

Lecture 23

CSE 331

Mar 30, 2020

Mergesort algorithm

Input: a_1, a_2, \dots, a_n

Output: Numbers in sorted order

MergeSort(a, n)

If $n = 1$ **return** the order a_1

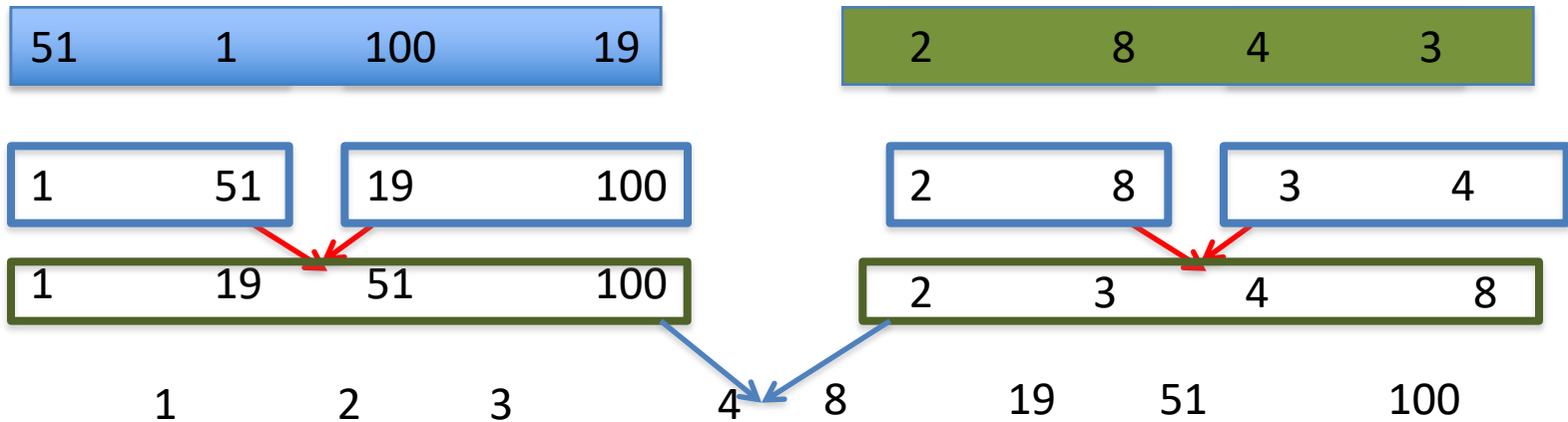
If $n = 2$ **return** the order $\min(a_1, a_2); \max(a_1, a_2)$

$a_L = a_1, \dots, a_{n/2}$

$a_R = a_{n/2+1}, \dots, a_n$

return MERGE (**MergeSort**($a_L, n/2$), **MergeSort**($a_R, n/2$))

An example run



MergeSort(a, n)

If $n = 1$ return the order a_1

If $n = 2$ return the order $\min(a_1, a_2); \max(a_1, a_2)$

$a_L = a_1, \dots, a_{n/2}$

$a_R = a_{n/2+1}, \dots, a_n$

return MERGE (MergeSort($a_L, n/2$), MergeSort($a_R, n/2$))

Correctness

Input: a_1, a_2, \dots, a_n

Output: Numbers in sorted order

MergeSort(a, n)

If $n = 1$ return the order a_1

If $n = 2$ return the order $\min(a_1, a_2); \max(a_1, a_2)$

$a_L = a_1, \dots, a_{n/2}$

$a_R = a_{n/2+1}, \dots, a_n$

return MERGE (MergeSort($a_L, n/2$) MergeSort($a_R, n/2$))

By
induction
on n

Inductive step follows from correctness of MERGE

Rest of today's agenda

Analyze runtime of mergesort algorithm