**EE602: Algorithms**

**Homework #2 Due: 9/19**

**Hualiang Li**

1. Exercises 3.1-2

（n + a）b = The most significant term is nb, so it is O(nb). Also, it is obviously greater than nb, so it is (nb). Therefore, it is

1. Exercises 3.1-4

a. 2n+1 = 2 \* 2n we choose c = 2 in the definition of Big O notation, we get it is O(2n).

b. 22n = 2n\*2n. There is no constant c such that for n > c, 2n\*2n < c\*2n.

1. Exercises 4.3.-1

We have to prove there is a constant c such that T(n) <= c\*n2.

T(n) = T(n-1) + n <= c(n-1)2 = cn2-2cn+c+n, if I pick c = 1; then T(n) <= n2-n+1 <=n2 for n >=1. So, T(n) is O(n2).

1. Exercises 4.4-9

We can assume that , Otherwise 1- would <= 1/2. Thus, it is easy to see that the depth of recursion tree is logn, and each level is cn. We can guess the leave is .

So, T(n) = cn \* logn + = .

1. Give asymptotic upper and lower bounds for T(n). Assume T(n) is constant for n <=2.
2. T(n) = 2T(n/2) + n3

a = 2, b = 2. nlogba = n. So, f(n) = 3), 2(n/2)3 <= 2 n3, we can apply Master theory 3, So, T(n) = .

1. T(n) = T(9n/10) + n

a = 1, b = 10/9. nlogba = 1, f(n) = n, so f(n) is polynomially larger than nlogba. T(n) = .

1. T(n) = 16T(n/4) + n2

A = 16, b= 4, nlogba = n2, f(n) = n2 = . Apply Master theory 2, T(n) = .

1. T(n) = 7T(n/3) + n2

A = 7, b = 3, nlogba=n1.5 f(n) = n2, polynomially larger than n1.5, T(n) =

1. T(n) = 7T(n/2) + n2

a = 7, b = 2, nlogba=n2.6 f(n) = n2 polynomially smaller than nlogba, T(n) =

1. T(n) = 2T(n/4) + n1/2

A = 2, b = 4, nlogba=n0.5 f(n) = n0.5=, so, Apply master theory 2, T(n) =

1. T(n) = T(n-1) + n

T(n) = T(n-2)+n-1+n = T(n-3) + n-2 + n-1 + n = T(0) + 1 + 2 +…+n = (1+n)n/2, This is

1. T(n) = T(n1/2) + 1

T(n) = T(n1/4) + 1 + 1 = T(n1/m) + logm so, T(n) = (logn)