2016 Algorithm HW3 Solutions

指導教授 ： 謝孫源 教授

助教： 盧緯 張耕華 楊順翔 許景添

Question 1(10pts)

Solution:

Let n be the total number of activites, $a_1, a_2, ..., a_n$.

```
GREEDY-ACTIVITY-SELECTOR-JMC(s,f)
    n = s.length
    A = {a_n}

    for m=n-1  to  1
        if f[m]<=s[k]                           //greedy step
            A={a_m} U A
            k=m

    return A
```

where n is the number of activities,
s is an n array and s[k] contains the starting time of a_k,
Assume s is monotonically increasing sorted array,
f is an n array and f[k] contains the finish time of a_k,

Question 1(10pts)

Solution:

- This algorithm iterates through the activities starting from the activit y with the latest Starting time. If the current activity has not finished before the last activity has started, then that activity is skipped and n ot added to optimal solution. However, if the candidate activity, ak, does finish before the last one starts, then that activity is added to th e solution. This is the greedy step and we know that the first activity with the latest starting time is going to be chosen before all the other ones because the array of activities is sorted in increasing order. In o rder for this approach to yield an optimal solution, it is sufficient to p rove that any activity with the latest starting time belongs to a maxi mum-size subset of mutually compatible activities of Sk.

- **Claim** : Consider any nonempty subproblem Sk and let am be an acti vity in Sk with the last starting time. Then am is included in some ma ximum-size subset of mutually compatible activities of Sk.

Question 1(10pts)

Solution:

Proof: Let a_i be an activity with starting time s_i and final time f_i. Let $\{a_1, a_2, ..., a_n\}$ be a set of activities monotonically increasing based on their starting time. That is, $s_1 \leq s_2 \leq s_3 \leq \ldots \leq s_n$. Let A_k be a maximum-size subset of mutually compatible activities S_k, and let a_j be the activity in A_k with the latest starting time.

Case 1: $a_j = a_m$

Then, since $a_j \in A_k$, a_j is in some maximum-size subset of mutually compatible activities of S_k

Case 2: $a_j \neq a_m$

Then set $A'_k = A_k - \{a_m\} + \{a_j\}$. Since A_k is some maximum-size subset of mutually compatible activities in S_k, then $f_1 \leq f_2 \leq f_3 \leq ... \leq s_m$. Since a_j and a_m are both activities with the latest starting time in S_k, $s_m = s_j$.Then we have that $f_1 \leq f_2 \leq f_3, ... \leq s_m = s_j$ and $|A'_k| = |A_k|$. Necessarily, A'_k must be a maximum-size subset of mutually compatible activities. Since $a_j \in A'_k$, a_j is in some maximum size subset of mutually compatible activities of S_k

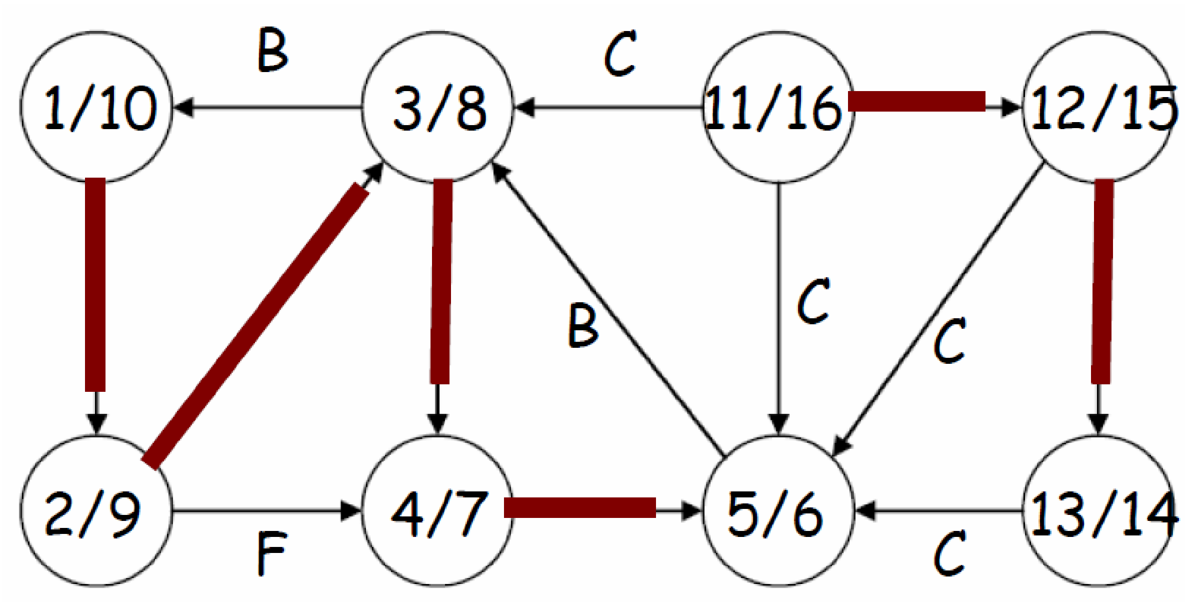
Question 2(10pts)

Solution:

- 兩個可以用 greedy 的問題 各 1 分
- 兩個不可以用 greedy 的問題 各 1 分
- 解釋差異 6 分

Question 3(10pts)

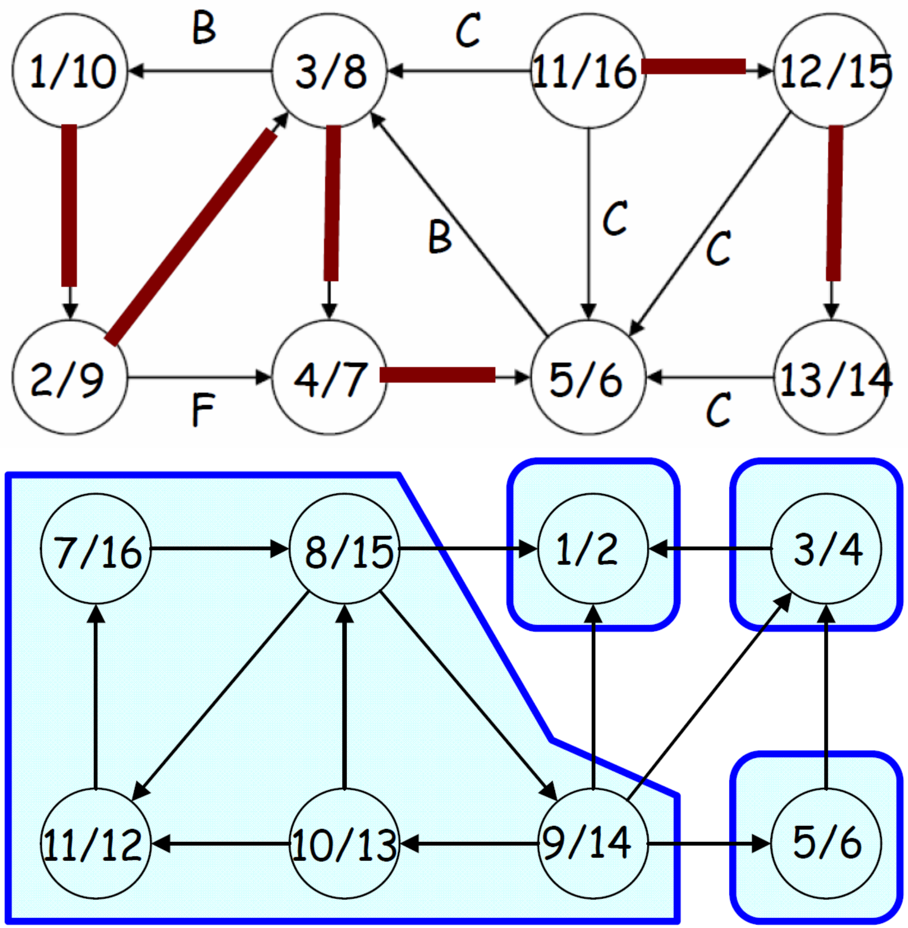
Solution:



- No, a directed graph is not acyclic because it has "back" edge s.

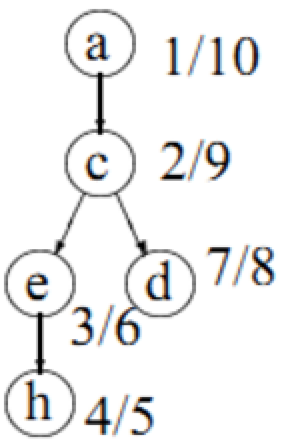
Question 4(10pts)

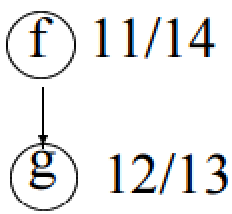
Solution:

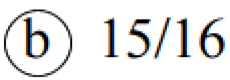


Question 5(10pts)

Solution:



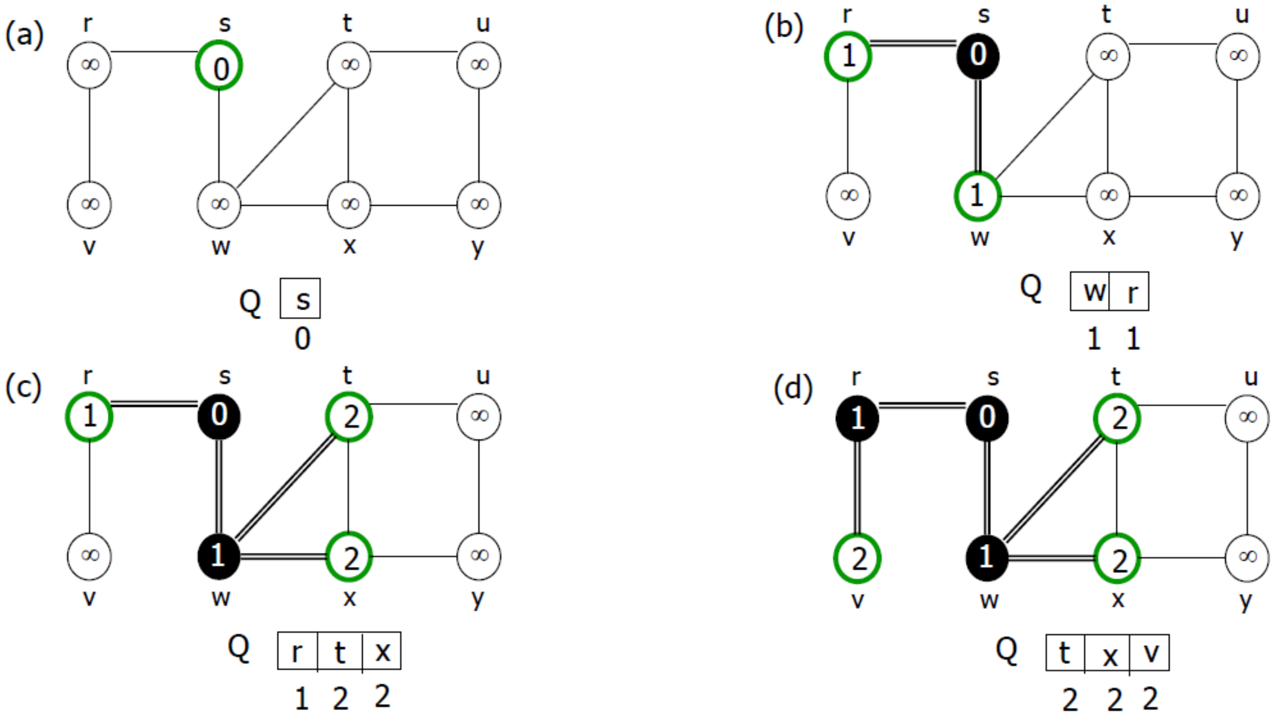




B, f, g, a, c, d, e, h

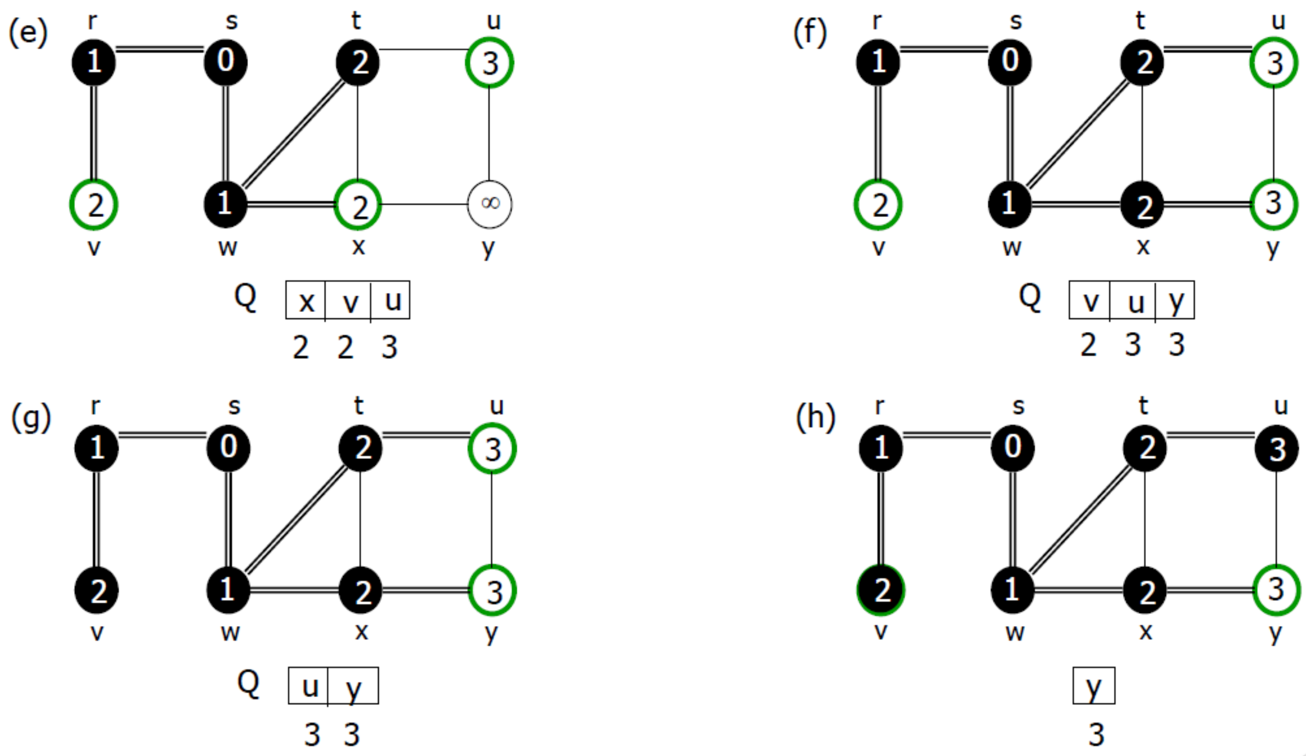
Question 6(10pts)

Solution:



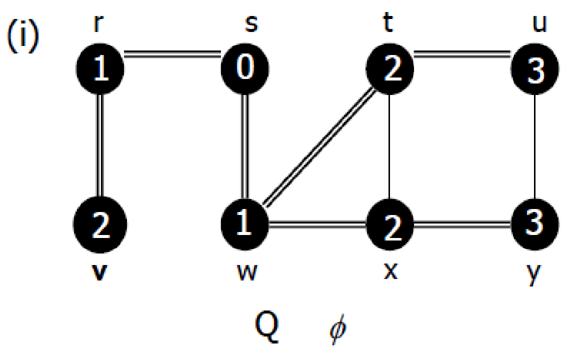
Question 6(10pts)

Solution:



Question 6(10pts)

Solution:

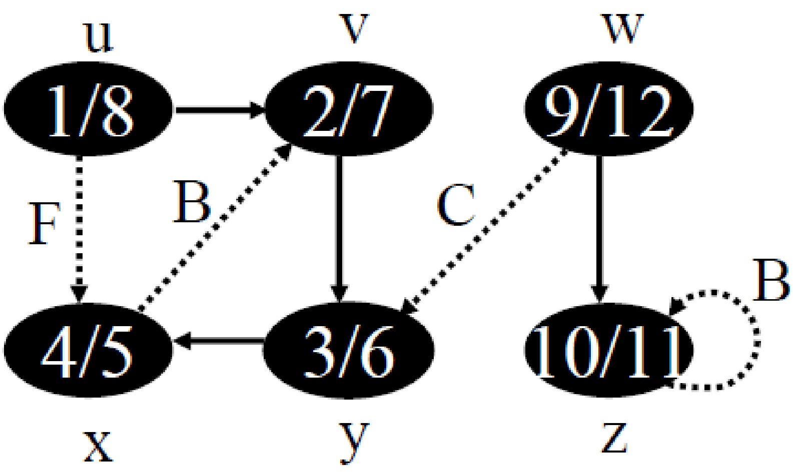


- 配分 (10%)
- 扣分方式
  - 寫得不夠詳細，依情況扣分。

Question 7(10pts)

Solution:

- False.
  The following is an example of DFS for directed graphs



Question 8(10pts)

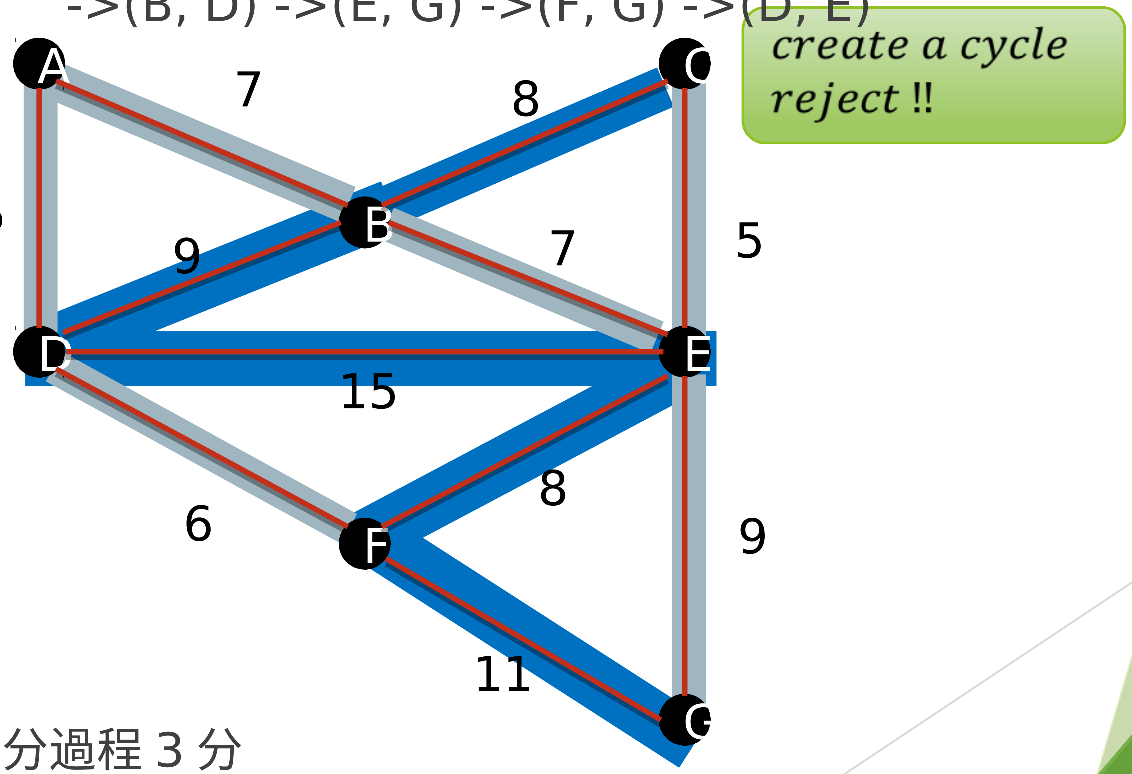
Solution:

- True.
  We can use Depth First Traversal to compute the finish times and then return the nodes in order of decreasing finishing times.
  We can also easily check for cycles as we do this and re port no sort is possible if a cycle exists.

Question 9(10pts)
Kruskal's algorithm (5pts)

- Sort the edges into non-decreasing order by weighted w

 ~~(A, D)~~->~~(C, E)~~->~~(D, F)~~ ->~~(A, B)~~ ->(B, E) ->(B, C) ->(E, F) ->(B, D) ->(E, G) ->(F, G) ->(D, E)



- 結果 2 分過程 3 分

Question 9(10pts) cont.
Multiple choices(5pts)

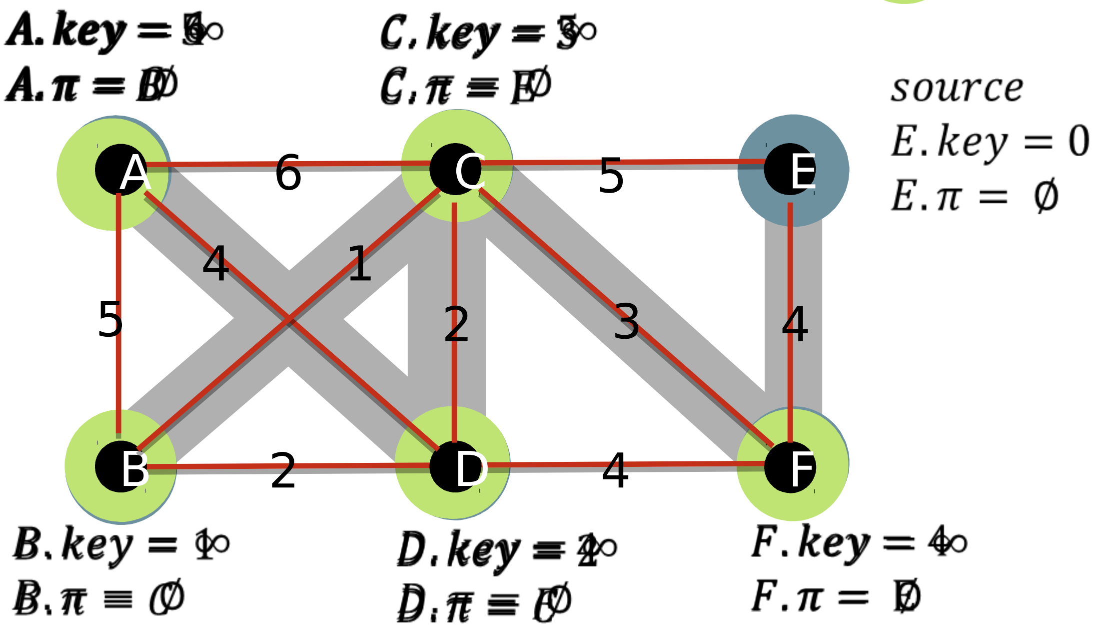
Solution:

- T (I) Because it chooses the smallest weight edge that does not create a cycle in the MST at each step
- F (II) Prim’ s algorithm
- F (III) non-decreasing
- T (IV) [CH23, p21]
- T (V) [CH23, p21]

Question 10(10pts) cont.
Prim’s algorithm (5pts)

Solution:

- s



Question 10(10pts) cont.
Multiple choices(5pts)

Solution:

- F (I) Because it takes the smallest key value in the node that holds in one data structure at each step
- T (II)
- F (III) non-decreasing, and this greedy strategy is used for Kruskal's algorithm
- T (IV)
- T (V) $O(VlgV + ElgV) = O(ElgV)$

For connected graph,

$\because\ |E| \geq |V| - 1$

$\therefore O(V) = O(E)$

$O(VlgV + ElgV) = O(2 * ElgV) = O(ElgV)$