Lecture 3, April 10, 2001.

## Examples

$T(n) = 2T(n/2) + \Theta(n)$

$n^{\lg_b a} = n^{\lg_2 2} = n$

$\Theta(n)/n = \Theta(1) = \Theta(\lg^0 n) \Rightarrow$ case 2 $\Rightarrow T(n) = \Theta(n \lg n)$

---

Strassen's matrix multiplication

$T(n) = 7T(n/2) + \Theta(n^2)$

$n^{\lg_b a} = n^{\lg_2 7}$

$\frac{\Theta(n^2)}{n^{\lg_2 7}} \approx O(n^{-0.8}) \Rightarrow$ case 1 $\Rightarrow T(n) = \Theta(n^{\lg_2 7})$ ← Better than $n^3$ !!!

---

$T(n) = 4T(n/2) + n^3$

$\frac{n^3}{n^{\lg_2 4}} = n \Rightarrow$ case 3 $\Rightarrow T(n) = \Theta(n^3)$

(*Note*: need to check the additional condition $4(n^3/2) \le cn^3$)

38

## Does the method always apply ?

$T(n) = 4T(n/2) + n^2/\lg n$

$$\frac{n^2/\lg n}{n^{\lg_2 4}} = \frac{1}{\lg n} \neq \begin{cases} O(n^{-e}),\ e > 0 \\ \Theta(\lg^k n),\ k \ge 0 \\ \Omega(n^e),\ e > 0 \end{cases}$$

ANSWER: $\Theta(n^2 \lg\lg n)$ (proof by substitution)

Upper bound: $4T(n/2) + n^2 \Rightarrow$ case 2 $\Rightarrow \Theta(n^2 \lg n)$

Lower bound: $4T(n/2) + n^{2-e} \Rightarrow$ case 1 $\Rightarrow \Theta(n^2)$

39

## Back to algorithms - Quicksort

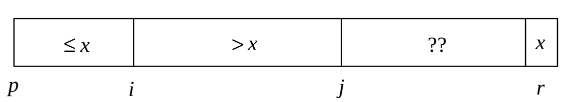
- **Quicksort**
  - » **Sort in place**
  - » **Very practical**
  - » **Divide & Conquer**

- **Algorithm:**
  - » **Divide into 2 arrays around the first element**
  - » **recursively sort each array**
  - » **merge/combine - trivial.**



## Partition routine

- ```
  Partition(A,p,r)
    x=A(p)
    i=p-1
    for j=p to r-1
      if A(j) <= x
        then i++, exchange A(i) and A(j)
    exchange A(i+1) and A(r)
    return (i+1)
  ```


  ```