Lecture 23

CSE 331

Oct 25, 2024

Details on 1-on-1 meetings

note @217 stop following 3 views Actions

Meetings to discuss CSE 331 performance

By Sunday night, I will email those who have a D or below in their mid-term grade (for more details on the grade see @216) to setup a one-on-one meeting to talk with me but I figured I should post the information about meeting times now rather than later.

Of course you can also come and talk about your 331 performance even if you have a temp grade higher than D (though students with a D or below will get preference).

I have locked out certain times over next week or so for **15 mins** meetings. Please note that **these are NOT walk-ins**: if no one signs up for a slot, I will NOT be on zoom then. If you want to come and talk with me, **please EMAIL me with ALL the slots below that work for you**. (Private posts on piazza will not work: please email me!) *Slots will be assigned on a first-come-first-serve basis. Also I might only be able to confirm your time after 11pm on the day* before *your scheduled slot.*

Note: These are my current availabilities-- some of the slots might be used up in some other non-CSE 331 meetings. So please send multiple choices for when you can meet.

To make things easier, **ALL meeting will be on zoom** (https://buffalo.zoom.us/j/95499374560?pwd=SrI2p86L6bI3PMl2uRtUjI1mpIP6qM.1)

Below are all the available slots (below the start times are listed: a slot that is already taken has a strike-through and italicized):

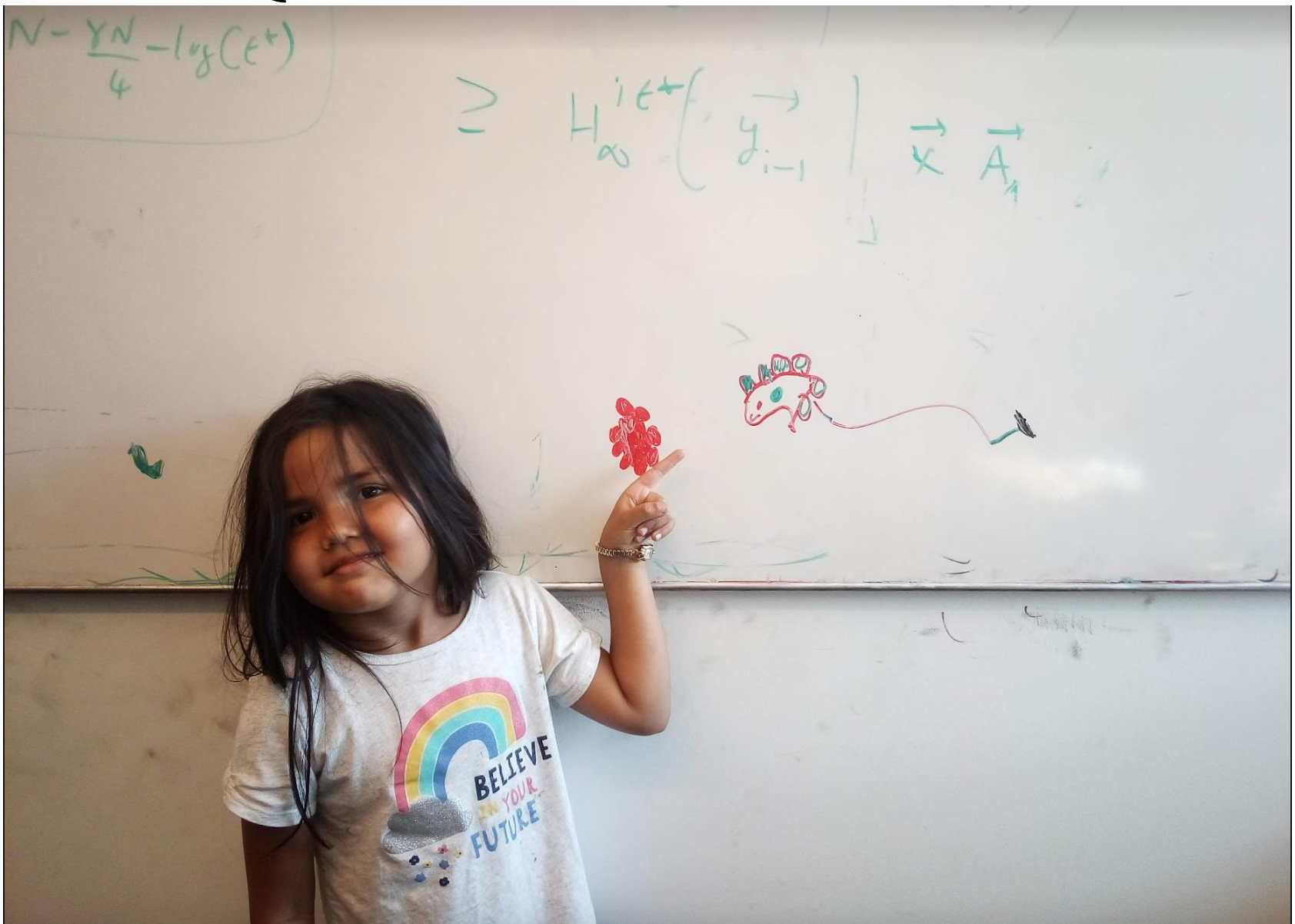
- **Tuesday (Oct 22):** 10:15am, 10:30am, 10:45am, 11:00am, 11:15am, 11:30am, 11:45am, 4:30pm, 4:45pm
- **Wednesday (Oct 23):** 12:45pm, 1:00pm, 1:15pm, 1:30pm, 1:45pm, 2:30pm, 2:45pm, 3:00pm, 3:15pm, 3:30pm, 3:45pm, 4:00pm, 4:15pm, 5:00pm, 5:15pm, 5:30pm, 5:45pm, 6:00pm, 6:15pm, 7:45pm, 8:00pm, 8:15pm, 8:30pm
- **Thursday (Oct 24):** 9:00am, 9:15am, 9:30am, 9:45am, 10:00am, 10:15am, 10:30am, 10:45am, 11:00am, 11:15am, 11:30am, 11:45am, 12:30pm, 12:45pm
- **Friday (Oct 25):** 9:00am, 9:15am, 9:30am, 9:45am, 12:30pm, 2:00pm, 2:15pm, 2:30pm, 2:45pm
- **Monday (Oct 28):** 12:30pm, 2:30pm, 2:45pm, 3:00pm, 3:15pm, 3:30pm, 3:45pm, 4:30pm, 4:45pm, 5:00pm, 5:15pm, 5:30pm, 5:45pm, 6:30pm, 6:45pm, 7:00pm, 7:15pm, 7:30pm, 7:45pm, 8:00pm, 8:15pm, 8:30pm

(If none of the times above work for you but you still want to meet, please email me and we can try and set up a time for the week of Oct 28.)

1st project deadline next Friday

Tue, Oct 15		**(HW 4 out)**
Wed, Oct 16	Dijkstra's algorithm F24 F23 F22 F21 F19 x²	[KT, Sec 4.4] Week 8 recitation notes
Fri, Oct 18	Correctness of Dijkstra's Algorithm F23 F22 F21 F19 F18 F17 x²	[KT, Sec 4.4] *Reading Assignment:* [KT, Sec 4.4]
Mon, Oct 21	Minimum Spanning Tree F23 F22 F21 F19 F18 F17 x²	[KT, Sec 4.5]
Tue, Oct 22		**(HW 4 in, HW 5 out)**
Wed, Oct 23	Cut Property Lemma F23 F22 F21 F19 F18 F17 x²	[KT, Sec 4.5] *Reading Assignment:* [KT, Sec 4.5, 4.6]
Fri, Oct 25	Mergesort F23 F22 F21 F19 F18 F17 x²	[KT, Sec 5.1]
Mon, Oct 28	Solving recurrence relations F23 F22 F21 F19 F18 F17 x²	[KT, Sec 5.1]
Tue, Oct 29		**(HW 5 in)**
Wed, Oct 30	Counting Inversions F23 F22 F21 F19 F18 F17 x²	[KT, Sec 5.3]
Fri, Nov 1	Multiplying large integers F23 F22 F21 F19 F18 F17 x²	[KT, Sec 5.5] **(Project (Problems 1 & 2 `Coding`) in)** *Reading Assignment:* Unraveling the mystery behind the identity
Mon, Nov 4	Closest Pair of Points F23 F22 F21 F19 F18 F17 x²	[KT, Sec 5.4] **(Project (Problems 1 & 2 `Reflection`) in)**

Questions/Comments?



Kruskal's Algorithm

Joseph B. Kruskal

Input: $G=(V,E)$, $c_e > 0$ for every e in E

$T = \emptyset$

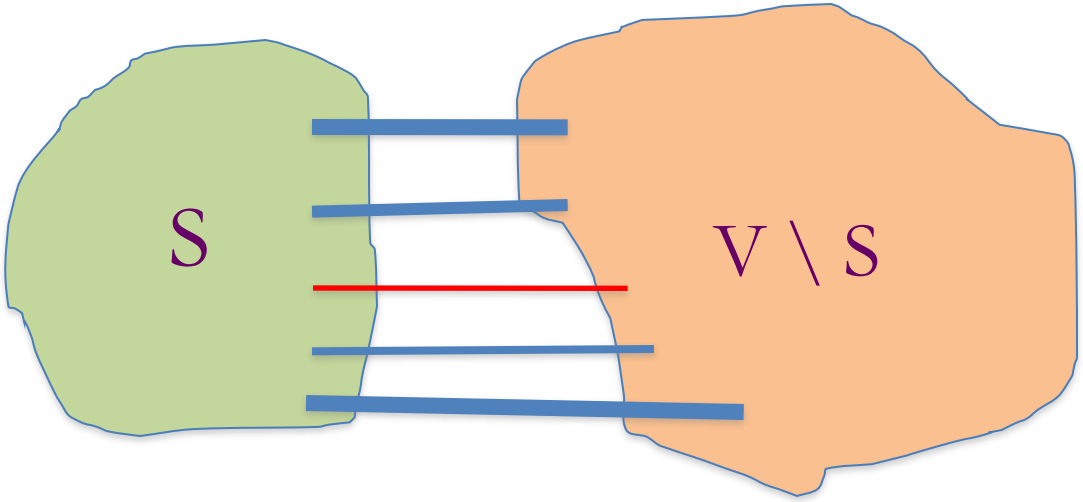
Sort edges in increasing order of their cost

Consider edges in sorted order

If an edge can be added to T without adding a cycle then add it to T

Cut Property Lemma for MSTs

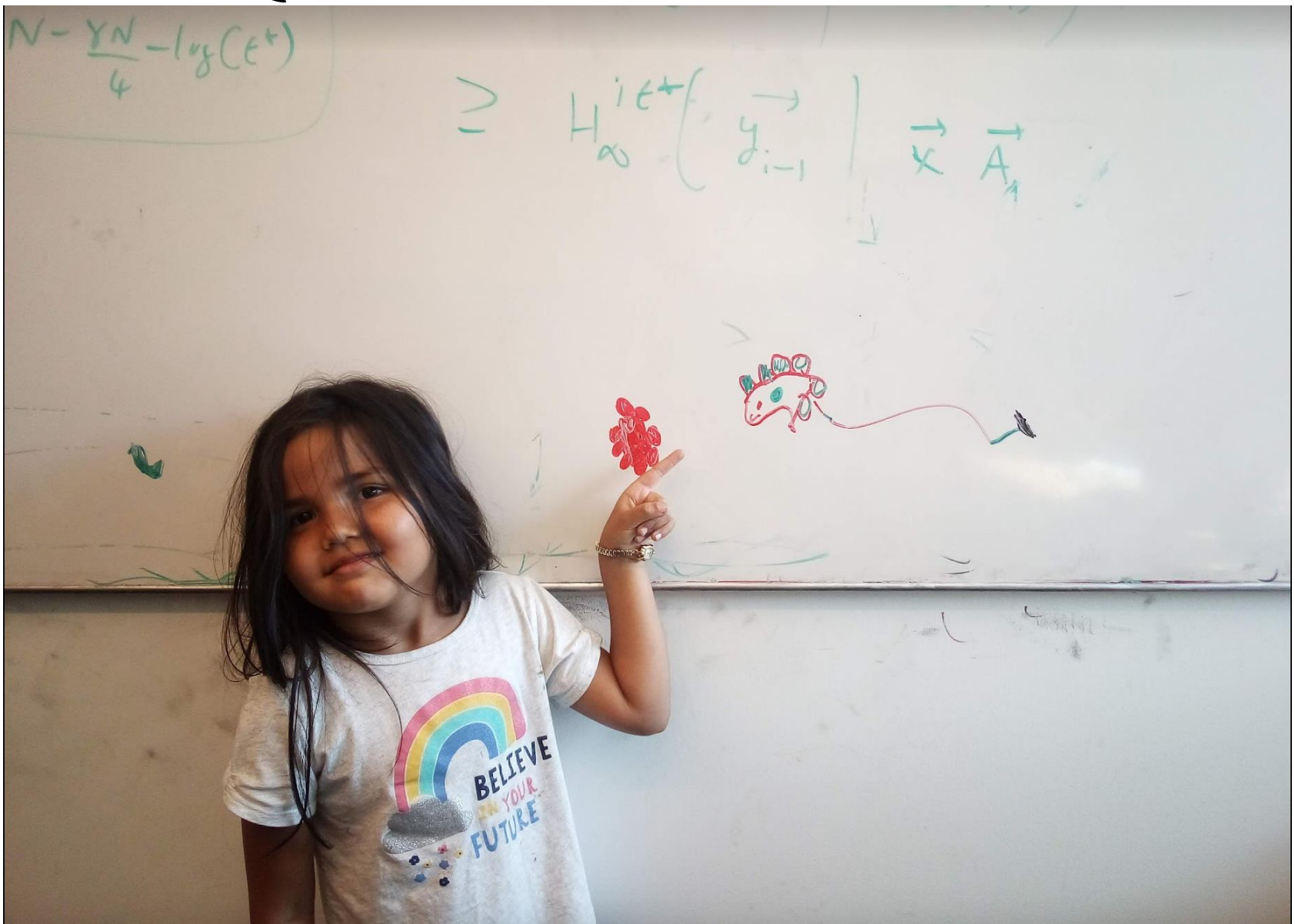
Condition: S and V\S are non-empty



Cheapest crossing edge is in **all** MSTs

Assumption: All edge costs are distinct

Questions/Comments?



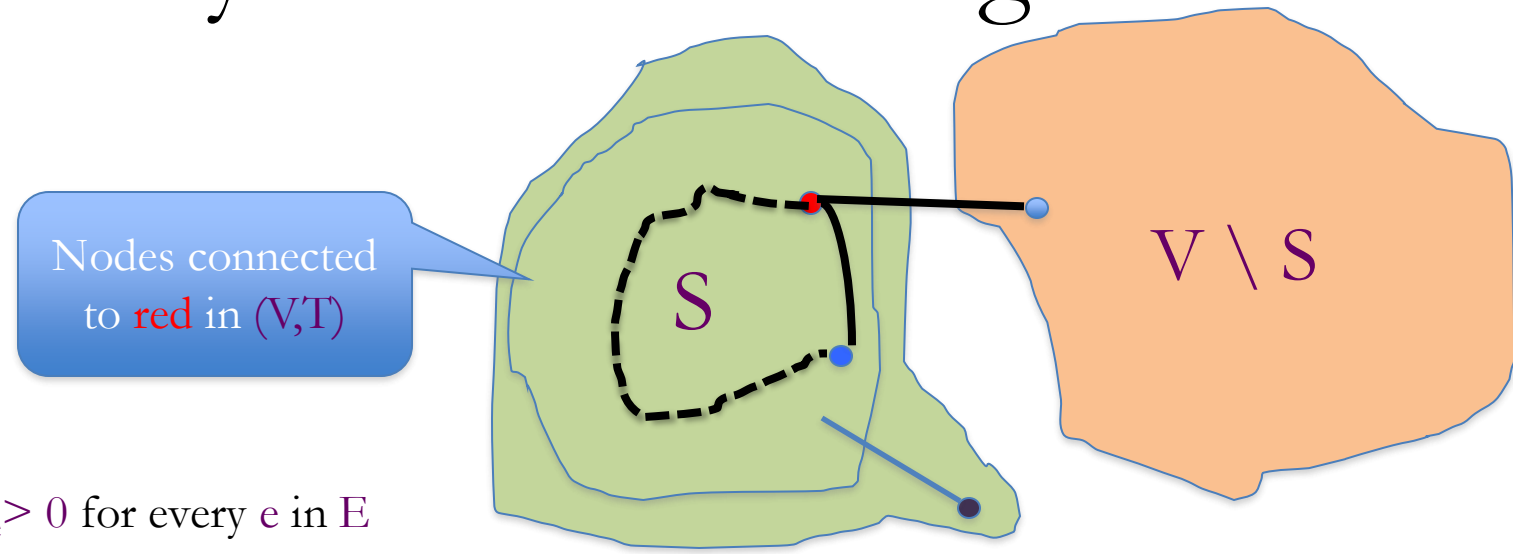
Today's agenda

Optimality of Kruskal's algorithm

Remove distinct edge weights assumption

Quick runtime analysis of Prim's+Kruskal's

Optimality of Kruskal's Algorithm



Input: G=(V,E), $c_e > 0$ for every e in E

T = Ø

Sort edges in increasing order of their cost

Consider edges in sorted order

If an edge can be added to T without adding a cycle then add it to T

S is non-empty

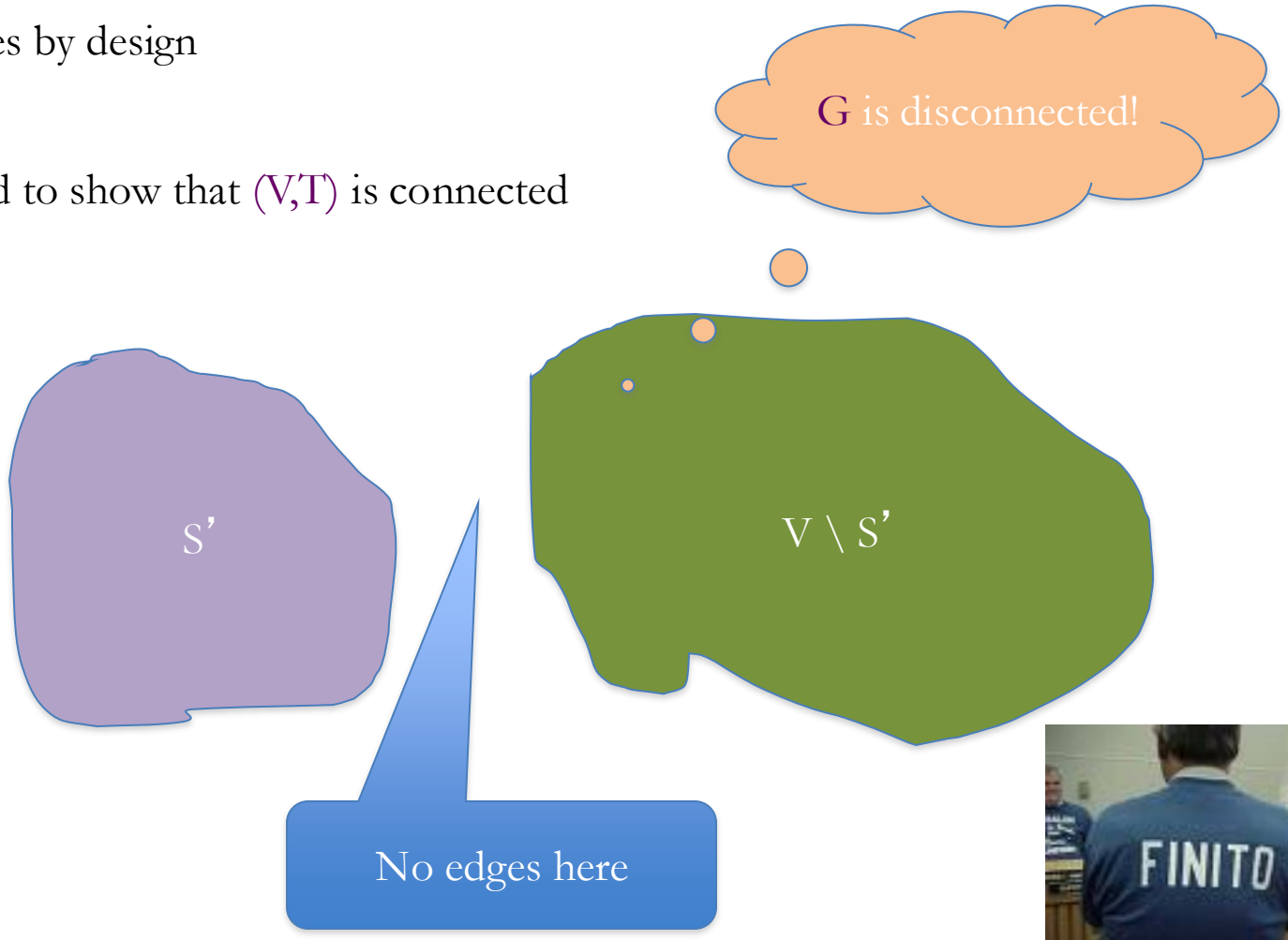
V\S is non-empty

First crossing edge considered

Is (V,T) a spanning tree?

No cycles by design

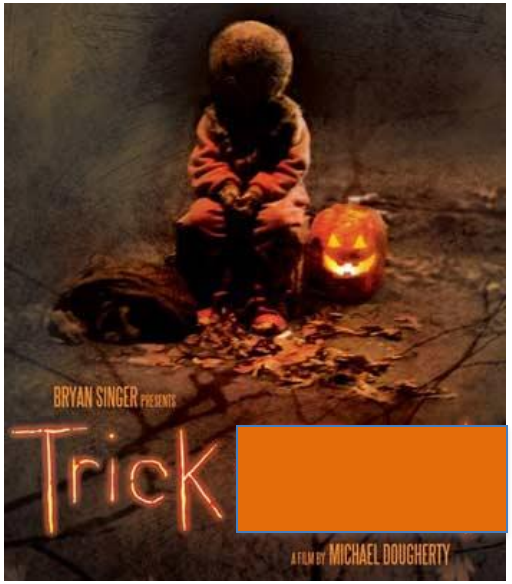
Just need to show that (V,T) is connected



Removing distinct cost assumption

Change all edge weights by very small amounts

Make sure that all edge weights are distinct

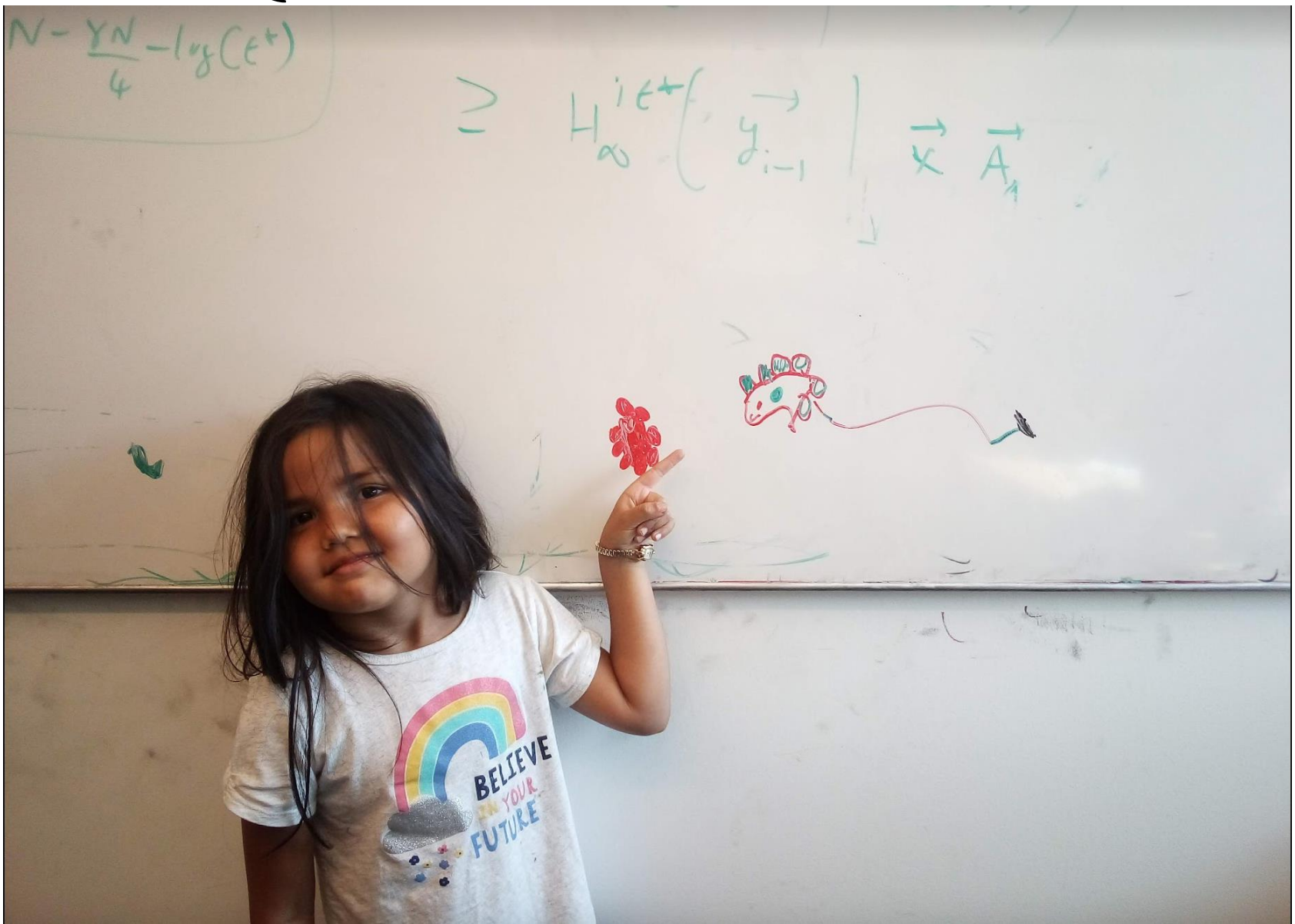


MST for "perturbed" weights is the same as for original

Changes have to be small enough so that this holds

EXERCISE: Figure out how to change costs

Questions/Comments?



Running time for Prim's algorithm

Similar to Dijkstra's algorithm

O(m log n)

Input: $G=(V,E)$, $c_e > 0$ for every e in E

$S = \{s\}$, $T = \emptyset$

While S is not the same as V

Among edges $e= (u,w)$ with u in S and w not in S, pick one with minimum cost

Add w to S, e to T

Running time for Kruskal's Algorithm

Can be implemented in $O(m \log n)$ time (Union-find DS)

Joseph B. Kruskal

Input: $G=(V,E)$, $c_e > 0$ for every e in E

$O(m^2)$ time overall

$T = \emptyset$

Sort edges in increasing order of their cost

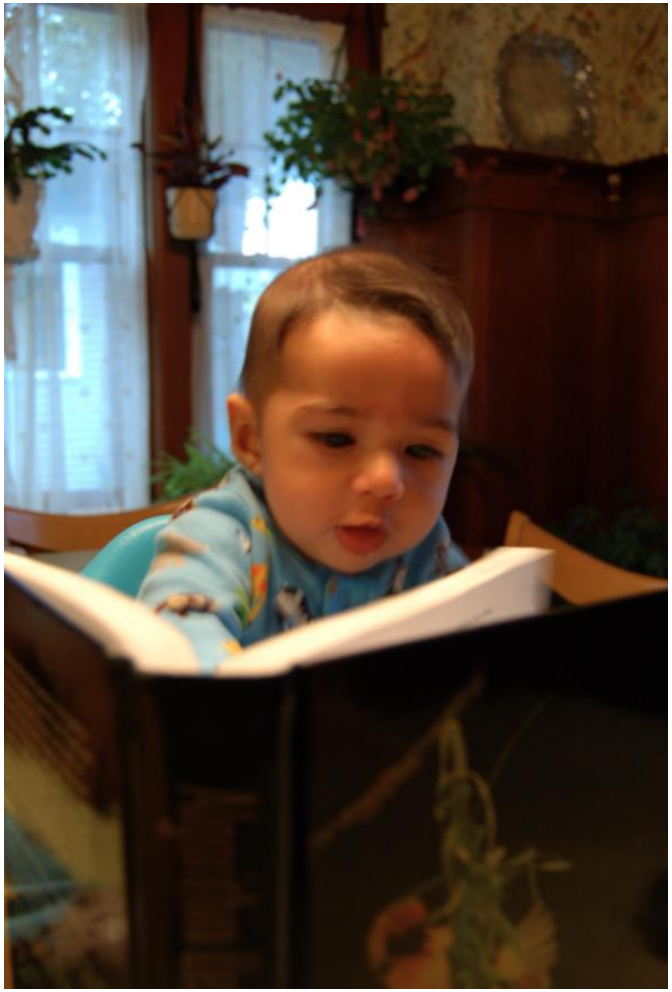
Consider edges in sorted order

If an edge can be added to T without adding a cycle then add it to T

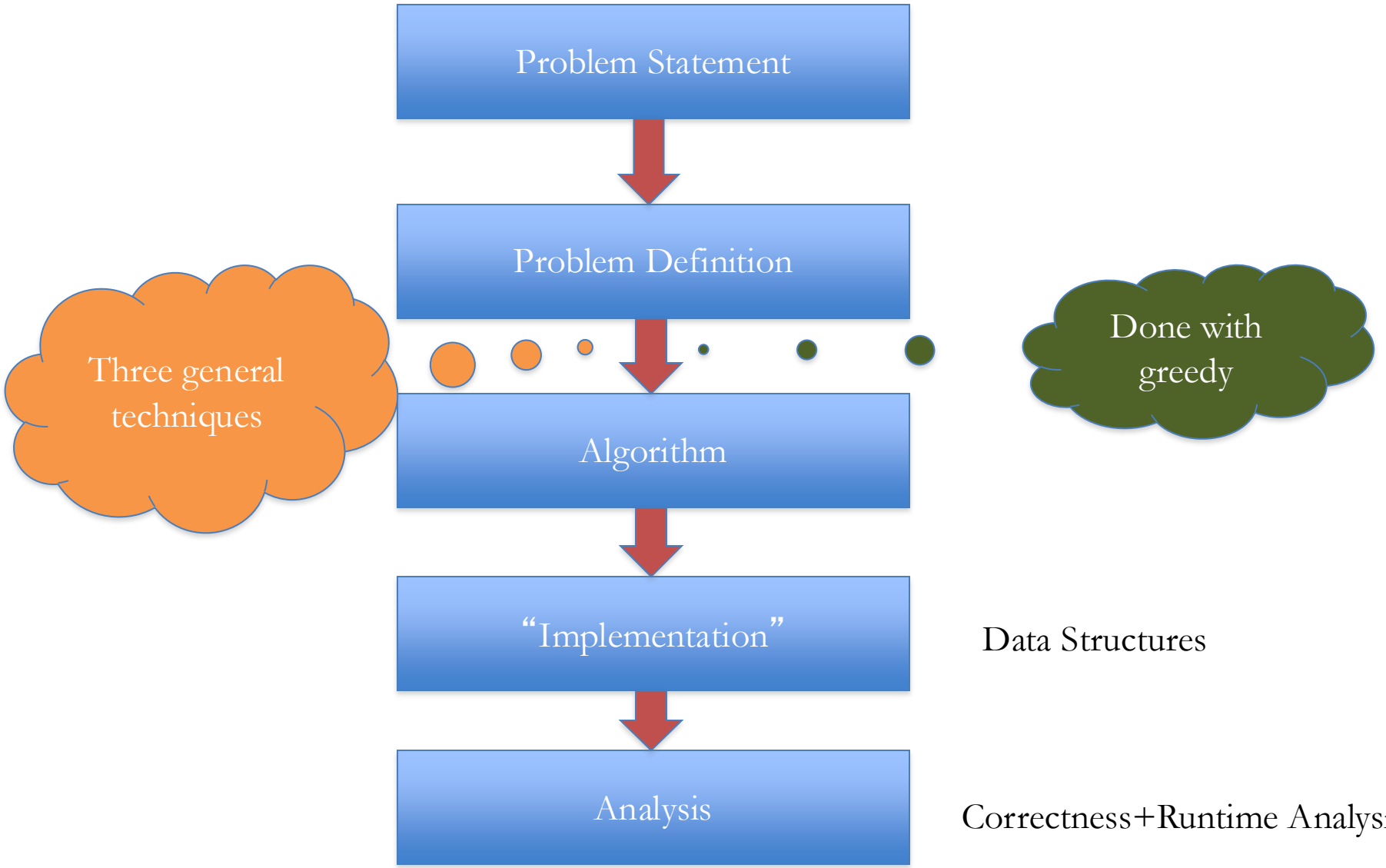
Can be verified in $O(m+n)$ time

Reading Assignment

Sec 4.5, 4.6 of [KT]



High Level view of the course



Trivia

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	3 4
3	3 4
2	3 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

Unless n=2

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

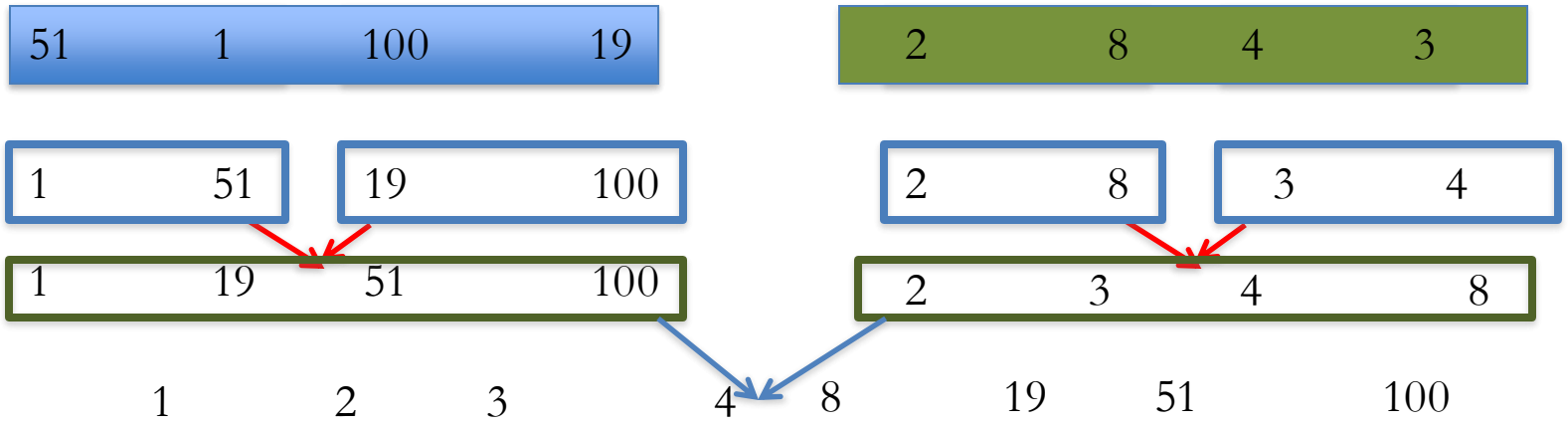
If $n = 2$ **return** the order $\min(a_1,a_2)$; $\max(a_1,a_2)$

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

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

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

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 $\min(a_1,a_2); \max(a_1,a_2)$

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

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

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

By induction on n

Inductive step follows from correctness of MERGE

Runtime analysis on the board…