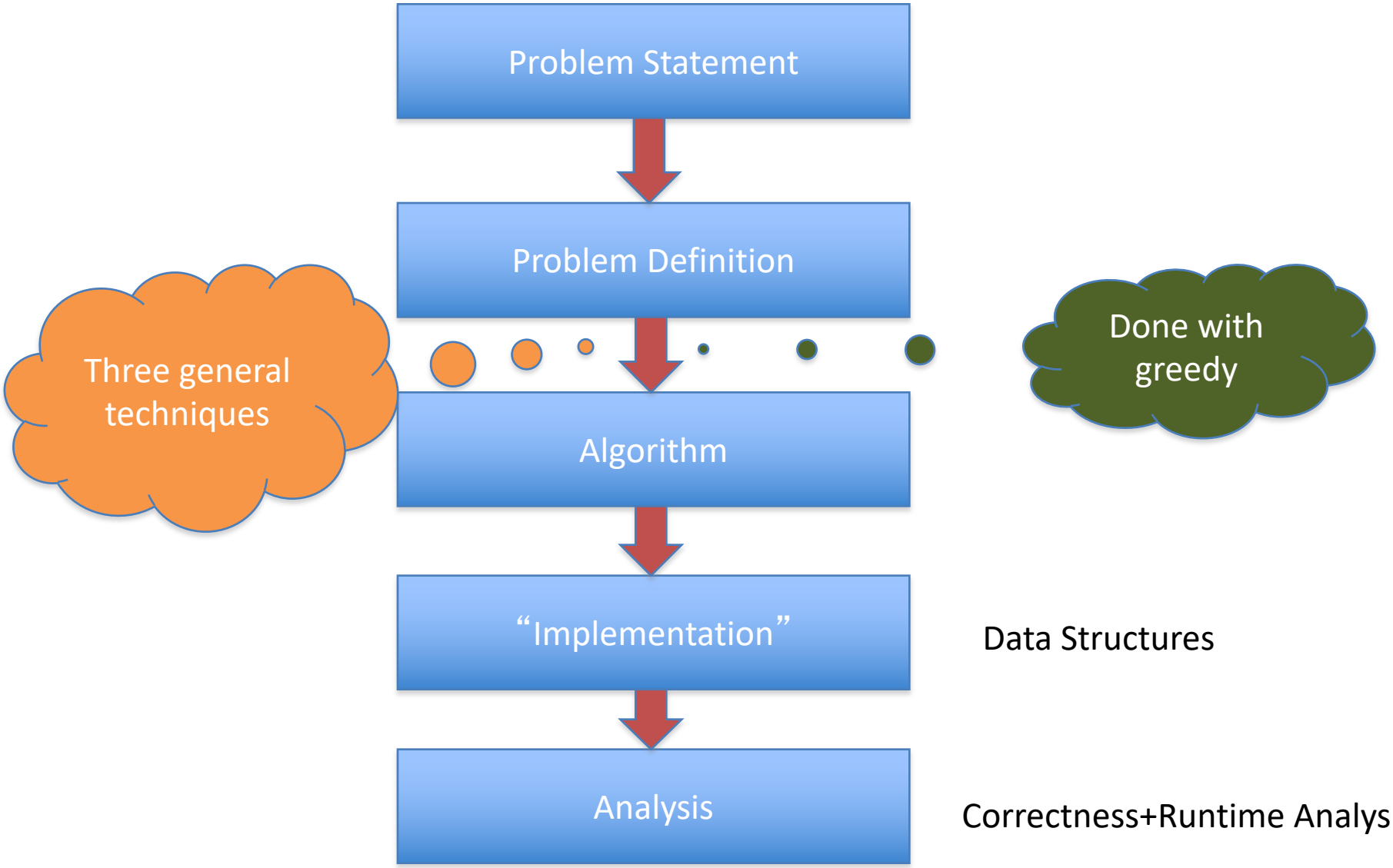
Lecture 22

CSE 331

Mar 27, 2020

High Level view of the course



Divide and Conquer

Divide up the problem into at least two sub-problems

Recursively solve the sub-problems

“Patch up” the solutions to the sub-problems for the final solution

Sorting

Given n numbers order them from smallest to largest

Works for any set of elements on which there is a total order

Insertion Sort

Input: $a_1, a_2, \ldots, a_n$

Output: $b_1, b_2, \ldots, b_n$

$O(n^2)$ overall

Make sure that all the processed numbers are sorted

$b_1 = a_1$

for $i = 2 \ldots n$

Find $1 \le j \le i$ s.t. a_i lies between b_{j-1} and b_j

Move b_j to b_{i-1} one cell "down"

$b_j = a_i$

O(log n)

O(n)

a	b
4	4
3	4
2	4
1	4

Other $O(n^2)$ sorting algorithms

Selection Sort: In every round pick the min among remaining numbers

Bubble sort: The smallest number “bubbles” up

Divide and Conquer

Divide up the problem into at least two sub-problems

Recursively solve the sub-problems

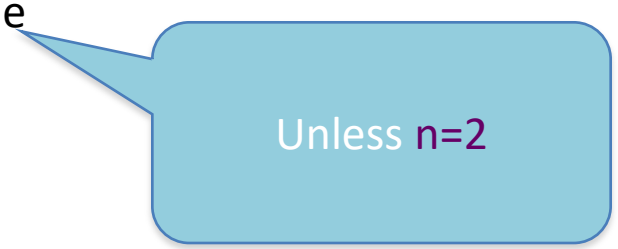
“Patch up” the solutions to the sub-problems for the final solution

Mergesort Algorithm

Divide up the numbers in the middle

Sort each half recursively

Merge the two sorted halves into one sorted output



How fast can sorted arrays be merged?

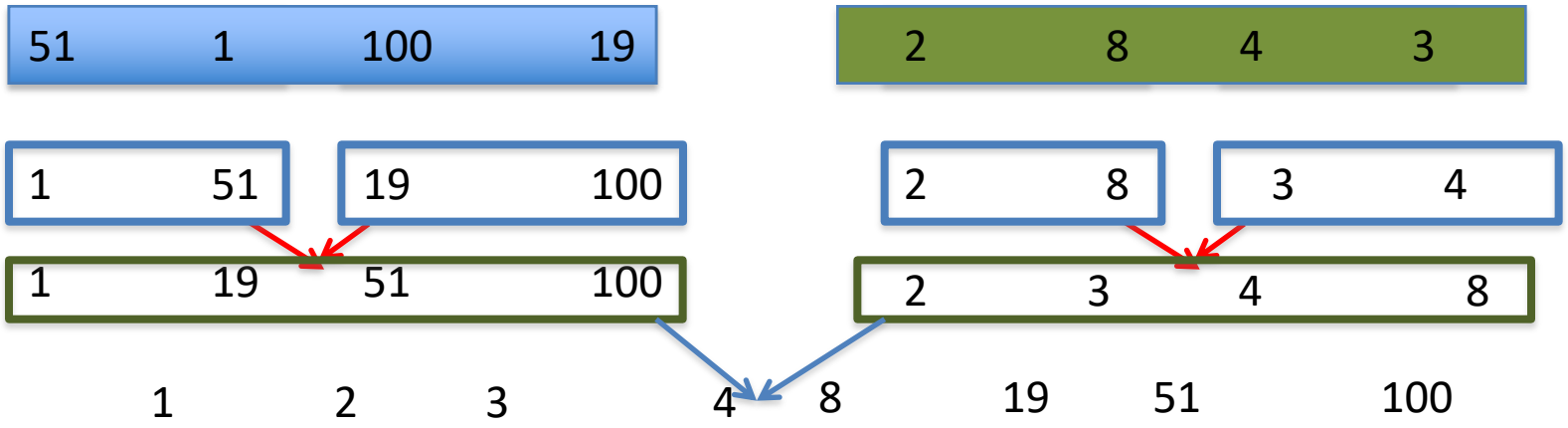
Mergesort algorithm

Input: $a_1, a_2, \ldots, a_n$

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,..., a_n/2

    a_R = a_n/2+1,..., 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)

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

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

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

Correctness

Input: $a_1, a_2, \ldots, a_n$

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,..., a_n/2

  a_R = a_n/2+1,..., 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