E)$，所以大不了 $O(EV)$

因此總時間是 $O(V+V+E+V+EV+1+V+EV)=O(EV)$





Example

(b)

　不行，因爲如果原圖形沒保證不會有 cycle 的話，我們就需要考慮前面原則 3 可能會違反，但是如果還是用同樣的方法，選出來的 edge 雖然依然會符合原則 1 和 2，但是卻無法確定會不會發生 3，因爲我們的方法只提供不違反 1 和 2 之下選出最多的 edge，但是即使今天選出來的 edge 是最多，其中違反 3 的卻佔了一大堆，不見得會比少選一些 edge，但是只有一些違反 3 來的好，因爲只要違反 3 的 edge 就不能提供減少 path 的效用，所以在有 cycle 的情況下直接套用這個方法是可能會有錯誤的。