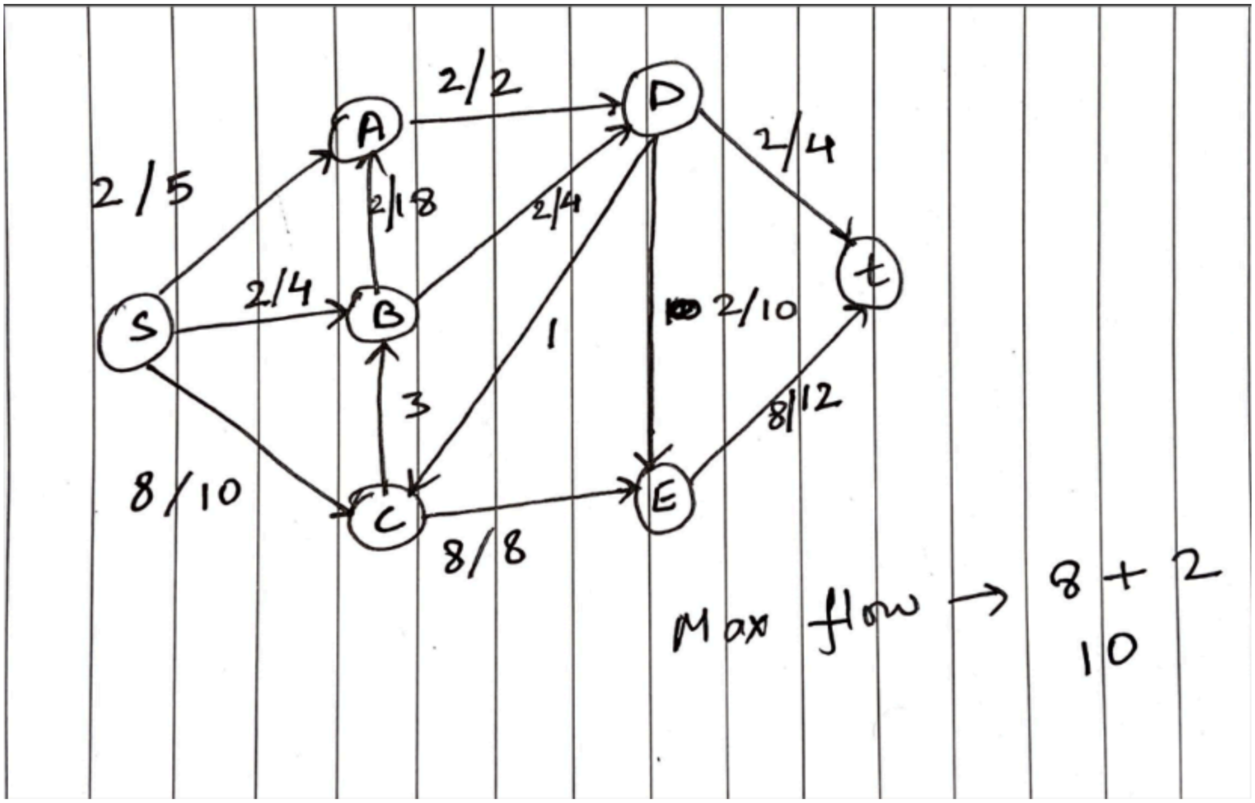
Friday, February 18, 2022

Assignment 3

CSCI 411

1.

a. & b.



c. If we cut the original graph in half, on the left side we are left with S, A, B, and C. S -> A == 5, S -> B == 2, C -> B == 3. If we add all these flows we get 10. And the flow from S -> C == 10. Hence, from above we can see that the max flow matches the cut.

2.

a. Given two sequences A and B,

A = “aggtab” and B = “gxtxayb”

Now if the last characters may or may not match. If the match then increment the length of the result by one and rprocess m - 1 and n - 1, m and n sizes of A and B respectively. If the last characters do not match, then the maximum between the maximum [m - 1][n] and [m][n - 1].

b.

```
findLCS(A, B):
    L[A.size + 1][B.size + 1]

    for (i = 0…..A.size):
        for (j = 0…..B.size):
            if (i is 0 or j is 0):
                L[i][j] = 0
            elif (A[i - 1] equals B[j - 1]):
                L[i][j] = L[i - 1][j - 1] + 1
            else:
                if L[i - 1][j] greater than L[i][j - 1]:
                        L[i][j] = L[i - 1][j]
                else:
                        L[i][j] = L[i][j - 1]
    return L[m][n]
```

c. Asymptotic run time is O(m*n), where m is the size of A and n is the size of B.

3.

a. For a given amount we need to find a way so that the denominations used to construct the amount is minimized.

For example, given coins = {1, 5, 6, 8} and target = 11. We can use one 6 and one 5 to build 11.

5 + 6 = 11

Hence, the vector returned/result should be {0, 1, 1, 0}.

b. We can use a table to log values of multiple combinations and at the end use that column to identify the minimum coins required.

For ex -

	0	1	2	3	4	5	6	7	8	9	10	11
1	0	1	2	3	4	5	6	7	8	9	10	11
5	0	1	2	3	4	1	2	3	4	5	2	3
6	0	1	2	3	4	1	1	2	3	4	2	2
8	0	1	2	3	4	1	1	2	1	2	2	2

```
makeChange(target, coins):
    Change {}
    buffer_t {}, buffer_r {}
    buffer_t(firstElement) = 0
    buffer_r(firstElement) = 0

    For i is 1 ….. target:
        buffer_t(i) = INT_MAX
        buffer_r(i) = -1

    For i is 0 ….. coinsSize:
            For j is 1 …… target:
                If j less than equal coins(i):
                    If buffer_t(j - coins(i)) is less than buffer_t(j):
                        buffer_t(j) = 1 + buffer_t(j - coins(i))
                        buffer_r(j) = i;

    If buffer_r is updated:
            While buffer_rSizeLastIndex not equal 0:
                    Denomination = buffer_r(buffer_rSizeLastIndex)
                    changeIndex(Denomination)++
     Return Change
```

c. The asymptotic runtime for this algorithm is O(m*n), where m is the size of coins vector and n is the target.