**CS430 Lecture 05 Activities**

Opening Questions

1. Define Big-O, Omega and Theta notation.

2. In your own words explain what a recurrence relation is, what do we use recurrence relations for, why do we solve recurrence relations?

Recurrence Relation Solution Approach - Guess and prove by induction

Guess (or are given Hint) at form of solution, prove it is the solution

* Using definition of BIG-O or θ
* Using Induction
  + Prove Base Case (if boundary condition given)
  + Assume true for some “n”
  + Prove true for a larger “n”

EXAMPLE

T(n) = 4T(n/2) + n guess T(n)=O(n3) ??

Assume T(k)<=ck3 for some k<n, use assumption with k=n/2, then prove it for k=n

T(n/2) <= c(n/2)3 merge with recurrence

T(n) <= 4 c(n/2)3 + n

T(n) <= c/2(n)3 + n

T(n) <= cn3 – (c/2(n)3 – n)

(c/2(n)3 – n) >0 if c>=2 and n>1

T(n) <= cn3 – (something positive)

T(n) <= cn3

T(n)=O(n3)

1. T(n) = 4T(n/2) + n3 guess T(n)=Θ(n3) ??

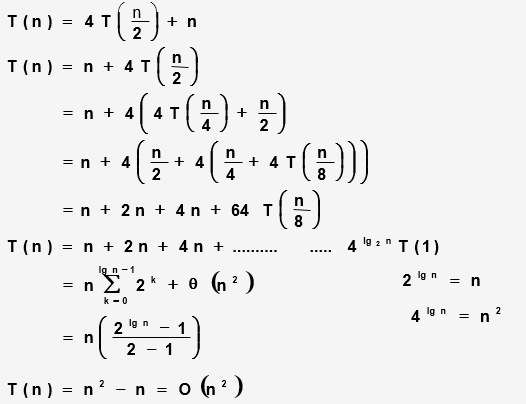
2. T(n) = 4T(n/2) + n guess T(n)=O(n2) ??

Recurrence Relation Solution Approach - Iteration Method (repeated substitution)

Convert the recurrence relation to summation using repeated substitution (Iterations)

* Keys to Iteration Method
  + # of times iterated to get T(1)
  + Find the pattern in terms and simplify to summation

EXAMPLE



3. T(n) = 3T(n/3) + lg n proof by iteration/repeated substitution

4. 

5. 

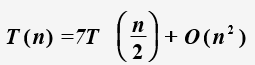
6. T(n) = T(n-1) + n

Recurrence Relation Solution Approach - Master Method

For solving recurrences of the form T(n) = aT(n/b) + f(n)

* Compare growth of f(n) to nlogba
  + Case1 f(n)<=cnlogba T(n) = Θ(nlogba)
  + Case2 c1nlogba<=f(n)<=c2nlogba T(n) = Θ(nlogba lg n)
  + Case3 f(n)>=cnlogba T(n) = Θ(f(n))

7. 

8.

9.