# Lecture 22

CSE 331

Oct 25, 2021

# Please have a face mask on

**Masking requirement**


*UR requires all students, employees and visitors – regardless of their vaccination status – to wear face coverings while inside campus buildings.*

https://www.buffalo.edu/coronavirus/health-and-safety/health-safety-guidelines.html

# Project deadlines coming up

| | | |
|---|---|---|
| Fri, Oct 29 | Counting Inversions F19 F18 F17 $x^2$ | [KT, Sec 5.3] (**Project (Problem 1** Coding **) in**) |
| Mon, Nov 1 | Multiplying large integers F19 F18 F17 $x^2$ | [KT, Sec 5.5] (**Project (Problem 1** Reflection **) in**)<br>*Reading Assignment:* Unraveling the mystery behind the identity |
| Wed, Nov 3 | Closest Pair of Points F19 F18 F17 $x^2$ | [KT, Sec 5.4] |
| Fri, Nov 5 | Kickass Property Lemma F19 F18 F17 $x^2$ | [KT, Sec 5.4] (**Project (Problem 2** Coding **) in**) |
| Mon, Nov 8 | Weighted Interval Scheduling F19 F17 $x^2$ | [KT, Sec 6.1] (**Project (Problem 2** Reflection **) in**) |

# Group formation instructions

# Autolab group submission for CSE 331 Project

The lowdown on submitting your project (especially the coding and reflection) problems as a group on Autolab.

Follow instructions **EXACTLY** as they are stated

The instruction below are for Coding Problem 1

**You will have to repeat the instructions below for EACH coding AND reflection problem on project on Autolab** (with the appropriate changes to the actual problem).

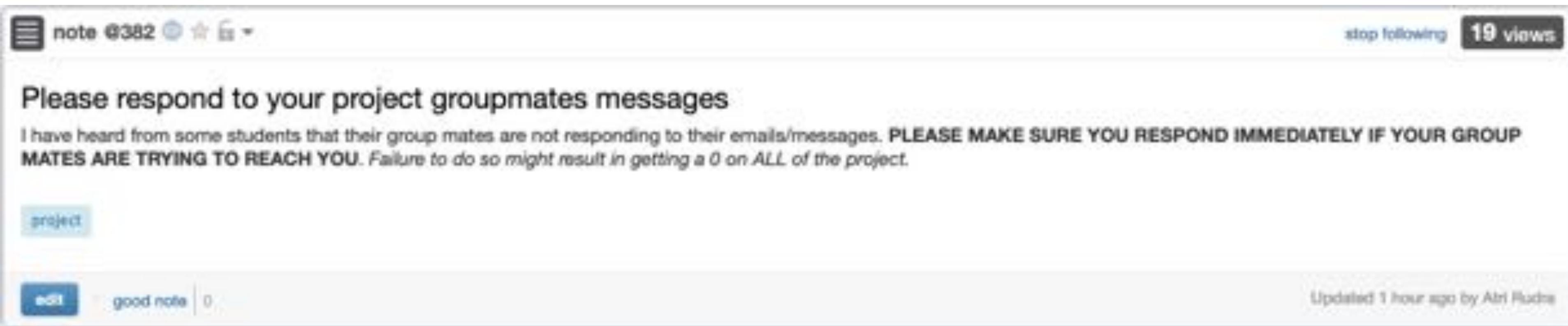
## Form your group on Autolab

Groups on Autolab will NOT be automatically created

You will have to form a group on Autolab by yourself (as a group). Read on for instructions on how to go about this.

Click to add notes

# Please be in touch w/ your group


note @382

stop following

19 views

**Please respond to your project groupmates messages**

I have heard from some students that their group mates are not responding to their emails/messages. **PLEASE MAKE SURE YOU RESPOND IMMEDIATELY IF YOUR GROUP MATES ARE TRYING TO REACH YOU.** *Failure to do so might result in getting a 0 on ALL of the project.*

project

edit

good note | 0

Updated 1 hour ago by Atri Rudra

# 1 on 1 meetings

note @367 · 88 views · Actions

## Meetings to discuss CSE 331 performance

I have emailed those who have a D+ or below in their mid-term grade (for more details on the grade see @359) to setup a one-on-one meeting to talk with me. 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 most likely will NOT be in my office/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.*

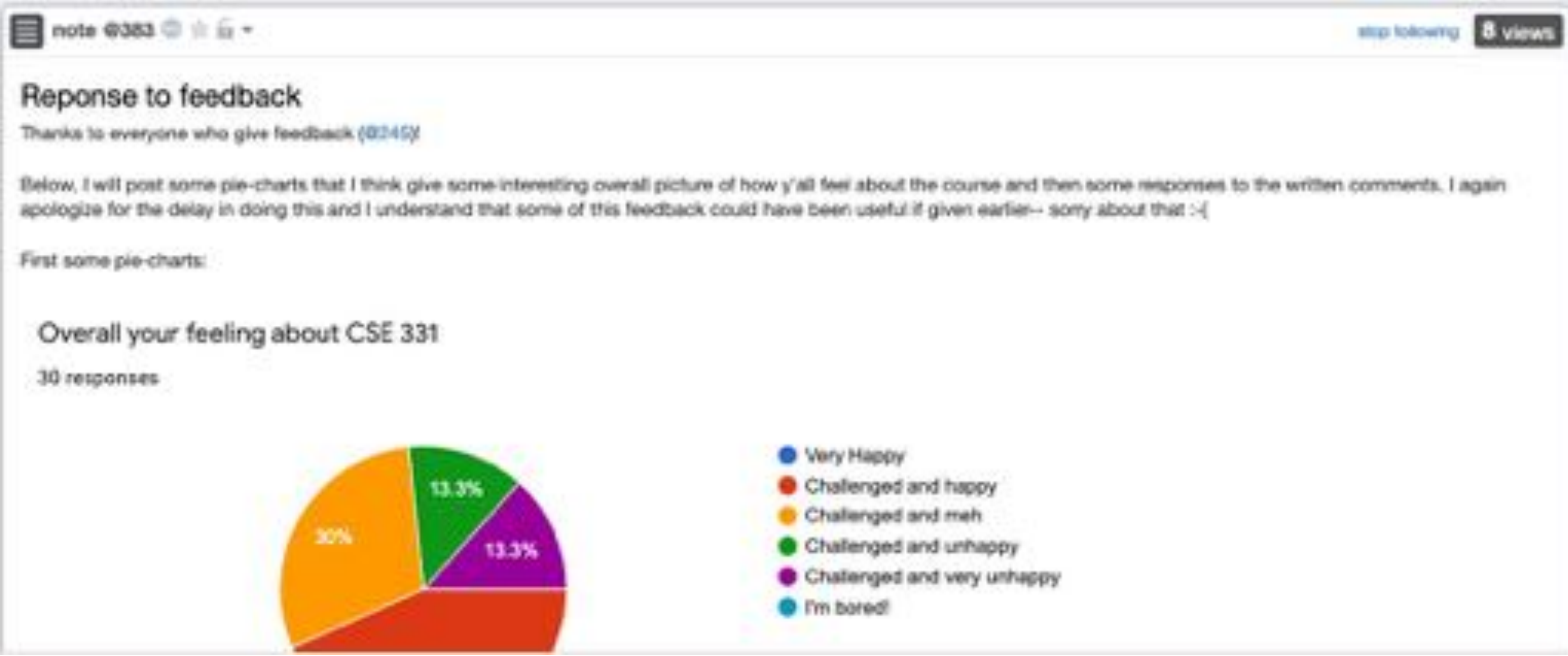
**Note:** These are my current availabilities-- some of the slots might be used up in some other non-CSE 331 meetings. So please send multiple choices for when you can meet.

We can have the meeting either in person (Davis 319) or on zoom (https://buffalo.zoom.us/j/[illegible]) **except for Th meeting which are virtual ONLY**.

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

- **Thursday (Oct 21) [VIRTUAL ONLY]:** 11:00am, 11:15am, 11:30am, 11:45am, 12:00pm, 12:15pm, 12:30pm, 12:45pm, 1:00pm, 1:15pm, 1:30pm, 1:45pm, 4:00pm, 4:15pm, 4:30pm
- **Friday (Oct 22):** 2:15pm, 2:30pm, 2:45pm, 4:00pm, 4:15pm, 4:30pm
- **Monday (Oct 25):** 9:30am, 9:45am, 2:00pm, 2:15pm, 2:30pm, 2:45pm, 3:00pm, 4:00pm
- **Tuesday (Oct 26):** 9:30am, 9:45am, 12:30pm, 12:45pm, 2:00pm, 2:15pm, 2:30pm, 2:45pm, 3:00pm, 3:15pm, 3:30pm, 3:45pm, 4:00pm, 4:15pm
- **Wednesday (Oct 27):** 1:30pm, 1:45pm, 2:00pm, 2:15pm, 2:30pm, 2:45pm, 3:00pm, 4:00pm, 4:15pm, 4:30pm

# Response to feedback


note @383

stop following 8 views

## Reponse to feedback

Thanks to everyone who give feedback (@345)!

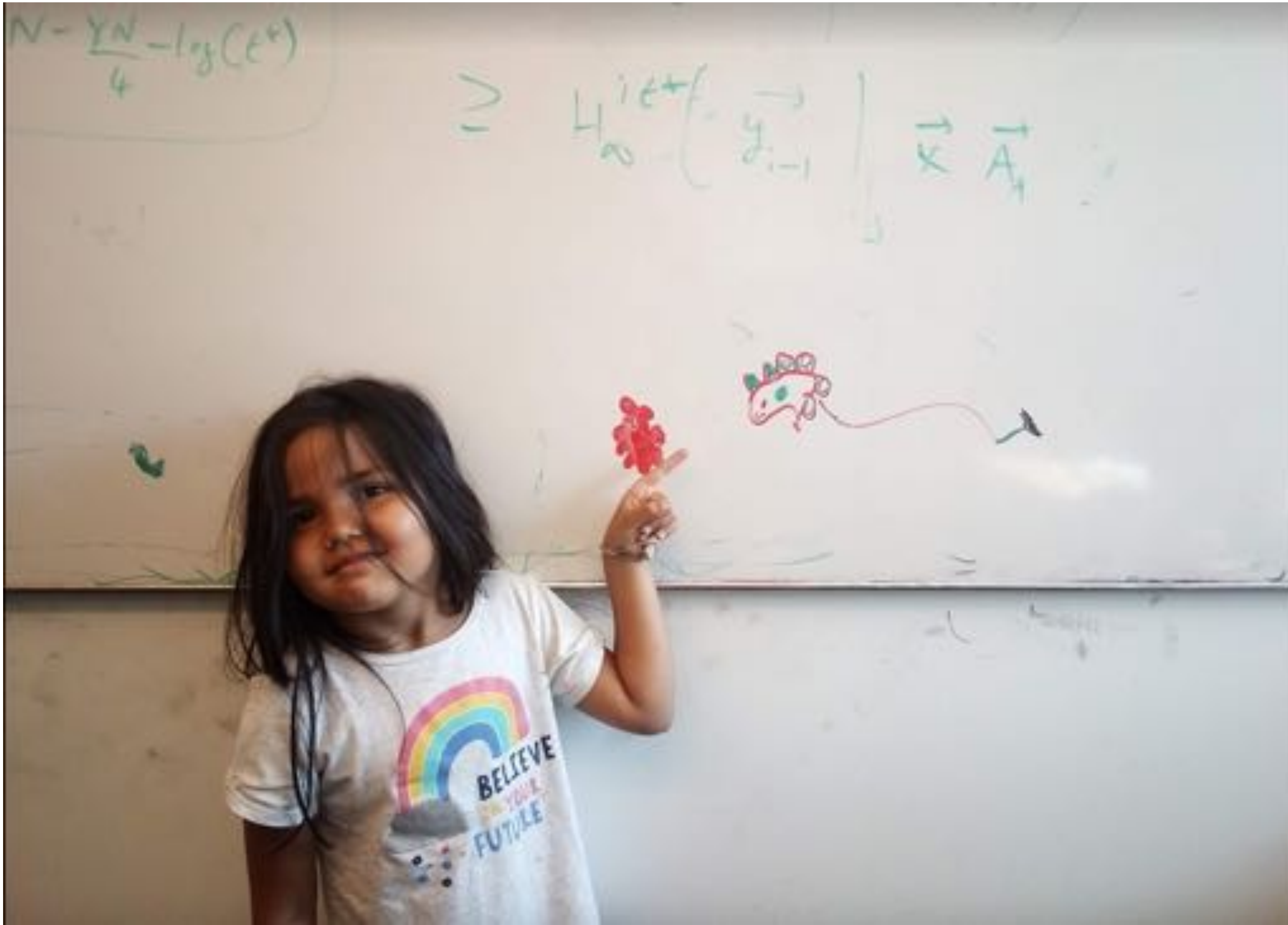
Below, I will post some pie-charts that I think give some interesting overall picture of how y'all feel about the course and then some responses to the written comments. I again apologize for the delay in doing this and I understand that some of this feedback could have been useful if given earlier-- sorry about that :-(

First some pie-charts:

Overall your feeling about CSE 331

30 responses

# Questions/Comments?



# Kruskal's Algorithm

Joseph B. Kruskal

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

$T = \emptyset$

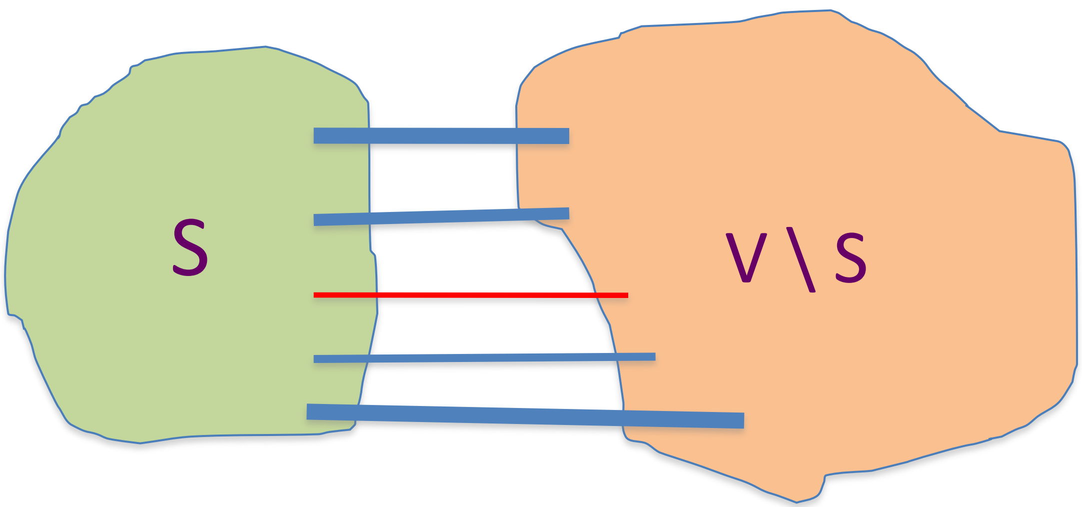
Sort edges in increasing order of their cost

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

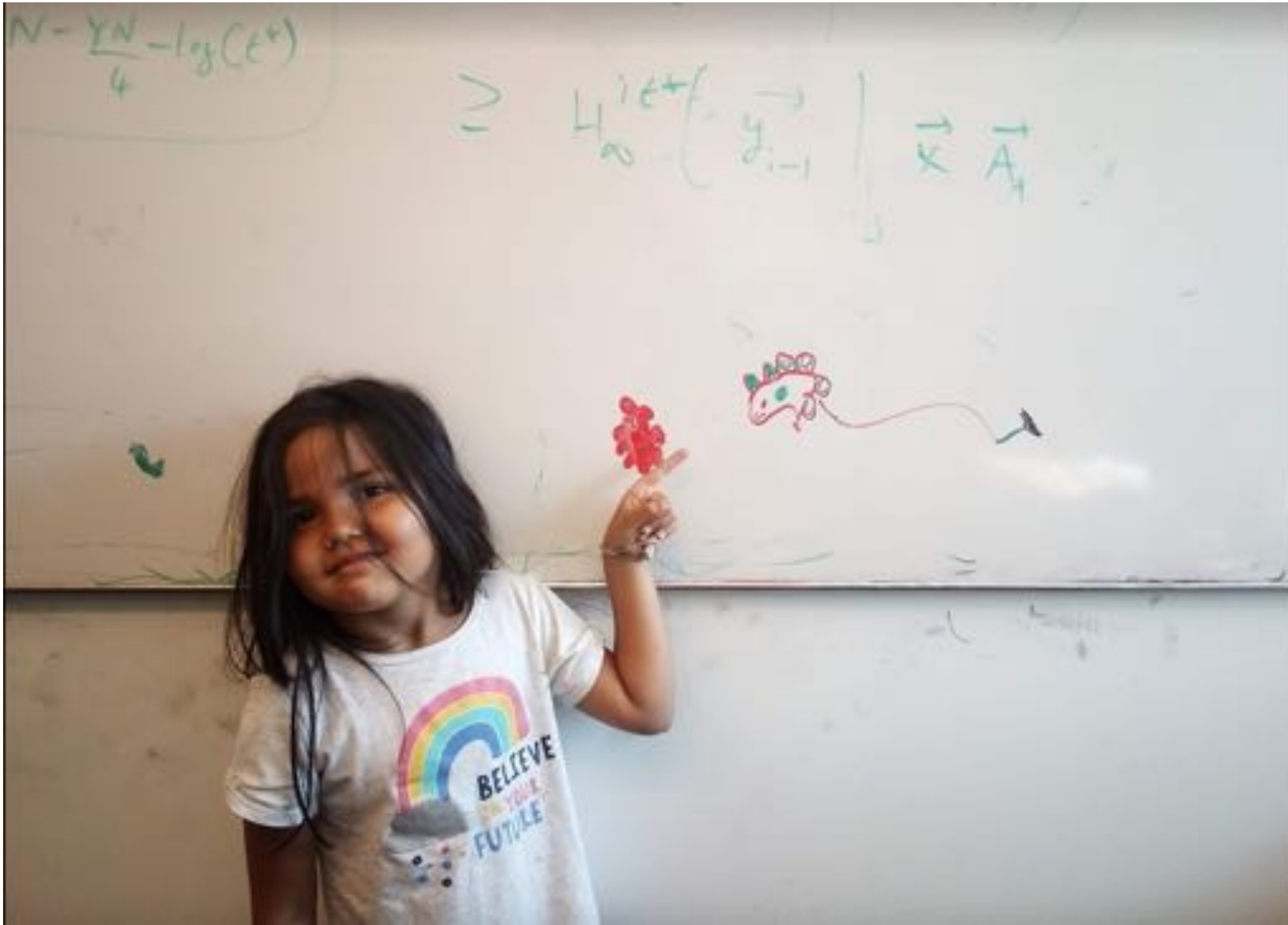
Condition: S and V\S are non-empty




Cheapest crossing edge is in **all** MSTs

Assumption: All edge costs are distinct

# Questions/Comments?



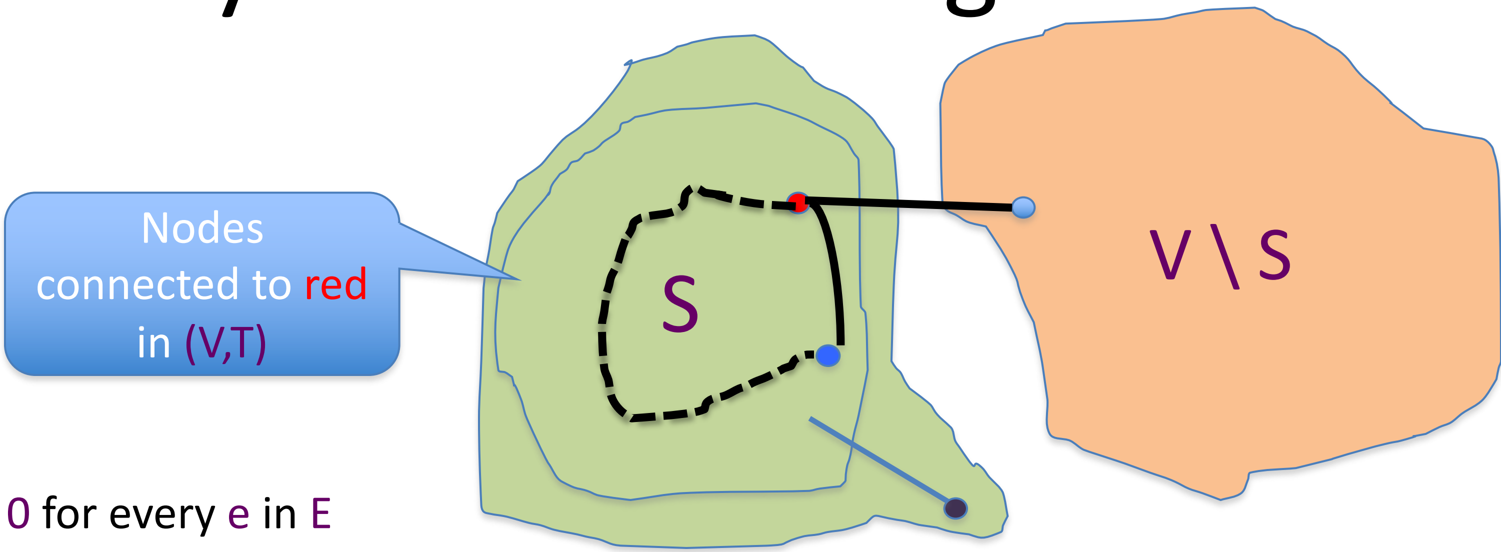
# Today’s agenda

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

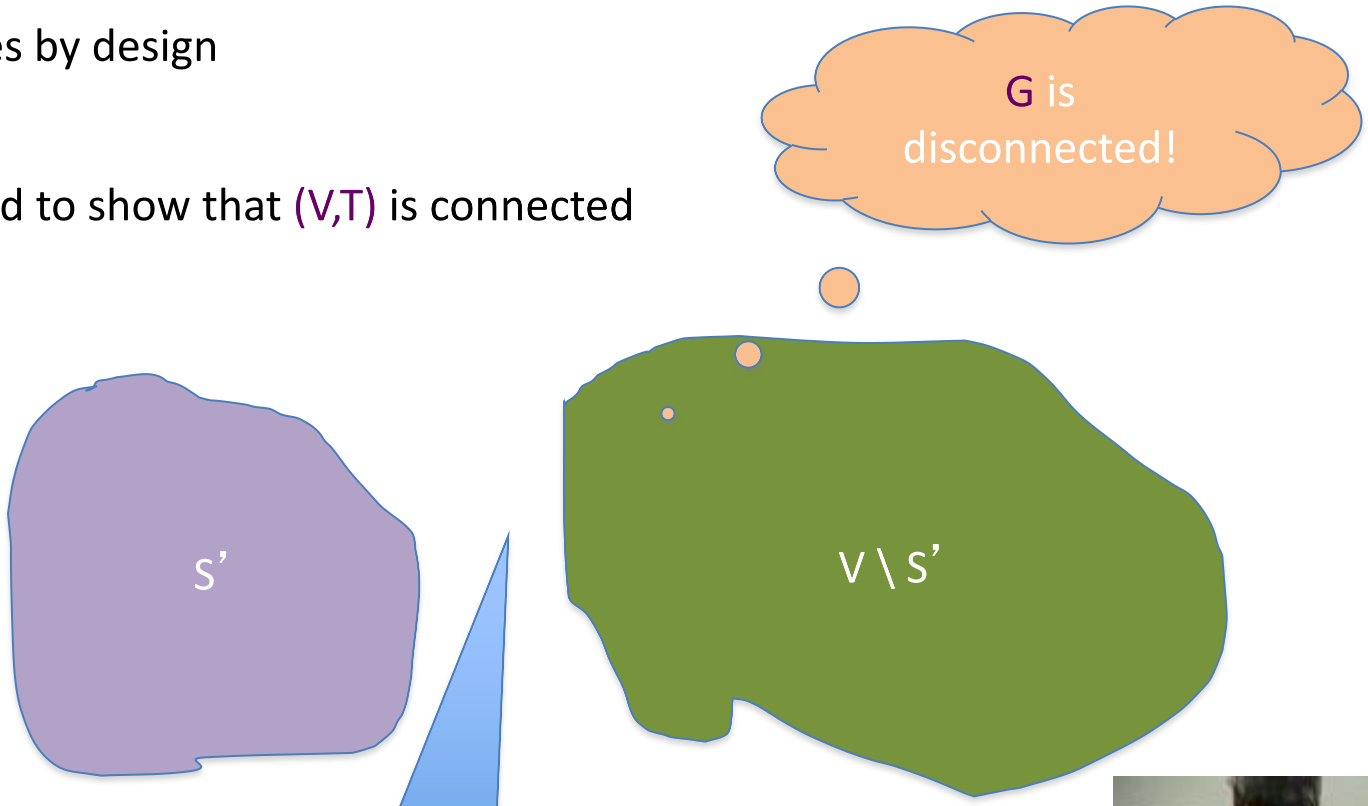
V\S is non-empty

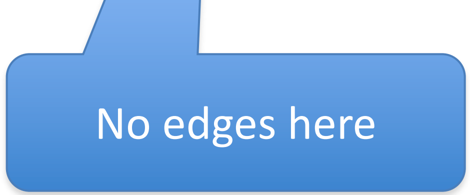
First crossing edge considered

# 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

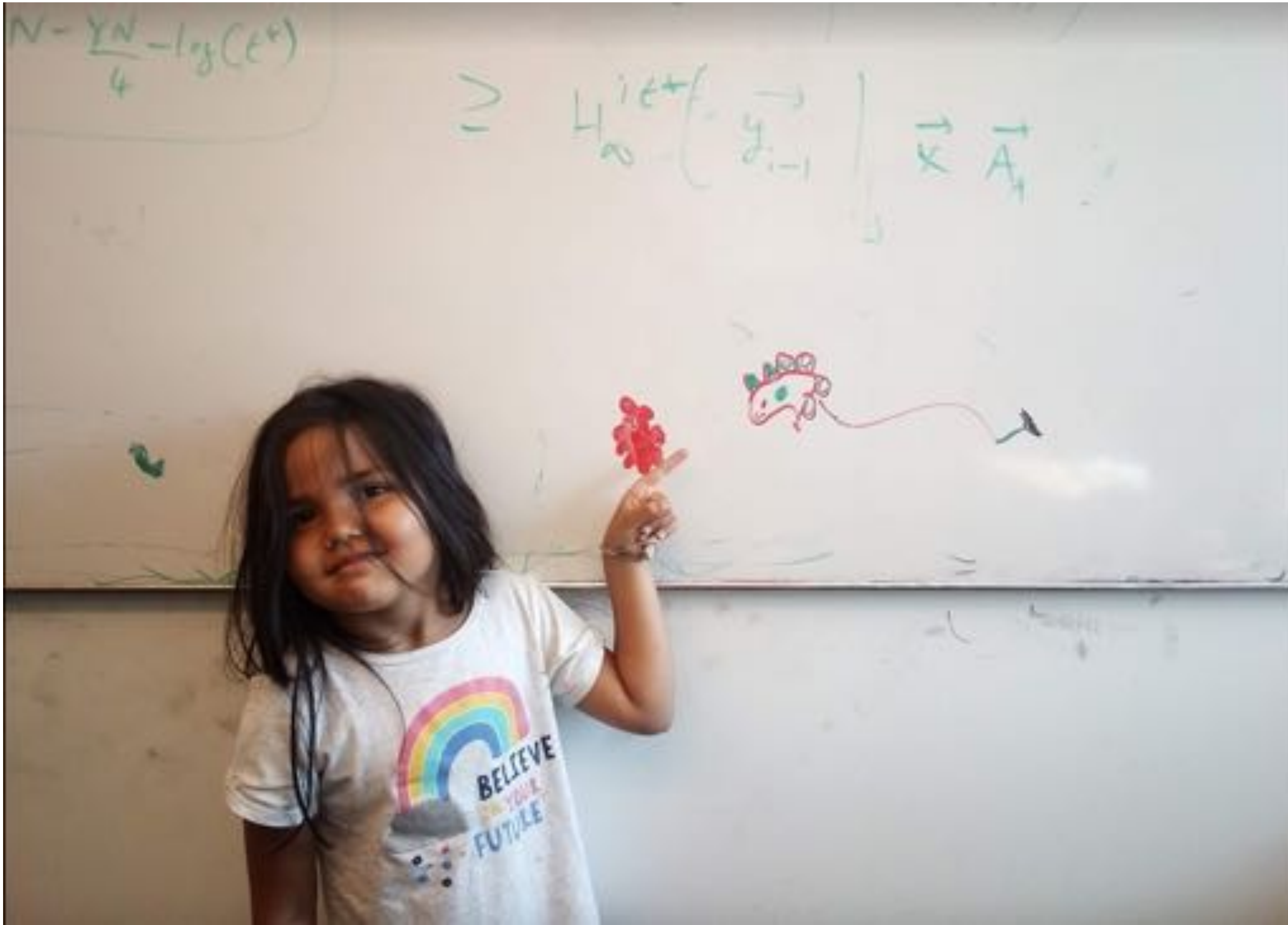


MST for "perturbed" weights is the same as for original

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

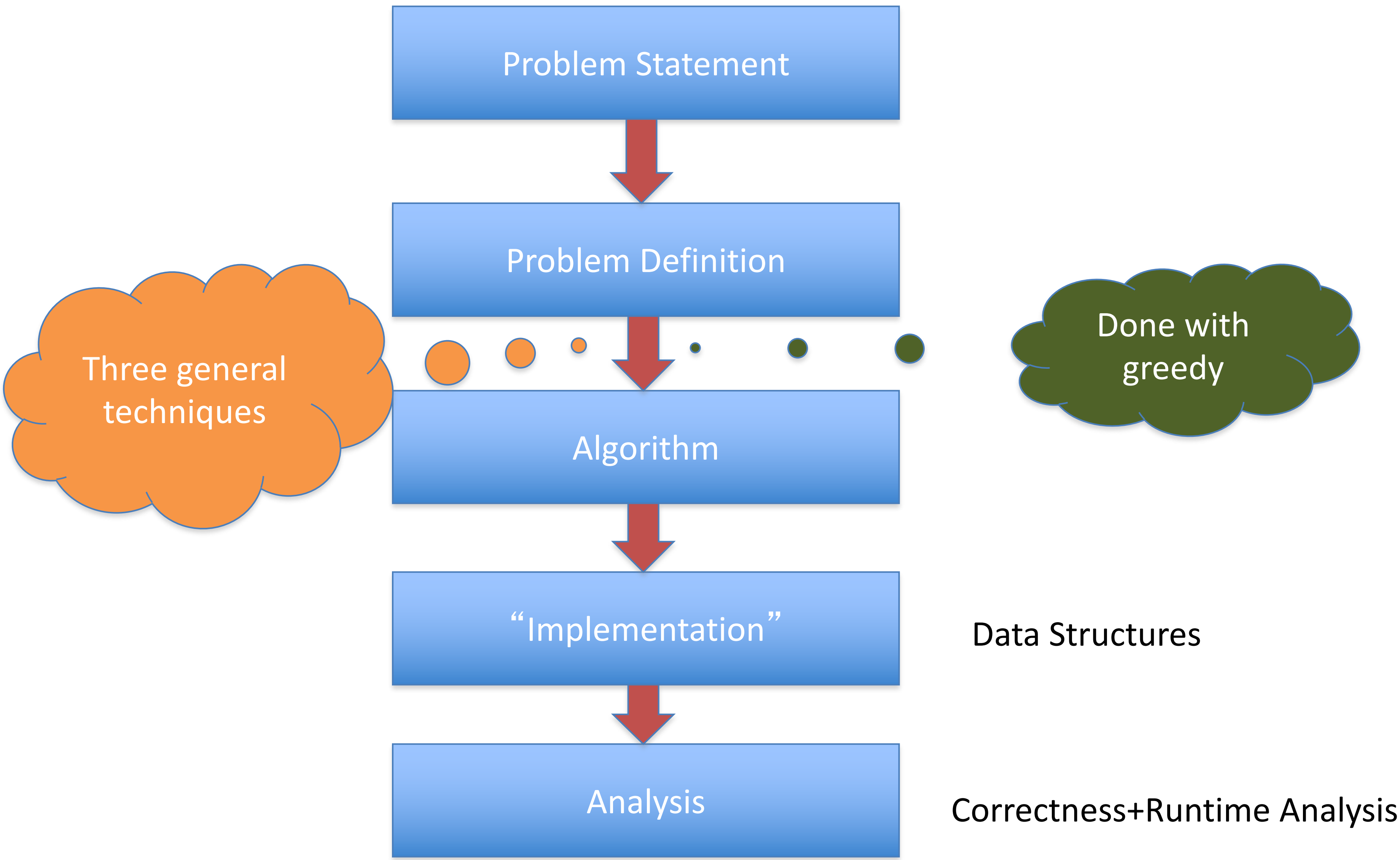
If an edge can be added to $T$ without adding a cycle then add it to $T$

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: $b_1, b_2, \ldots, b_n$

$b_1 = a_1$

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”

    $b_j = a_i$

$O(n^2)$ overall

$O(\log n)$

$O(n)$

Make sure that all the processed numbers are sorted

| a | b |
|---|---|
| 4 | [illegible] |
| 3 | [illegible] |
| 2 | [illegible] |
| 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

Unless n=2

Sort each half recursively

Merge the two sorted halves into one sorted output

# How fast can sorted arrays be merged?

Group talk time

# Mergesort algorithm

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

Output: Numbers in sorted order

MergeSort( a, n )

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

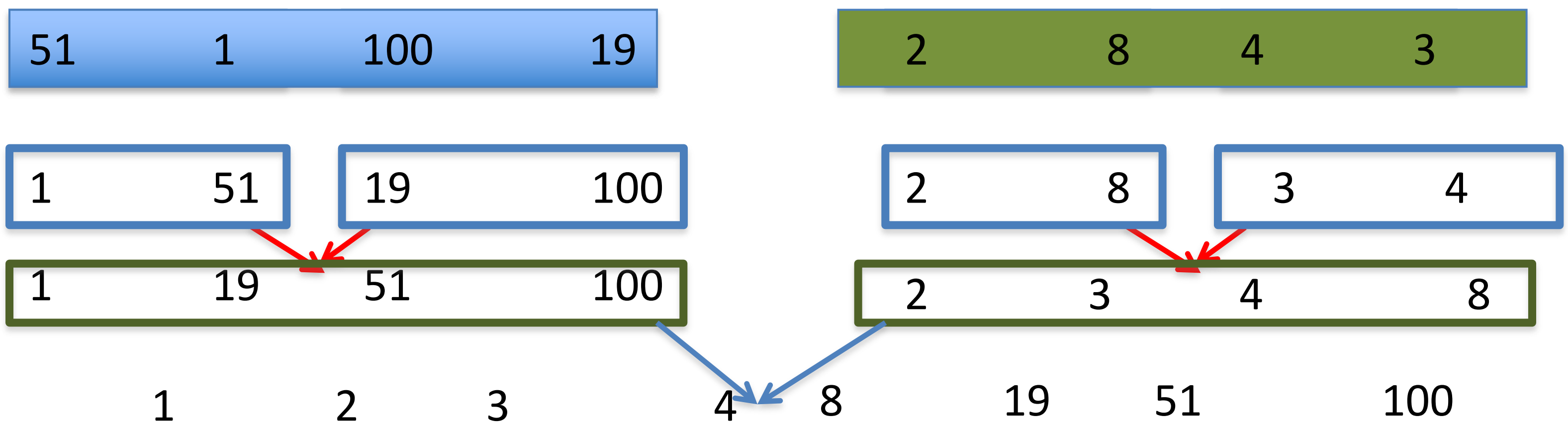
If $n = 2$ **return** the order $\min(a_1,a_2); \max(a_1,a_2)$

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

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

**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…