

# CSE 421 Algorithms

Richard Anderson  
Lecture 13, Winter 2019  
Recurrences, Part 2

## Announcements

- Midterm
  - Wednesday, February 13, in class, closed book
  - Through section 5.2

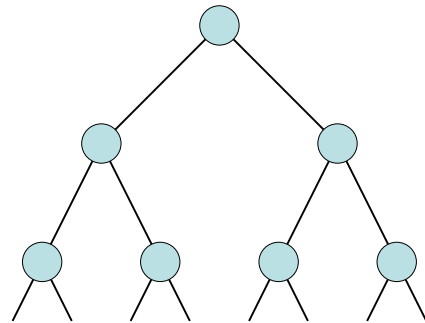


## Recurrence Examples

- $T(n) = 2 T(n/2) + cn$ 
  - $O(n \log n)$
- $T(n) = T(n/2) + cn$ 
  - $O(n)$
- More useful facts:
  - $\log_k n = \log_2 n / \log_2 k$
  - $k^{\log n} = n^{\log k}$

$$\sum_{i=0}^n x^i = \frac{1 - x^{n+1}}{1 - x}$$

## Unrolling the recurrence



## Recursive Matrix Multiplication

Multiply 2 x 2 Matrices:  
 $\begin{bmatrix} r & s \\ t & u \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} e & g \\ f & h \end{bmatrix}$

$$\begin{aligned} r &= ae + bf \\ s &= ag + bh \\ t &= ce + df \\ u &= cg + dh \end{aligned}$$

A  $N \times N$  matrix can be viewed as a  $2 \times 2$  matrix with entries that are  $(N/2) \times (N/2)$  matrices.

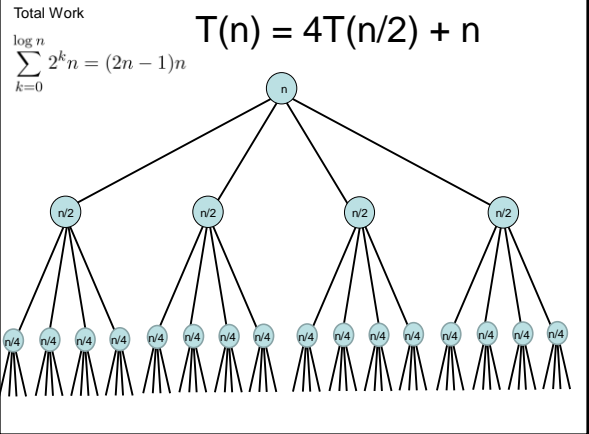
The recursive matrix multiplication algorithm recursively multiplies the  $(N/2) \times (N/2)$  matrices and combines them using the equations for multiplying  $2 \times 2$  matrices

## Recursive Matrix Multiplication

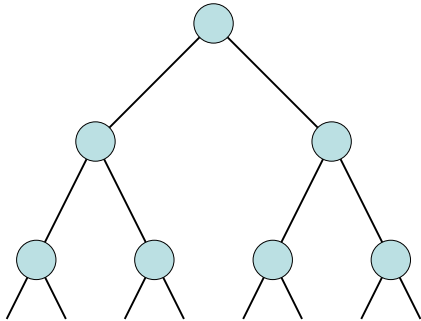
- How many recursive calls are made at each level?
- How much work in combining the results?
- What is the recurrence?

What is the run time for the recursive Matrix Multiplication Algorithm?

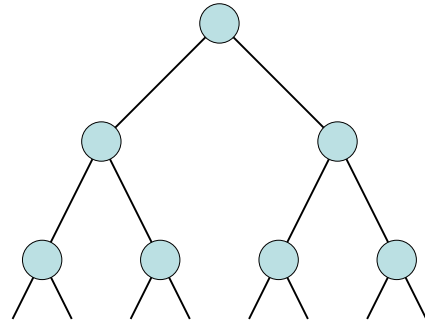
- Recurrence:



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



$$T(n) = 2T(n/2) + n^{1/2}$$



## Recurrences

- Three basic behaviors
  - Dominated by initial case
  - Dominated by base case
  - All cases equal – we care about the depth

## What you really need to know about recurrences

- Work per level changes geometrically with the level
- Geometrically increasing ( $x > 1$ )
  - The bottom level wins
- Geometrically decreasing ( $x < 1$ )
  - The top level wins
- Balanced ( $x = 1$ )
  - Equal contribution

### Classify the following recurrences (Increasing, Decreasing, Balanced)

- $T(n) = n + 5T(n/8)$
- $T(n) = n + 9T(n/8)$
- $T(n) = n^2 + 4T(n/2)$
- $T(n) = n^3 + 7T(n/2)$
- $T(n) = n^{1/2} + 3T(n/4)$

### Strassen's Algorithm

Multiply 2 x 2 Matrices:

$$\begin{vmatrix} r & s \\ t & u \end{vmatrix} = \begin{vmatrix} a & b \\ c & d \end{vmatrix} \begin{vmatrix} e & g \\ f & h \end{vmatrix}$$

Where:

$$p_1 = (b + d)(f + g)$$

$$p_2 = (c + d)e$$

$$p_3 = a(g - h)$$

$$p_4 = d(f - e)$$

$$p_5 = (a - b)h$$

$$p_6 = (c - d)(e + g)$$

$$p_7 = (b - d)(f + h)$$

$$r = p_1 + p_4 - p_5 + p_7$$

$$s = p_3 + p_5$$

$$t = p_2 + p_6$$

$$u = p_1 + p_3 - p_2 + p_7$$

### Recurrence for Strassen's Algorithms

- $T(n) = 7 T(n/2) + cn^2$
- What is the runtime?

### BFPRT Recurrence

$$T(n) \leq T(3n/4) + T(n/5) + 20n$$

What bound do you expect?