Divide-and-Conquer

Inge Li Gørtz

Thank you to Kevin Wayne for inspiration to slides

Divide-and-Conquer

- Divide -and-Conquer.
    - Break up problem into several parts.
    - Solve each part recursively.
    - Combine solutions to subproblems into overall solution.

- Today
    - Mergesort (recap)
    - Recurrence relations
    - Integer multiplication

Mergesort

Recurrence relations

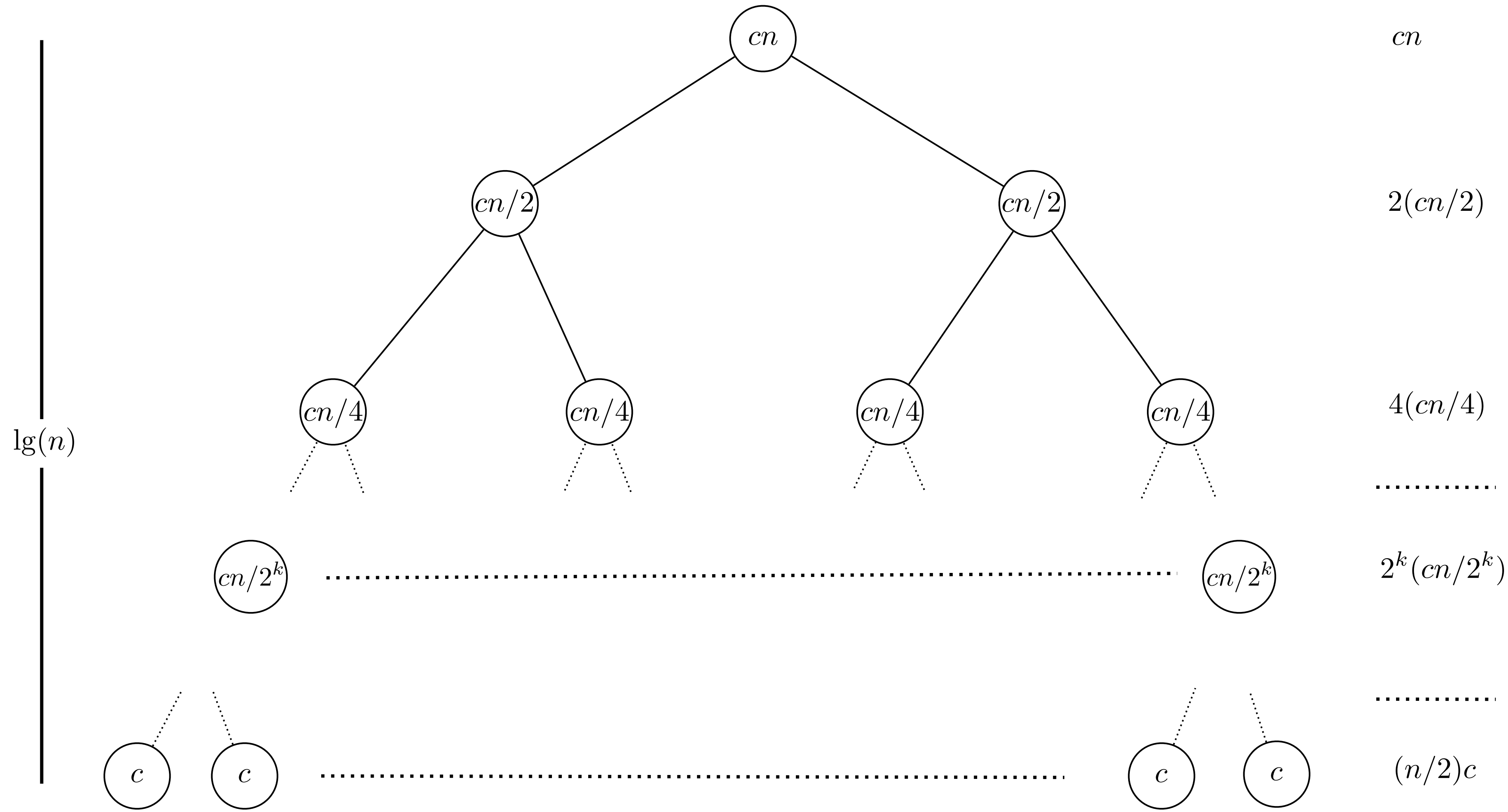
- T(n) = running time of mergesort on input of size n
- Mergesort recurrence:

$$T(n) \leq \begin{cases} 2T(n/2) + cn & \text{if } n > 2 \\ c & \text{otherwise} \end{cases}$$

- Solving the recurrence:
  - Recursion tree
  - Substitution

Mergesort recurrence: recursion tree

$$T(n) \leq \begin{cases} 2T(n/2) + cn & \text{if } n > 2 \\ c & \text{otherwise} \end{cases}$$



Mergesort recurrence: substitution

$$T(n) \leq \begin{cases} 2T(n/2) + cn & \text{if } n > 2 \\ c & \text{otherwise} \end{cases}$$

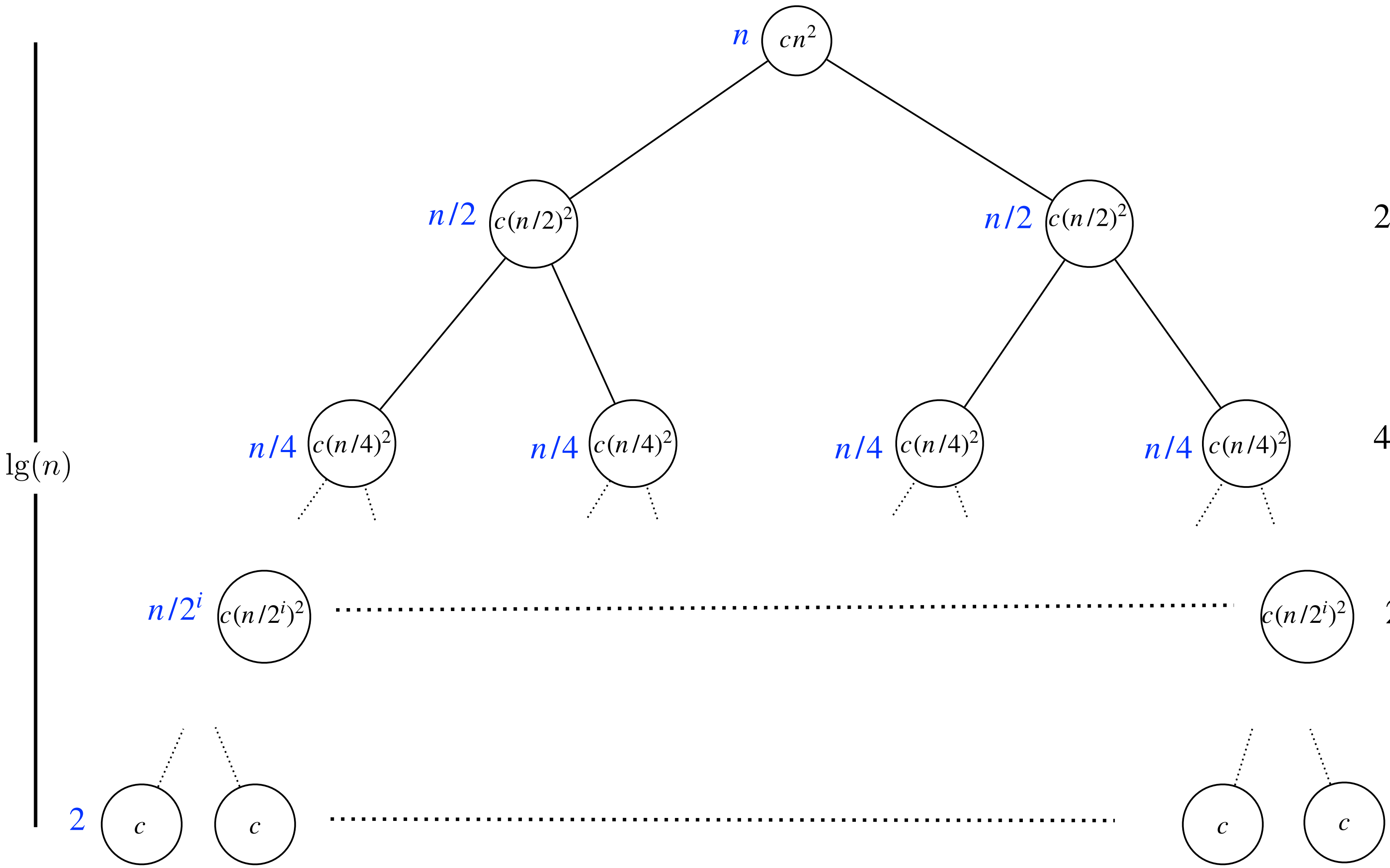
- Substitute $T(n)$ with $kn \lg n$ and use induction to prove $T(n) \leq n \lg nk$.
- Base case ($n = 2$):
  - By definition $T(2) = c$.
  - Substitution: $k \cdot 2 \lg 2 = 2k \geq c = T(2)$ if $k \geq c/2.$
- Induction: Assume $T(m) \leq km \lg m$ for $m < n$.

$$\begin{aligned} T(n) &\leq 2T(n/2) + cn \\ &\leq 2k(n/2)\lg(n/2) + cn \\ &= kn(\lg n - 1) + cn \\ &= kn \lg n - kn + cn \\ &\leq kn \lg n \quad \text{if} \quad k \geq c. \end{aligned}$$

More Recurrence Relations

More recurrences

$$T(n) \le \begin{cases} 2T(n/2) + cn^2 & \texttt{if} \quad n > 2 \\ c & \texttt{otherwise} \end{cases}$$



cn^2

$2c(n/2)^2 = \frac{cn^2}{2}$

$4c(n/4)^2 = \frac{cn^2}{4}$

....

$2^i c(n/2^i)^2 = \frac{cn^2}{2^i}$

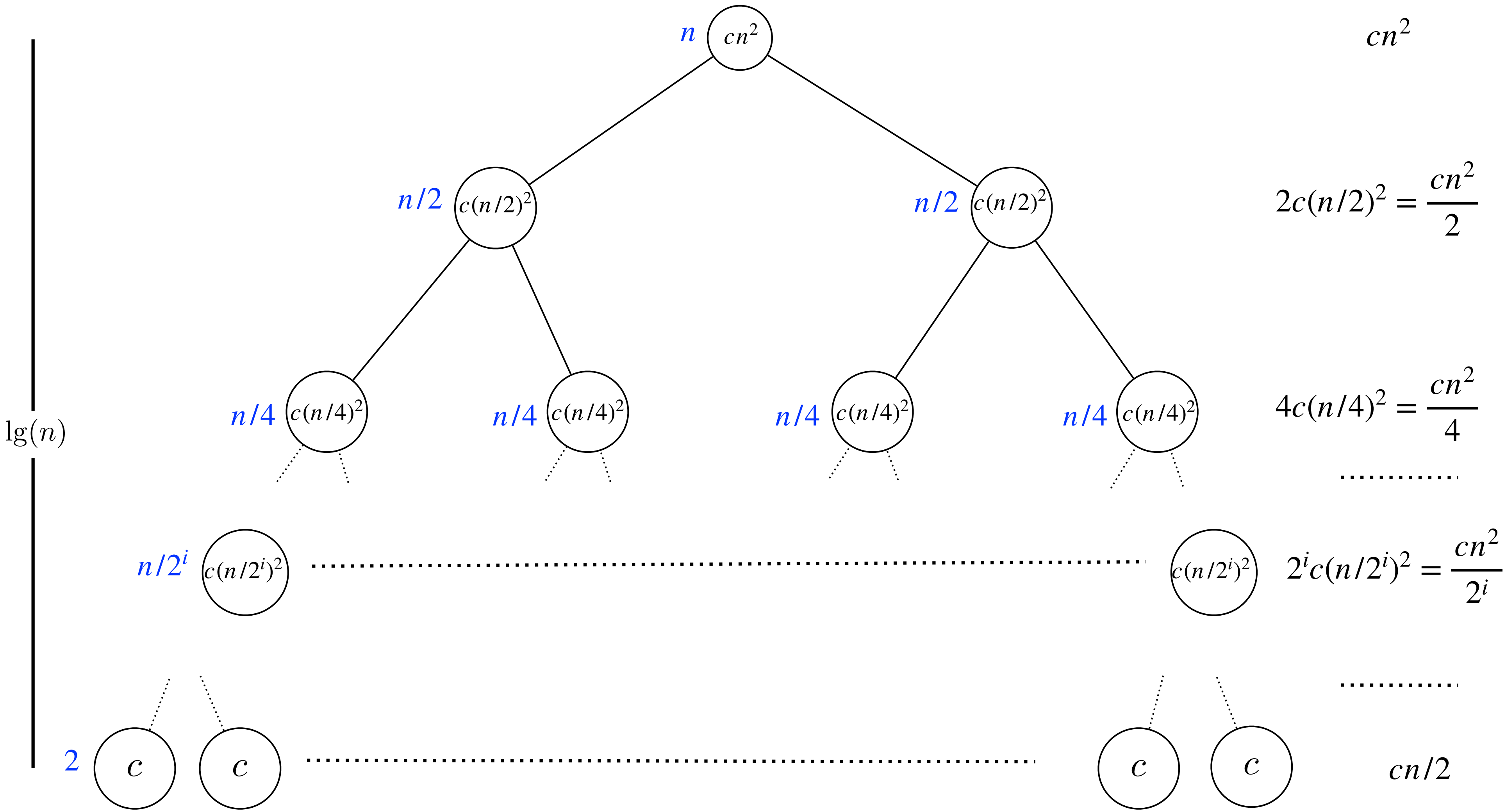
....

$cn/2$

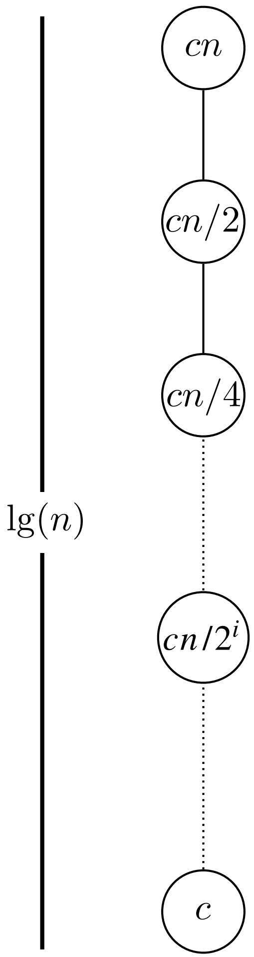
More recurrences

$$T(n) \le \begin{cases} 2T(n/2) + cn^2 & \texttt{if} \quad n > 2 \\ c & \texttt{otherwise} \end{cases}$$

$$T(n) \le \sum_{i=0}^{\log_2 n} \frac{cn^2}{2^i} \le cn^2 \sum_{i=0}^{\log_2 n} \frac{1}{2^i} \le 2cn^2$$



More recurrence relations: 1 subproblem



$$T(n) \le \begin{cases} T(n/2) + cn & \text{if } n > 2 \\ c & \text{otherwise} \end{cases}$$

- Summing over all levels:

$$T(n) \le \sum_{i=0}^{\lg n - 1} \frac{cn}{2^i} = cn \sum_{i=0}^{\lg n-1} \frac{1}{2^i} \le 2cn = O(n)$$

- Substitution: Guess $T(n) \le kn$
  - Base case:

$$k \cdot 2 \ge c = T(2) \quad \text{if} \quad k \ge c/2.$$

  - Assume $T(m) \le km$ for $m < n$.

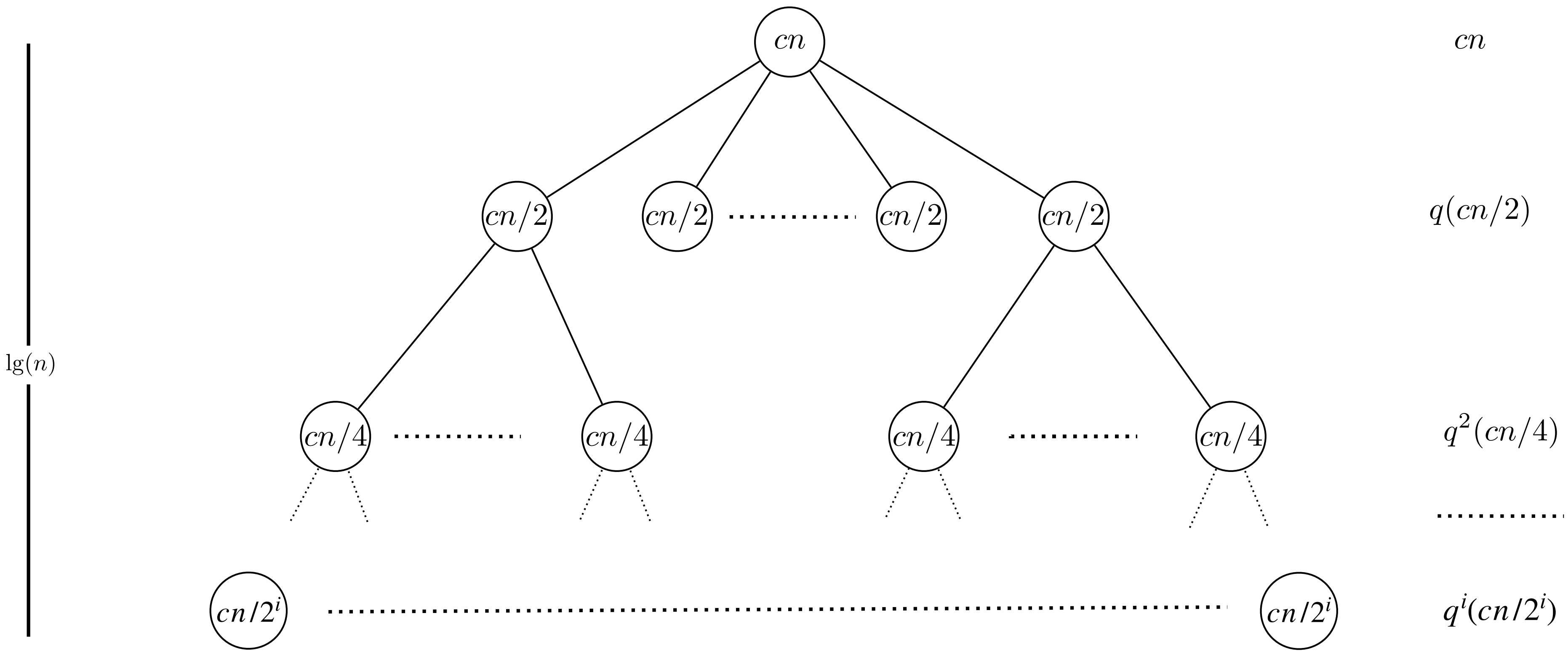
$$T(n) \le T(n/2) + cn \le k(n/2) + cn = (k/2)n + cn$$

$$\le kn \quad \text{if} \quad c \le k/2.$$

More than 2 subproblems

- q subproblems of size n/2.

$$T(n) \leq \begin{cases} qT(n/2) + cn & \text{if } n > 2 \\ c & \text{otherwise} \end{cases}$$



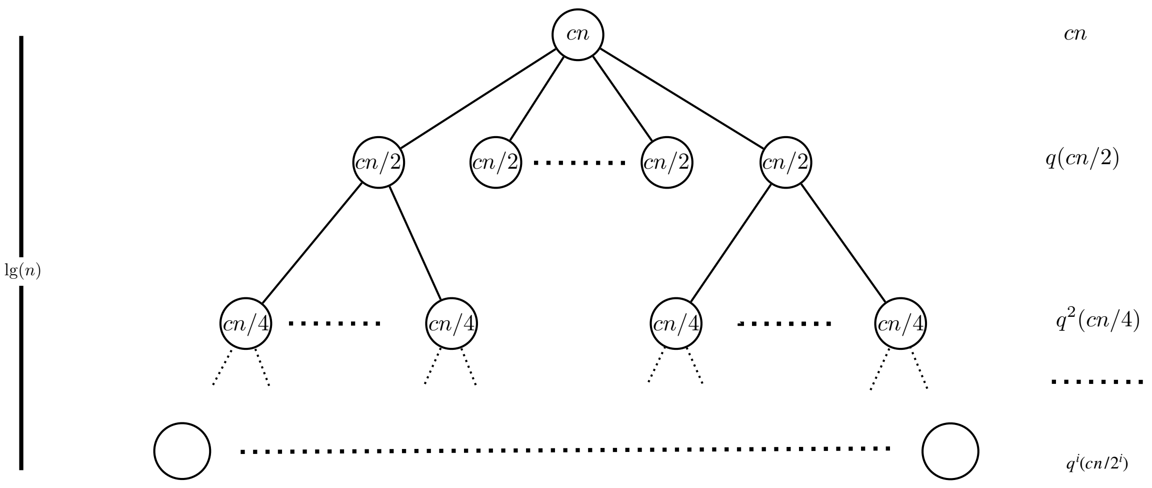
More than 2 subproblems

- q subproblems of size n/2.

$$T(n) \le \begin{cases} qT(n/2) + cn & \text{if } n > 2 \\ c & \text{otherwise} \end{cases}$$

- Summing over all levels:

$$T(n) \le \sum_{j=0}^{\lg n - 1} \left(\frac{q}{2}\right)^j cn = cn \sum_{j=0}^{\lg n - 1} \left(\frac{q}{2}\right)^j$$



Geometric series.

for $x \neq 1 : \sum_{i=0}^{m} x^i = \frac{x^{m+1} - 1}{x - 1}$

for $x < 1 : \sum_{i=0}^{\infty} x^i = \frac{1}{1 - x}$

More than 2 subproblems

Proof of $cn \sum_{j=0}^{\lg n - 1} \left(\frac{q}{2}\right)^j = O(n^{\lg q})$

Geometric series.

for $x \neq 1 : \sum_{i=0}^{m} x^i = \frac{x^{m+1} - 1}{x - 1}$

for $x < 1 : \sum_{i=0}^{\infty} x^i = \frac{1}{1 - x}$

Use geometric series: $$cn \sum_{j=0}^{\lg n - 1} \left(\frac{q}{2}\right)^j = cn \frac{\left(\frac{q}{2}\right)^{\lg n} - 1}{\frac{q}{2} - 1}$$

Reduce $$\left(\frac{q}{2}\right)^{\lg n} = \frac{q^{\lg n}}{2^{\lg n}} = \frac{q^{\lg n}}{n}$$

Now:

$$cn \frac{\left(\frac{q}{2}\right)^{\lg n} - 1}{\frac{q}{2} - 1} = cn \frac{\frac{q^{\lg n}}{n} - 1}{\frac{q-2}{2}} = \frac{2c}{q-2} n \left(\frac{q^{\lg n}}{n} - 1\right) = \underbrace{\boxed{\frac{2c}{q-2}}}_{\text{constant}} \left(q^{\lg n} - n\right) = O(q^{\lg n})$$

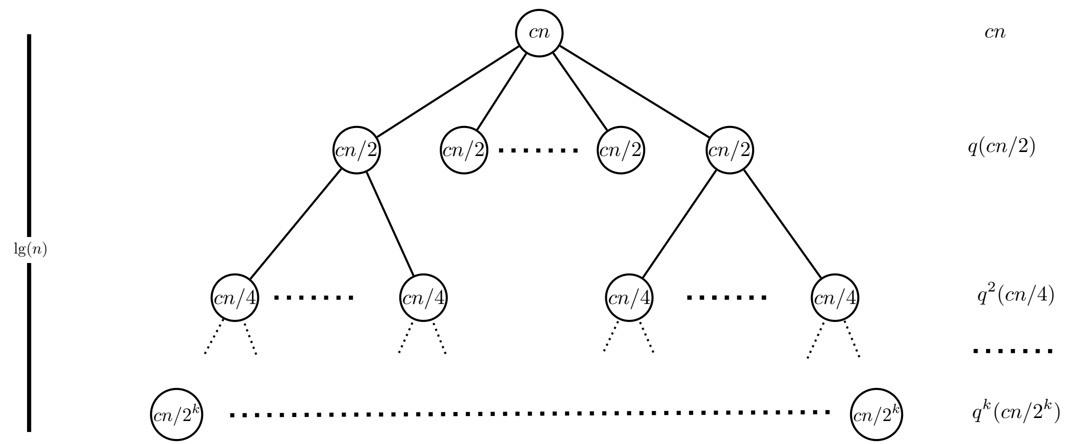
More than 2 subproblems

- q subproblems of size n/2.

$$T(n) \leq \begin{cases} qT(n/2) + cn & \text{if } n > 2 \\ c & \text{otherwise} \end{cases}$$

- Summing over all levels:

$$T(n) \;\leq\; \sum_{j=0}^{\lg n-1} \left(\frac{q}{2}\right)^j cn \;=\; cn \sum_{j=0}^{\lg n-1} \left(\frac{q}{2}\right)^j \;=\; O(n^{\lg q})$$



Geometric series.

for $x \neq 1 : \sum_{i=0}^{m} x^i = \frac{x^{m+1} - 1}{x - 1}$

for $x < 1 : \sum_{i=0}^{\infty} x^i = \frac{1}{1 - x}$

Integer Multiplication

Integer multiplication

- Add. Given two n-bit integers a and b, compute a + b.
- School method. $\Theta(n)$ bit operations.

1	0	1	1	1	
	1	0	0	1	1
+	1	0	1	1	1
1	0	1	0	1	0

- Multiply. Given two n-bit integers a and b, compute a × b.
- School method. $\Theta(n^2)$ bit operations.

1	1	0	×	1	1	1
				0	0	0
+			1	1	1	0
+		1	1	1	0	0
	1	0	1	0	1	0

Integer multiplication: warmup

- Divide-and-conquer: divide the n-bit integers into two.

$$x = \underbrace{1000}_{x_1}\underbrace{1101}_{x_0} \qquad y = \underbrace{1110}_{y_1}\underbrace{0001}_{y_0}$$

$$x = 2^{n/2} \cdot x_1 + x_0$$
$$y = 2^{n/2} \cdot y_1 + y_0$$

- First try:

$$x \cdot y = (2^{n/2} \cdot x_1 + x_0)(2^{n/2} \cdot y_1 + y_0) = 2^n \cdot x_1 y_1 + 2^{n/2} \cdot (x_1 y_0 + x_0 y_1) + x_0 y_0$$

  - Multiply four n/2-bit integers (recursively)
  - Add two n/2-bit integers
  - Shift and add to obtain result.

$$T(n) = \underbrace{4T(n/2)}_{\text{recursive calls}} + \underbrace{cn}_{\text{add, shift}}$$

$$T(n) = O(n^{\lg 4}) = O(n^2)$$

Integer multiplication: Karatsuba

- Divide-and-conquer: divide the n-bit integers into two.

$$x = \underbrace{1000}_{x_1}\underbrace{1101}_{x_0} \qquad y = \underbrace{1110}_{y_1}\underbrace{0001}_{y_0}$$

$$x = 2^{n/2} \cdot x_1 + x_0$$
$$y = 2^{n/2} \cdot y_1 + y_0$$

$$\begin{aligned} x \cdot y &= 2^n \cdot x_1 y_1 + 2^{n/2} \cdot (x_1 y_0 + x_0 y_1) + x_0 y_0 \\ &= 2^n \cdot \underset{1}{x_1 y_1} + 2^{n/2} \cdot \big(\underset{2}{(x_1 + x_0)(y_1 + y_0)} - \underset{1}{x_1 y_1} - \underset{3}{x_0 y_0}\big) + \underset{3}{x_0 y_0} \end{aligned}$$

- Karatsuba:
  - Recursively compute *three* products of n/2-bit integers:
    - $x_1 y_1, (x_1 + x_0)(y_1 + y_0), x_0 y_0$
  - Shift, add, and subtract to obtain result.

$$(x_1 + x_0)(y_1 + y_0) =$$
$$x_1 y_1 + x_1 y_0 + x_0 y_1 + x_0 y_0$$
$$\Rightarrow$$
$$x_1 y_0 + x_0 y_1 =$$
$$(x_1 + x_0)(y_1 + y_0) - x_1 y_1 - x_0 y_0$$

$$T(n) = \underbrace{3T(n/2)}_{\text{recursive calls}} + \underbrace{cn}_{\text{add, shift}}$$

$$T(n) = O(n^{\lg 3}) = O(n^{1.59})$$