David Hatcher

COT4400

Dr. Hendrix

8/30/2020

Homework 1

1. Prove F(n) > 1.5^(n) for all n >= 11.

   Base Case 1: n = 11, F(11) = 89 > 1.5^(n) = 86.4975

   Base Case 1: n = 12, F(12) = 144 > 1.5^(n) = 194.6195

   Assume: F(k) > 1.5^k, for some arbitrary number k

   Suppose: F(k + 1) > 1.5^(k + 1)

   F(k + 1) = F(k) + F(k – 1)

   Since F(k) > 1.5^(K) is true based on our assumption, then we look at F(k – 1)

   F(k – 1) > 1.5^(k – 1)

   Which comes out to

   F(k) + F(k – 1) > 1.5^(k) + 1.5^(k – 1)

   Since we know that F(k) > 1.5^(k) is true, and subsequently F(k) > 1.5^(k – 1) is also true.

   And we know that F(k – 1) > 1.5^(k – 1) is true based on our assumption, then we also know that F(k + 1) = F(k) + F(k – 1) > 1.5^(k) + 1.5^(k – 1) is also true

   Now suppose we have some number x, where 12 <= x <= k

   Assume: F(x) > 1.5^(x)

   F(x + 1) > 1.5^(x + 1), where the minimum value for x + 1 is 13 and the maximum value is k + 1

   Since F(x + 1) = F(x) + F(x – 1), which we know will always be greater than 1.5^(x + 1) from the proof of F(k + 1) > 1.5^(k + 1) then we know that for any value, n, between 11 and k, F(n) > 1.5^(n) will always be true.

2. Base Case: For some arbitrary numbers x1 and y1.
   Suppose: DiffSwap is correct for all values x1 and y1.

   Line 2:

   x2 = x1 – y1

   Line 3:

   y3 = y1 + x2

Line 4:

x4 = y3 – x2

Line 5:

Return x4 and y3

We can argue that this algorithm is correct because it returns x4 which is equal to (y1 + (x1 – y1)) – (x1 – y1) = y1 + x1 – y1 – x2 + y1 = y1. The algorithm also returns y3 which is equal to y1 + x1 – y1 = x1. So this does in fact return the value for y1 as x and the value for x1 as y.

3. Prove that recursive IsSubstring for true for all values of n.

- Base Case: n <= m
- Line 2:
  - If Statement: Proof By Cases
    - Case 1: n < m
      - Line 3
        - The algorithm would terminate and return false
        - We can argue that the algorithm is correct as it terminates and since a string who's length is less than that if the string we're looking for inside can't contain the string we're looking.
    - Line 4
      - Case 2: n = m
        - Line 5
        - Case 3: a = b
          - The algorithm would terminate and return true
          - We can argue that the algorithm is correct as b is a substring of a and the algorithm returns true and terminates
        - Line 7
        - Case 4: a != b
          - The algorithm would recursively call it self again with n-1 as the new value for n which would be less than m.
          - The algorithm would then terminate and return false because of our Case 1 statement. Which we can argue is correct based on the same information from Case 1.
- Base Case: n > m
- Suppose that RIS(Recursive IsSubstring) is correct for all words of size k for some arbitrary value and k > m)
- Let a be some string with length of k + 1 and b is a string with length of m which is some arbitrary constant
- Line 2: if statement will evaluate as false and continue to line 4
  - Line 4
  - Case 5: The substring b is contained in string a
    - Case 6: The substring b is at the start of string a

        - Line 5:
        - If b is at the start of a then the algorithm will terminate and return true
        - We can argue that this is correct as the algorithm terminated and returned true and b contained in a.
    - Case 7: The substring b is not at the start of string a
        - Line 4,5, and 6 will be skipped and the algorithm will continue at line 7
        - Line 7:
            - The algorithm will recursively call the algorithm with a[2..k+1] as a, k as n, b as b, and m as m.
            - The length of a, n, is size k and via the assumption that the that RIS is correct for all words of size k we can assume that if b is contained in a it will be processed correctly
- Case 8: The substring b is not contained in string a
    - If the substring b is not contained in substring a then recursively call itself, decrementing n, the algorithm will eventually reach the base case where $n <= m$ and return false based on the proof for case 1.