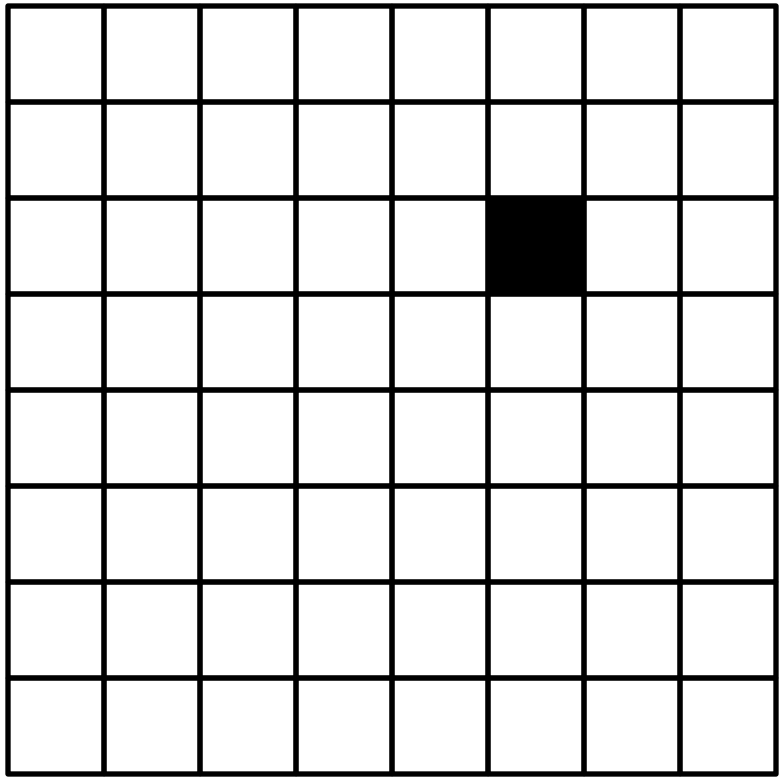
CS4102 Algorithms

Spring 2020

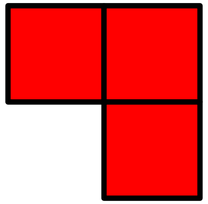
Warm up

Can you cover an 8×8 grid with 1 square missing using "trominoes?"

Can you cover this?



With these?



Office Hours

- Weekly:
  - Mondays and Wednesdays, 11am-12pm
  - Tuesdays 3-4pm
- This Week Only:
  - Friday 1-3pm

Today's Keywords

- Recursion
- Recurrences
- Asymptotic notation
- Divide and Conquer
- Trominoes
- Merge Sort

CLRS Readings

- Chapters 3 & 4

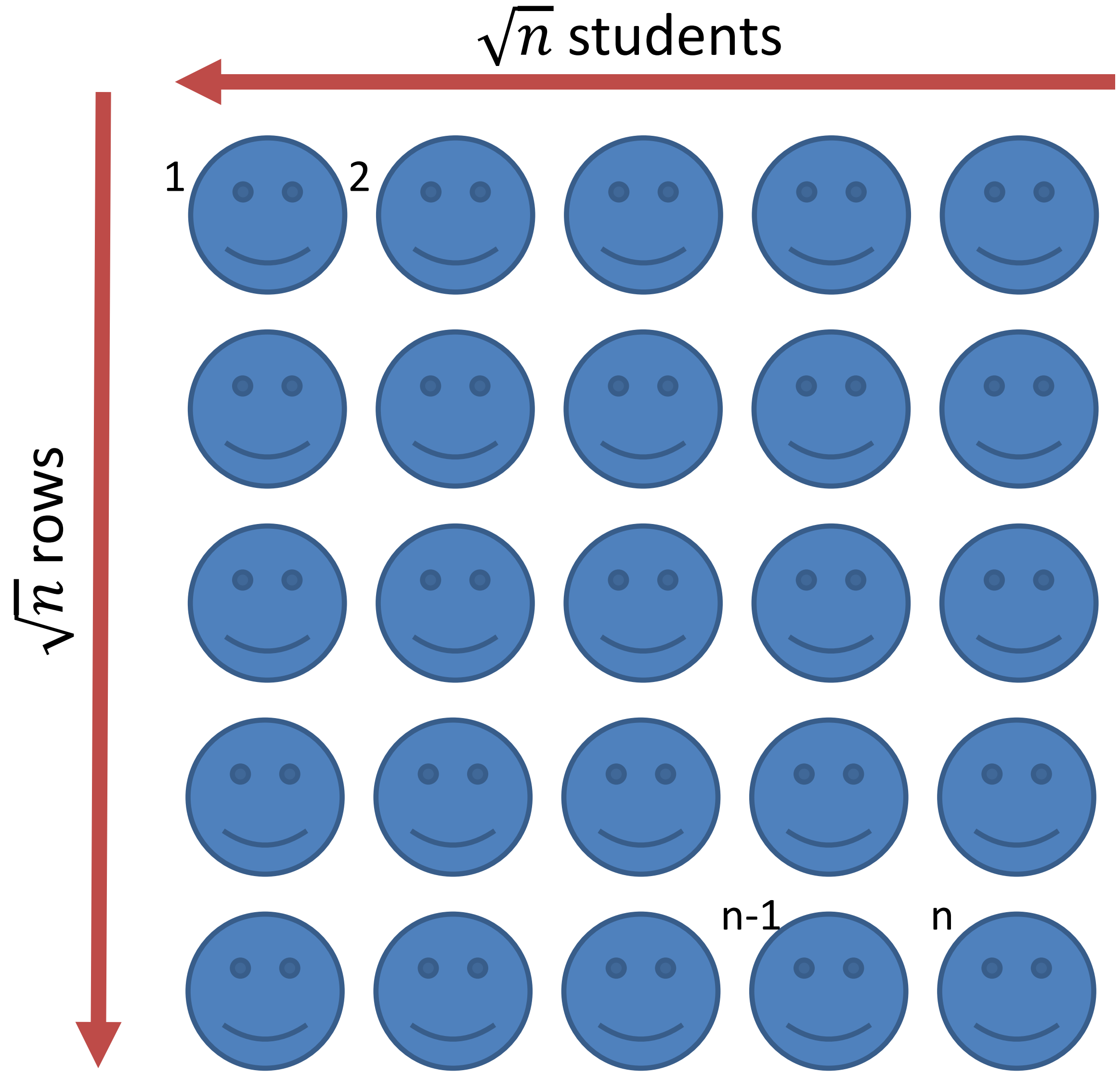
Homeworks

- Hw0 due 11pm Tuesday, Jan 21
  - Submit 2 attachments (zip and pdf)
- Hw1 released next week
  - Written (use Latex!)
  - Asymptotic notation
  - Recurrences
  - Divide and conquer

Attendance

- How many people are here today?
- Naïve algorithm
  1. Everyone stand
  2. Professor walks around counting people
  3. When counted, sit down
- Run time?
  - Class of n students
  - O(n)
- Other suggestions?

Good Attendance



$O(\sqrt{n})$

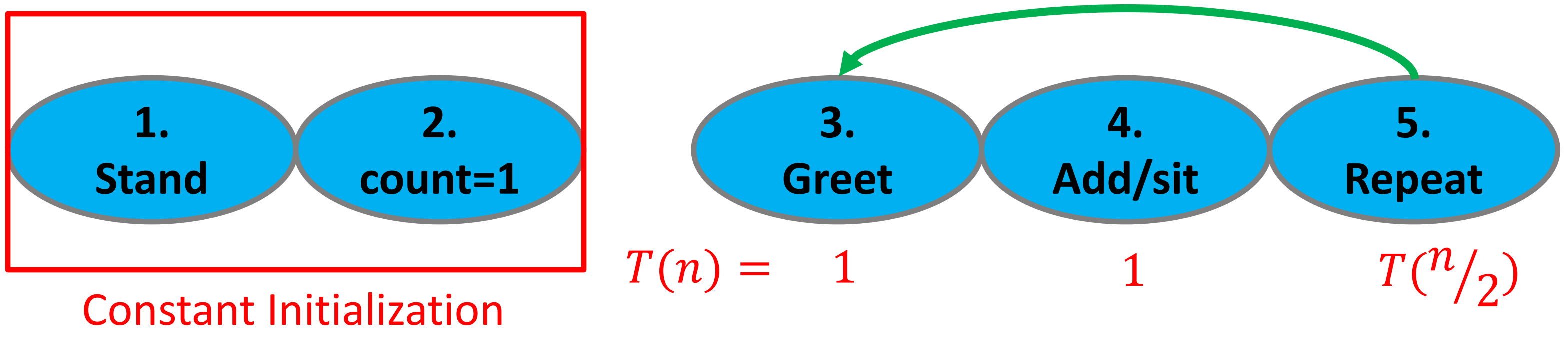
Better Attendance

141

What was the run time of this algorithm?
What are we going to count?

1. Everyone Stand
2. Initialize your “count” to 1
3. Greet a neighbor who is standing: share your name, full date of birth(pause if odd one out)
4. If you are older: give “count” to younger and sit.
   Else if you are younger: add your “count” with older’s
5. If you are standing and have a standing neighbor, go to 3

Attendance Algorithm Analysis



$$T(n) = 1 + 1 + T(^n/_2)$$

How can we "solve" this?

$$T(1) = 3$$

Base case?

Do not need to be exact, asymptotic bound is fine.

Why?

Let's solve the recurrence!

$T(1) = 3$

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

$2 + T(n/4)$

$2 + T(n/8)$

$\ldots$

3

k

Special case: $n = 2^k$

k = log n

$$T(n) = 3 + \sum_{i=0}^{k=\log_2 n} 2 = 2\log_2 n + 3$$

What if $n \neq 2^k$?

- More people in the room → more time
  - $\forall\ 0 < n < m, T(n) < T(m)$, where $2^{\mathrm{k}} < n < 2^{k+1} = m$,
    i.e., $\mathrm{k} < \log n < k + 1$
  - $T(n) \leq T(m) = T\left(2^{k+1}\right) = T\left(2^{\lceil \log_2 n \rceil}\right) = 2\lceil \log_2 n \rceil + 3 = O(\log n)$

These are unimportant. Why?

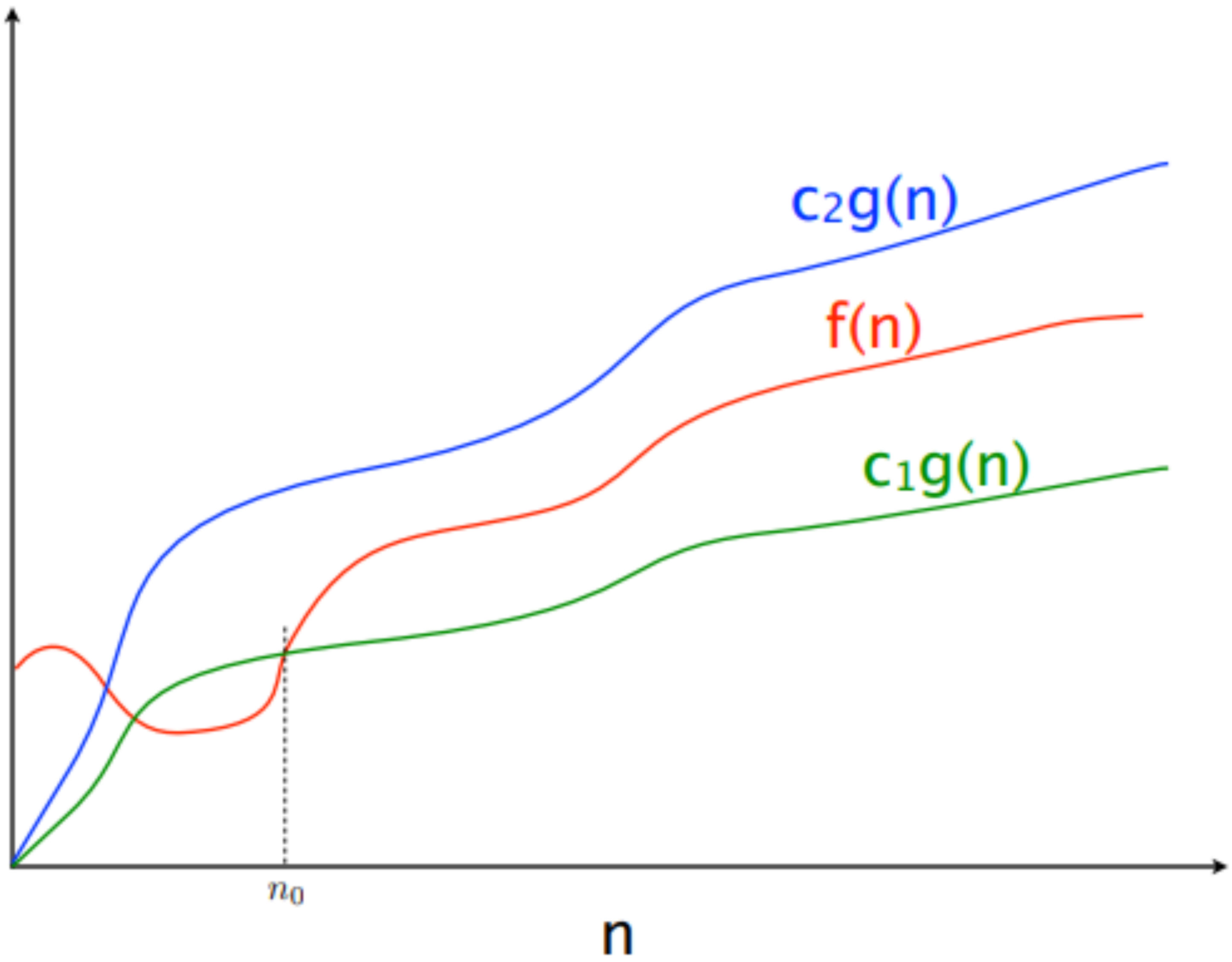
Asymptotic Notation*

- $O(g(n))$
  - At most within constant of g for large n
  - {functions $f \mid \exists$ constants $c, n_0 > 0$ s.t. $\forall n > n_0, f(n) \leq c \cdot g(n)$}

- $\Omega(g(n))$
  - At least within constant of g for large n
  - {functions $f \mid \exists$ constants $c, n_0 > 0$ s.t. $\forall n > n_0, f(n) \geq c \cdot g(n)$}

- $\Theta(g(n))$
  - "Tightly" within constant of g for large n
  - $\Omega(g(n)) \cap O(g(n))$

*CLRS Chapter 3



$f(n) \in O(g(n))$

$f(n) \in \Theta(g(n))$

$f(n) \in \Omega(g(n))$

Asymptotic Notation Example

- Show: $n \log n \in O(n^2)$

to show: $\exists c, n_0 > 0 : \forall n > n_0 \quad n \log n \leq c \cdot n^2$.

Let $c = 1$, $n_0 = 1$. Then $n_0 \log n_0 \leq n_0^2 \cdot c$

$0 = 1 \log 1 \leq 1^2 \cdot 1 = 1$. $0 \leq 1$.

$\forall n > 1$, we know $\log(n) \leq n$ (by def of log). Multiply by n. $\Rightarrow n \log n \leq n^2$

Therefore $n \log n \in O(n^2)$

Asymptotic Notation Example

- To Show: $n \log n \in O(n^2)$

  Direct Proof!

  - **Technique:** Find $c, n_0 > 0$ s.t. $\forall n > n_0, n \log n \leq c \cdot n^2$
  - **Proof:** Let $c = 1, n_0 = 1$. Then,

$$n_0 \log n_0 = (1) \log (1) = 0,$$
$$c\, n_0^2 = 1 \cdot 1^2 = 1,$$
$$0 \leq 1.$$

$$\forall n \geq 1, \log(n) < n \Rightarrow n \log n \leq n^2 \quad \square$$

Asymptotic Notation Example

- Show: $n^2 \notin O(n)$

Assume $n^2 \in O(n)$. Then $\exists\, c, n_0 > 0$ s.t. $\forall n > n_0$ $n^2 \le c \cdot n$

Consider $n = \max(c, n_0) + 1$. That is, $n > c$ and $n > n_0$.

Then $n^2 = n \cdot n > c \cdot n$. contradiction.

Asymptotic Notation Example

- To Show: $n^2 \notin O(n)$
  - **Technique: Contradiction**
  - **Proof:** Assume $n^2 \in O(n)$. Then $\exists c, n_0 > 0 \text{ s.t. } \forall n > n_0, n^2 \leq cn$
    Let us derive constant c. For all $n > n_0 > 0$, we know:
    $cn \geq n^2$,
    $c \geq n$.

    Since c is dependent on n, it is not a constant.
    Contradiction. Therefore $n^2 \notin O(n)$. □

Proof by Contradiction!

Proof Techniques

- Direct Proof ✓
  - From the assumptions and definitions, directly derive the statement
- Proof by Contradiction ✓
  - Assume the statement is true, then find a contradiction
- Proof by Cases
- Induction

Asymptotic Notation

- $o(g(n))$
  - Below *any* constant factor of g for large n
  - $\{\text{functions } f : \forall \text{ constants } c > 0, \exists n_0 \text{ s.t. } \forall n > n_0, f(n) < c \cdot g(n)\}$
  - Set of functions that always grow more slowly than $g(n)$

$\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$

- $\omega(g(n))$
  - Above *any* constant factor of g for large n
  - $\{\text{functions } f : \forall \text{ constants } c > 0, \exists n_0 \text{ s.t. } \forall n > n_0, f(n) > c \cdot g(n)\}$
  - Set of functions that always grow more quickly than $g(n)$

$\lim_{n \to \infty} \frac{g(n)}{f(n)} = 0$

- $\theta(g(n))$?
  - $o(g(n)) \cap \omega(g(n)) = \emptyset$

Asymptotic Notation Example

- $o(g(n)) = \{\text{functions } f : \forall \text{ constants } c > 0, \exists n_0 \text{ s.t. } \forall n > n_0, f(n) < c \cdot g(n)\}$
- Show: $n \log n \in o(n^2)$

Show: $\forall c > 0 \; \exists n_0 \geq 1 : \forall n > n_0 \quad n \log n < c n^2.$

Then $\frac{n \log n}{n^2} < c$

$\frac{\log n}{n} < c.$

$\lim_{n \to \infty} \frac{\log n}{n} = 0$

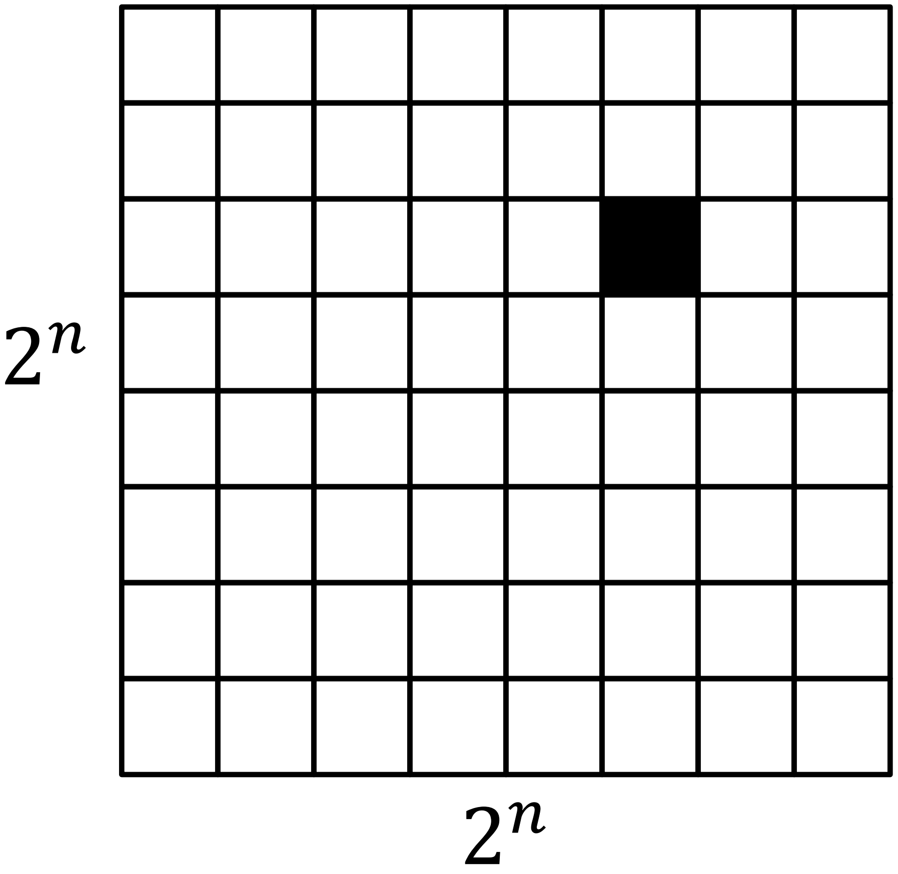
So $n \log n \in o(n^2)$

Asymptotic Notation Example

- $o(g(n)) = \{\text{functions } f : \forall \text{ constants } c > 0, \exists n_0 \text{ s.t. } \forall n > n_0, f(n) < c \cdot g(n)\}$
- Show: $n \log n \in o(n^2)$
  - given any c find a $n_0 > 0$ s.t. $\forall n > n_0, n \log n < c \cdot n^2$
  - Find a value of n in terms of c:
    - $n \log n < c \cdot n^2$
    - $\log n < c \cdot n$
    - $\frac{\log n}{n} < c$
  - For a given c, select any value of n_0 such that $\frac{\log n}{n} < c$

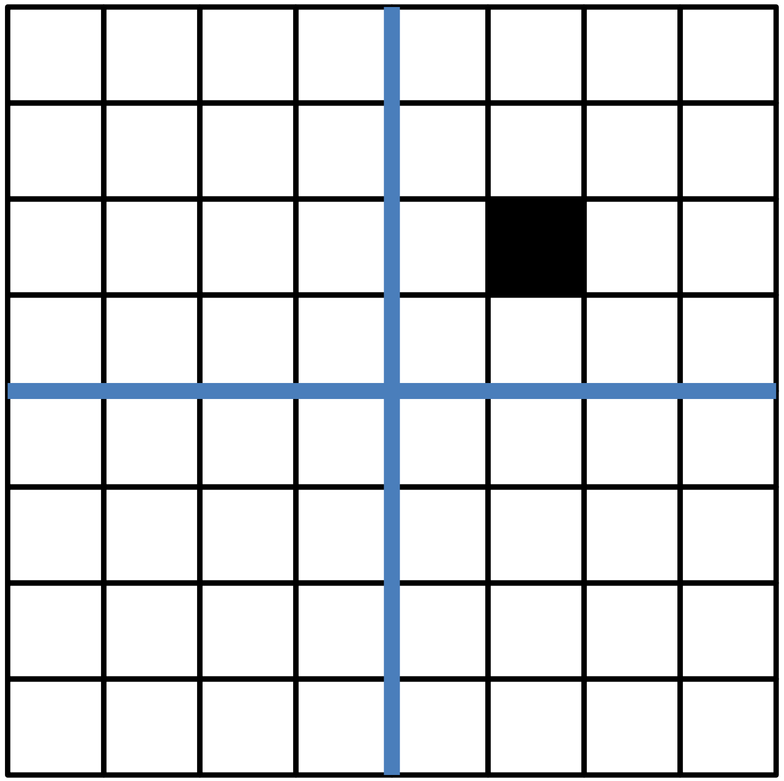
Equivalently: $\lim_{n \to \infty} \frac{n \log n}{n^2} = \lim_{n \to \infty} \frac{\log n}{n} = 0$

Trominoes Puzzle Solution



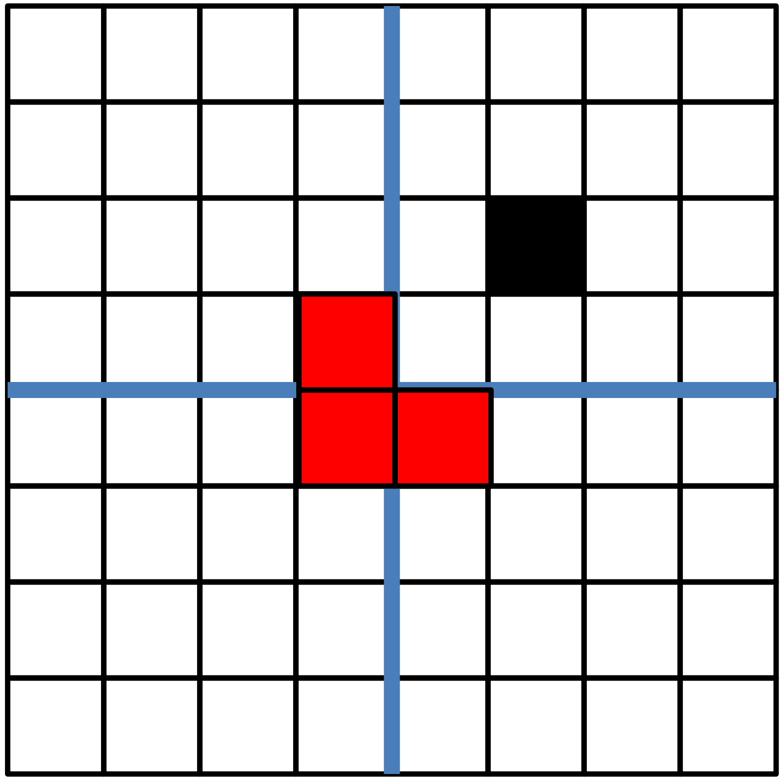
What about larger boards?

Trominoes Puzzle Solution



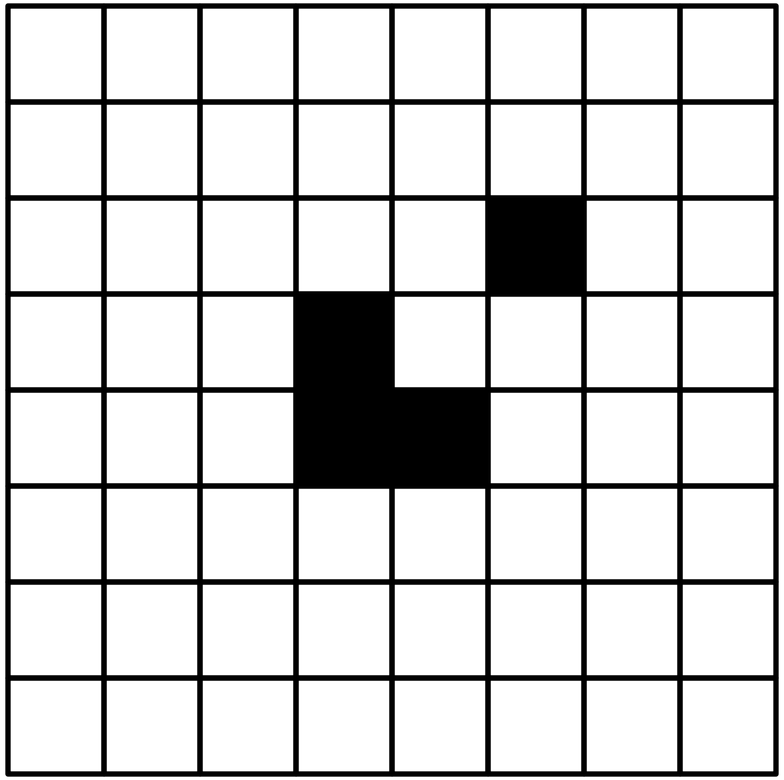
Divide the board into quadrants

Trominoes Puzzle Solution



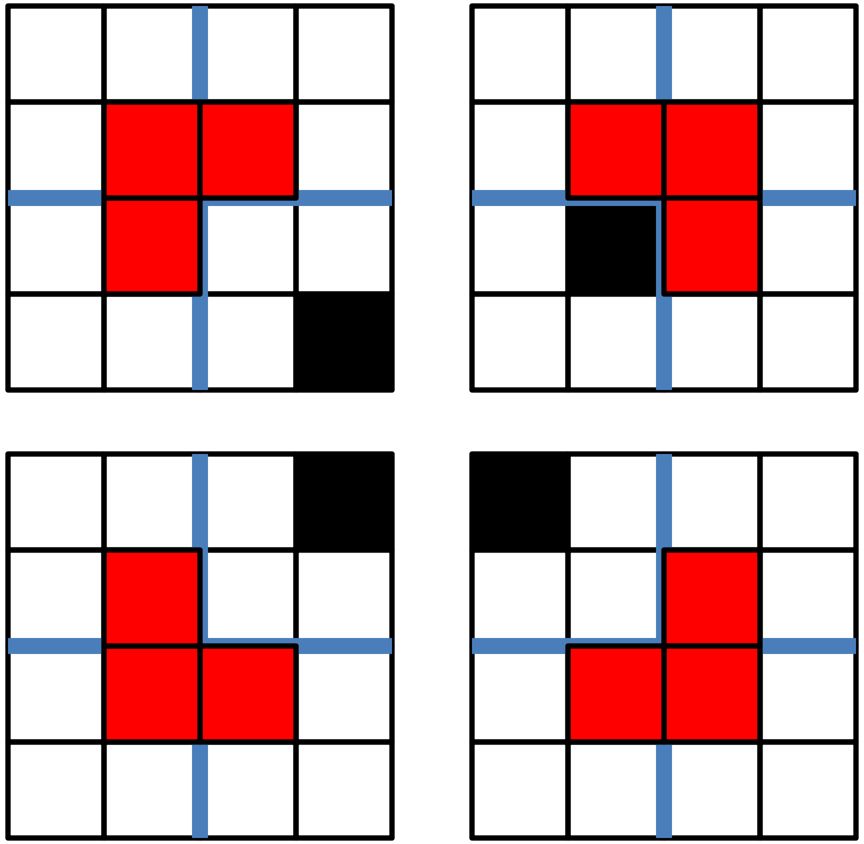
Place a tromino to occupy the three quadrants without the missing piece

Trominoes Puzzle Solution



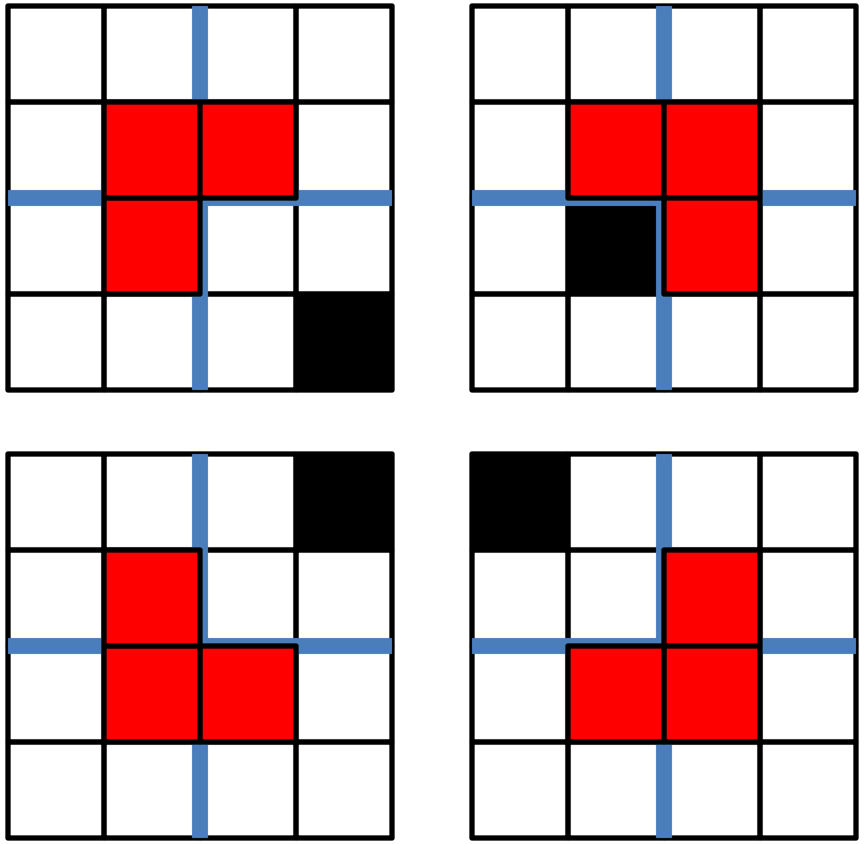
Each quadrant is now a smaller subproblem

Trominoes Puzzle Solution



Solve **Recursively**

Divide and Conquer



Our first algorithmic technique!

Trominoes Puzzle Solution