EL9343 Data Structure and Algorithm
Homework 5

Qiming Zhang NetID: qz718

Xiaojie Zha NetID: xz1776

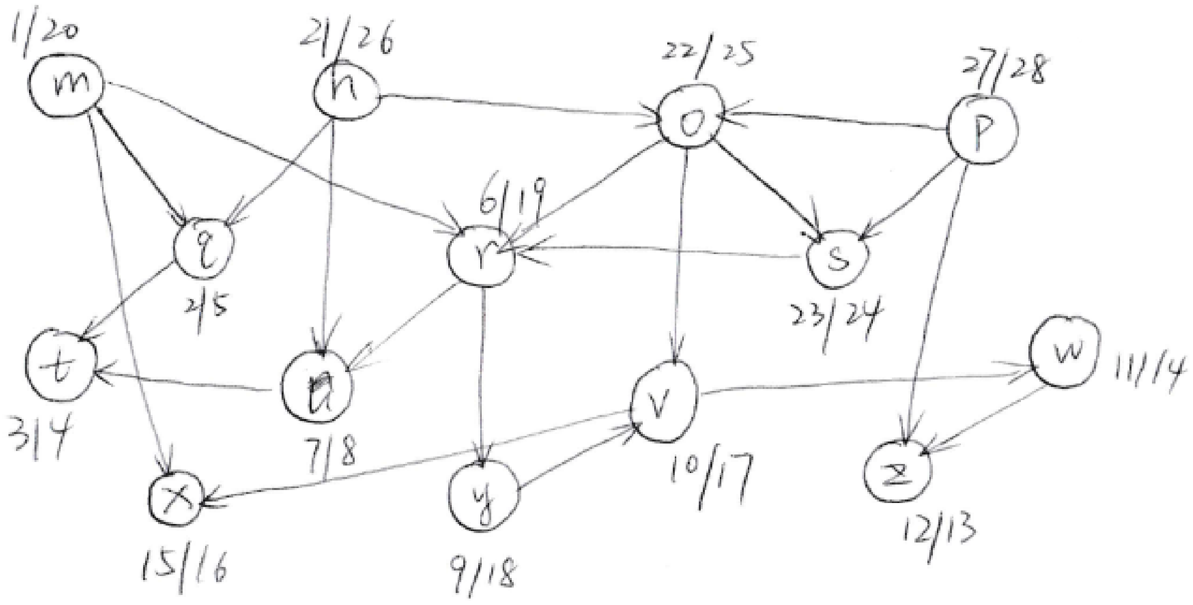
Yuhan Zhang NetID: yz2903

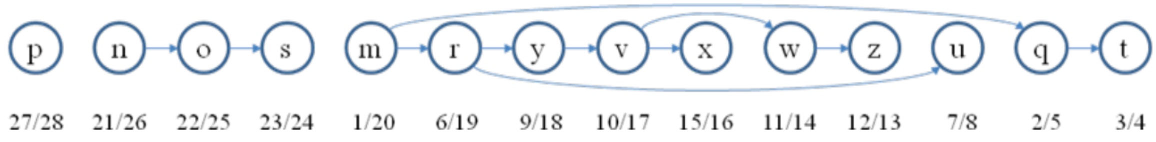
Yixuan Li NetID: yl3768

Yiyan Zhang NetID: yz3050

December 1, 2016

1 problem 1





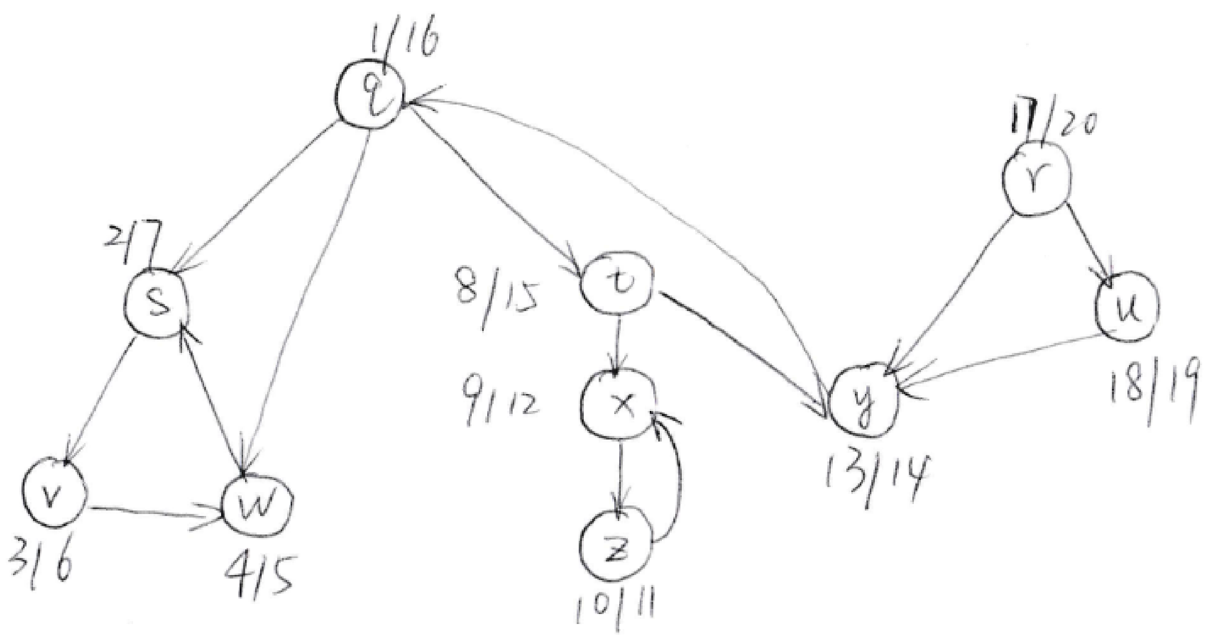
The topological order is p, n, o, s, m, r, y, v, x, w, z, u, q, t.

2 problem 2

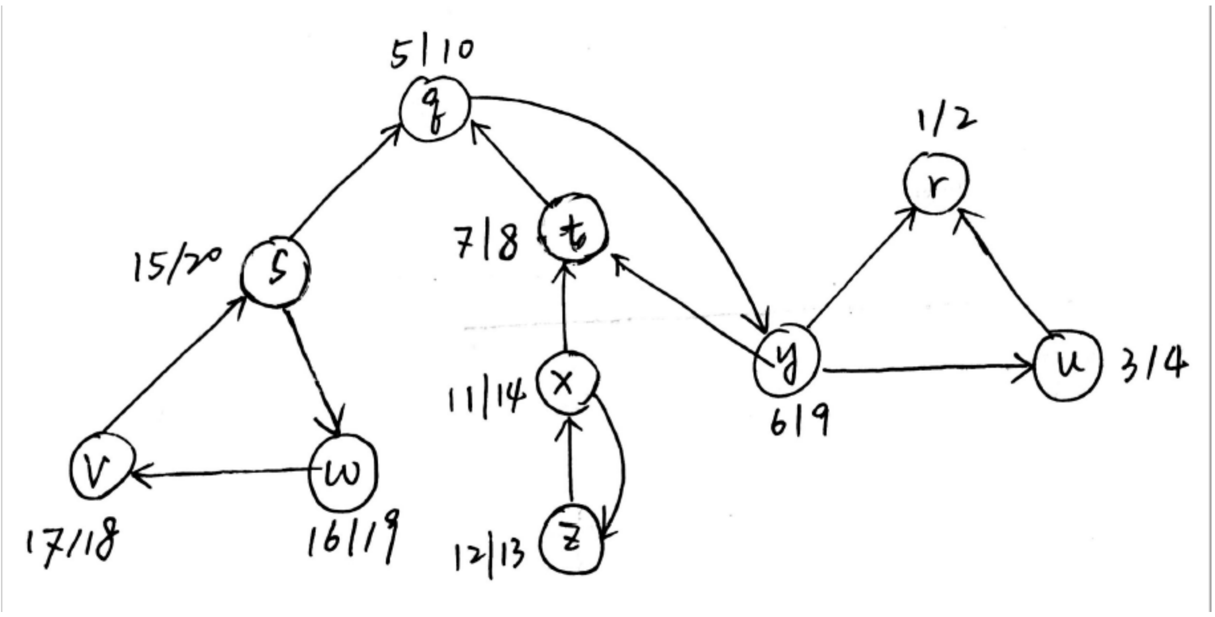
```
for each node v, we set path[v] = 0
set path[t] = 1 //t is the destination
call topological sort
change it to the reverse order
for each node v:
    for each successor w of v:
        set path[v] = path[v] + path[w]
return path[s] // s is the origin
```

3 problem 3

The output finish time of line 1 is



After we calculate $DFS(G^T)$,



strongly-connected components are {r}, {u}, {q, y, t}, {x, z}, {s, w, v}

4 problem 4

length i	1	2	3	4
price p_i	1	5	8	9

When i = 3, the density is the largest. The greedy result is 3 and 1 and yields 9 profit. But it is not the optimal solution, which is 2 and 2 and yields 10 profit.

5 problem 5

```
MODIFIED-CUT-ROD(p,n,c)
let r[0..n] be a new array
r[0] = 0
for j = 1 to n
        q = p[j]
        for i = 1 to j-1
                q = max(q, p[i]+ r[j-i]-c)
        r[j] = q
return r[n]
```

We add cost c into the inner for loop. We also have to handle the case in which we make no cuts (when i equals j). We modify the inner for loop to run from i to j-1 instead of to j.The assignment q = p[j] is for the case of no cuts.

6 problem 6

The LCS is $< 1, 0, 0, 1, 1, 0 >$, Or It can be $< 1, 0, 1, 0, 1, 0 >$

7 problem 7

```
PRINT_LCS(c, x, y, i, j)
    if i = 0 || j = 0
        return
    if x[i] = y[j]
        PRINT_LCS(c, x, y, i-1, j-1)
        print x[i]
    elif c[i-1, j] >= c[i, j-1]
        PRINT_LCS(c, x, y, i-1, j)
    else
        PRINT_LCS(c, x, y, i, j-1)
```

8 problem 8

```
DYNAMIC_ACTIVITY_SELECTOR(S):
    initialize c[i,j] = 0
    for i = 1 to n
        do for j = 2 to n
            do if i >= j
                then c[i,j] = 0
            else
                for k = i+1 to j-1
                    do if c[i,j] < c[i,k] + c[k,j] + 1
                        then c[i,j] = c[i,k] + c[k,j] + 1
                            s[i,j] = k
```

We use dynamic programming from bottom to the top to solve this problem. Since there are three layers of for loop, the running time is $O(n^3)$

9 problem 9

```
GREEDY–ACTIVITY–SELECTOR(s , f)
    n = length[s]
    A = {an}
    i = n
    for m = n−1 to 1
        do if fi >= si
            then A = A U {am}
                i = m
    return A
```

The proof of optimal solution is pretty similar to choose the earliest finished activity. We both need to proof the optimal substructure and greedy-choice property. Again we could also use contradiction to proof greedy-choice property and optimal substructure.

10 problem 10

Use greedy strategy. We take the item with maximum value. And it has the smallest weight and the largest ratio of value/weight. It's not possible to take any other item from the rest of the items since the largest value one is also the lightest one. Thus we could get the optimal solution by following this strategy.