CS 580
ALGORITHM DESIGN AND ANALYSIS

Divide and Conquer 1

Vassilis Zikas

PLAN

- Discuss D&C
- MergeSort
- Master Theorem
- Counting inversions

THE STRATEGY

- Template:
  1. Break input into k subproblems that are themselves smaller instances of the same problem
  2. Recursively solve the subproblems
  3. Merge the solutions
- Typically all the work is #3

INTRO: MERGESORT

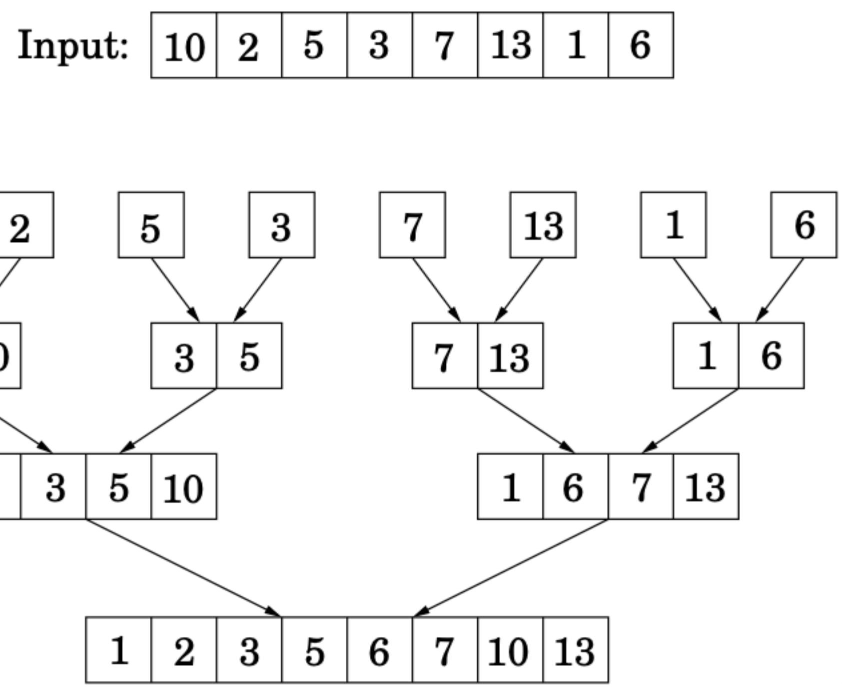
- Input:
  - n numbers
- Desired output:
  - the numbers, sorted in an increasing order

MERGESORT

- Divide the numbers into two sets
- Recurse
- Merge the two solutions

Jon von Neumann (1945)

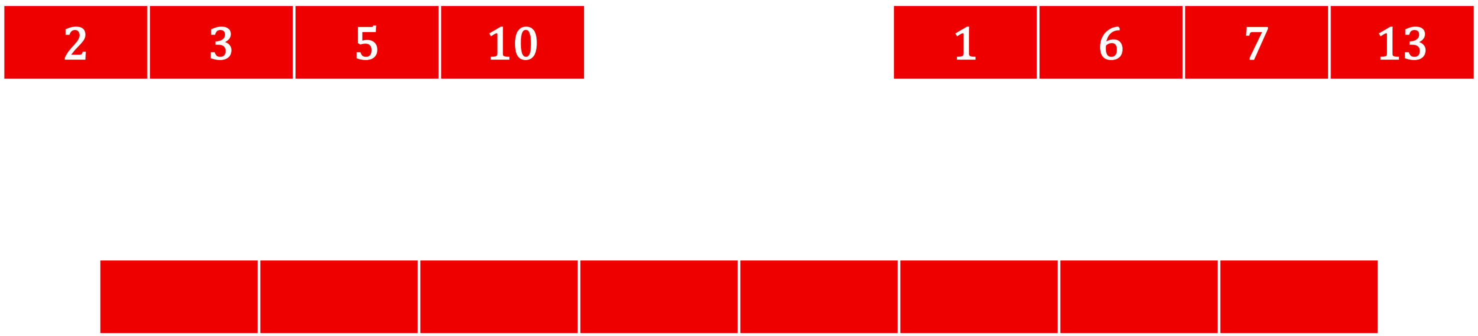
MERGESORT



Input: 10 2 5 3 7 13 1 6
10
2
5
3
7
13
1
6
2 10
3 5
7 13
1 6
2 3 5 10
1 6 7 13
1 2 3 5 6 7 10 13

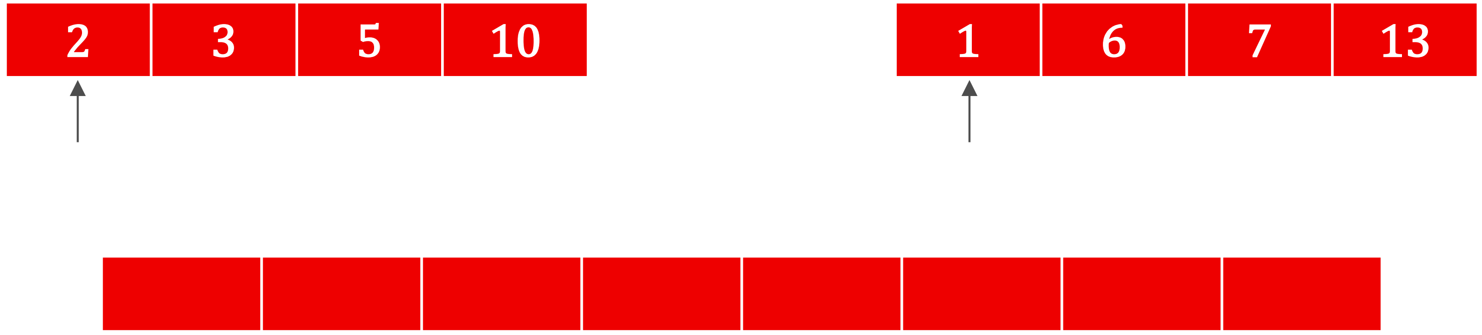
MERGESORT

- How can we merge efficiently?



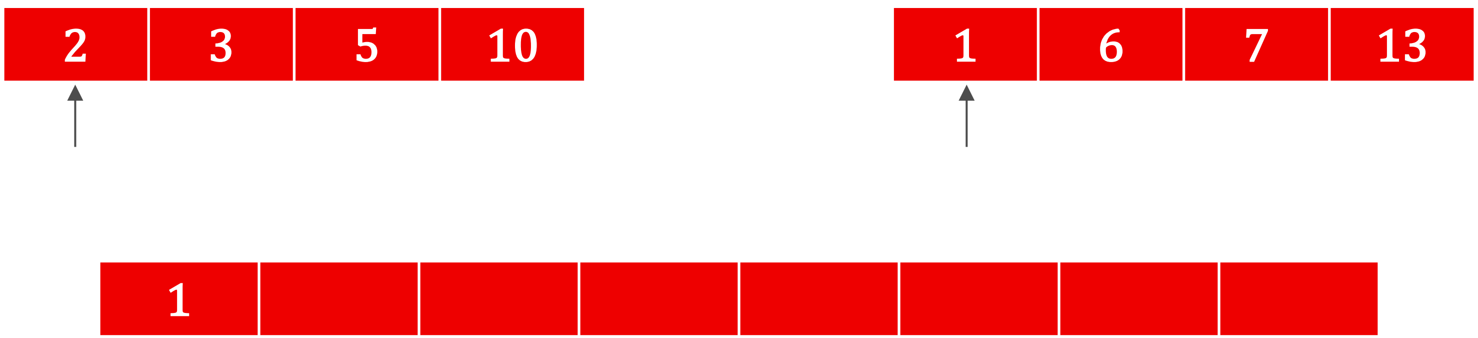
MERGESORT

- How can we merge efficiently?



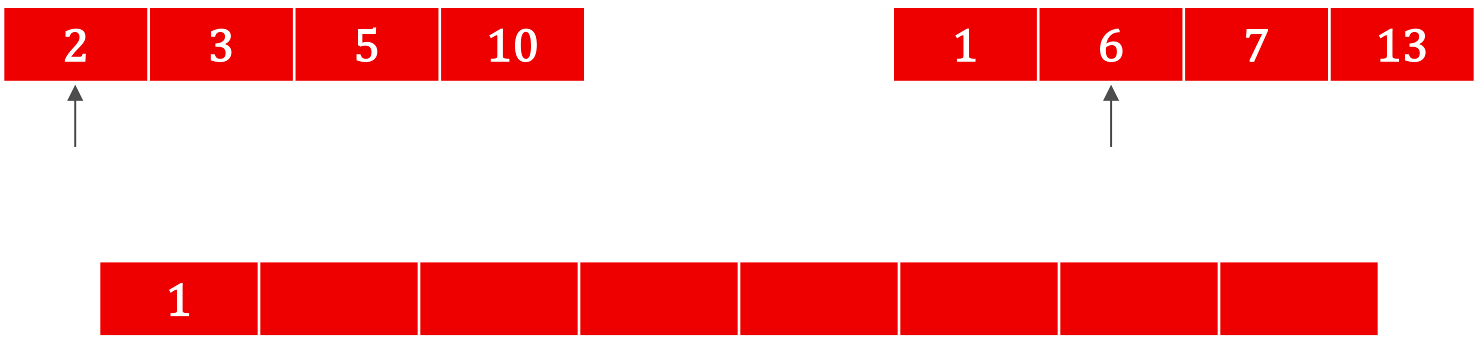
MERGESORT

- How can we merge efficiently?



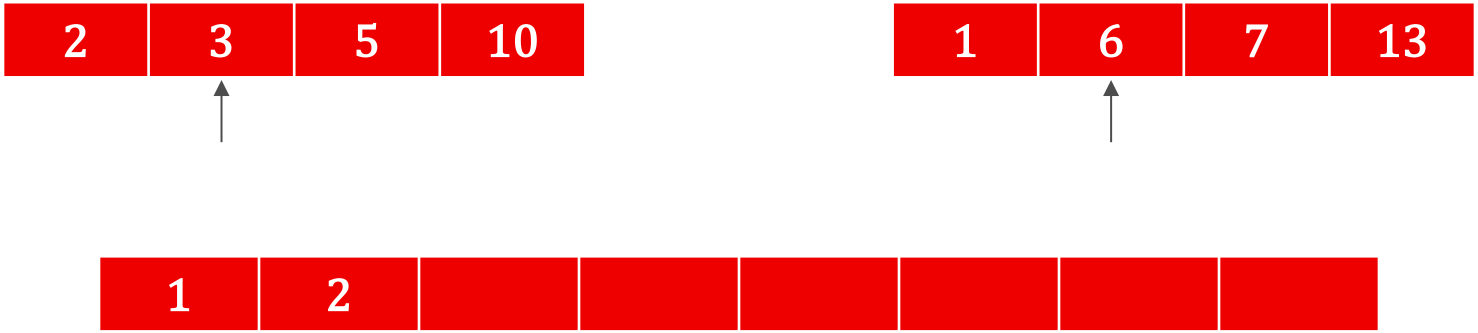
MERGESORT

- How can we merge efficiently?



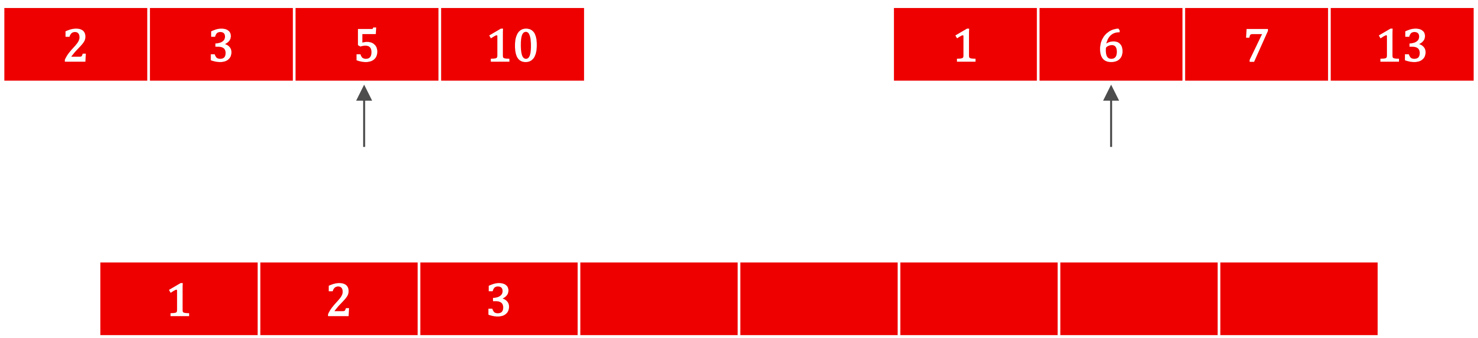
MERGESORT

- How can we merge efficiently?



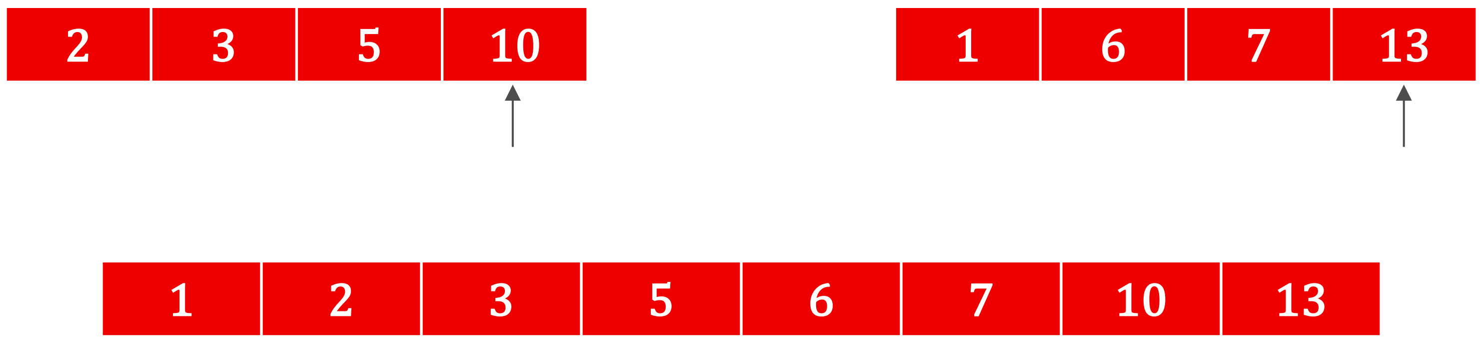
MERGESORT

- How can we merge efficiently?



MERGESORT

- How can we merge efficiently?



- Time $O(n)$

MERGESORT

- Time $T(n)$:
  - Divide - $O(1)$
  - Recurse - $2T(\frac{n}{2})$
  - Merge - $O(n)$
- $T(n) = 2T(\frac{n}{2}) + O(n)$
- $T(n) \in O(nlog(n))$
  - We'll see why next

MASTER THEOREM

<u>**Theorem**</u>: If $T(n) = aT\left(\left\lceil \frac{n}{b} \right\rceil\right) + \mathrm{cn^d}$, for some constants $a, c > 0$, $b > 1$, and $d \geq 0$, then

$$T(n) = \begin{cases} O\left(n^d\right) & \text{if } d > \log_b(a) \\ O(n^d \log(n)) & \text{if } d = \log_b(a) \\ O\left(n^{\log_b(a)}\right) & \text{if } d < \log_b(a) \end{cases}$$

MASTER THEOREM

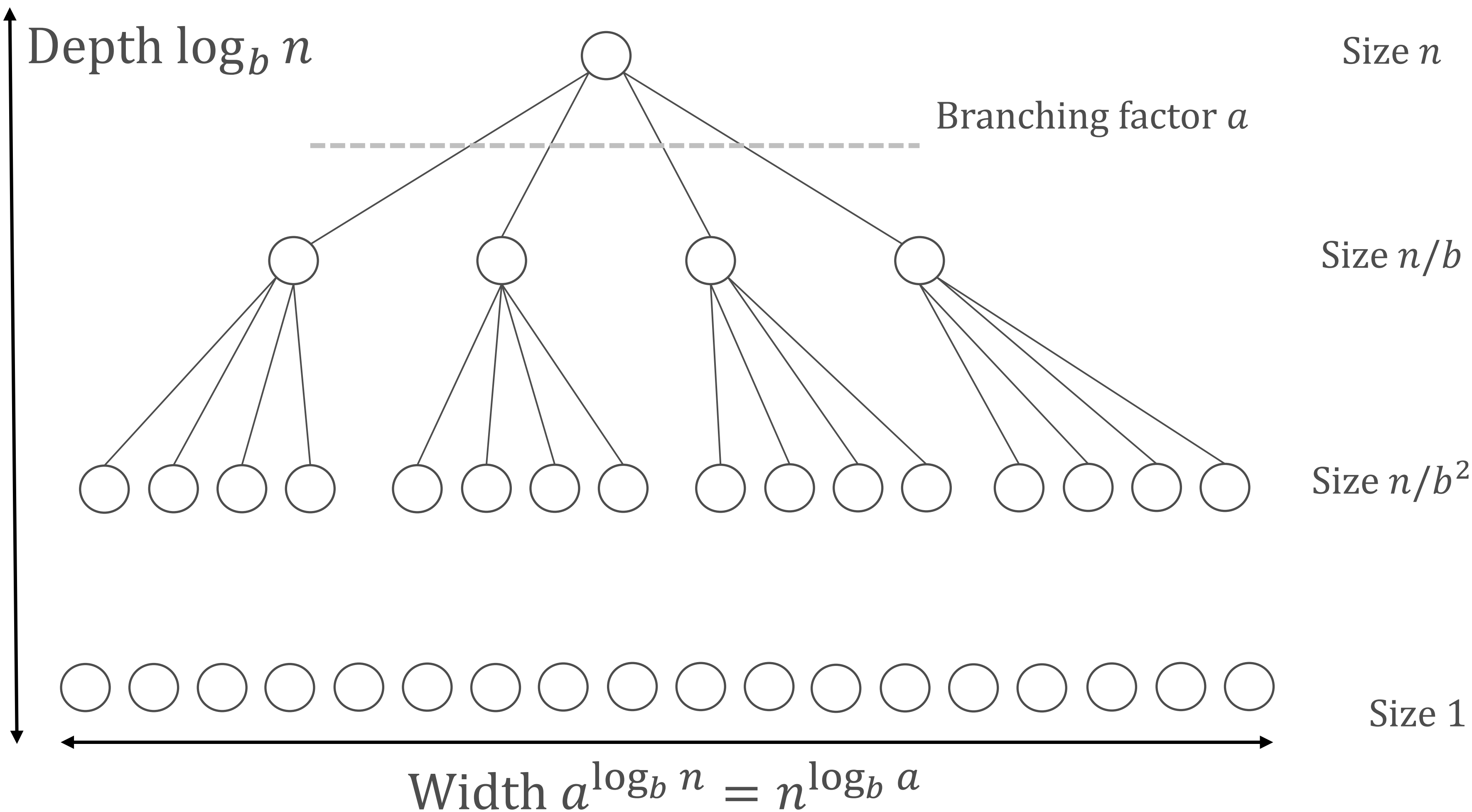
<u>Theorem</u>: If $T(n) = aT\left(\left\lceil \frac{n}{b} \right\rceil\right) + \mathrm{cn}^{\mathrm{d}}$, for some constants $a, c > 0$, $b > 1$, and $d \geq 0$, then

$$T(n) = \begin{cases} O\left(n^{d}\right) & \text{if } d > \log_b(a) \\ O\left(n^{d}\log(n)\right) & \text{if } d = \log_b(a) \\ O\left(n^{\log_b(a)}\right) & \text{if } d < \log_b(a) \end{cases}$$

Comparing n^d with $a^{\log_b n}$

MASTER THEOREM

- Let's assume that n is a power of b



MASTER THEOREM

- Total work:
  - cn^d in level zero
  - $a \cdot c\left(\frac{n}{b}\right)^d$ in level one
  - $a^k c\left(\frac{n}{b^k}\right)^d = cn^d\left(\frac{a}{b^d}\right)^k$ in level k
- Overall:

$$\sum_{k=0}^{\log_b n} c\, n^d \left(\frac{a}{b^d}\right)^k$$

MASTER THEOREM

- Let $r = a/b^d$
- Overall work:

$$cn^d(\,1 + r + r^2 + r^3 + \cdots + r^{\log_b n}\,)$$

1. $r < 1$
   - Series converges to $\frac{1}{(1-r)}$
   - Total work at most $\frac{cn^d}{1-r} \in O(n^d)$

MASTER THEOREM

- Let $r = a/b^d$
- Overall work:

$$cn^d(\, 1 + r + r^2 + r^3 + \cdots + r^{\log_b n}\,)$$

2. $r = 1$
   - Series adds up to $\log_b n$
   - Total work at most $cn^d \log_b n \in O(n^d logn)$

MASTER THEOREM

- Let $r = a/b^d$
- Overall work:

$$cn^d(\ 1 + r + r^2 + r^3 + \cdots + r^{\log_b n}\ )$$

3. $r > 1$
   - Series dominated by last term

$$(\ 1 + r + \cdots + r^{\log_b n}\ ) = r^{\log_b n}(\frac{1}{r^{\log_b n}} + \ \ldots + \frac{1}{r} + 1)$$

   - Total work at most

$$\frac{cn^d r^{\log_b n}}{1 - 1/r} \in O\left(n^d \left(\frac{a}{b^d}\right)^{\log_b n}\right) = O\left(\frac{n^d a^{\log_b n}}{n^d}\right)$$

Which is $O(n^{\log_b a})$

MASTER THEOREM

<u>**Theorem**</u>: If $T(n) = aT\left(\left\lceil \frac{n}{b} \right\rceil\right) + \mathrm{cn}^{\mathrm{d}}$, for some constants $a, c > 0$, $b > 1$, and $d \geq 0$, then

$$T(n) = \begin{cases} O\left(n^{d}\right) & \text{if } d > \log_b(a) \\ O(n^{d}\log(n)) & \text{if } d = \log_b(a) \\ O\left(n^{\log_b(a)}\right) & \text{if } d < \log_b(a) \end{cases}$$

MASTER THEOREM

<u>**Theorem**</u>: If $T(n) = aT\left(\left\lceil \frac{n}{b} \right\rceil\right) + \mathrm{O}(\mathrm{n}^{\mathrm{d}})$, for some constants $a > 0, b > 1$, and $d \geq 0$, then

$$T(n) = \begin{cases} O\left(n^{d}\right) & \text{if } d > \log_b(a) \\ O(n^{d}\log(n)) & \text{if } d = \log_b(a) \\ O(n^{\log_b(a)}) & \text{if } d < \log_b(a) \end{cases}$$

MERGESORT

- Analysis:
  - 2 subproblems: $a = 2$
  - The size of each subproblem is half the original: $b = 2$
  - Time to merge is linear: $d = 1$
  - $\log_b a = 1 = d$
  - Running time is $O\left(n^d \log n\right) = O(nlogn)$

COUNTING INVERSIONS

- Music site tries to match your song preferences with others
  - You rank n songs
  - Music sites consults database to find people with similar tastes
- Similarity metric: number of inversions between two rankings
  - My rank: $1,2,3, \ldots, n$
  - Your rank: $a_1, a_2, \ldots, a_n$
  - Songs i, j inverted if $i < j$ but $a_i > a_j$

COUNTING INVERSIONS

Songs

	A	B	C	D	E
Me	1	2	3	4	5
You	1	3	4	2	5

Inversions

3-2, 4-2

COUNTING INVERSIONS

- Also called the Kendall tau distance or bubble-sort distance
  - Many applications in voting theory

COUNTING INVERSIONS

1	5	4	8	10	2	6	9	12	11	3	7

COUNTING INVERSIONS

1	5	4	8	10	2	6	9	12	11	3	7

1	5	4	8	10	2

6	9	12	11	3	7

$2T\left({}^{n}/_{2}\right)$

COUNTING INVERSIONS

1	5	4	8	10	2	6	9	12	11	3	7

1	5	4	8	10	2

6	9	12	11	3	7

$2T\left({}^{n}/_{2}\right)$

5 Inversions:
5-4, 5-2, 4-2,8-2,10-2

8 Inversions:
6-3, 9-3, 9-7, 12-11, 12-3,12-7, 11-3, 11-7

COUNTING INVERSIONS

1	5	4	8	10	2	6	9	12	11	3	7

1	5	4	8	10	2

6	9	12	11	3	7

$2T(^n/_2)$

5 Inversions:
5-4, 5-2, 4-2,8-2,10-2

8 Inversions:
6-3, 9-3, 9-7, 12-11, 12-3,12-7, 11-3, 11-7

- Merge??
- There is a blue green inversion (a_i, a_j), $a_i > a_j$, every time a_i is blue and a_j is green
- It'll be easier to sort on the way

COUNTING INVERSIONS

1	5	4	8	10	2	6	9	12	11	3	7

1	2	4	5	8	10

3	6	7	9	11	12

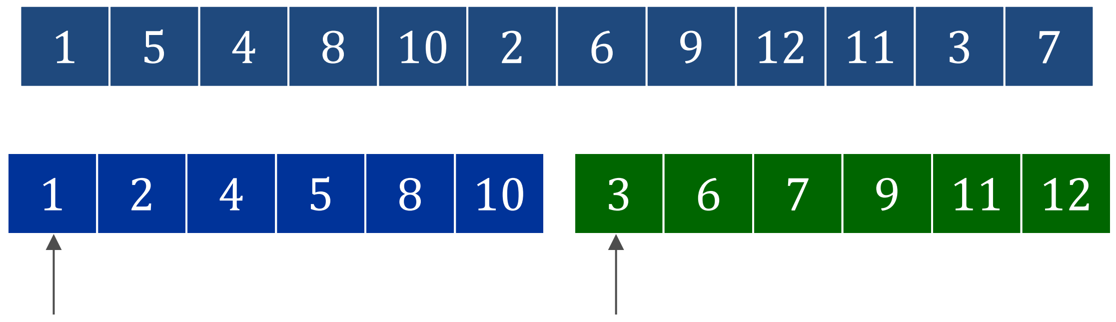
$2T\left({}^{n}/_{2}\right)$

5 inversions so far

8 inversions so far

- Assume each half comes back sorted

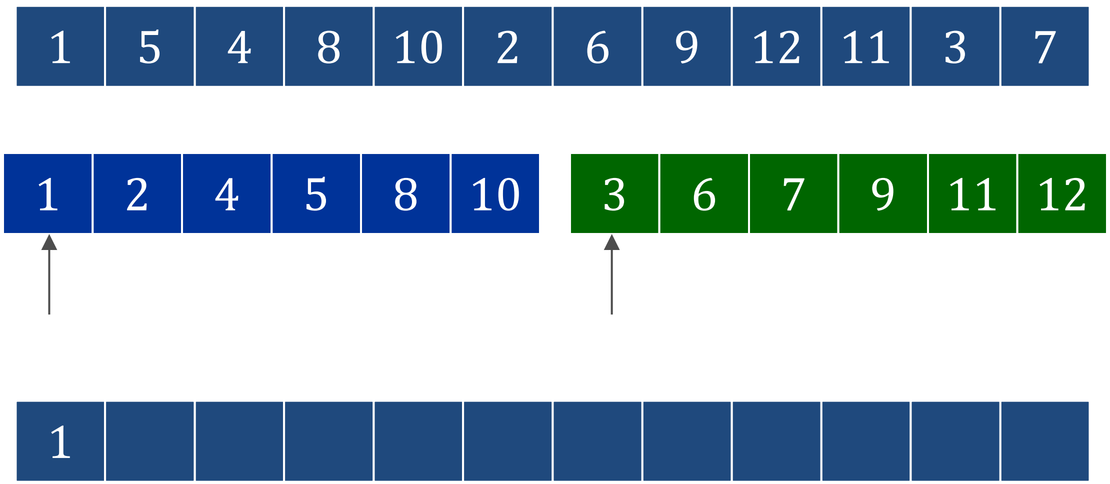
COUNTING INVERSIONS



$2T\left({}^{n}/_{2}\right)$

- Inversions so far: 5+8+0

COUNTING INVERSIONS



$2T\left(\frac{n}{2}\right)$

- Inversions so far: 5+8+0

COUNTING INVERSIONS

1	5	4	8	10	2	6	9	12	11	3	7

1	2	4	5	8	10

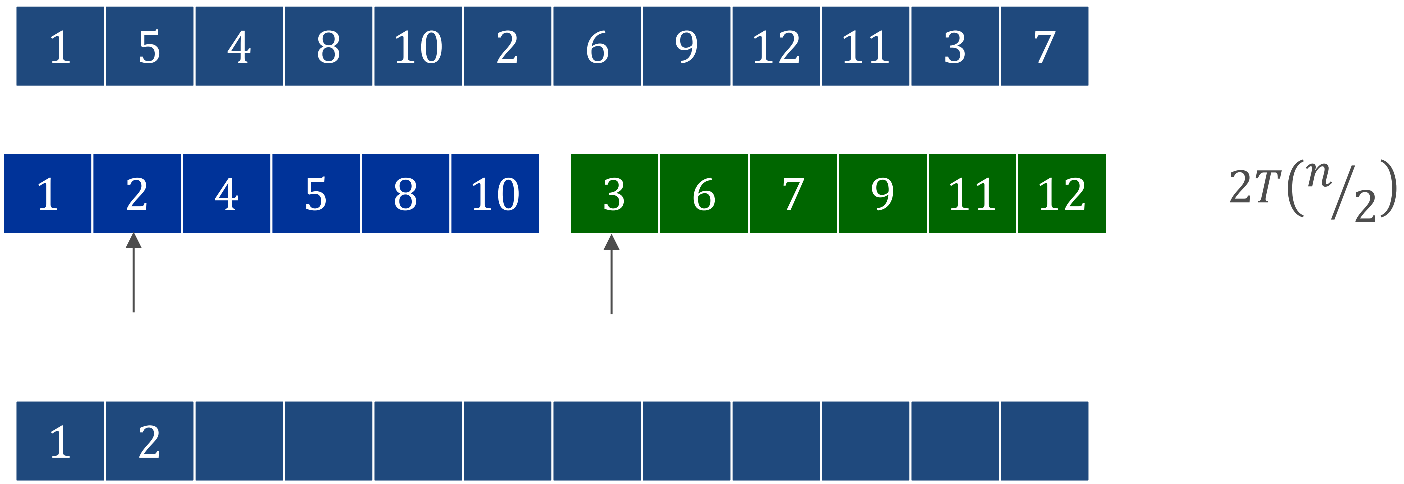
3	6	7	9	11	12

$2T\left({}^{n}/_{2}\right)$

1											

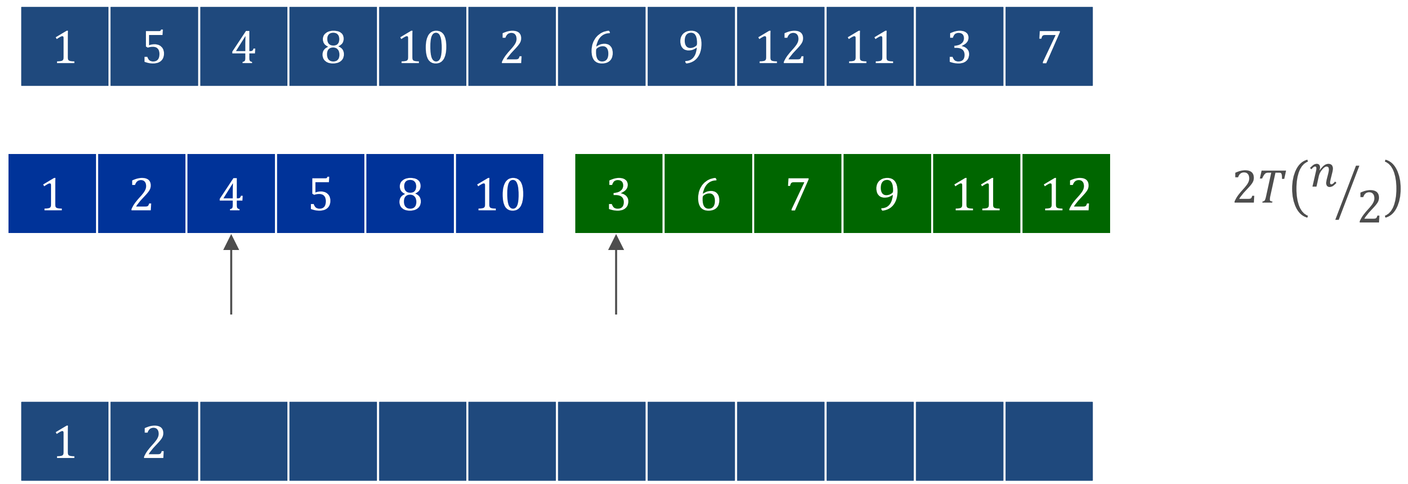
- Inversions so far: 5+8+0

COUNTING INVERSIONS



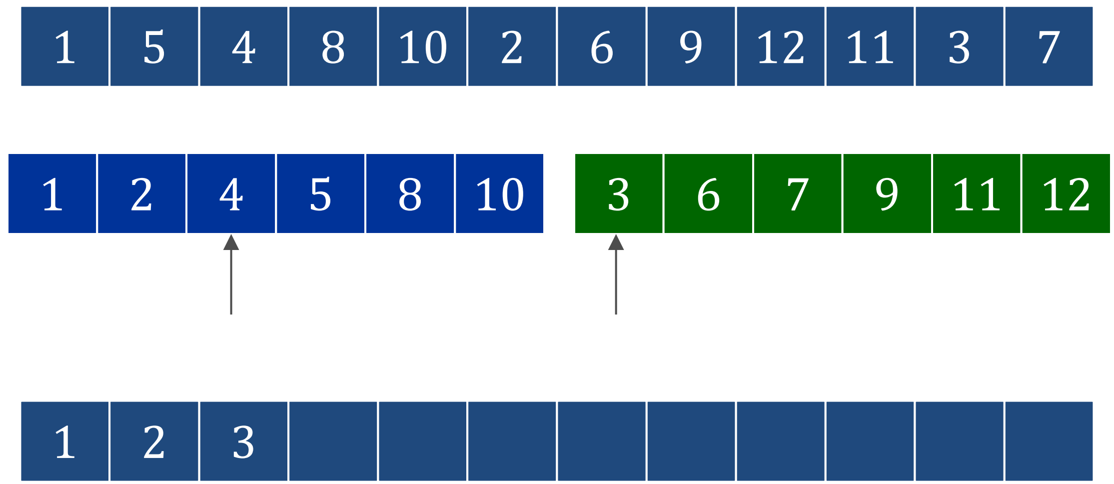
- Inversions so far: 5+8+0

COUNTING INVERSIONS



- Inversions so far: 5+8+0

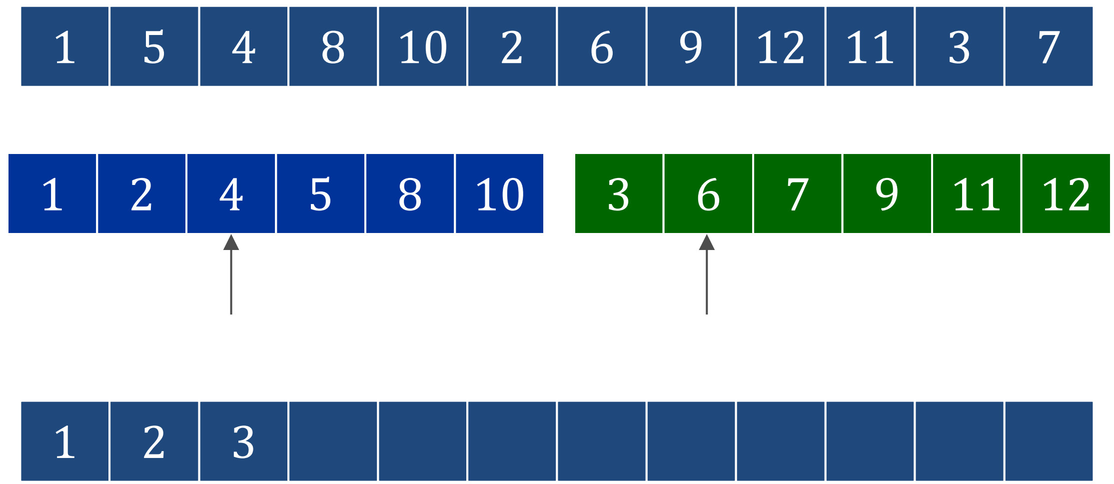
COUNTING INVERSIONS



$2T\left(^{n}/_{2}\right)$

- Inversions so far: 5+8+4

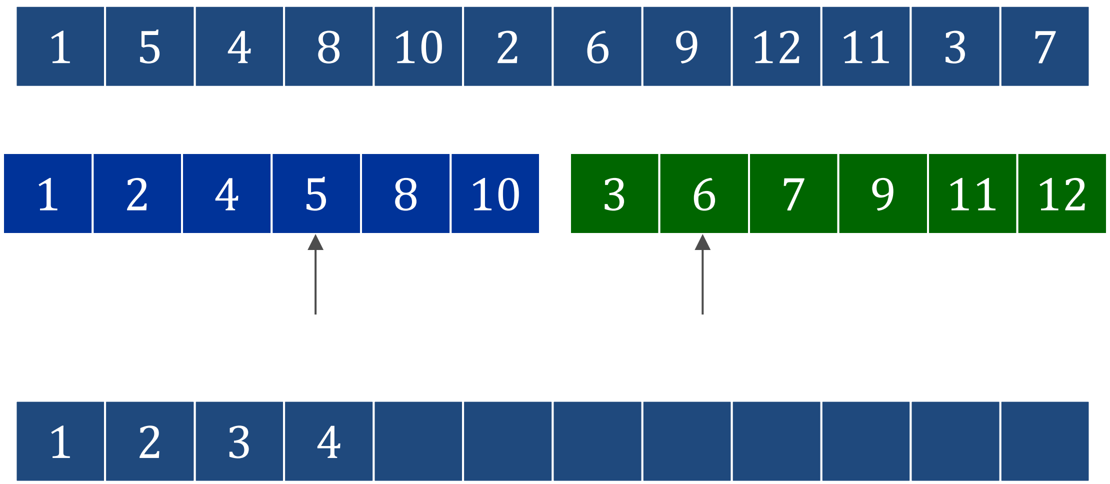
COUNTING INVERSIONS



$2T\left({}^{n}/_{2}\right)$

- Inversions so far: 5+8+4

COUNTING INVERSIONS



$2T(n/2)$

- Inversions so far: 5+8+4

COUNTING INVERSIONS

1	5	4	8	10	2	6	9	12	11	3	7

1	2	4	5	8	10

3	6	7	9	11	12
4	2	2	1		

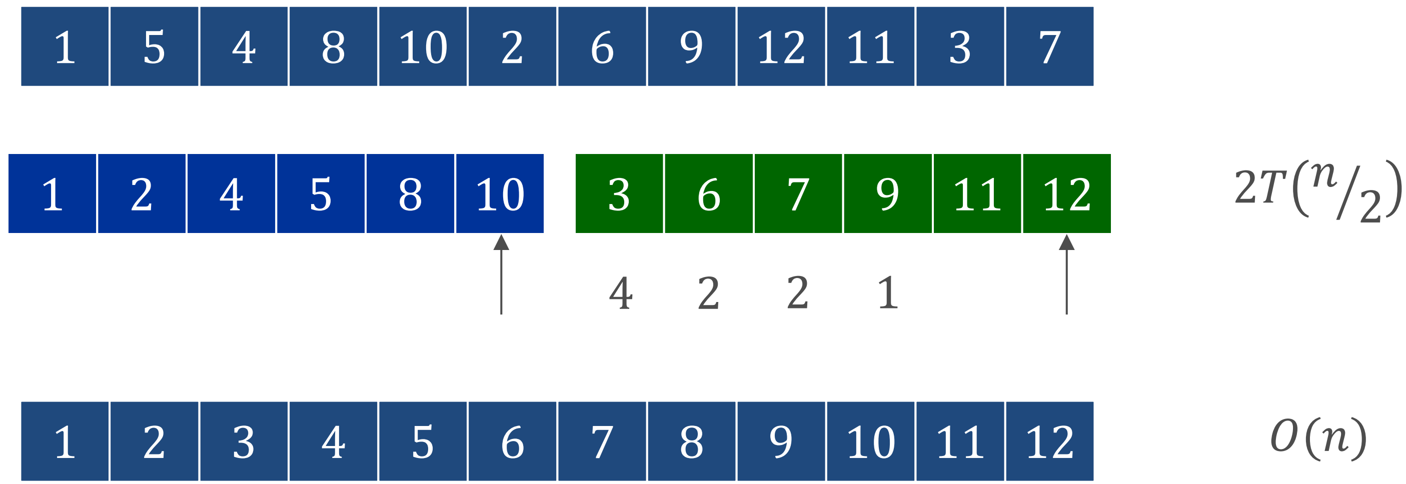
$2T\left({}^{n}/_{2}\right)$

1	2	3	4	5	6	7	8	9	10	11	12

$O(n)$

- Inversions so far: 5+8+9

COUNTING INVERSIONS



- Inversions so far: 5+8+9
- 2 subproblems of size $n/2$, plus linear time:
  - Master Theorem: $O(nlog(n))$

SUMMARY

- Mergesort (5.1 in KT)
- Master Theorem (5.2 KT)
- Counting Inversions (5.3 in KT)