Data Structures and Algorithm Analysis (CSC317)

“Divide and conquer. Then, a merger.”

Divide and conquer (part 2)

Classical example: matrix multiplication

$$\begin{bmatrix} a_{11} a_{12} a_{13} a_{14} \\ a_{21} a_{22} a_{23} a_{24} \\ a_{31} a_{32} a_{33} a_{34} \\ a_{41} a_{42} a_{43} a_{44} \end{bmatrix} \begin{bmatrix} b_{11} b_{12} b_{13} b_{14} \\ b_{21} b_{22} b_{23} b_{24} \\ b_{31} b_{32} b_{33} b_{34} \\ b_{41} b_{42} b_{43} b_{44} \end{bmatrix} = \begin{bmatrix} c_{11} c_{12} c_{13} c_{14} \\ c_{21} c_{22} c_{23} c_{24} \\ c_{31} c_{32} c_{33} c_{34} \\ c_{41} c_{42} c_{43} c_{44} \end{bmatrix}$$

Classical example: matrix multiplication

$$\begin{bmatrix} a_{11}a_{12}a_{13}a_{14} \\ a_{21}a_{22}a_{23}a_{24} \\ a_{31}a_{32}a_{33}a_{34} \\ a_{41}a_{42}a_{43}a_{44} \end{bmatrix}\begin{bmatrix} b_{11}b_{12}b_{13}b_{14} \\ b_{21}b_{22}b_{23}b_{24} \\ b_{31}b_{32}b_{33}b_{34} \\ b_{41}b_{42}b_{43}b_{44} \end{bmatrix}=\begin{bmatrix} c_{11}c_{12}c_{13}c_{14} \\ c_{21}c_{22}c_{23}c_{24} \\ c_{31}c_{32}c_{33}c_{34} \\ c_{41}c_{42}c_{43}c_{44} \end{bmatrix}$$

$$c_{11} = a_{11}b_{11} + a_{12}b_{21} + a_{13}b_{31} + a_{14}b_{41};$$

...

Classical example: matrix multiplication

$$\begin{bmatrix} a_{11}a_{12}a_{13}a_{14} \\ a_{21}a_{22}a_{23}a_{24} \\ a_{31}a_{32}a_{33}a_{34} \\ a_{41}a_{42}a_{43}a_{44} \end{bmatrix} \begin{bmatrix} b_{11}b_{12}b_{13}b_{14} \\ b_{21}b_{22}b_{23}b_{24} \\ b_{31}b_{32}b_{33}b_{34} \\ b_{41}b_{42}b_{43}b_{44} \end{bmatrix} = \begin{bmatrix} c_{11}c_{12}c_{13}c_{14} \\ c_{21}c_{22}c_{23}c_{24} \\ c_{31}c_{32}c_{33}c_{34} \\ c_{41}c_{42}c_{43}c_{44} \end{bmatrix}$$

n by n matrix

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$$

- Run time?

Classical example: matrix multiplication

$$\begin{bmatrix} a_{11}a_{12}a_{13}a_{14} \\ a_{21}a_{22}a_{23}a_{24} \\ a_{31}a_{32}a_{33}a_{34} \\ a_{41}a_{42}a_{43}a_{44} \end{bmatrix}\begin{bmatrix} b_{11}b_{12}b_{13}b_{14} \\ b_{21}b_{22}b_{23}b_{24} \\ b_{31}b_{32}b_{33}b_{34} \\ b_{41}b_{42}b_{43}b_{44} \end{bmatrix}=\begin{bmatrix} c_{11}c_{12}c_{13}c_{14} \\ c_{21}c_{22}c_{23}c_{24} \\ c_{31}c_{32}c_{33}c_{34} \\ c_{41}c_{42}c_{43}c_{44} \end{bmatrix}$$

n by n matrix

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$$

- Run time? $O(n^3)$

Answer: Naïve implementation

Classical example: matrix multiplication

Square-Matrix-Multiply(A,B)

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$$

1. n = A.rows
2. Let C be a new n by n matrix
3. **For** i=1 to n
4.   **For** j=1 to n
5.   cij = 0
6.   **For** k=1 to n
7.   cij = cij+ aik bkj
8. **Return** C

$O(n^3)$

Classical example: matrix multiplication

- Run time?

Answer: Naïve implementation $O(n^3)$

Can we do better?

Classical example: matrix multiplication

$$\begin{bmatrix} a_{11} a_{12} a_{13} a_{14} \\ a_{21} a_{22} a_{23} a_{24} \\ a_{31} a_{32} a_{33} a_{34} \\ a_{41} a_{42} a_{43} a_{44} \end{bmatrix} \begin{bmatrix} b_{11} b_{12} b_{13} b_{14} \\ b_{21} b_{22} b_{23} b_{24} \\ b_{31} b_{32} b_{33} b_{34} \\ b_{41} b_{42} b_{43} b_{44} \end{bmatrix} = \begin{bmatrix} c_{11} c_{12} c_{13} c_{14} \\ c_{21} c_{22} c_{23} c_{24} \\ c_{31} c_{32} c_{33} c_{34} \\ c_{41} c_{42} c_{43} c_{44} \end{bmatrix}$$

- Divide and conquer? How?

Classical example: matrix multiplication

$$\begin{bmatrix} a_{11} a_{12} a_{13} a_{14} \\ a_{21} a_{22} a_{23} a_{24} \\ a_{31} a_{32} a_{33} a_{34} \\ a_{41} a_{42} a_{43} a_{44} \end{bmatrix} \begin{bmatrix} b_{11} b_{12} b_{13} b_{14} \\ b_{21} b_{22} b_{23} b_{24} \\ b_{31} b_{32} b_{33} b_{34} \\ b_{41} b_{42} b_{43} b_{44} \end{bmatrix} = \begin{bmatrix} c_{11} c_{12} c_{13} c_{14} \\ c_{21} c_{22} c_{23} c_{24} \\ c_{31} c_{32} c_{33} c_{34} \\ c_{41} c_{42} c_{43} c_{44} \end{bmatrix}$$

- Divide and conquer?
- Can’t break in half like array
- But can break into 4 pieces

Classical example: matrix multiplication

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ b_{21} & b_{22} & b_{23} & b_{24} \\ b_{31} & b_{32} & b_{33} & b_{34} \\ b_{41} & b_{42} & b_{43} & b_{44} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & c_{34} \\ c_{41} & c_{42} & c_{43} & c_{44} \end{bmatrix}$$

- Divide and conquer?
- Can’t break in half like array
- But can break into 4 pieces

Classical example: matrix multiplication

$$\left[\begin{array}{cc|cc} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ \hline a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{array}\right] \left[\begin{array}{cc|cc} b_{11} & b_{12} & b_{13} & b_{14} \\ b_{21} & b_{22} & b_{23} & b_{24} \\ \hline b_{31} & b_{32} & b_{33} & b_{34} \\ b_{41} & b_{42} & b_{43} & b_{44} \end{array}\right] = \left[\begin{array}{cc|cc} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ \hline c_{31} & c_{32} & c_{33} & c_{34} \\ c_{41} & c_{42} & c_{43} & c_{44} \end{array}\right]$$

- Divide and conquer?
- Can’t break in half like array
- But can break into 4 pieces

Example on the board…

Classical example: matrix multiplication

$$A_{11} = \begin{bmatrix} a_{11}a_{12} \\ a_{21}a_{22} \end{bmatrix}; A_{12} = \begin{bmatrix} a_{13}a_{14} \\ a_{23}a_{24} \end{bmatrix}; A_{21} = \begin{bmatrix} a_{31}a_{32} \\ a_{41}a_{42} \end{bmatrix}; A_{12} = \begin{bmatrix} a_{33}a_{34} \\ a_{43}a_{44} \end{bmatrix}$$

$$A = \begin{bmatrix} A_{11}A_{12} \\ A_{21}A_{22} \end{bmatrix}; B = \begin{bmatrix} B_{11}B_{12} \\ B_{21}B_{22} \end{bmatrix}; C = \begin{bmatrix} C_{11}C_{12} \\ C_{21}C_{22} \end{bmatrix}$$

$$C = AB$$

- Now each capital A;B;C is a whole square matrix (need not be a single element)

Classical example: matrix multiplication

$$A_{11} = \begin{bmatrix} a_{11}a_{12} \\ a_{21}a_{22} \end{bmatrix}; A_{12} = \begin{bmatrix} a_{13}a_{14} \\ a_{23}a_{24} \end{bmatrix}; A_{21} = \begin{bmatrix} a_{31}a_{32} \\ a_{41}a_{42} \end{bmatrix}; A_{12} = \begin{bmatrix} a_{33}a_{34} \\ a_{43}a_{44} \end{bmatrix}$$

$$A = \begin{bmatrix} A_{11}A_{12} \\ A_{21}A_{22} \end{bmatrix}; B = \begin{bmatrix} B_{11}B_{12} \\ B_{21}B_{22} \end{bmatrix}; C = \begin{bmatrix} C_{11}C_{12} \\ C_{21}C_{22} \end{bmatrix}$$

$$C = AB$$

- Now each capital A;B;C is a whole square matrix (need not be a single element)
- Write out what the $C_{11};C_{12};C_{21};C_{22}$ elements are in terms of A and B?

Classical example: matrix multiplication

$$A = \begin{bmatrix} A_{11} A_{12} \\ A_{21} A_{22} \end{bmatrix}; B = \begin{bmatrix} B_{11} B_{12} \\ B_{21} B_{22} \end{bmatrix}; C = \begin{bmatrix} C_{11} C_{12} \\ C_{21} C_{22} \end{bmatrix}$$

$$C = AB$$

$$C_{11} = A_{11}B_{11} + A_{12}B_{21}$$

$$C_{12} = A_{11}B_{12} + A_{12}B_{22}$$

$$C_{21} = A_{21}B_{11} + A_{22}B_{21}$$

$$C_{22} = A_{21}B_{12} + A_{22}B_{22}$$

Classical example: matrix multiplication

$$A = \begin{bmatrix} A_{11} A_{12} \\ A_{21} A_{22} \end{bmatrix}; B = \begin{bmatrix} B_{11} B_{12} \\ B_{21} B_{22} \end{bmatrix}; C = \begin{bmatrix} C_{11} C_{12} \\ C_{21} C_{22} \end{bmatrix}$$

$$C = AB$$

$$C_{11} = A_{11}B_{11} + A_{12}B_{21}$$

$$C_{12} = A_{11}B_{12} + A_{12}B_{22}$$

$$C_{21} = A_{21}B_{11} + A_{22}B_{21}$$

$$C_{22} = A_{21}B_{12} + A_{22}B_{22}$$

- How many recursive calls to matrix multiply?

Classical example: matrix multiplication

$$A = \begin{bmatrix} A_{11} A_{12} \\ A_{21} A_{22} \end{bmatrix}; B = \begin{bmatrix} B_{11} B_{12} \\ B_{21} B_{22} \end{bmatrix}; C = \begin{bmatrix} C_{11} C_{12} \\ C_{21} C_{22} \end{bmatrix}$$

$$C = AB$$

$$C_{11} = A_{11}B_{11} + A_{12}B_{21}$$

$$C_{12} = A_{11}B_{12} + A_{12}B_{22}$$

$$C_{21} = A_{21}B_{11} + A_{22}B_{21}$$

$$C_{22} = A_{21}B_{12} + A_{22}B_{22}$$

- How many recursive calls to matrix multiply?
  8 total multiplications = 8 recursive calls

Classical example: matrix multiplication

$$A = \begin{bmatrix} A_{11} A_{12} \\ A_{21} A_{22} \end{bmatrix}; B = \begin{bmatrix} B_{11} B_{12} \\ B_{21} B_{22} \end{bmatrix}; C = \begin{bmatrix} C_{11} C_{12} \\ C_{21} C_{22} \end{bmatrix}$$

$$C = AB$$

$$C_{11} = A_{11}B_{11} + A_{12}B_{21}$$

$$C_{12} = A_{11}B_{12} + A_{12}B_{22}$$

$$C_{21} = A_{21}B_{11} + A_{22}B_{21}$$

$$C_{22} = A_{21}B_{12} + A_{22}B_{22}$$

- Cost of combining recursions?

Classical example: matrix multiplication

$$A = \begin{bmatrix} A_{11} A_{12} \\ A_{21} A_{22} \end{bmatrix}; B = \begin{bmatrix} B_{11} B_{12} \\ B_{21} B_{22} \end{bmatrix}; C = \begin{bmatrix} C_{11} C_{12} \\ C_{21} C_{22} \end{bmatrix}$$

$$C = AB$$

$$C_{11} = A_{11}B_{11} + A_{12}B_{21}$$

$$C_{12} = A_{11}B_{12} + A_{12}B_{22}$$

$$C_{21} = A_{21}B_{11} + A_{22}B_{21}$$

$$C_{22} = A_{21}B_{12} + A_{22}B_{22}$$

- Cost of combining recursions?
  4 matrix additions, each matrix size $\frac{n^2}{4}$

Classical example: matrix multiplication

$$A = \begin{bmatrix} A_{11} A_{12} \\ A_{21} A_{22} \end{bmatrix}; B = \begin{bmatrix} B_{11} B_{12} \\ B_{21} B_{22} \end{bmatrix}; C = \begin{bmatrix} C_{11} C_{12} \\ C_{21} C_{22} \end{bmatrix}$$

$$C = AB$$

Divide: constant time

Conquer: $8T(\frac{n}{2})$ 8 matrix multiplications of size $\frac{n}{2}$

Combine: $\Theta(n^2)$ 4 matrix additions, each matrix size $\frac{n^2}{4}$

$$T(n) = 8T(\frac{n}{2}) + \Theta(n^2)$$

Classical example: matrix multiplication

$$A = \begin{bmatrix} a_{11} a_{12} \\ a_{21} a_{22} \end{bmatrix}; B = \begin{bmatrix} b_{11} b_{12} \\ b_{21} b_{22} \end{bmatrix}; C = \begin{bmatrix} c_{11} c_{12} \\ c_{21} c_{22} \end{bmatrix}$$

$$C = AB$$

Divide: constant time

Conquer: $8T(\frac{n}{2})$ 8 matrix multiplications of size $\frac{n}{2}$

Combine: $\Theta(n^2)$ 4 matrix additions, each matrix size $\frac{n^2}{4}$

Cost? Guess?

Classical example: matrix multiplication

Divide: constant time

Conquer: $8T(\frac{n}{2})$ 8 matrix multiplications of size $\frac{n}{2}$

Combine: $\Theta(n^2)$ 4 matrix additions, each matrix size $\frac{n^2}{4}$

Cost? Guess?

Answer: Bad news! $\Theta(n^3)$

Same cost as naïve multiplication!

Convince yourself why after we go through solving recurrences

Strassen's method

Clever way to compute only 7 matrix multiplications and therefore 7 recursions

By defining new matrices that are sums and differences of the original

$$T(n) = 7T(\frac{n}{2}) + \Theta(n^2)$$

Strassen's method

Clever way to compute only 7 matrix multiplications
and therefore 7 recursions

By defining new matrices that are sums and differences
of the original

$$T(n) = 7T(\frac{n}{2}) + \Theta(n^2)$$

We'll later be able to go back and see that this is good news!

$$\Theta(n^{\log_2 7})$$

Strassen's method

What is the clever solution? First define additions and subtractions of the submatrices…

$$S_1 = B_{12} - B_{22}$$

$$S_2 = A_{11} + A_{12}$$

$$\dots$$

$$S_{10} = B_{11} + B_{12}$$

These are 10 additions and subtractions…

$$10\left(\frac{n}{2}\right)^2 = \Theta(n^2)$$

Strassen's method

What is the clever solution? Next, matrix multiplications…

$$P_1 = A_{11}S_1$$

$$P_2 = S_2 B_{22}$$

$$P_3 = S_3 B_{11}$$

$$\dots$$

$$P_7 = S_9 S_{10}$$

- These are 7 multiplications (7 recursions instead of 8!)

$$7T(\frac{n}{2})$$

Strassen's method

What is the clever solution? Next, the C matrix elements are additions and subtractions of the 7 matrix multiplications…

$$C_{11} = P_5 + P_4 - P_2 + P_6;$$

$$C_{12} = P_1 + P_2$$

....

- Additions and subtractions

$$\Theta(n^2)$$

Strassen's method

What is the clever solution?

$$P_1 = A_{11}S_1$$

$$P_2 = S_2B_{22}$$

$$P_3 = S_3B_{11}$$

$$\ldots$$

$$P_7 = S_9S_{10}$$

- These are 7 multiplications (7 recursions instead of 8!)

  $7T(\frac{n}{2})$

- Elements of C are then additions and subtractions of the P

  $\Theta(n^2)$

Strassen’s method

How did Strassen come up with this??

Strassen's method

We'll just show one example that this actually works

$$C_{12} = P_1 + P_2 = A_{11}B_{12} - A_{11}B_{22} + A_{11}B_{22} + A_{12}B_{22}$$

Strassen’s method

We’ll just show one example that this actually works

$$C_{12} = P_1 + P_2 = A_{11}B_{12} - \cancel{A_{11}B_{22}} + \cancel{A_{11}B_{22}} + A_{12}B_{22} =$$
$$A_{11}B_{12} + A_{12}B_{22}$$

As in regular multiplication!

Goals

What kind of recurrences arise in algorithms and how do we solve more generally (than what we saw for merge sort)?

- More recurrence examples
- Revisit recursion trees more generally
- Master theorem as “cookbook recipe” for range of cases
- (Substitution method)

Matrix multiplication – if recursion helps or hurts not always intuitive

Naïve:

$$\Theta(n^3)$$

Simple divide and conquer:

$$T(n) = 8T(\frac{n}{2}) + \Theta(n^2) = \Theta(n^3)$$ No gain!

Strassen's divide and conquer method:

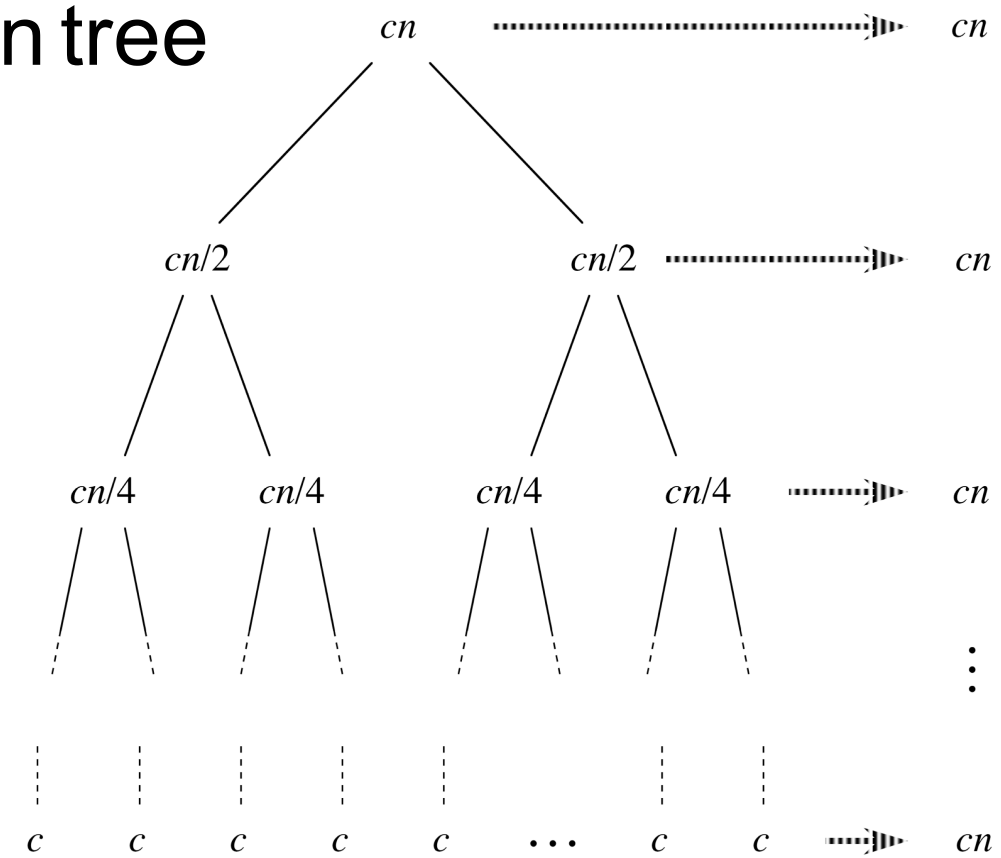
$$T(n) = 7T(\frac{n}{2}) + \Theta(n^2) = \Theta(n^{\log_2 7})$$

Matrix multiplication – if recursion helps or hurts not always intuitive

We'd like to better understand what determines cost of divide and conquer approaches

Merge sort: recursion tree

$$T(n) = 2T(\frac{n}{2}) + cn$$

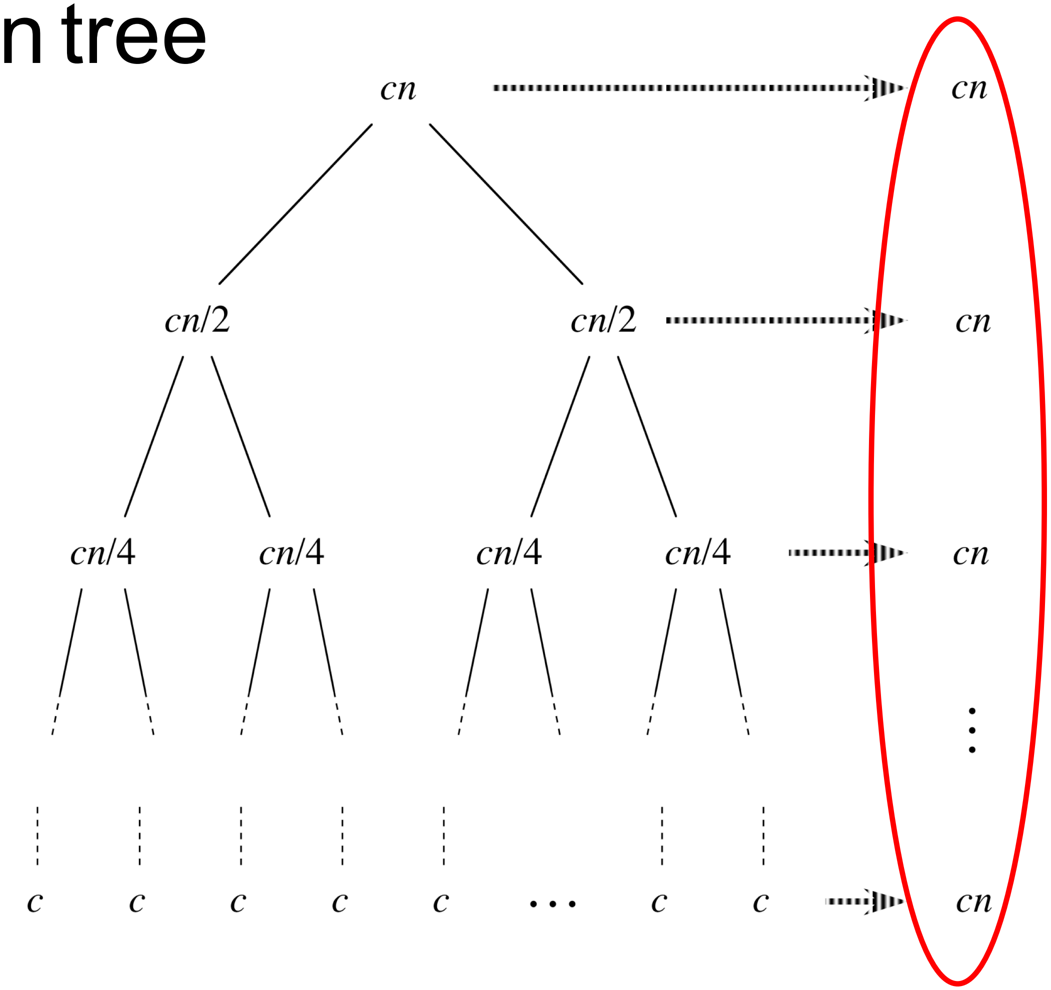


Number of levels: $k = \log_2(n)$

Work at each level: cn

Merge sort: recursion tree

$$T(n) = 2T(\frac{n}{2}) + cn$$



"Easy" because "work" the same at each level

Total: $cn\log_2(n)$

Recursion example

$$T(n) = 3T(\frac{n}{4}) + cn^2$$

$$T(n)$$

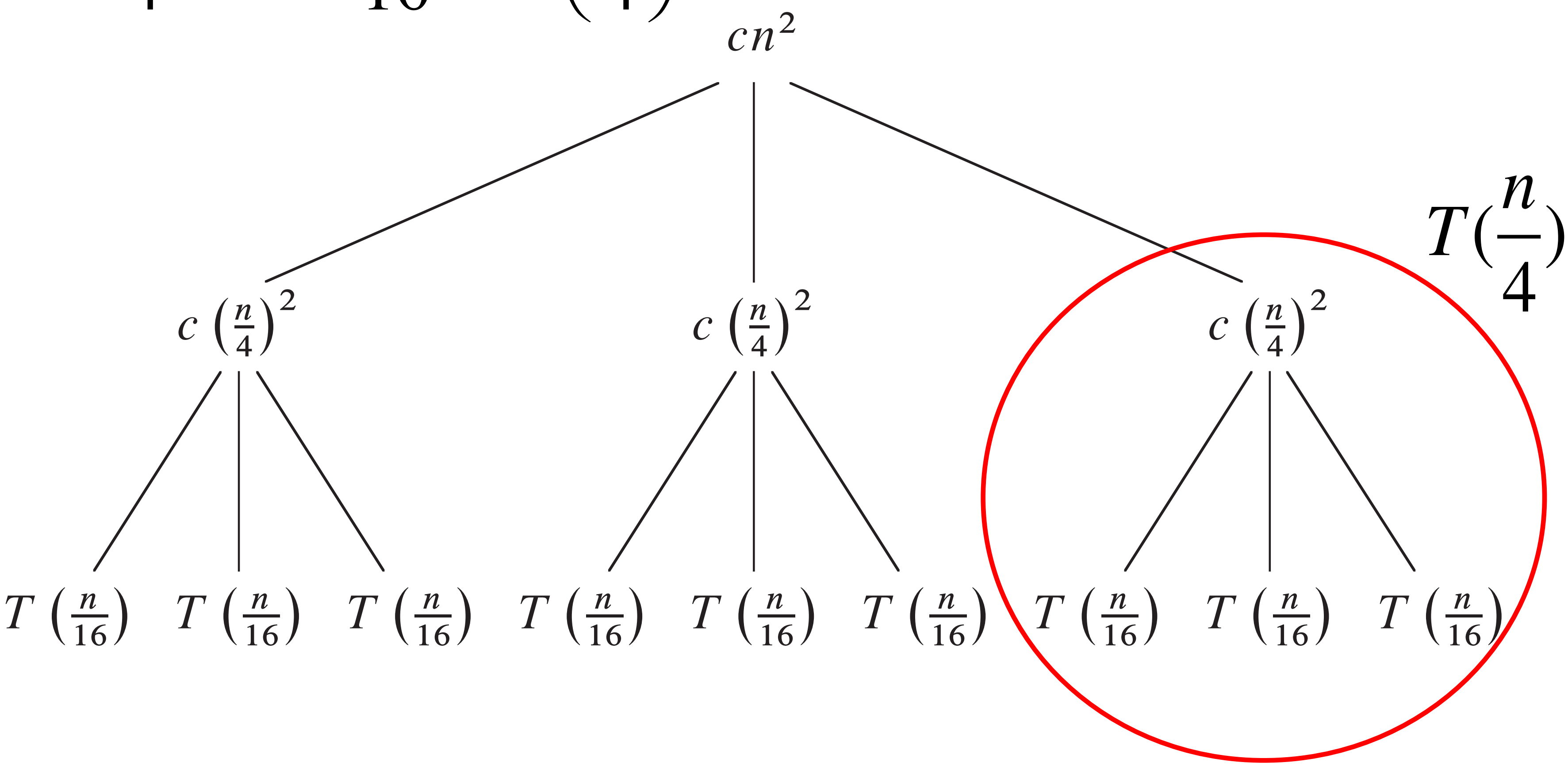
$$cn^2$$

$$T\left(\frac{n}{4}\right) \quad T\left(\frac{n}{4}\right) \quad T\left(\frac{n}{4}\right)$$

Recursion example

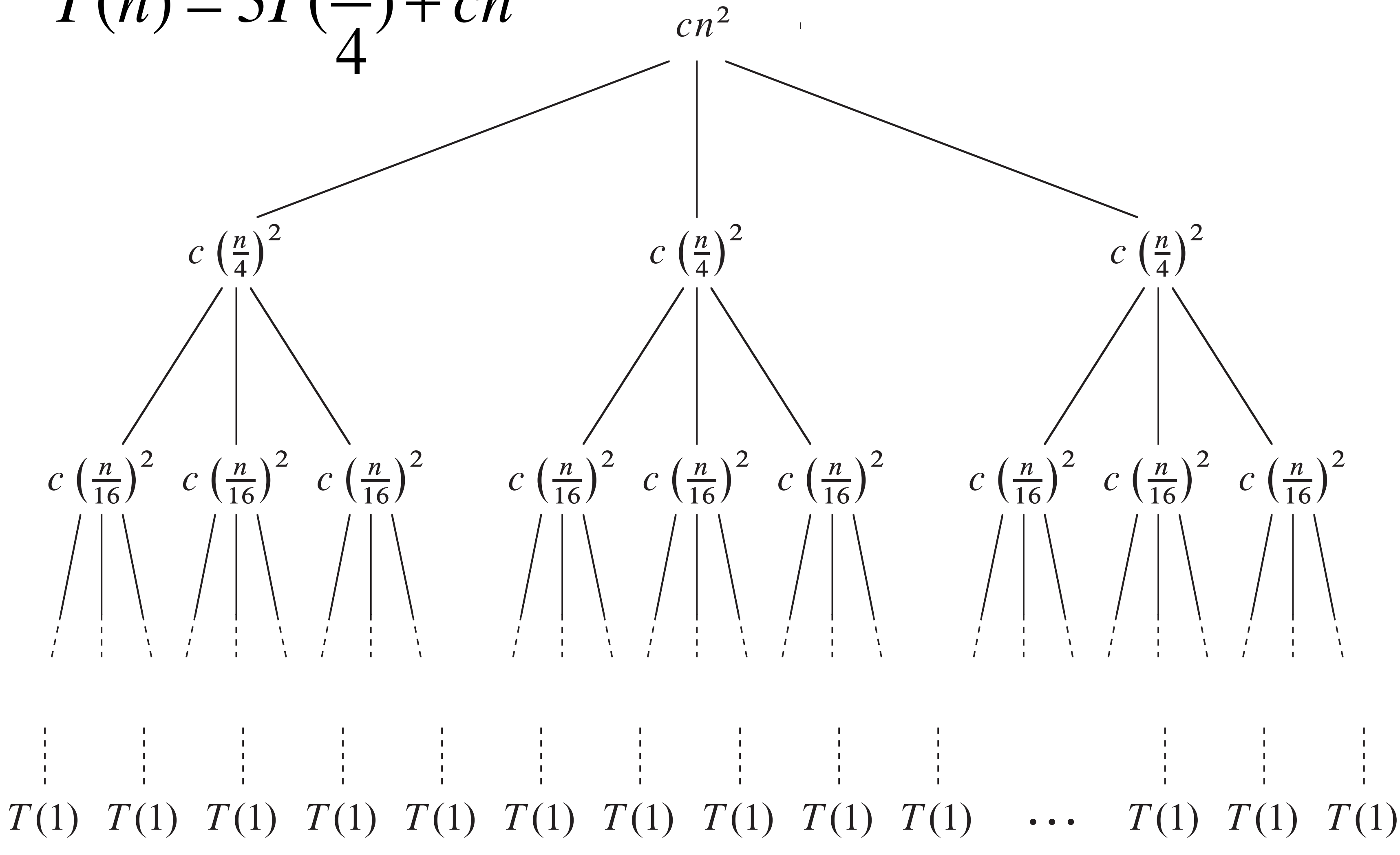
$$T(n) = 3T(\frac{n}{4}) + cn^2$$

$$T(\frac{n}{4}) = 3T(\frac{n}{16}) + c\left(\frac{n}{4}\right)^2$$



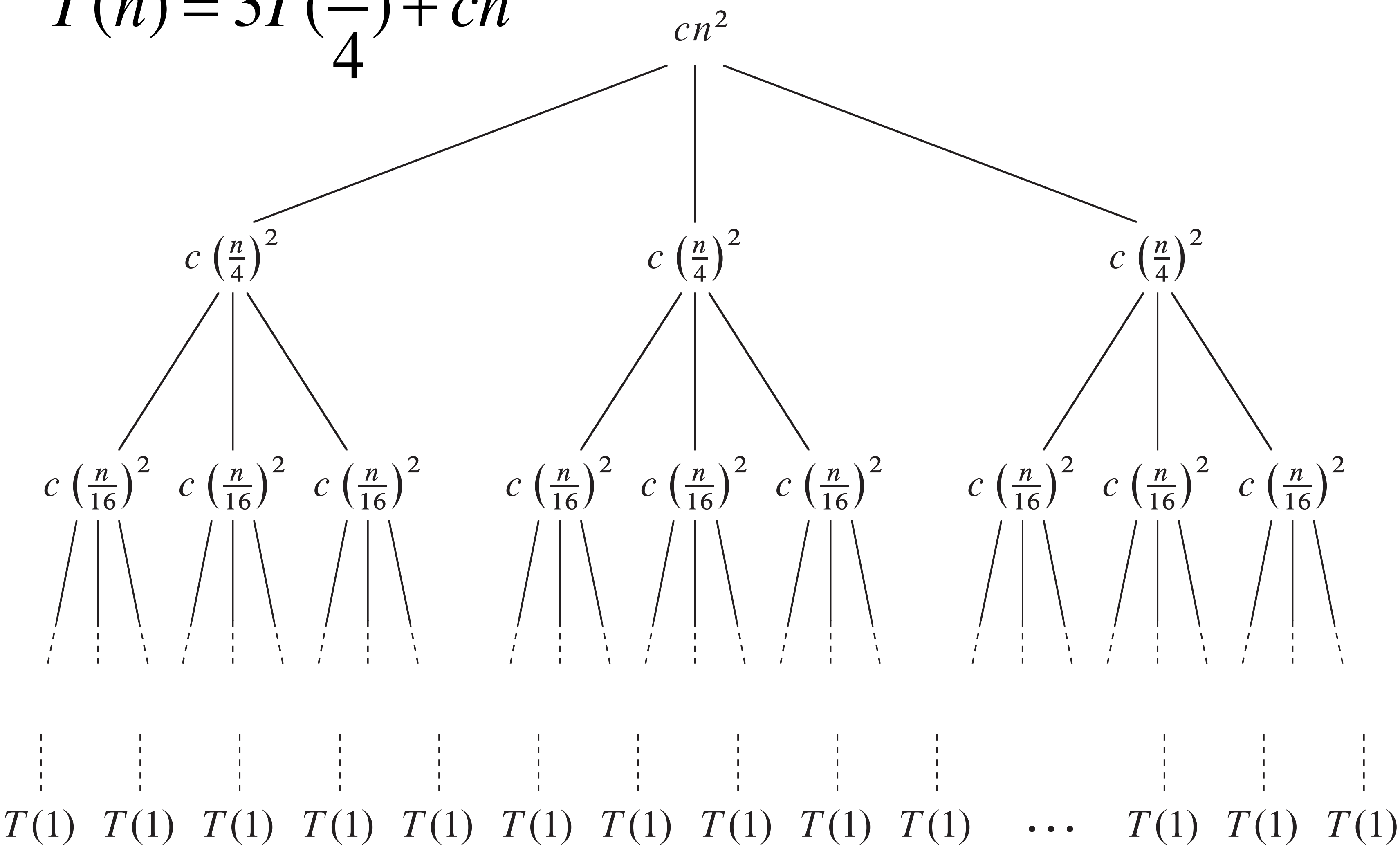
Recursion example

$$T(n) = 3T(\frac{n}{4}) + cn^2$$



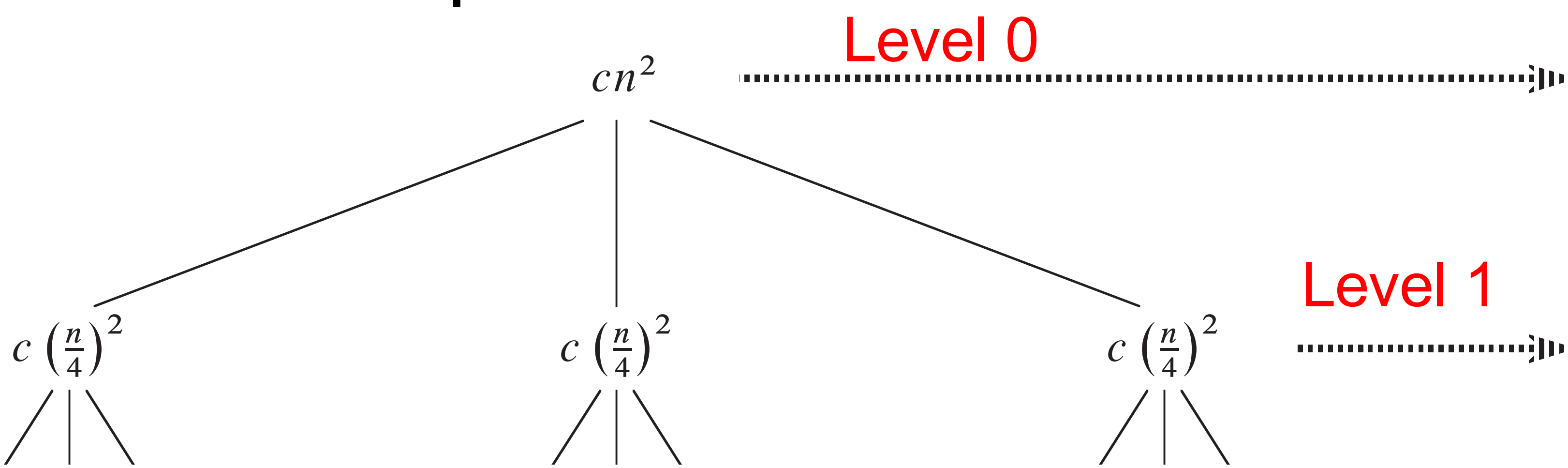
Recursion example

$$T(n) = 3T(\frac{n}{4}) + cn^2$$



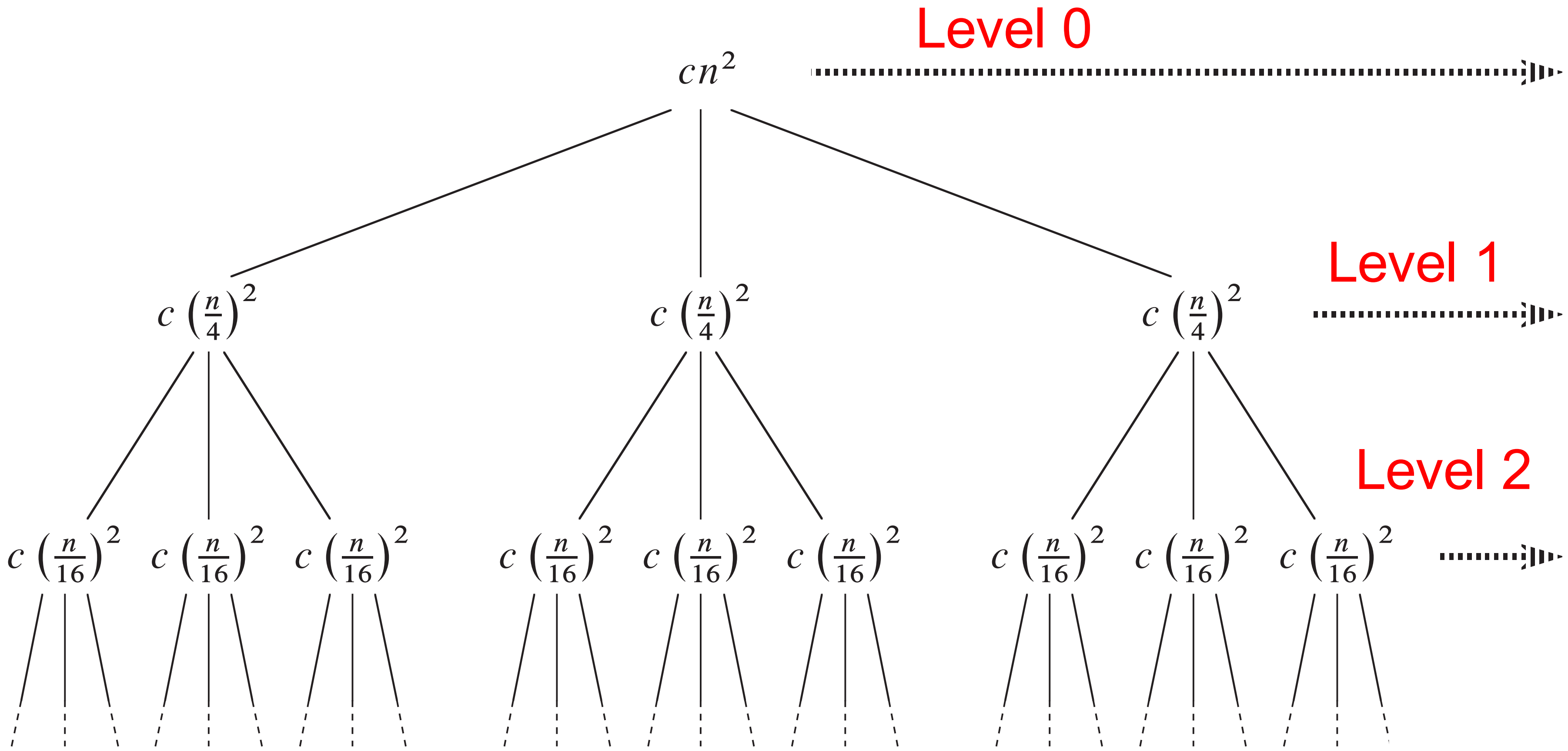
Number of levels?

Recursion example



Level 1: subproblem size $\quad \dfrac{n}{4} = \dfrac{n}{4^1}$

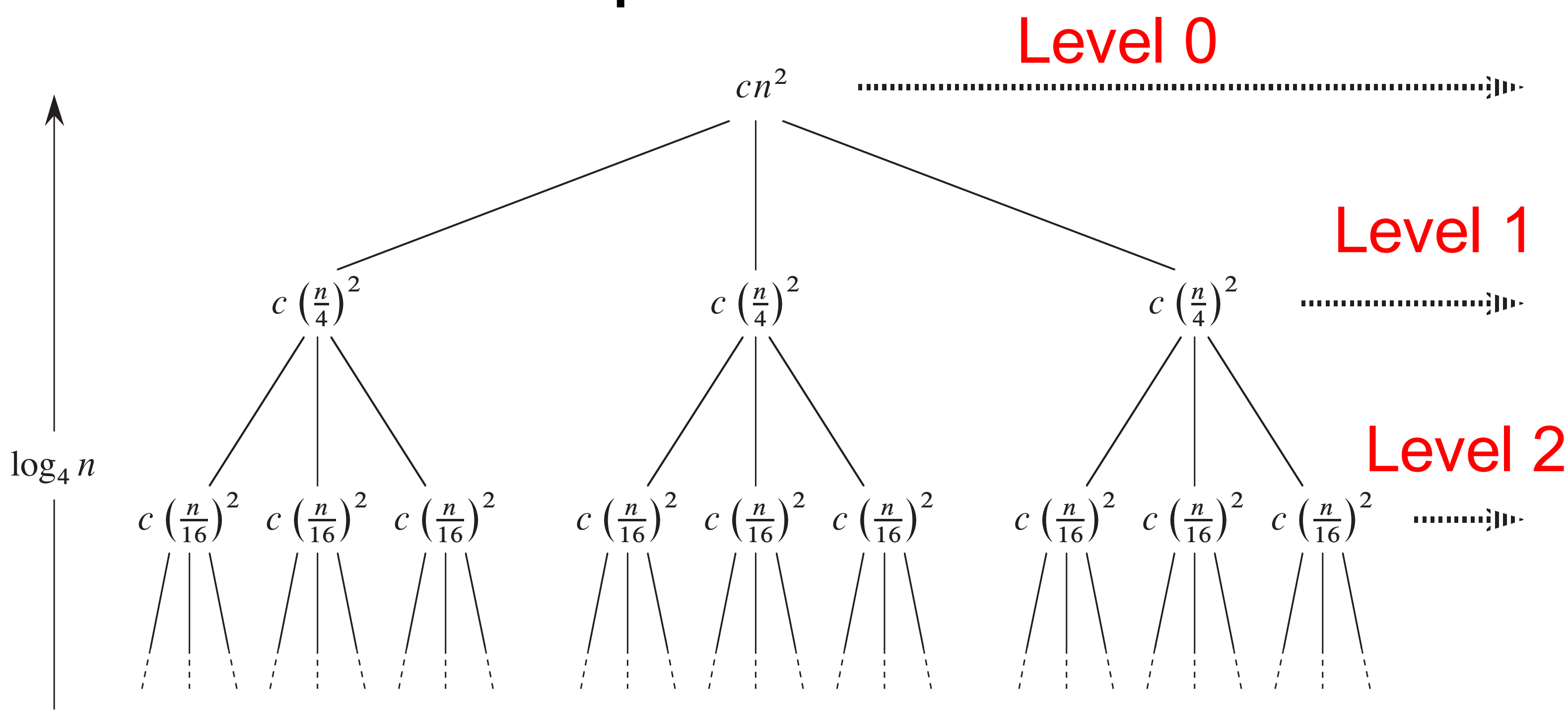
Recursion example



Level 1: subproblem size $\frac{n}{4} = \frac{n}{4^1}$

Level 2: subproblem size $\frac{n}{16} = \frac{n}{4^2}$

Recursion example

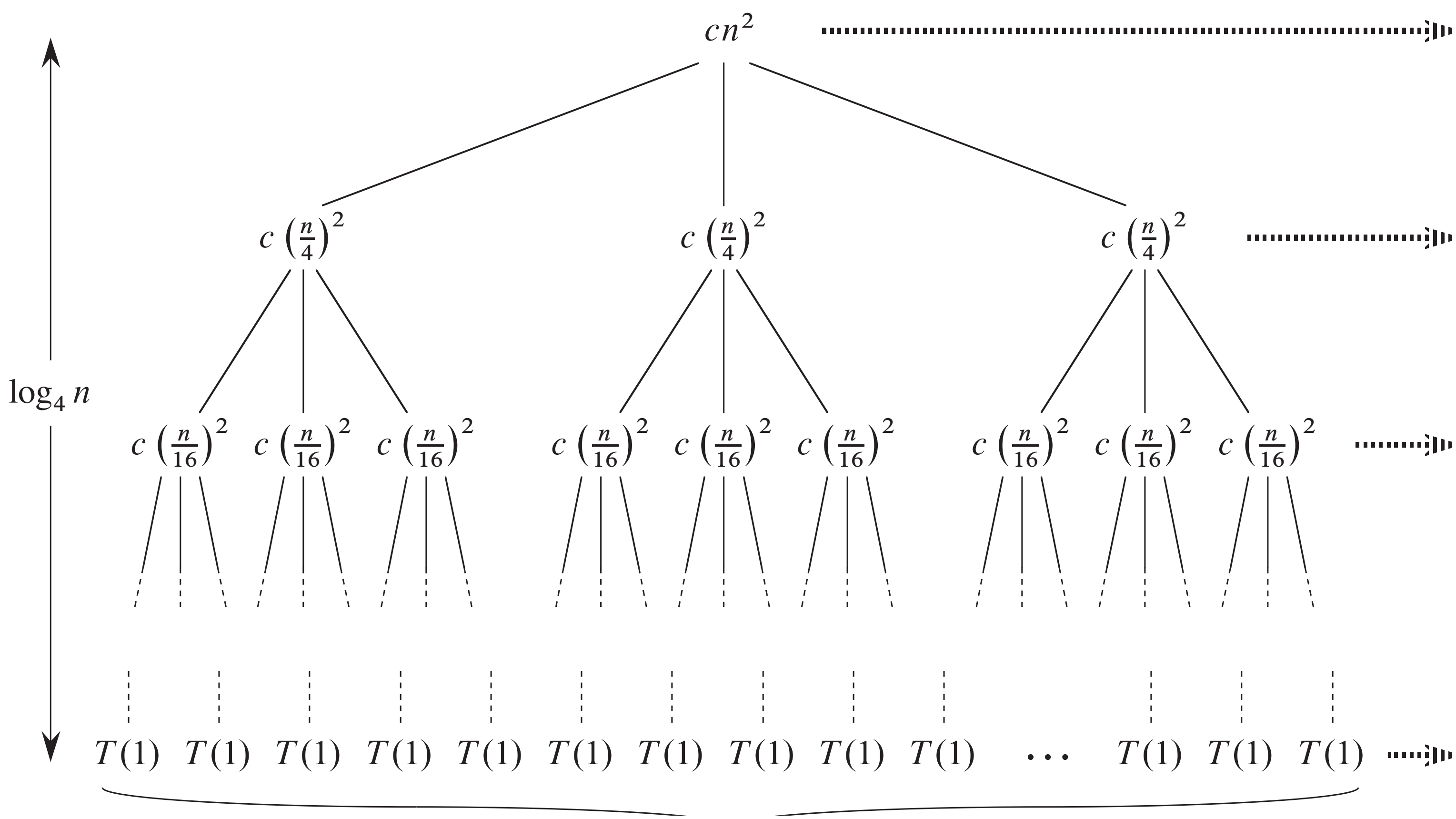


Level 1: subproblem size $\frac{n}{4} = \frac{n}{4^1}$

Level 2: subproblem size $\frac{n}{16} = \frac{n}{4^2}$

Level k: subproblem size $\frac{n}{4^k}$

Recursion example



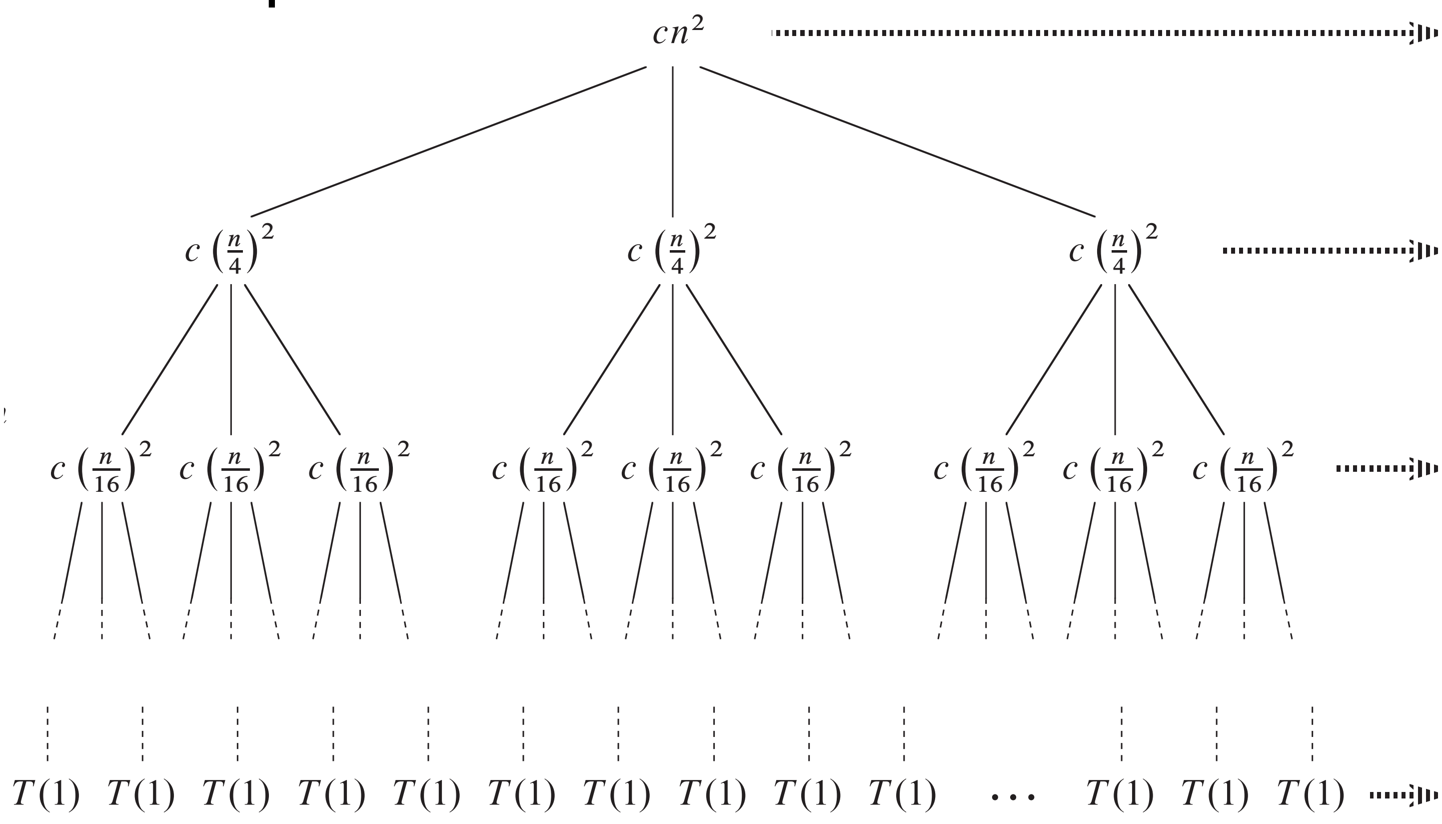
At the last level, the subproblem size is 1

$$\frac{n}{4^k} = 1;$$

$$k = \log_4 n$$

Recursion example

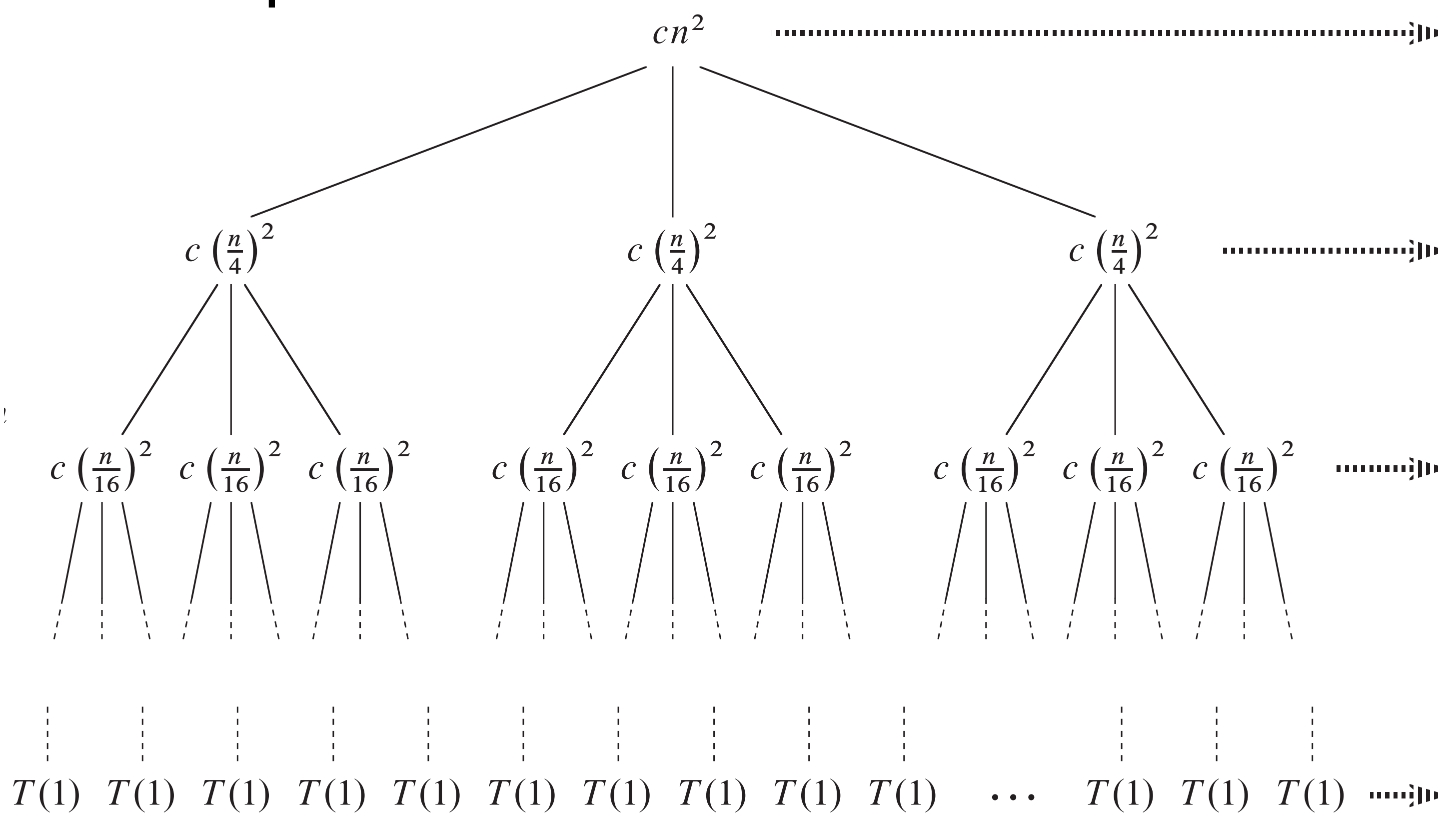
$$T(n) = 3T(\frac{n}{4}) + cn^2$$



Number of levels? $\log_4 n$

Recursion example

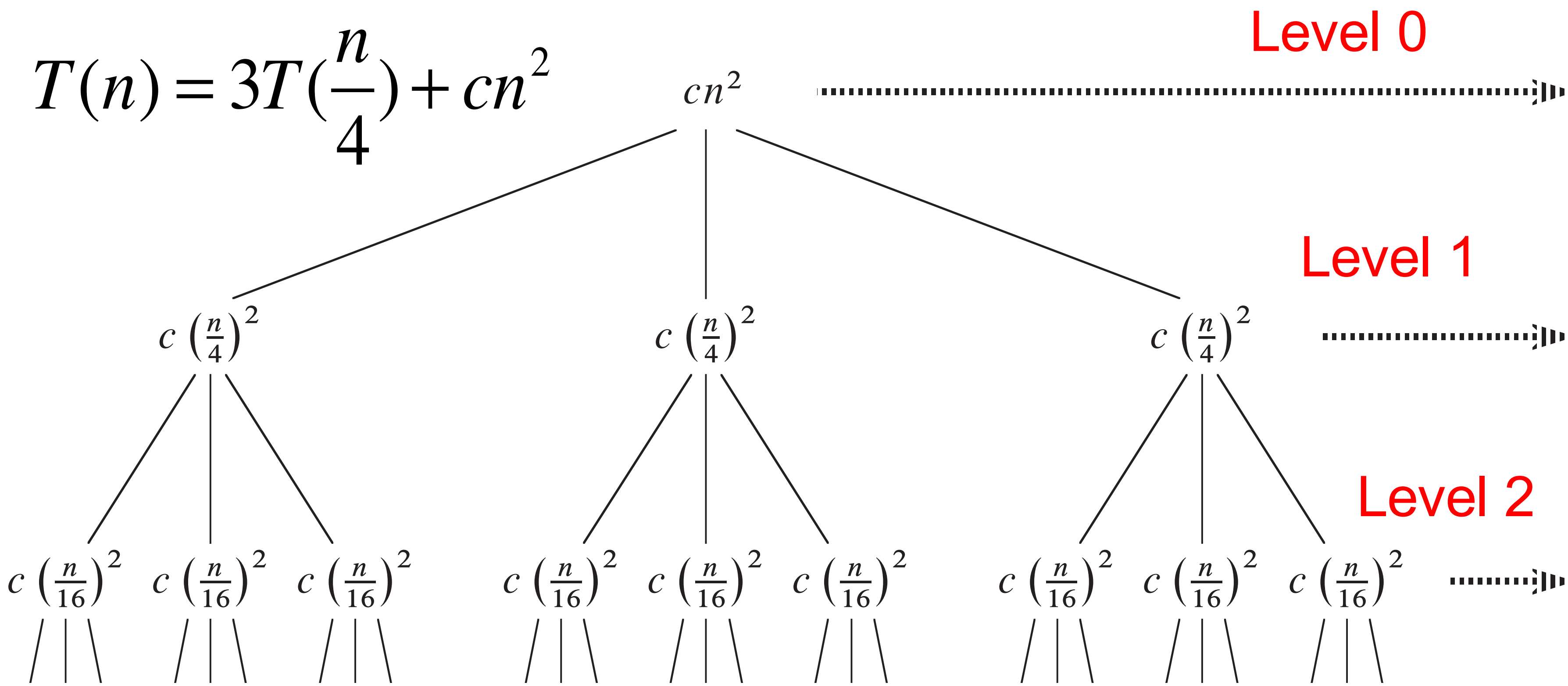
$$T(n) = 3T(\frac{n}{4}) + cn^2$$



Total work?

Recursion example

$$T(n) = 3T(\frac{n}{4}) + cn^2$$



Cost at level 1 $\quad 3^1 c\left(\frac{n}{4^1}\right)^2$

Cost at level 2 $\quad 3^2 c\left(\frac{n}{4^2}\right)^2$

Recursion example

$$T(n) = 3T(\frac{n}{4}) + cn^2$$

Cost at level 1 $\quad 3^1 c\left(\frac{n}{4^1}\right)^2$

Cost at level 2 $\quad 3^2 c\left(\frac{n}{4^2}\right)^2$

Cost at level 3 $\quad 3^3 c\left(\frac{n}{4^3}\right)^2$

Recursion example

$$T(n) = 3T(\frac{n}{4}) + cn^2$$

Cost at root cn^2

Cost at level 1 $3^1 c\left(\frac{n}{4^1}\right)^2$

Cost at level 2 $3^2 c\left(\frac{n}{4^2}\right)^2$

Cost at level 3 $3^3 c\left(\frac{n}{4^3}\right)^2$

As we go to deeper levels of the tree, is the cost?

A. Increasing

B. Decreasing

C. Same

Recursion example

$$T(n) = 3T(\frac{n}{4}) + cn^2$$

Cost at root $\quad cn^2$

Cost at level 1 $\quad 3^1 c\left(\frac{n}{4^1}\right)^2 = \frac{3}{4}cn^2$

Cost at level 2 $\quad 3^2 c\left(\frac{n}{4^2}\right)^2 = \left(\frac{3}{16}\right)^2 cn^2$

Cost at level 3 $\quad 3^3 c\left(\frac{n}{4^3}\right)^2 = \left(\frac{3}{16}\right)^3 cn^2$

As we go to deeper levels of the tree, is the cost?

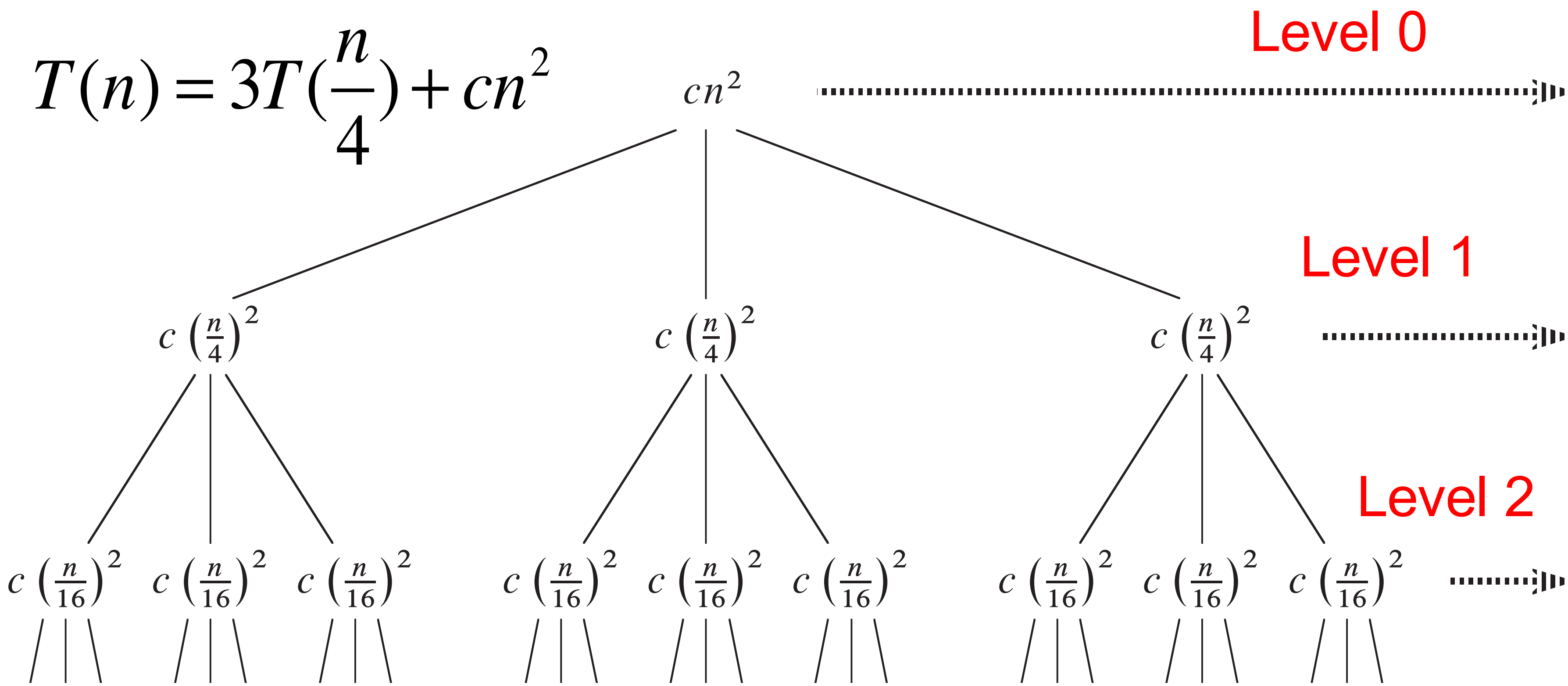
A. Increasing

B. Decreasing

C. Same

Recursion example

$$T(n) = 3T(\frac{n}{4}) + cn^2$$

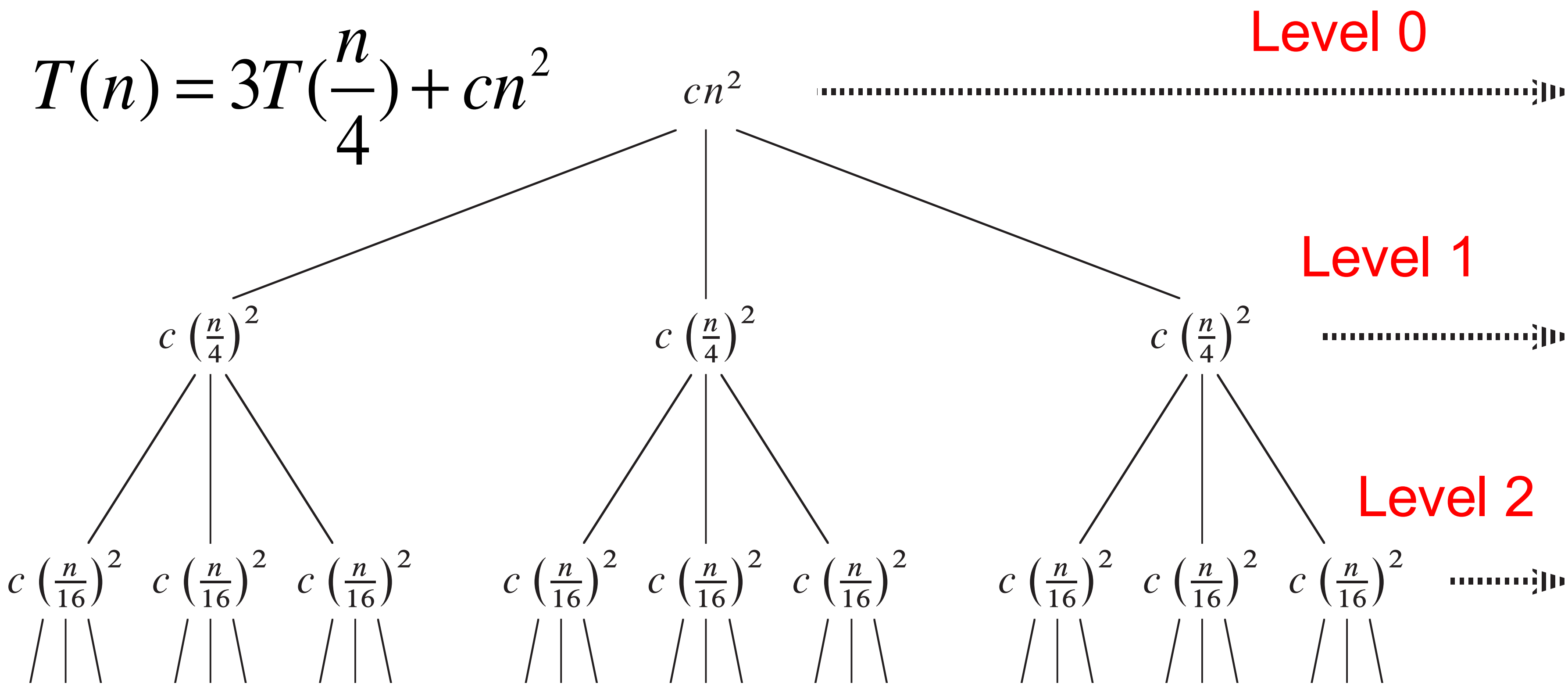


Cost at level 1 $\quad 3^1 c\left(\frac{n}{4^1}\right)^2$

Cost at level k $\quad 3^k c\left(\frac{n}{4^k}\right)^2 = 3^k c\left(\frac{n^2}{16^k}\right) = \left(\frac{3}{16}\right)^k cn^2$

Recursion example

$$T(n) = 3T(\frac{n}{4}) + cn^2$$

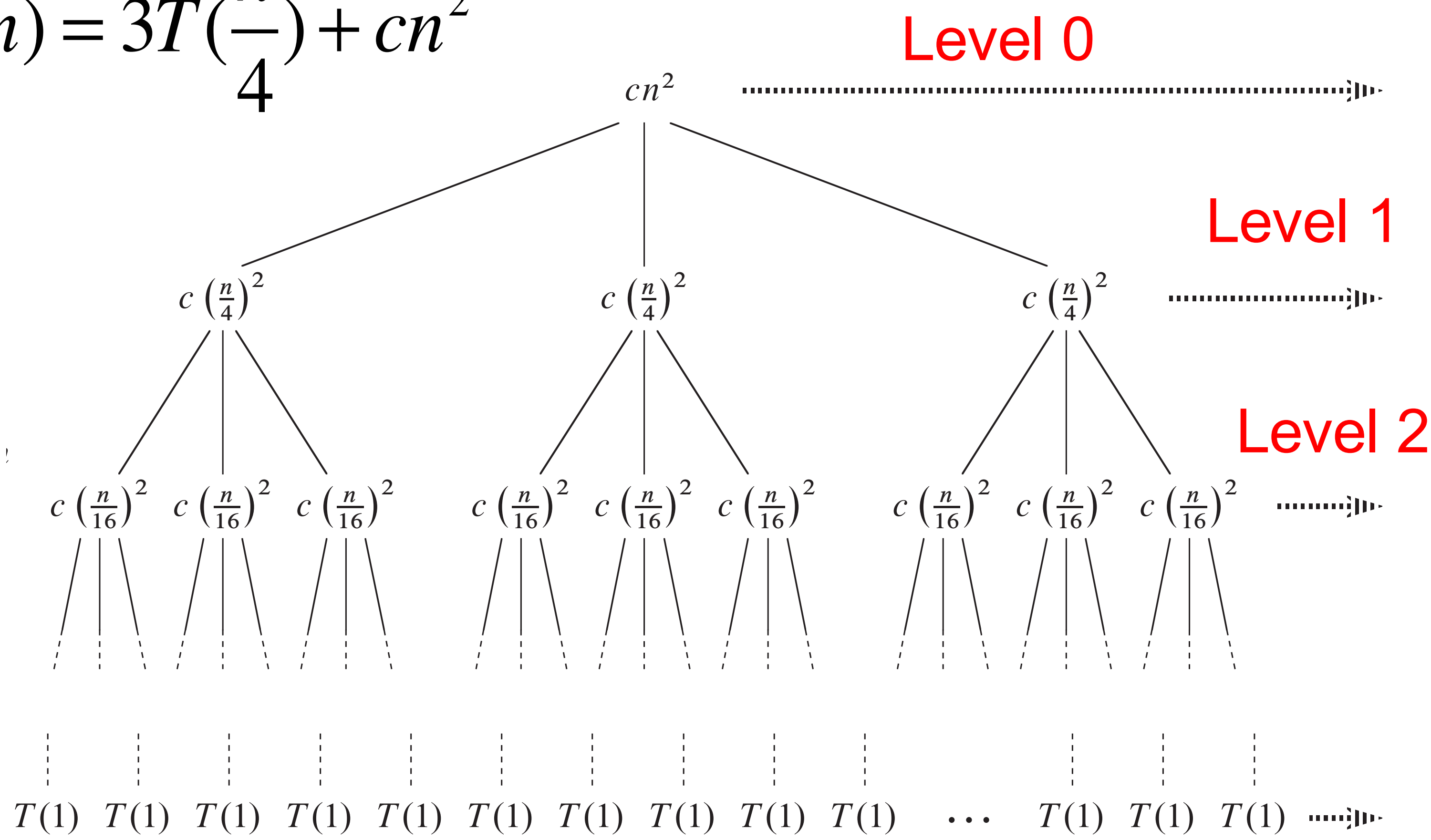


Cost at last level? We know we are down to subproblem size of 1

We just need to know number of nodes

Recursion example

$$T(n) = 3T(\frac{n}{4}) + cn^2$$



Number nodes at level k = last level

$$3^k = 3^{\log_4 n}$$

Recursion example

Remember that:

$$a^{\log_b n} = n^{\log_b a}$$

because

$$\log_b(a^{\log_b n}) = \log_b(n^{\log_b a})$$

Number nodes at level k = last level

$$3^k = 3^{\log_4 n}$$

Recursion example

Remember that:

$$a^{\log_b n} = n^{\log_b a}$$

because

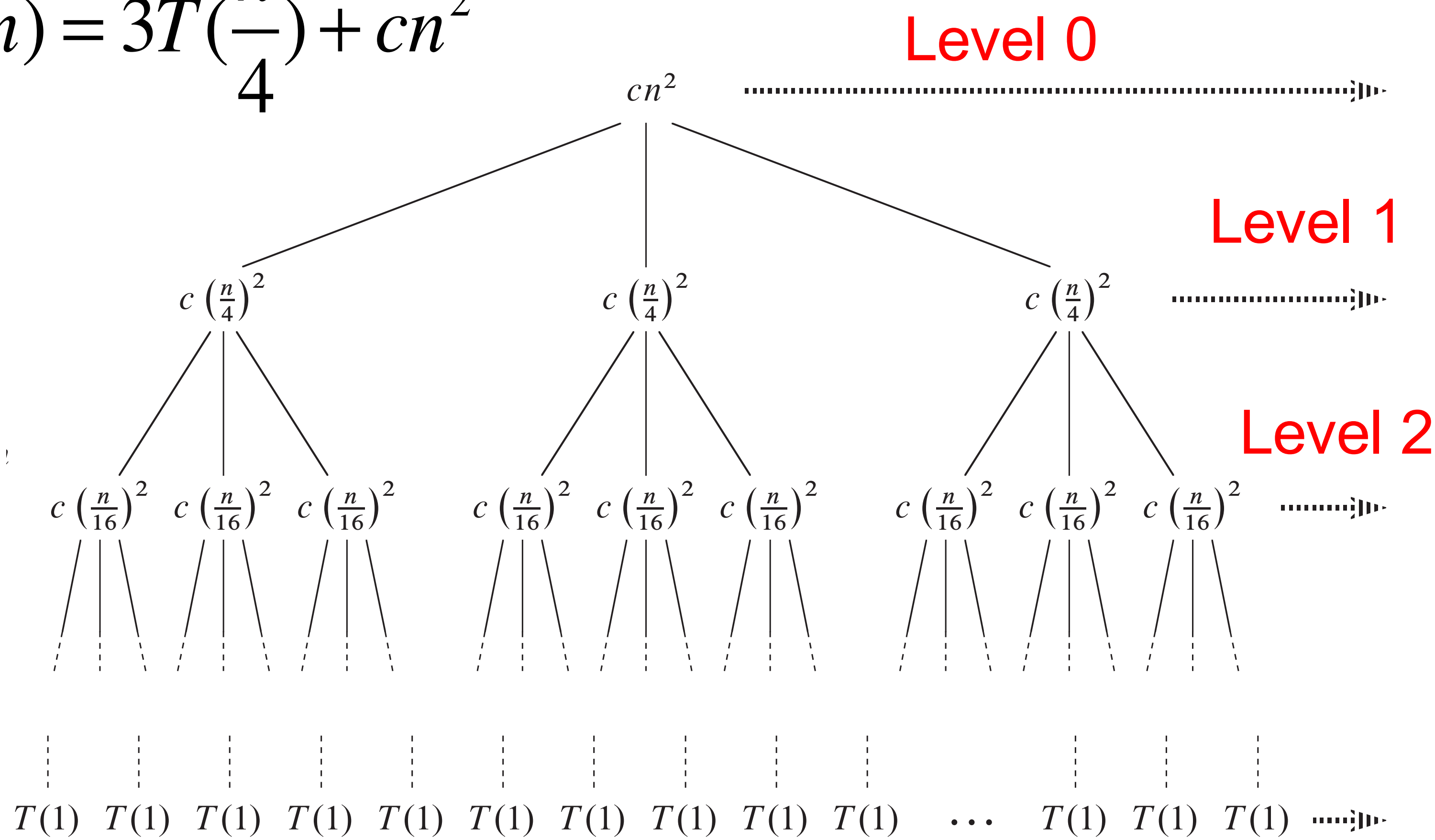
$$\log_b(a^{\log_b n}) = \log_b(n^{\log_b a})$$

Number nodes at level k = last level

$$3^k = 3^{\log_4 n} = n^{\log_4 3}$$

Recursion example

$$T(n) = 3T\left(\frac{n}{4}\right) + cn^2$$



Cost at last level $$T(1)3^{\log_4 n} = T(1)n^{\log_4 3} = \Theta(n^{\log_4 3})$$

Recursion example

$$T(n) = 3T(\frac{n}{4}) + cn^2$$

Total work?

$$T(n) = \left(\frac{3}{16}\right)cn^2 + \left(\frac{3}{16}\right)^2 cn^2 + .. + \left(\frac{3}{16}\right)^{\log_4 n - 1} cn^2$$

$$+\Theta\left(n^{\log_4 3}\right)$$

All other levels

Last level

Recursion example

$$T(n) = 3T(\frac{n}{4}) + cn^2$$

Total work?

$$T(n) = \left(\frac{3}{16}\right)cn^2 + \left(\frac{3}{16}\right)^2 cn^2 + .. + \left(\frac{3}{16}\right)^{\log_4 n - 1} cn^2$$

$$+\Theta\left(n^{\log_4 3}\right)$$

$$= \Theta\left(n^{\log_4 3}\right) + \sum_{k=1}^{\log_4 n - 1} \left(\frac{3}{16}\right)^k cn^2$$

Last level

All other levels

Summation…

$$\sum_{k=1}^{\log_4 n-1}\left(\frac{3}{16}\right)^k cn^2 = cn^2\sum_{k=1}^{\log_4 n-1}\left(\frac{3}{16}\right)^k$$

All other levels

Summation…

$$\sum_{k=1}^{\log_4 n-1}\left(\frac{3}{16}\right)^k cn^2 = cn^2\sum_{k=1}^{\log_4 n-1}\left(\frac{3}{16}\right)^k$$

All other levels

Summation…

$$\sum_{k=1}^{\log_4 n-1}\left(\frac{3}{16}\right)^k cn^2 = cn^2\sum_{k=1}^{\log_4 n-1}\left(\frac{3}{16}\right)^k$$

All other levels

This can be solved exactly (messy), but we also know that for an Infinite summation series and $|x| < 1$

$$\sum_{k=0}^{\infty} x^k = \frac{1}{1-x}$$

Infinitely decreasing series

Summation…

$$\sum_{k=1}^{\log_4 n-1}\left(\frac{3}{16}\right)^k cn^2 = cn^2 \sum_{k=1}^{\log_4 n-1}\left(\frac{3}{16}\right)^k$$

All other levels

This can be solved exactly (messy), but we also know that for an Infinite summation series and $|x| < 1$

$$\sum_{k=0}^{\infty} x^k = \frac{1}{1-x}$$

Infinitely decreasing series

Our series doesn't go to infinity. What to do?

Summation…

$$\sum_{k=1}^{\log_4 n-1}\left(\frac{3}{16}\right)^k cn^2 = cn^2\sum_{k=1}^{\log_4 n-1}\left(\frac{3}{16}\right)^k$$

All other levels

This can be solved exactly (messy), but we also know that for an Infinite summation series and $|x|<1$

$$\sum_{k=0}^{\infty} x^k = \frac{1}{1-x}$$

Infinitely decreasing series

Our series doesn't go to infinity. What to do? Upper bound.

Summation…

$$\sum_{k=1}^{\log_4 n-1}\left(\frac{3}{16}\right)^k cn^2 = cn^2\sum_{k=1}^{\log_4 n-1}\left(\frac{3}{16}\right)^k <$$

$$cn^2\sum_{k=1}^{\infty}\left(\frac{3}{16}\right)^k$$

Infinitely decreasing series

Our series doesn't go to infinity. What to do?
Upper bound.

Summation…

$$\sum_{k=1}^{\log_4 n-1}\left(\frac{3}{16}\right)^k cn^2 = cn^2\sum_{k=1}^{\log_4 n-1}\left(\frac{3}{16}\right)^k <$$

$$cn^2\sum_{k=1}^{\infty}\left(\frac{3}{16}\right)^k = ?$$

Infinitely decreasing series $\displaystyle\sum_{k=0}^{\infty} x^k = \frac{1}{1-x}$

Bound equal to?

Summation…

$$\sum_{k=1}^{\log_4 n-1}\left(\frac{3}{16}\right)^k cn^2 = cn^2\sum_{k=1}^{\log_4 n-1}\left(\frac{3}{16}\right)^k <$$

$$cn^2\sum_{k=1}^{\infty}\left(\frac{3}{16}\right)^k = cn^2\left(\frac{1}{1-\frac{3}{16}}\right)$$

Infinitely decreasing series

Recursion example

$$T(n) = 3T(\frac{n}{4}) + cn^2$$

Total work?

Last level

All other levels

$$T(n) = \Theta\left(n^{\log_4 3}\right) + \sum_{k=1}^{\log_4 n - 1} \left(\frac{3}{16}\right)^k cn^2 <$$

$$\Theta\left(n^{\log_4 3}\right) + \frac{1}{1 - \left(\frac{3}{16}\right)} cn^2$$

Recursion example

$$T(n) = 3T(\frac{n}{4}) + cn^2$$

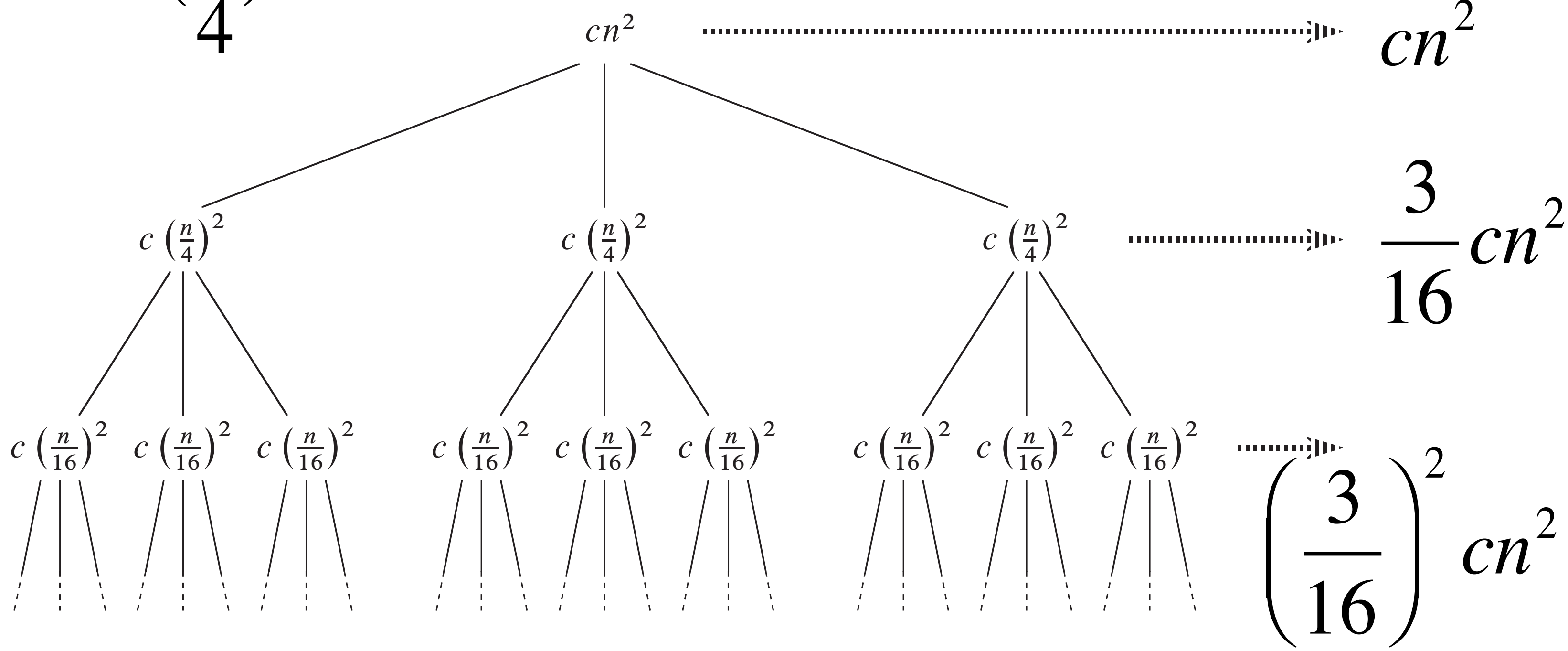
Total work?

$$T(n) = \Theta\left(n^{\log_4 3}\right) + \sum_{k=1}^{\log_4 n - 1} \left(\frac{3}{16}\right)^k cn^2 <$$

$$\Theta\left(n^{\log_4 3}\right) + \frac{1}{1 - \left(\frac{3}{16}\right)} cn^2 = O(n^2)$$

Recursion example summary

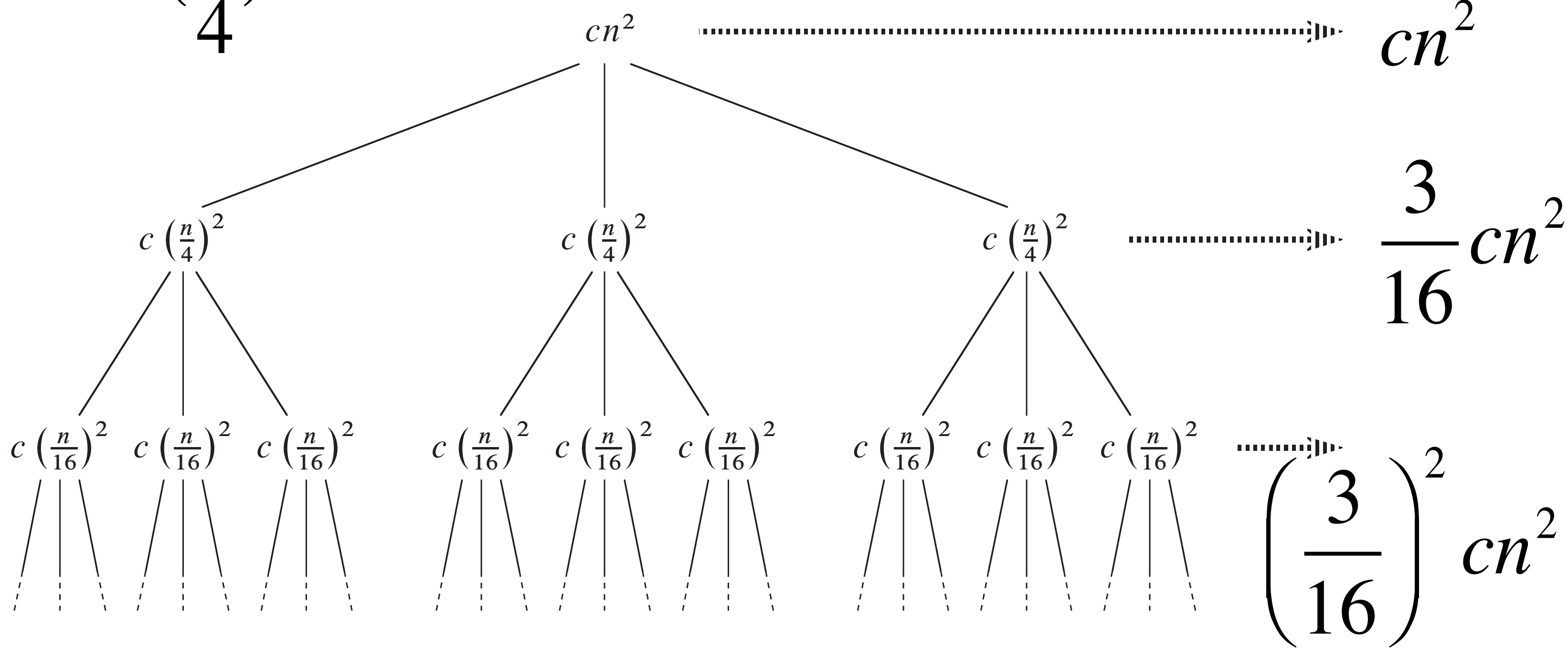
$$T(n) = 3T(\frac{n}{4}) + cn^2$$



Total work $O(n^2)$

Recursion example summary

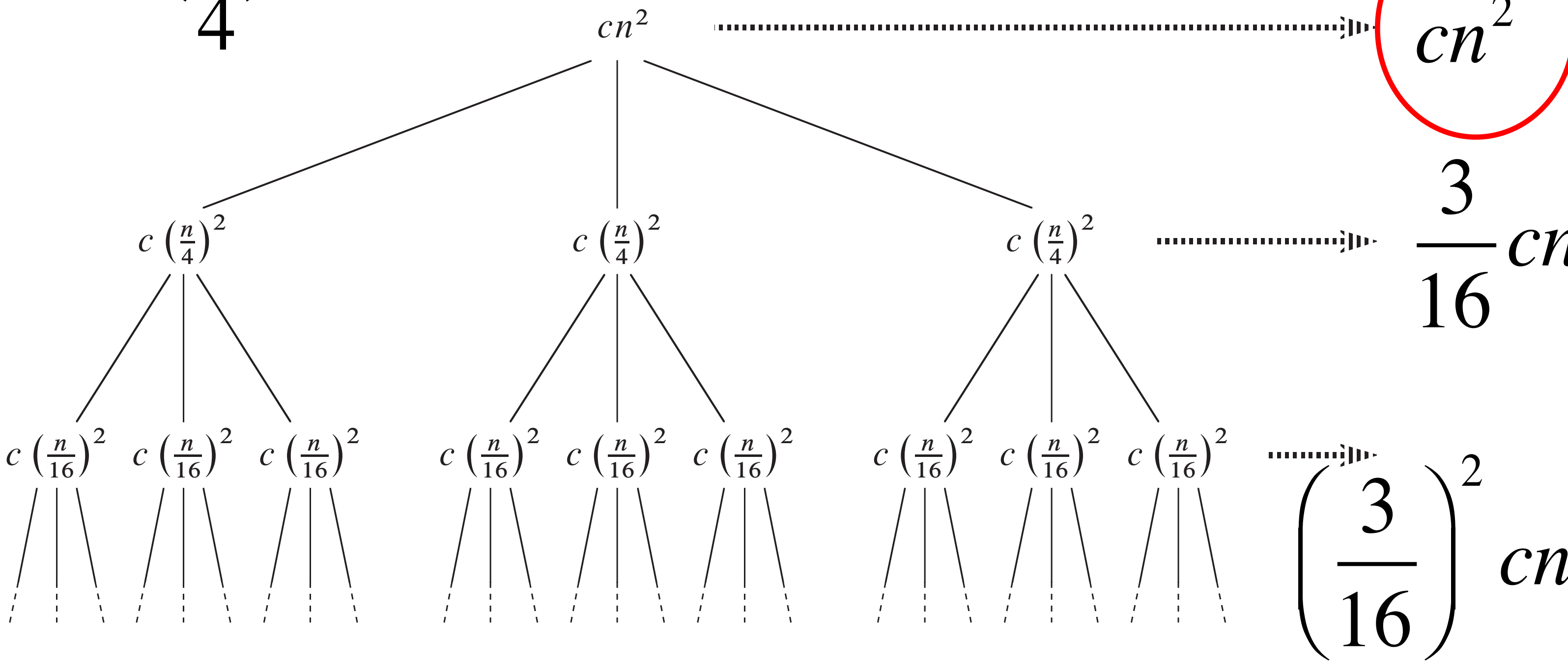
$$T(n) = 3T(\frac{n}{4}) + cn^2$$



Total work $\Theta(n^2)$ Actually a tight bound
Why?

Recursion example summary

$$T(n) = 3T(\frac{n}{4}) + cn^2$$



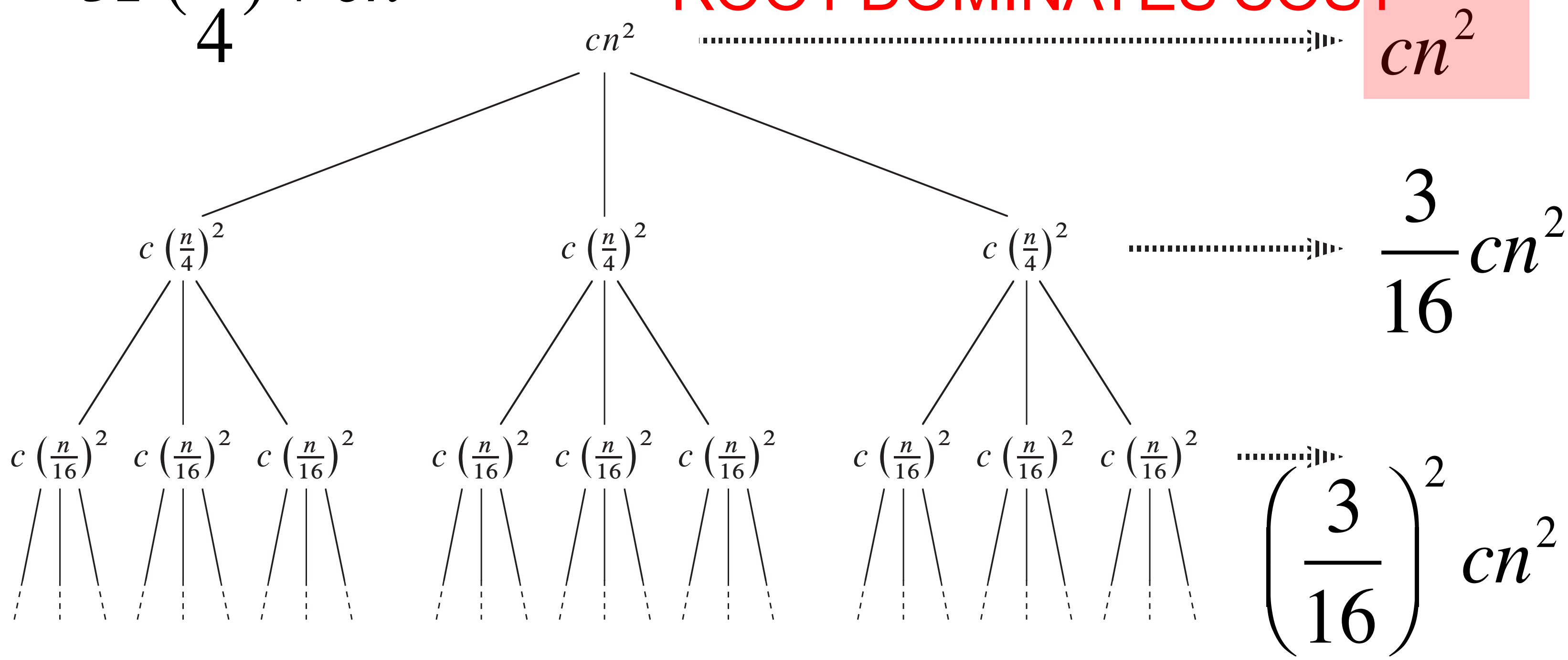
Total work $\Theta(n^2)$ Actually a tight bound

Why?

Recursion example

$$T(n) = 3T\left(\frac{n}{4}\right) + cn^2$$

ROOT DOMINATES COST



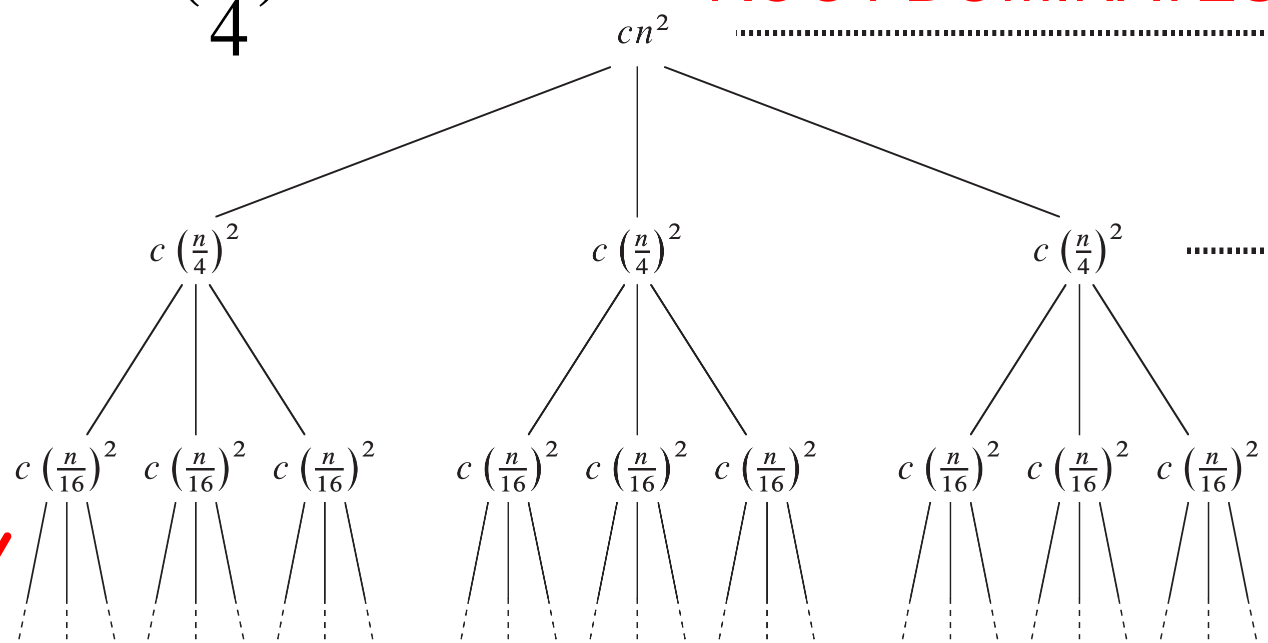
Total work $\Theta(n^2)$ Actually a tight bound

Recursion example

$$T(n) = 3T(\frac{n}{4}) + cn^2$$

ROOT DOMINATES COST

Cost decr



$$\boxed{cn^2}$$

$$\frac{3}{16}cn^2$$

$$\left(\frac{3}{16}\right)^2 cn^2$$

Total work $\Theta(n^2)$ Actually a tight bound

Another recursion example

$$T(n) = 2T(\frac{n}{2}) + c$$

On the board…

Summary of 3 recursion examples

Summary of 3 recursion examples

$$T(n) = 2T(\frac{n}{2}) + cn = \Theta(n \log n)$$

Equal cost at all levels – like merge sort

Summary of 3 recursion examples

$$T(n) = 2T(\frac{n}{2}) + cn = \Theta(n \log n)$$

Equal cost at all levels – like merge sort

$$T(n) = 3T(\frac{n}{4}) + cn^2 = \theta(n^2)$$

Root dominated

Summary of 3 recursion examples

$$T(n) = 2T(\frac{n}{2}) + cn = \Theta(n \log n)$$

Equal cost at all levels – like merge sort

ALSO

$$T(n) = 3T(\frac{n}{4}) + cn^2 = \theta(n^2) \qquad T(n) = 2T(\frac{n}{2}) + cn^2 = \theta(n^2)$$

Root dominated

Summary of 3 recursion examples

$$T(n) = 2T(\frac{n}{2}) + cn = \Theta(n \log n)$$

Equal cost at all levels – like merge sort

$$T(n) = 3T(\frac{n}{4}) + cn^2 = \theta(n^2) \quad T(n) = 2T(\frac{n}{2}) + cn^2 = \theta(n^2)$$

Root dominated

$$T(n) = 2T(\frac{n}{2}) + c = \Theta(n)$$

Leaves dominated

Summary of 3 recursion examples

$$T(n) = 2T(\frac{n}{2}) + cn = \Theta(n \log n)$$

Equal cost at all levels – like merge sort

$$T(n) = 3T(\frac{n}{4}) + cn^2 = \theta(n^2) \qquad T(n) = 2T(\frac{n}{2}) + cn^2 = \theta(n^2)$$

Root dominated

$$T(n) = 2T(\frac{n}{2}) + c = \Theta(n)$$

Leaves dominated

What factors appear important?

Summary of 3 recursion examples

$$T(n) = 2T(\frac{n}{2}) + \boxed{cn} = \Theta(n \log n)$$

Equal cost at all levels – like merge sort

$$T(n) = 3T(\frac{n}{4}) + cn^2 = \theta(n^2) \qquad T(n) = 2T(\frac{n}{2}) + \boxed{cn^2} = \theta(n^2)$$

Root dominated

$$T(n) = 2T(\frac{n}{2}) + \boxed{c} = \Theta(n)$$

Leaves dominated

What factors appear important?
Notice differences here…

Summary of 3 recursion examples

$$T(n) = 2T(\frac{n}{2}) + cn = \Theta(n \log n)$$

Equal cost at all levels – like merge sort

$$T(n) = 3T(\frac{n}{4}) + cn^2 = \theta(n^2) \qquad T(n) = 2T(\frac{n}{2}) + cn^2 = \theta(n^2)$$

Root dominated

$$T(n) = 2T(\frac{n}{2}) + c = \Theta(n)$$

Leaves dominated

What factors appear important?

What else might be important?

Goals

What kind of recurrences arise in algorithms
and how do we solve more generally (than what
we saw for merge sort)?

- More recurrence examples
- Revisit recursion trees more generally
- Master theorem as "recipe" for range of cases (next class)
- Substitution method