CSE373 Assignment 1

Aditya Balwani
SBUID: 109353920

April 18, 2016

1 Optimal Network Transmission Combination

2 Mining For Gold

At each position we can either go up or right. So the maximum gold value that can be can be found is the maximum value out of the max gold value obtained by going up and the max gold value obtained by going right plus the gold value of the current spot. If we can't go up then the max value is obtained by going right and if we can't go right then max value is from going up. At the top right corner the max value is the value of the position itself. At the end we return the gold vale of T[0,0] where

$$T[i,j] - \begin{cases} Gold[i,j] & \text{if } (i == M-1 \wedge j == N-1) \\ Gold[i,j] + T[i,j+1] & \text{if } (i == M-1) \\ Gold[i,j] + T[i+1,j] & \text{if } (j == N-1) \\ Gold[i,j] + max(T[i,j+1], T[i+1,j]) & \text{otherwise} \end{cases}$$

The asymptotic running time is O(mn)
The asymptotice space requirement is O(mn)

Psuedocode:

3 Maximum Product

4 String Decomposition

Lets say a and b are the 2 strings and m and n are the length of Our recursive function takes in all 3 strings: a, b and c. If a and b are both of length 0 then we return false. If the first lett of c is equal to the first letter of a then we we return the value returned by the recursive call containing a[1:n],b and c[1:m+n] . If not then we compare the first letter of b and c and if equal then the make the recursive call with a, b[1:n], c[1:m+n]. If that is also unequal then we return false.

$$T[i,j] \begin{cases} \text{false} & \text{if } m \text{ and } n \text{ are both 0} \\ T[i-1,j] & \text{if } a[0] == c[0] \\ T[i,j-1] & \text{if } b[0] == c[0] \\ \text{false} & \text{otherwise} \end{cases}$$

The asymptotic running time is O(mn)
The asymptotice space requirement is O(mn)

Psuedocode:

```
def isStringDecomposition(a, b, c, i, j):
    if(len(a) == 0 && len(b) == 0):
        T[0,0] = False
        return false
    if(a[0] == c[0]):
        if(T[i-1, j] != None):
            return T[i-1, j]
        else:
```

```
            success = isStringDecomposition(a[1:i], b, c[1:i+j], i-1, j)
            T[i-1, j] = success
            return success
    elif(b[0] == c[0]):
        if(T[i, j-1] != None):
            return T[i, j-1]
        else:
            success = isStringDecomposition(a, b[1:j], c[1:i+j], i, j-1)
            T[i, j-1] = success
            return success
    T[i, j] = False
    return False
```

5 Two salesmen

We get the following recursive structure to this question.
Let $C = distance(A)/2$
Let T[i] be the cost of splitting the list of cities A[0]...[i-1] into 2 arrays the we have

$$T[i] = min_{t \in [0,i]} \sum_{l=t}^{i-1} |distance(l) - C|$$