Kent Mark

3/30/21

Com S 311 – Homework 3

Homework 3

1. I do not know how to answer this question.

2. If this function returns 1 Riley wins, otherwise Morgan wins

    integer 2d array arr[integer variable n + 1][integer variable m + 1];

    integer function func(integer variable n, integer variable m) {

    //n = red cookies, m = green cookies

    if(amount of red cookies is less than 0 or amount of green cookies is less than 0) {

    return 1;
    }

    if(amount of red cookies is equal to 0 or amount of green cookies is equal to 0) {

    return 0;
    }

    if(amount of red cookies is equal to 1 or amount of green cookies equal to 1) {

    return 0;
    }

    if(integer 2d array arr[n][m] is equal 1) {

    return integer 2d array arr[n][m];
    }

    //make a recursive call to consider all three scenarios
    return integer 2d array arr[n][m] = (func(n – 3, m - 3) and func(n – 2, m – 4)) or (func(n – 4, m – 2) and func(n – 3, m – 3));
    Time complexity of this algorithm is $O(n^2)$

3. LCS is a brute force method which has a time complexity of $O(n^2)$. Algorithms with this time complexity can be optimized using dynamic programming which makes use of two properties: optimal substructure and overlapping subproblems. To show that LCS makes use of optimal substructure:

   Let the lengths of X and Y be X[0…A – 1] and Y[0…B – 1] with lengths A and B respectively. If the last characters of the two sequences match then 1 is added to the diagonal element.

   For example:

   if X[A – 1] == Y[B – 1] then B(X[0…A – 1], Y[0…B – 1] = 1 + B(X[0…A – 1], Y[0…B – 1])

   If they don't match then find the max of the left and top element of that particular element

   For example:

   if X[A – 1] != Y[B – 1] then MAX (B(X[0…A – 2], Y[0…B – 1], B(X[0…A – 1], Y[0…B – 2]

   This proves that LCS utilizes optimal substructure due to the fact that the main problem can be solved by first solving its subproblems.

4. The statement in this problem is true. I know this because if it were not true then that would mean there is a vertex v where the minimum spanning tree does not use the edge e with the least weight. As such, in that scenario, e would be added to the minimum spanning tree and v would be removed, which we have already established in the problem statement has a higher weight. This creates a logical contradiction which is why I say that this problem has to be true.