Divide-and-Conquer

Inge Li Gørtz

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

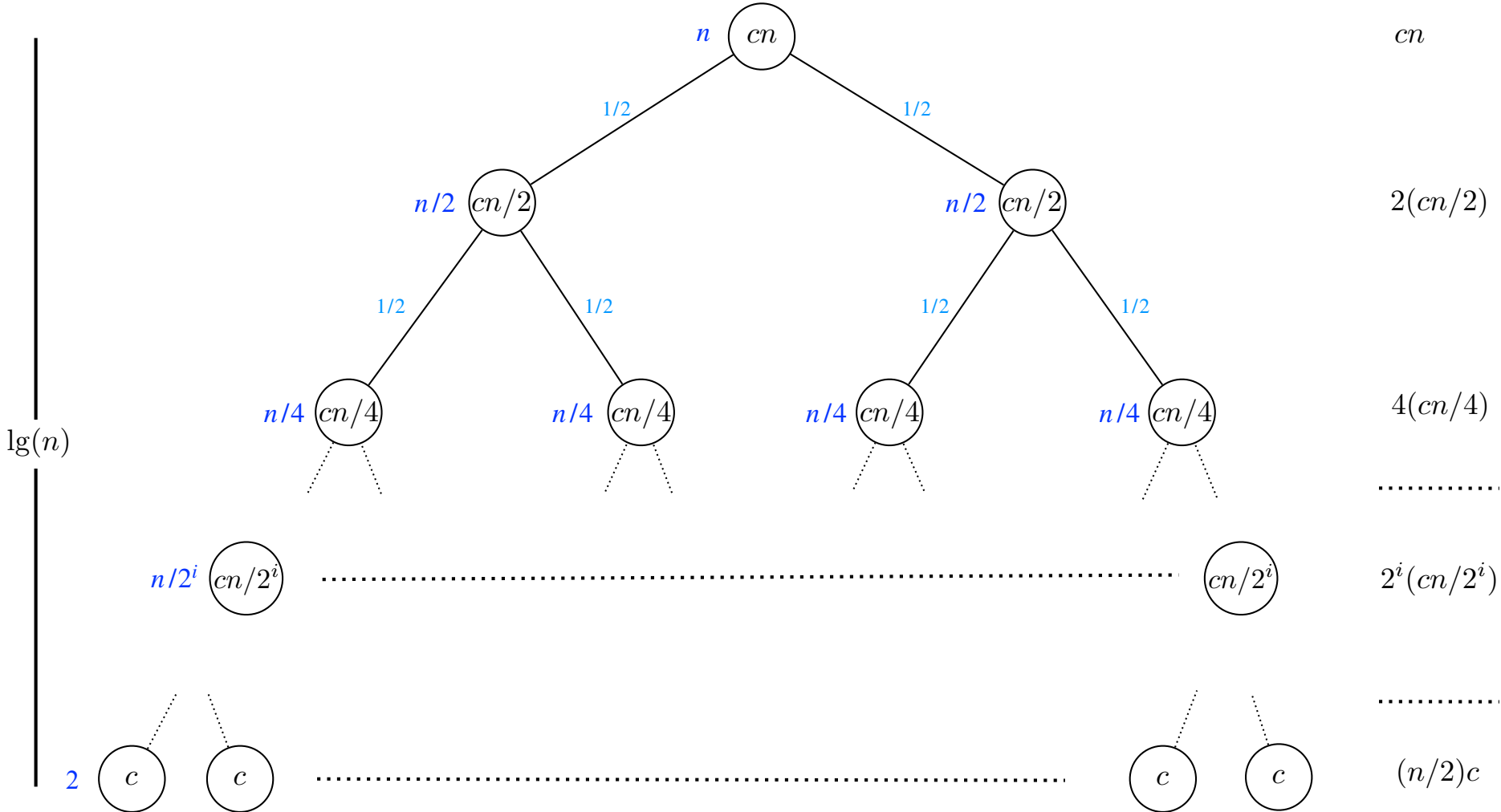 - Integer multiplication

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

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

Mergesort recurrence: substitution

$$T(n) \leq \begin{cases} 2T(n/2) + cn & \text{if } n > 2 \\ c & \text{otherwise} \end{cases}$$
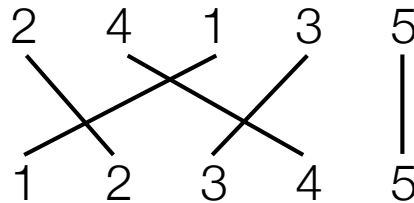
- Substitute $T(n)$ with $kn \lg n$ and use induction to prove $T(n) \leq n \lg nk$.

- 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 $m < n$.

$$T(n) \leq 2T(n/2) + cn$$

$$\leq 2k(n/2)\lg(n/2) + cn$$

$$= kn(\lg n - 1) + cn$$

$$= kn \lg n - kn + cn$$

$$\leq kn \lg n \quad \text{if} \quad k \geq c.$$

Counting Inversions

Counting Inversions

- Given sequence (permutation) $a_1, a_2, \ldots, a_n$ of the numbers from 1 to n.

- Inversion: a_i and a_j inverted if $i < j$ and $a_i > a_j$.

- Applications:

  - Comparing preferences (e.g. on a music site).

  - Voting theory

  - Collaborative filtering

  - Measuring the "sortedness" of an array.

  - Sensitivity of Google's ranking function.

Counting Inversions

- Given sequence (permutation) $a_1, a_2, \ldots, a_n$ of the numbers from 1 to n.

- Inversion: a_i and a_j inverted if $i < j$ and $a_i > a_j$.

10	8	14	4	15	1	9	5	16	7	3	12	13	2	11	6

- Brute-force:

  - Compare each a_i with each a_j, where $i < j$.

Counting Inversions

- Given sequence (permutation) $a_1, a_2, \ldots, a_n$ of the numbers from 1 to n.

- Inversion: a_i and a_j inverted if $i < j$ and $a_i > a_j$.

10	8	14	4	15	1	9	5	16	7	3	12	13	2	11	6

- Brute-force:

  - Compare each a_i with each a_j, where $i < j$.

Counting Inversions

- Given sequence (permutation) $a_1, a_2, \ldots, a_n$ of the numbers from 1 to n.

- Inversion: a_i and a_j inverted if $i < j$ and $a_i > a_j$.

10	8	14	4	15	1	9	5	16	7	3	12	13	2	11	6

- Brute-force:

  - Compare each a_i with each a_j, where $i < j$.

Counting Inversions

- Given sequence (permutation) $a_1, a_2, \ldots, a_n$ of the numbers from 1 to n.

- Inversion: a_i and a_j inverted if $i < j$ and $a_i > a_j$.

10	8	14	4	15	1	9	5	16	7	3	12	13	2	11	6

- Brute-force:

  - Compare each a_i with each a_j, where $i < j$.

Counting Inversions

- Given sequence (permutation) $a_1, a_2, \ldots, a_n$ of the numbers from 1 to n.

- Inversion: a_i and a_j inverted if $i < j$ and $a_i > a_j$.

10	8	14	*4*	15	*1*	9	*5*	16	*7*	*3*	12	13	*2*	11	*6*

- Brute-force:

  - Compare each a_i with each a_j, where $i < j$.

Counting Inversions

- Given sequence (permutation) $a_1, a_2, \ldots, a_n$ of the numbers from 1 to n.

- Inversion: a_i and a_j inverted if $i < j$ and $a_i > a_j$.

10	8	14	4	15	1	9	5	16	7	3	12	13	2	11	6

- Brute-force:

  - Compare each a_i with each a_j, where $i < j$.

Counting Inversions

- Given sequence (permutation) $a_1, a_2, \ldots, a_n$ of the numbers from 1 to n.

- Inversion: a_i and a_j inverted if $i < j$ and $a_i > a_j$.

| 10 | 8 | 14 | 4 | 15 | 1 | 9 | 5 | 16 | 7 | 3 | 12 | 13 | 2 | 11 | 6 |

- Brute-force:

  - Compare each a_i with each a_j, where $i < j$.

  - Time: $O(n^2)$

Counting Inversions

- Given sequence (permutation) $a_1, a_2, \ldots, a_n$ of the numbers from 1 to n.

- Inversion: a_i and a_j inverted if $i < j$ and $a_i > a_j$.

10	8	14	4	15	1	9	5	16	7	3	12	13	2	11	6

- Divide-and-Conquer:

  - Divide: Split list in two.

Counting Inversions

- Given sequence (permutation) $a_1, a_2, \ldots, a_n$ of the numbers from 1 to n.

- Inversion: a_i and a_j inverted if $i < j$ and $a_i > a_j$.

10	8	14	4	15	1	9	5	16	7	3	12	13	2	11	6

10	8	14	4	15	1	9	5		16	7	3	12	13	2	11	6

- Divide-and-Conquer:

    - Divide: Split list in two.

    - Conquer: recursively count inversions in each half.

Counting Inversions

- Given sequence (permutation) $a_1, a_2, \ldots, a_n$ of the numbers from 1 to n.

- Inversion: a_i and a_j inverted if $i < j$ and $a_i > a_j$.

10	8	14	4	15	1	9	5	16	7	3	12	13	2	11	6

10	8	14	4	15	1	9	5

17

16	7	3	12	13	2	11	6

18

- Divide-and-Conquer:

  - Divide: Split list in two.

  - Conquer: recursively count inversions in each half.

Counting Inversions

- Given sequence (permutation) $a_1, a_2, \ldots, a_n$ of the numbers from 1 to n.

- Inversion: a_i and a_j inverted if $i < j$ and $a_i > a_j$.

10	8	14	4	15	1	9	5	16	7	3	12	13	2	11	6

10	8	14	4	15	1	9	5

17

16	7	3	12	13	2	11	6

18

- Divide-and-Conquer:

  - Divide: Split list in two.

  - Conquer: recursively count inversions in each half.

  - Combine:

   - count inversions where a_i and a_j are in different halves

    - return sum.

$$17 + 18 + 30 = 65$$

Counting Inversions

- Given sequence (permutation) $a_1, a_2, \ldots, a_n$ of the numbers from 1 to n.

- Inversion: a_i and a_j inverted if $i < j$ and $a_i > a_j$.

10	8	14	4	15	1	9	5	16	7	3	12	13	2	11	6

10	8	14	4	15	1	9	5

17

16	7	3	12	13	2	11	6

18

- Divide-and-Conquer:

  - Divide: Split list in two.   *Divide: O(1)*

  - Conquer: recursively count inversions in each half.   *Conquer: 2T(n/2)*

  - Combine: *Combine: ???*

    - count inversions where a_i and a_j are in different halves

    - return sum.

      $$17 + 18 + 30 = 65$$
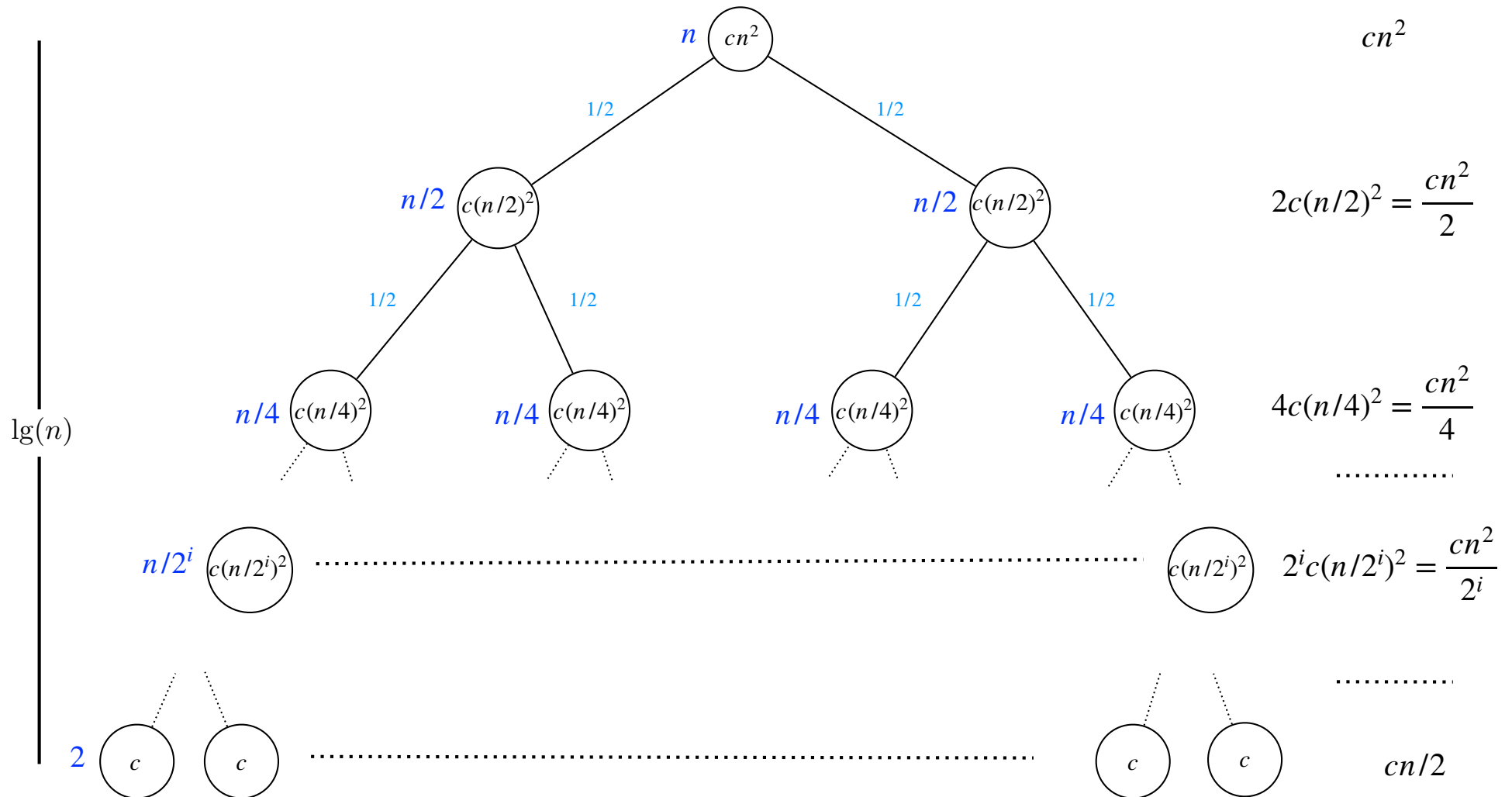
Another recurrence

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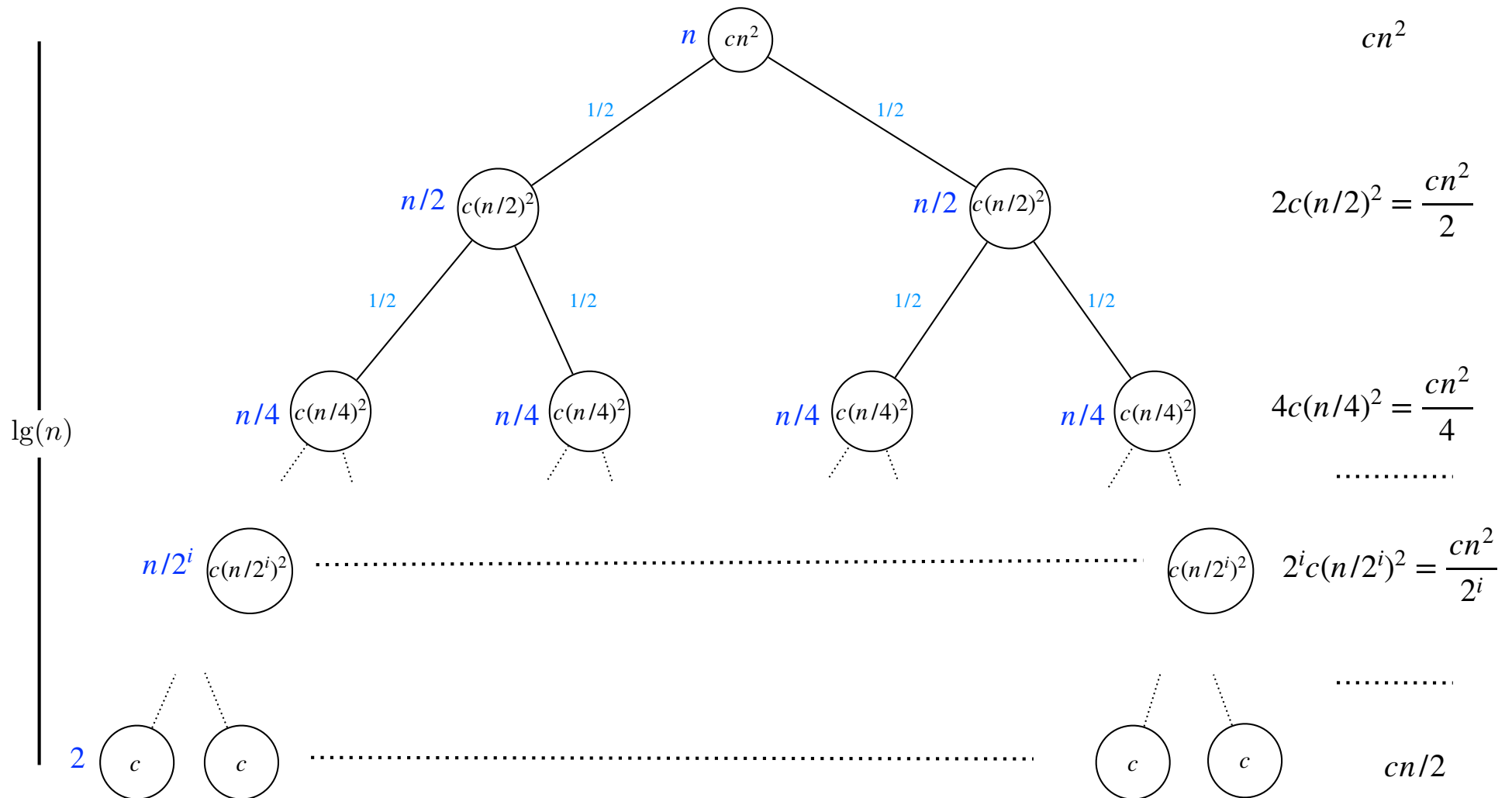
More recurrences

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

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

n cn^2 cn^2

$1/2$ $1/2$

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

$1/2$ $1/2$ $1/2$ $1/2$

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

$\lg(n)$

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

2 c c c c $cn/2$

Counting Inversions

- Given sequence (permutation) $a_1, a_2, \ldots, a_n$ of the numbers from 1 to n.

- Inversion: a_i and a_j inverted if $i < j$ and $a_i > a_j$.

10	8	14	4	15	1	9	5	16	7	3	12	13	2	11	6

10	8	14	4	15	1	9	5

17

16	7	3	12	13	2	11	6

18

- Divide-and-Conquer:

  - Divide: Split list in two.   *Divide: O(1)*

  - Conquer: recursively count inversions in each half.   *Conquer: 2T(n/2)*

  - Combine: *Combine: ???*

    - count inversions where a_i and a_j are in different halves

    - return sum.

$$17 + 18 + 30 = 65$$

Counting Inversions: Combine

- Combine: count inversions where a_i and a_j are in different halves.

Counting Inversions: Combine

- Combine: count inversions where a_i and a_j are in different halves.
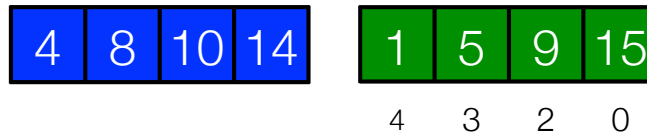
  - Assume each half sorted.

10	8	14	4		15	1	9	5

Counting Inversions: Combine

- Combine: count inversions where a_i and a_j are in different halves.

  - Assume each half sorted.

| 4 | 8 | 10 | 14 | | 1 | 5 | 9 | 15 |

Counting Inversions: Combine

- Combine: count inversions where a_i and a_j are in different halves.

  - Assume each half sorted.

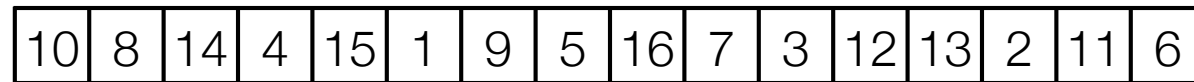| 4	8	10	14		1	5	9	15
					4	3	2	0

Counting Inversions: Combine

- Combine: count inversions where a_i and a_j are in different halves.

  - Assume each half sorted.

  - Merge sorted halves into sorted whole while counting inversions.

4	8	10	14		1	5	9	15

    4   3   2   0

1	4	5	8	9	10	14	15
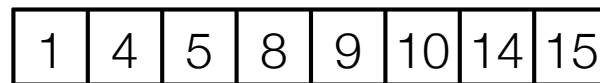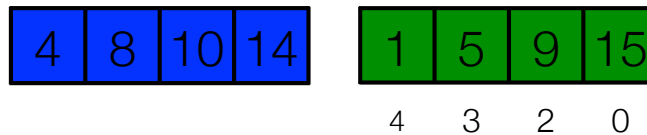
Inversions: 4 + 3 + 2 + 0 = 9

Counting Inversions

- Given sequence (permutation) $a_1, a_2, \ldots, a_n$ of the numbers from 1 to n.

- Inversion: a_i and a_j inverted if $i < j$ and $a_i > a_j$.

| 10 | 8 | 14 | 4 | 15 | 1 | 9 | 5 | 16 | 7 | 3 | 12 | 13 | 2 | 11 | 6 |

| 10 | 8 | 14 | 4 | 15 | 1 | 9 | 5 |
17

| 16 | 7 | 3 | 12 | 13 | 2 | 11 | 6 |
18

- Divide-and-Conquer:

  - Divide: Split list in two.   *Divide: O(1)*

  - Conquer: recursively count inversions in each half.   *Conquer: 2T(n/2)*

  - Combine: *Combine: O(n)*

    - Merge-and-Count.

Counting Inversions: Implementation

```
Sort-and-Count(L):

    if list L has one element:
        return (0, L)

    divide the list L into two halves A and B
    (i_A, A) = Sort-and-Count(A)
    (i_B, B) = Sort-and-Count(B)
    (i_L, L) = Merge-and-Count(A, B)

    i = i_A + i_B + i_L

    return (i, L)
```

- Pre-condition (Merge-and-Count): A and B are sorted.

- Post-condition (Sort-and-Count, Merge-and-Count): L is sorted.

More Recurrence Relations

More recurrence relations: 1 subproblem

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

- Summing over all levels:

$$T(n) \leq \sum_{i=0}^{\lg n - 1} \frac{cn}{2^i} = cn \sum_{i=0}^{\lg n - 1} \frac{1}{2^i} \leq 2cn = O(n)$$
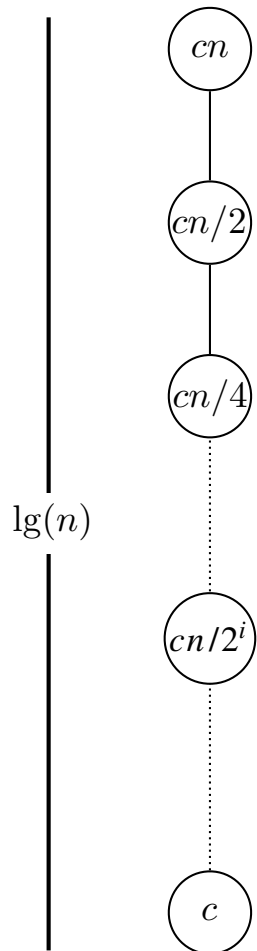
- Substitution: Guess $T(n) \leq kn$

  - Base case:

$$k \cdot 2 \geq c = T(2) \quad \text{if} \quad k \geq c/2.$$
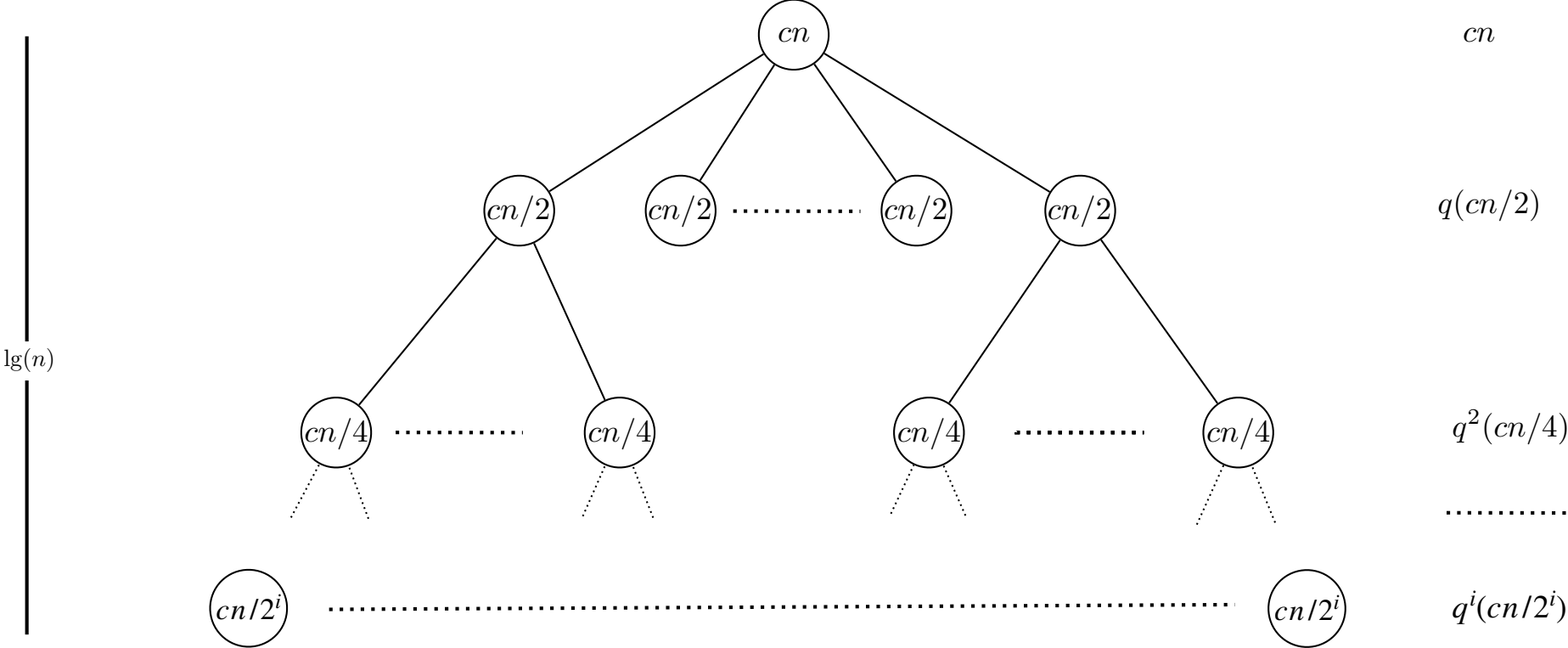
  - Assume $T(m) \leq km$ for $m < n$.

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

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

(Tree diagram on left side showing nodes labeled: cn, $cn/2$, $cn/4$, $cn/2^i$, c, with total height $\lg(n)$)
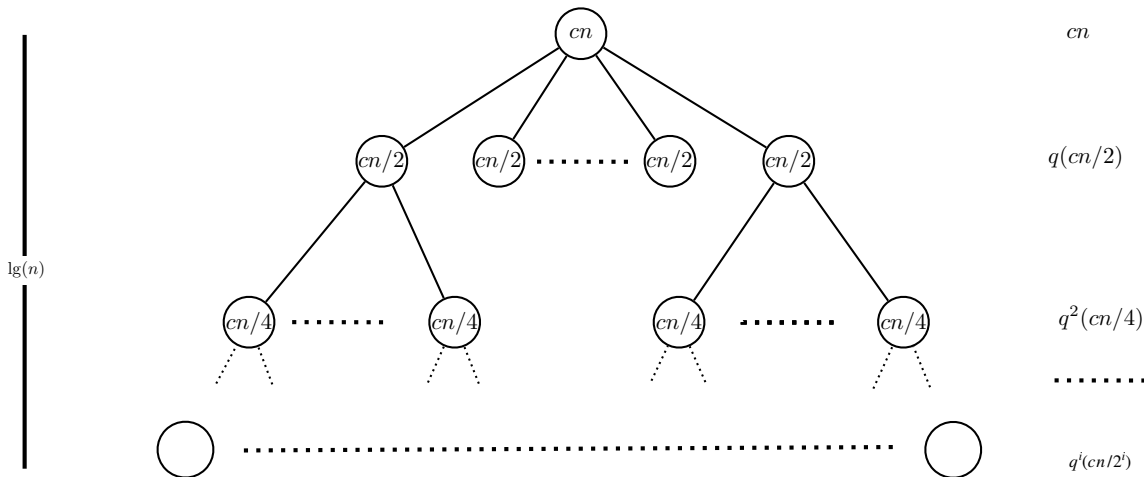
More than 2 subproblems

- q subproblems of size n/2.

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

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:

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

Geometric series.

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

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

More than 2 subproblems

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

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

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

Now:

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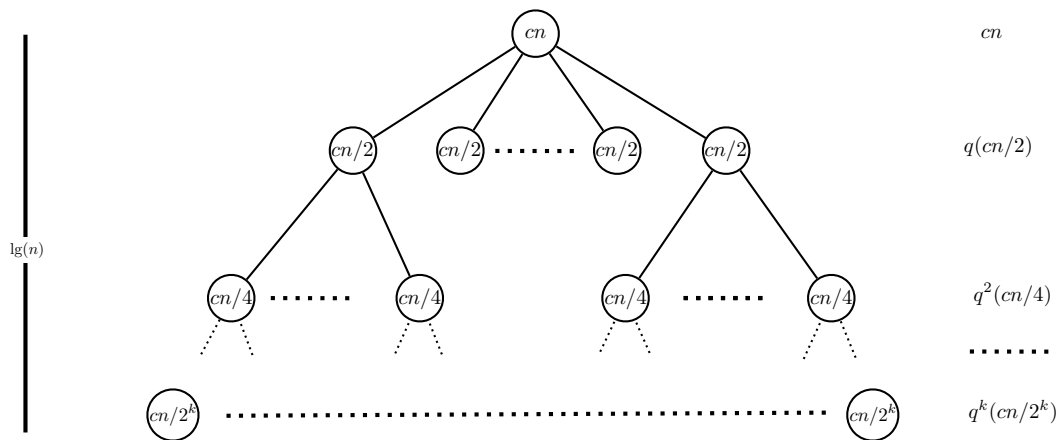
constant

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

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

Subproblems of different sizes

$$T(n) = \begin{cases} T(3n/4) + T(n/2) + f(n) & \text{if } n > 4 \\ c & \text{otherwise} \end{cases}$$

$f(n)$

$f(3n/4) + f(n/2)$

$f(9n/16) + 2f(3n/8) + f(n/4)$