Lab11

Santiago Flores

1. Recursive trace for gcd(a,b) function

First call: gcd(8,6) return 2--> final answer

8%6=2 call-> gcd(6,2) gcd(b, a%b) return 2 ^

6%2=0 call->gcd(2,0) gcd(b, a%b) return 2 ^

Since b = 0 return a. a =2 each call returns 2 to get the final answer of 2

1. Correctness of the recursive function gcd(a,b)

Variant: a , b

Threshold: b = 0

Base Case: when b is zero return a

Recursive case: a is replaced with b and b is replaced by a%b gcd(a,b)--> gcd(b,a%b)

Since each iteration decreases b by using % the function will always find a smaller solution until

It reaches the base case, and it will always reach the base case of zero.

1. Time complexity of towers of Hanoi analysis

Since there are two recursive calls the first part of the equation can be denoted as

T(n) = 2T(n-1). The function also holds one no-recursive call which is just a constant k

Leaving the formula to be T(n) = 2T(n-1)+k using this formula we can find the time complexity

T(0) = k constant

T(1) = 2T(1-1)+k= 2T(0)+k = 2k+k

T(2) = 2T(2-1)+k =2T(1)+k = 4k+2k+k

T(3) = 2T(3-1)+k = 2T(2)+k = 8k+4k+2k+k

We see that each iteration is multiplied by 2 and can denote the complexity to be O(2^n)