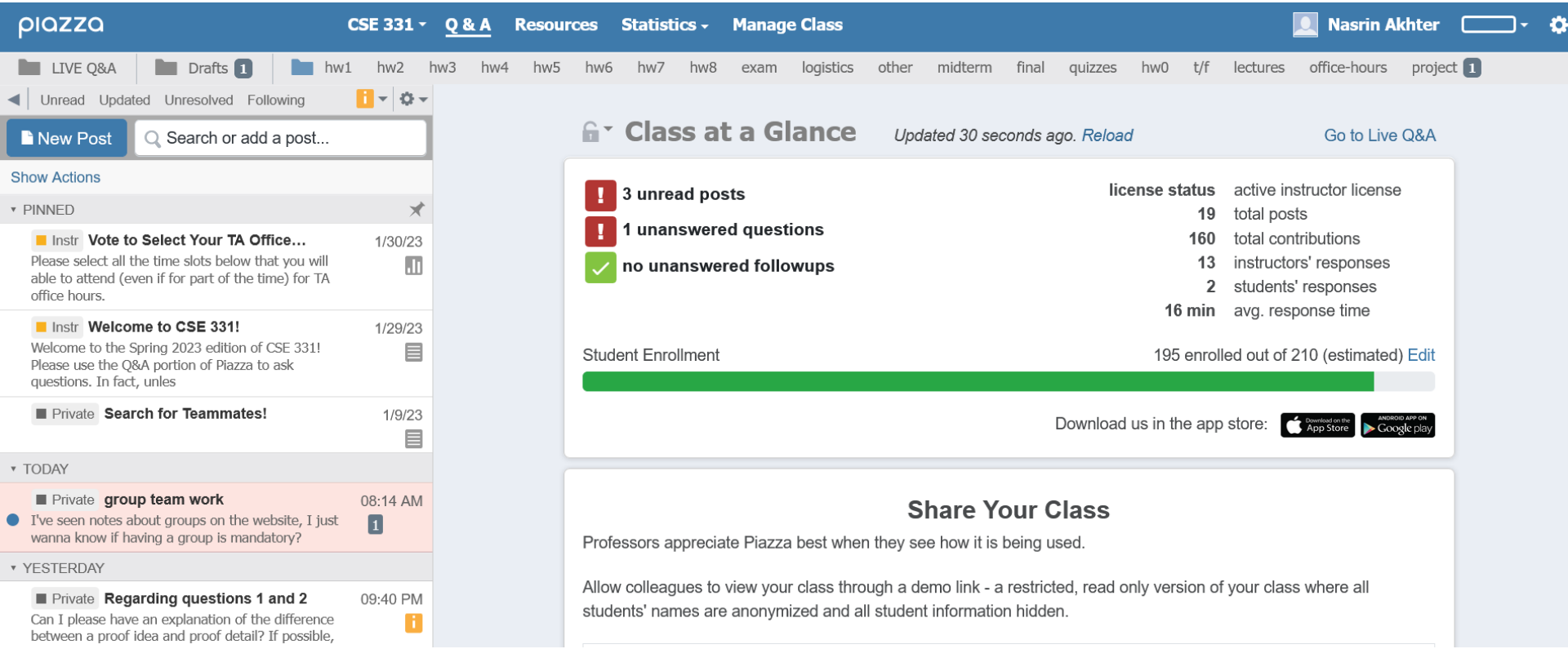
Lecture 2

CSE 331

Enroll on Piazza



https://piazza.com/buffalo/spring2023/cse331/home

Read the syllabus CAREFULLY!

Syllabus Quiz

Admin Options

CA Options

Options

View handin history

View writeup

Download handout

Due: **May 15th 2023, 11:59 pm**

Last day to handin: **May 15th 2023, 11:59 pm**

No graded material will be handed back till you pass the syllabus quiz!

Autolab FAQ

Autolab

Details on Autolab, which will be used for all homework submissions in CSE 331.

The main link

We will be using the UB CSE extension to Autolab for submission and (auto)grading of CSE 331 homeworks. You can access Autolab via https://autograder.cse.buffalo.edu/.

Signing up

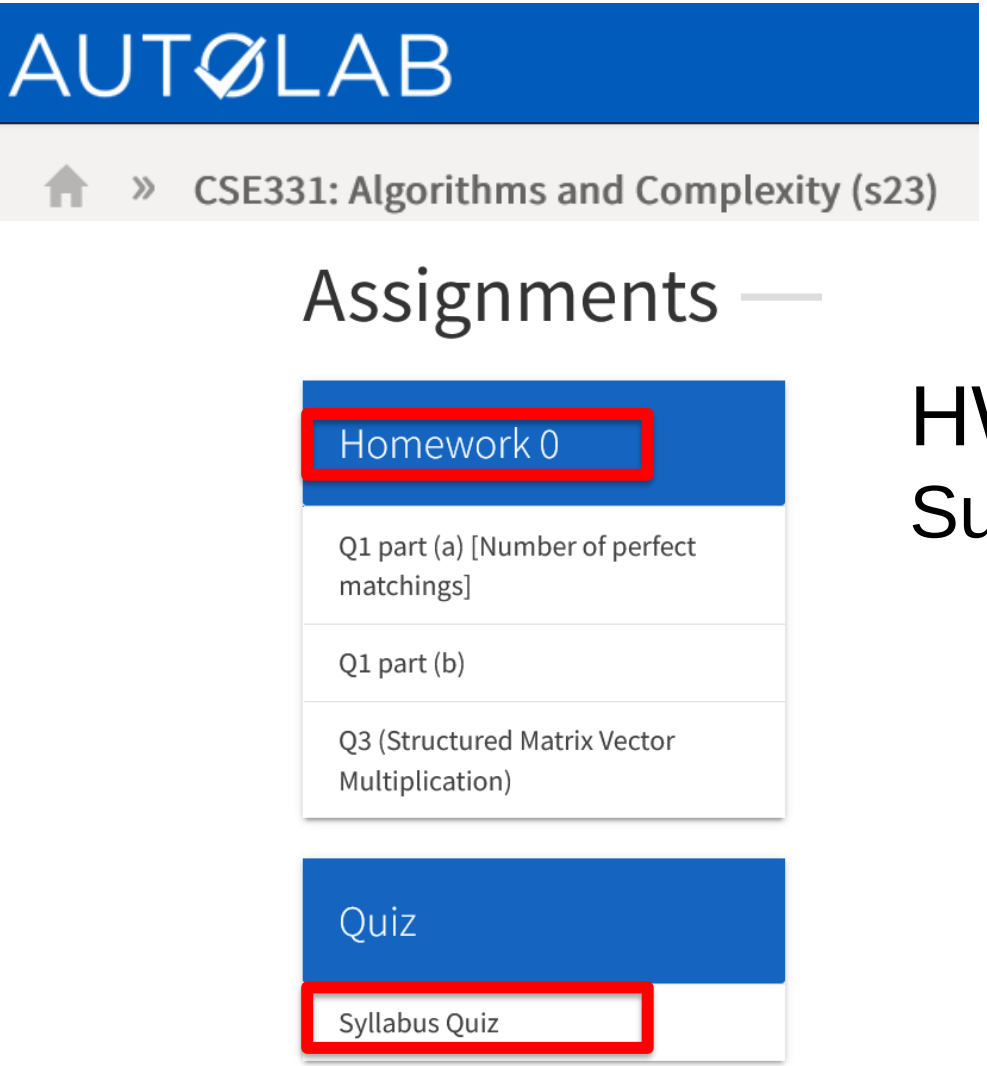
Follow these steps to setup an account on Autolab (unless you already have one in which case you'll use your existing account):

1. Go to this page and click on the Sign in with MyUB link. A new account will automatically be created for you.
2. I believe Autolab should now be using your preferred name instead of your official UB first and last name. **If this is not the case, please let us know ASAP.**
3. We will have leader boards for all the programming assignments. For anonymity, all students are identified by their chosen nicknames. So please make sure you pick an appropriate one (you can change your nickname at any point in time).
4. After you have done the above steps, you wait.

What happens next

Here are the steps that we need to take on our end:

HW 0 is Out



HW0 is out!

Submit Q1 and Q3 (not Q2).

HW 0 is Out

Q1 part (a) [Number of perfect matchings]

Admin Options

CA Options

Options

View handin history

View writeup

Download handout

Due: **February 6th 2023, 9:00 am**

Last day to handin: **February 6th 2023, 9:00 am**

Sources *:

! PDF only please

Collaborators *:

* denotes required fields. The submission cannot be completed without filling out the required fields.

HW 0 is Out

Q1 part (b)

Admin Options

CA Options

Options

View handin history

View writeup

Download handout

Due: **February 6th 2023, 9:01 am**

Last day to handin: **February 6th 2023, 9:01 am**

Sources *:

Collaborators *:

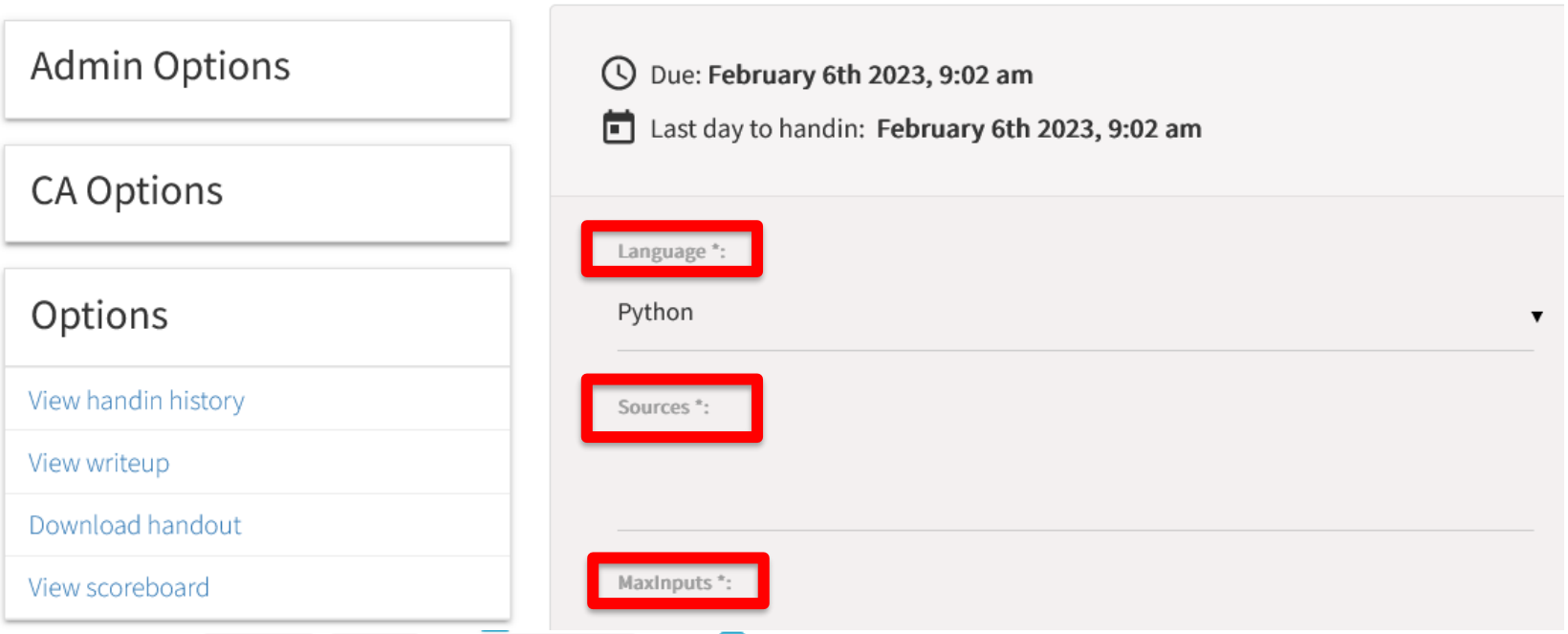
! PDF only please

* denotes required fields. The submission cannot be completed without filling out the required fields.

I affirm that I have complied with this course's academic integrity policy as defined in the syllabus.

HW 0 is Out

Q3 (Structured Matrix Vector Multiplication)



- What do the textfields `MaxInputs`, `Sources`, and `Collaborators` mean?

`MaxInputs` is only relevant to the third (programming) question, where you can specify the number of testcases you want the Grader to test your code on (for partial credit). This is especially useful if your code times out if you run it on all 10 inputs. `Sources` and `Collaborators` applies to all assignments and their use is outlined on the Homework Policy page.

https://cse.buffalo.edu/~nasrinak/cse331/SP23/autolab.html

Allowed Sources

Allowed sources

You can **ONLY** use the following sources for reference once you start working on the homework problems:

1. **the Kleinberg-Tardos textbook,**

> ### Other textbooks are not allowed
> While you can use other textbooks (e.g. those listed in the syllabus) to better understand the lecture material, you **cannot** use them once you start working on the homeworks.

2. **any material linked from this webpage or the CSE 331 piazza page (including any discussion in the Q&A section),**

> ### One-click rule
> When using webpages that are allowed as sources, you **cannot** click on link on that source. (Otherwise within a constant number of clicks one can reach any webpage one wants.)

3. **specific *mathematical* result from a previous course,**
4. **anything discussed in the lectures, recitations and/or office hours** and
5. **any notes that you might have taken during class or recitation.**

> ### Everything else is not allowed
> Note that the above list covers all the allowed sources and **everything else is not allowed.** In particular, *YOU ARE NOT SUPPOSED TO SEARCH FOR SOLUTIONS ON THE*

… even for programming Q

CSE 331 | Syllabus | Piazza | Schedule | Homeworks | Autolab | Mini Project | Support Pages | Youtube channel

- All discussions and posts on piazza.

Basic programming references

C++ Sources

- cppreference.com (and all pages within the website).

Java Sources

- Oracle Java Documentation (and all pages within the website).

Python Sources

- Python 3.5.2 documentation (and all pages within the website).

Asymptotic Analysis

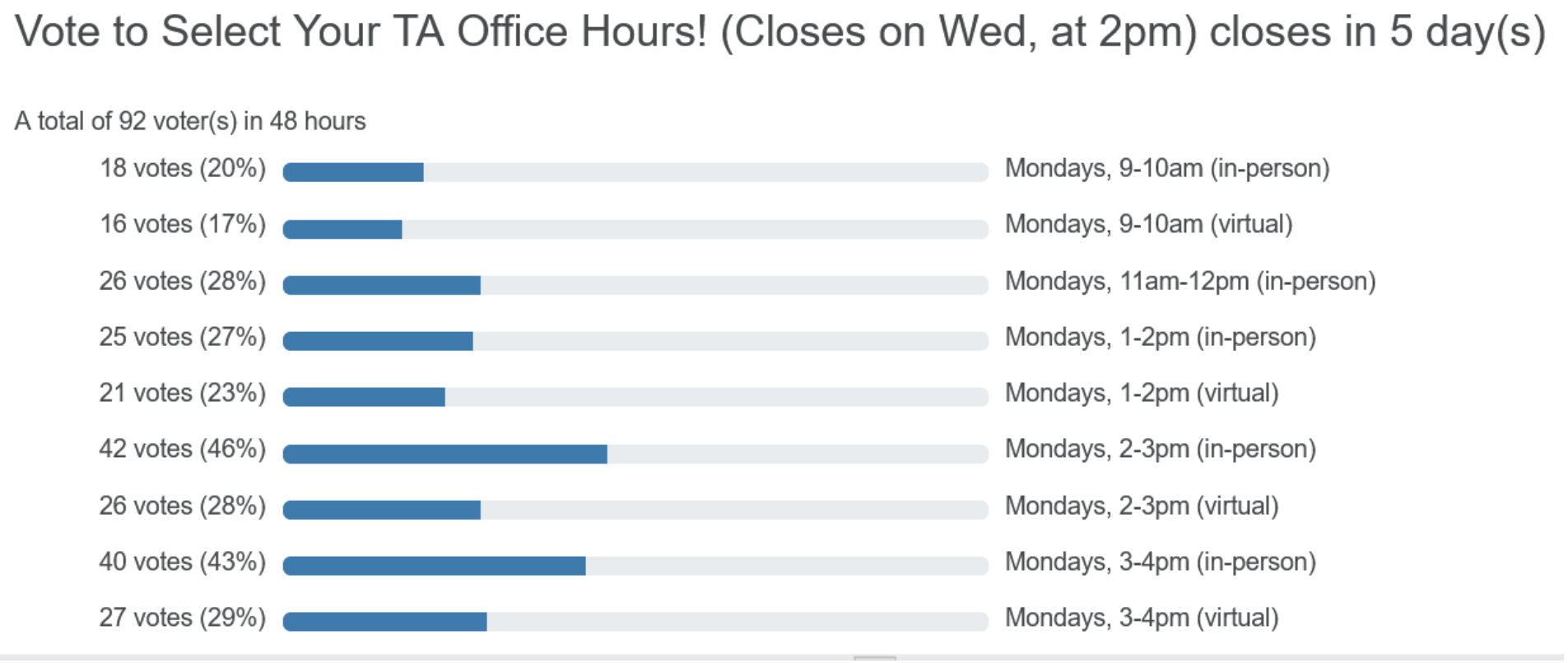
- Big-O cheat sheet.

Wikipedia Pages

Below are some approved Wikipedia pages (in addition to those that are already linked to in other pages in the CSE 331 Fall 2018 web page.

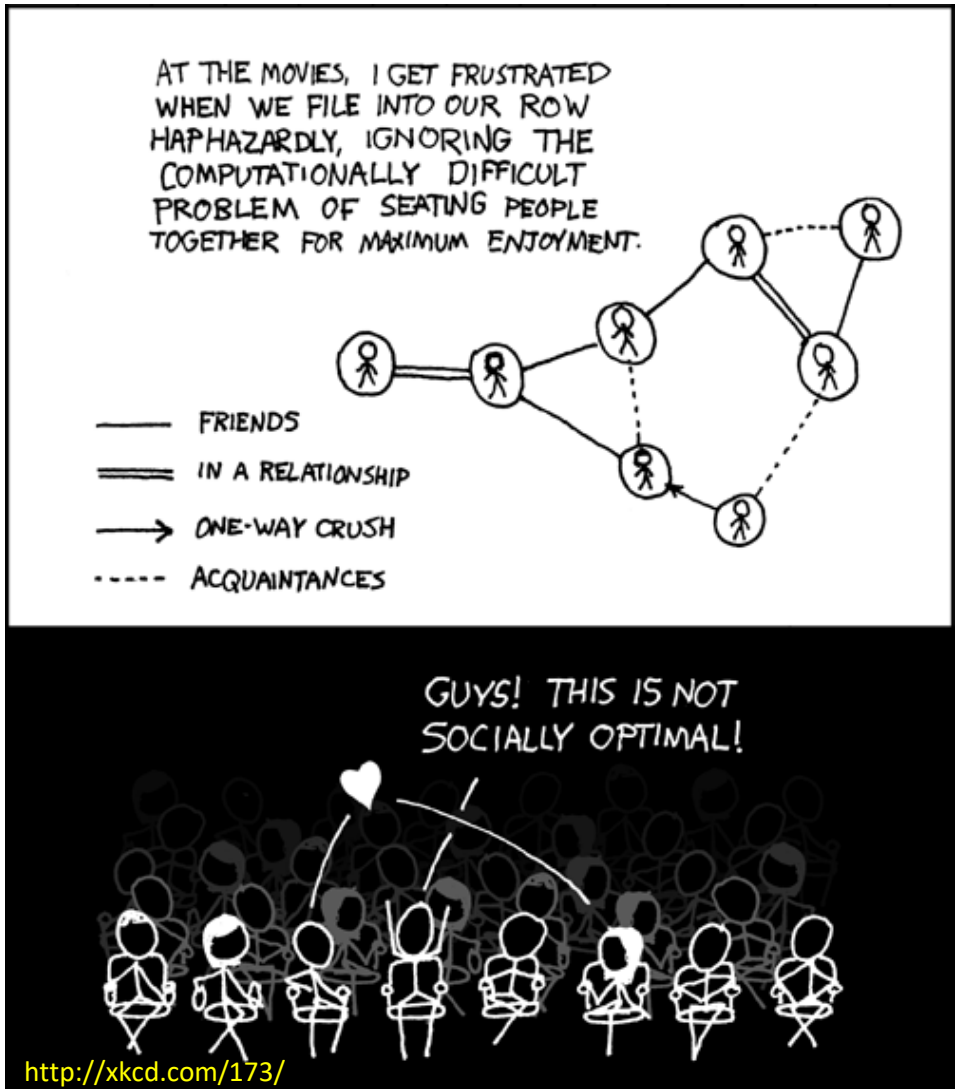
- Gale Shapley algorithm.
- DFS.
- Dijkstra's Algorithm.
- Prim's algorithm

TA Office hours finalize tomorrow

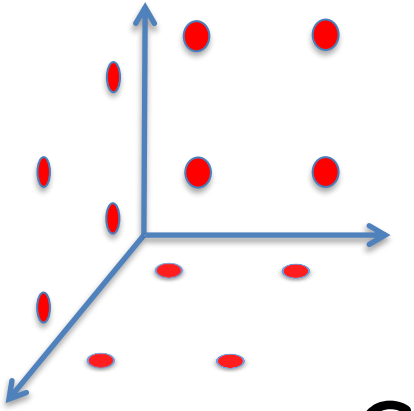


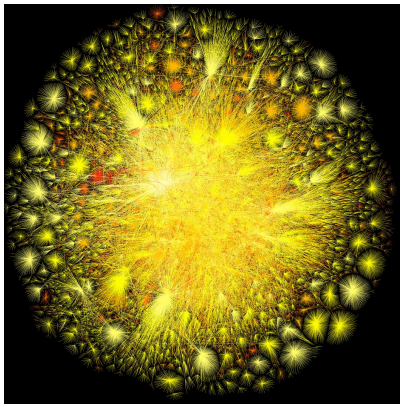
Questions/Comments?

This course: how to solve problems!



Why should I care ?





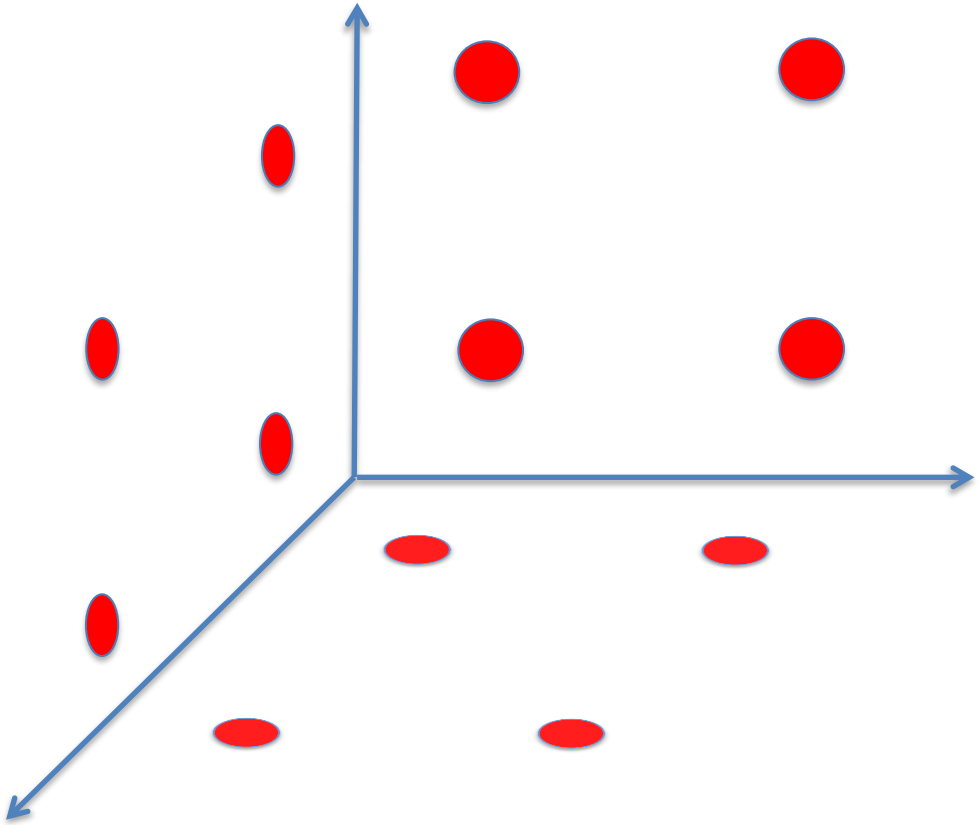
Combining Shadows to Understanding the network

relationalAI

rAI

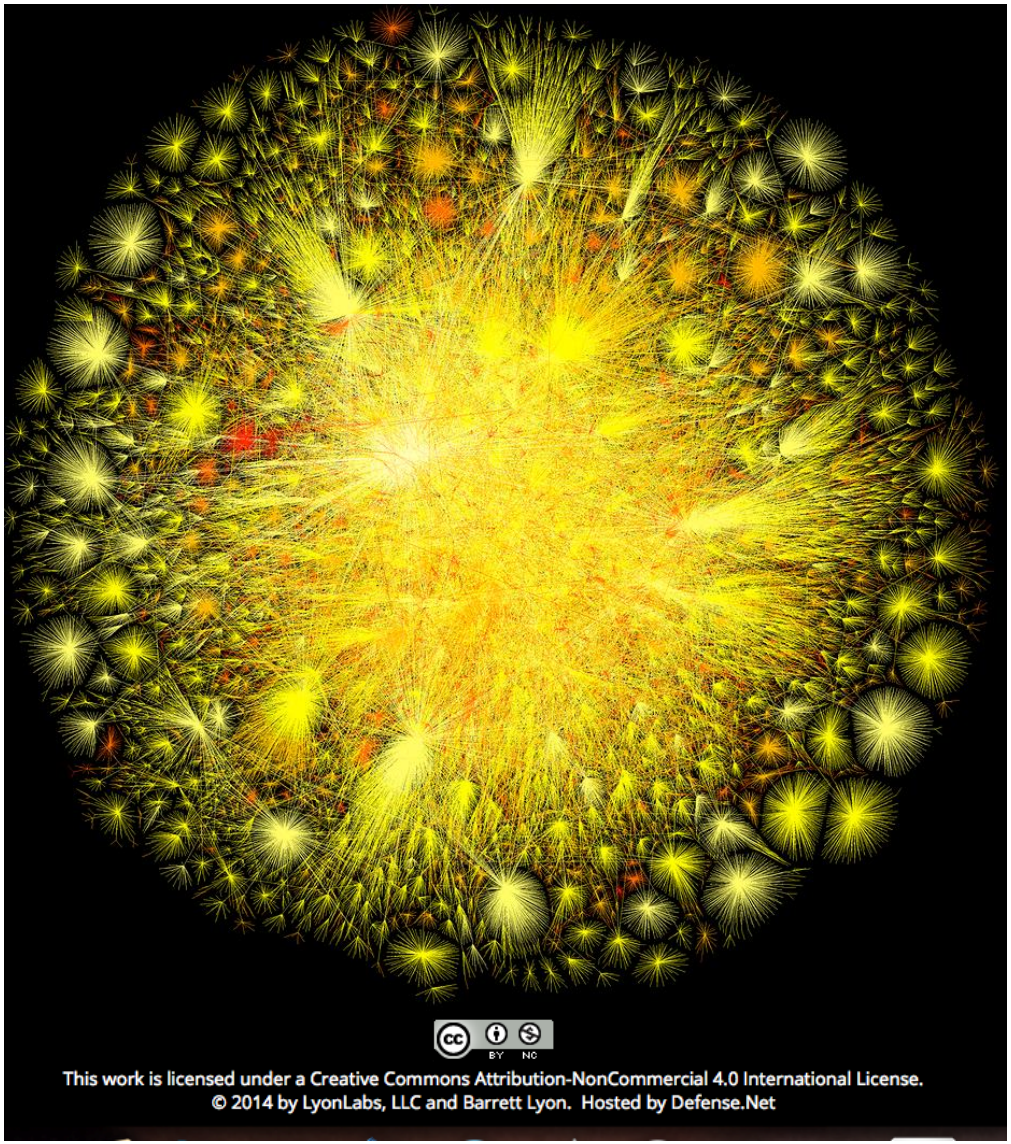
Stanford University

The key technical problem

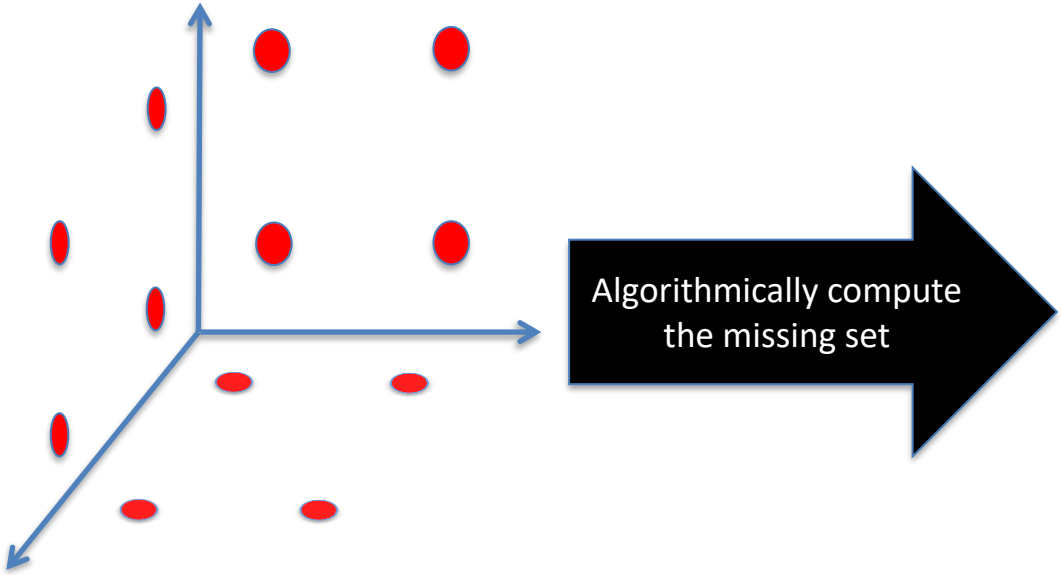


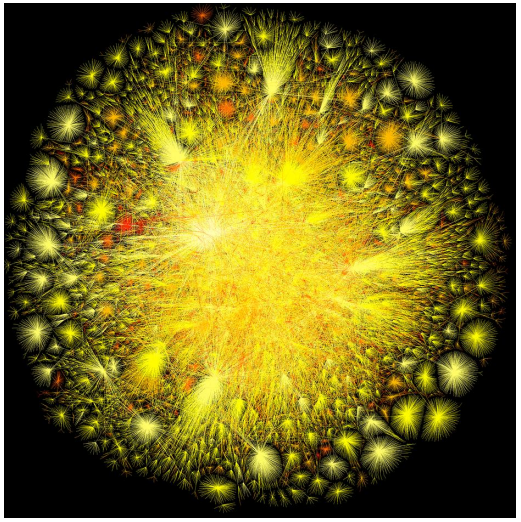
Given the three projections, what is the largest size of the original set of points?

Detecting Communities



Conquering Shadows to Conquering the Internet





The proof is in the performance

EMPTYHEADED




10x faster

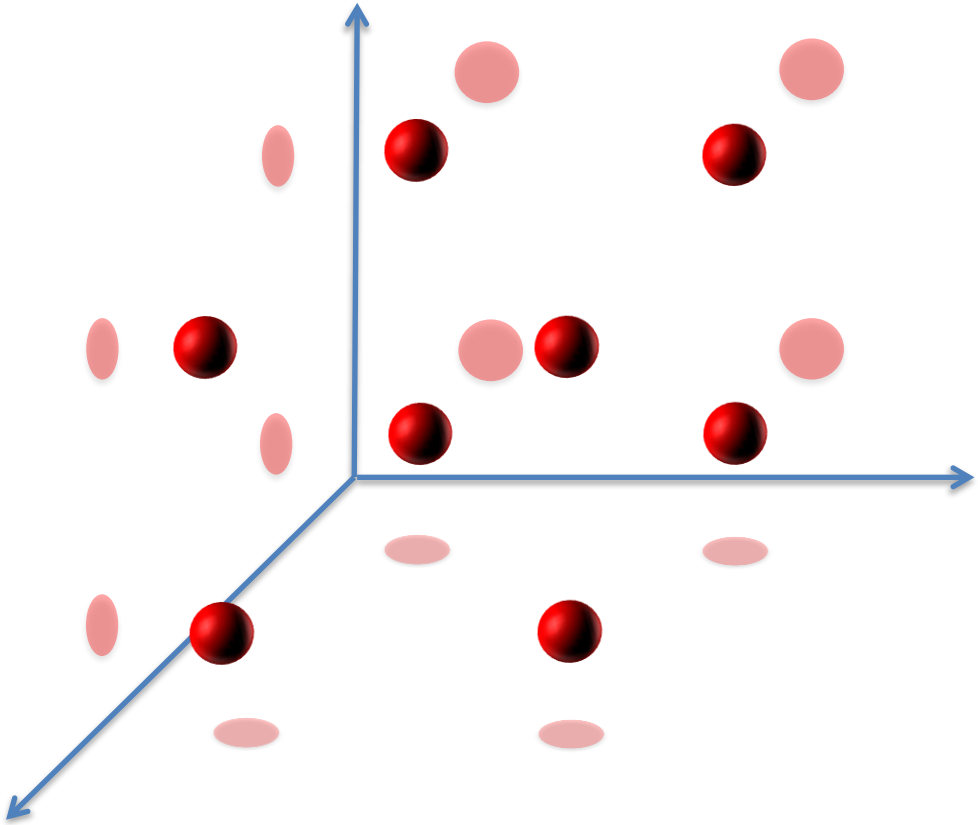
ORACLE

Dato GraphLab

LogicBlox

Better algorithm with little hacking will beat a worse algorithm with tons of hacking

The key technical problem



Highly trivial: $4^3 = 64$

Still trivial: $4^2 = 16$

Correct answer: $4^{1.5} = 8$

If detecting communities is not for you

From someone who got a Google job

“You can let your algorithms class know that the phone interviews are essentially like **a difficult algorithms test**.

Lots of data structures, specifying the algorithm, analyzing the run time and space requirements... And all on the phone and **you're supposed to talk through your thought process.”**

Coding jobs will be done by AI

stack**sort**

In a recent xkcd's alt text, Randall Munroe suggested **stacksort**, a sort that searches StackOverflow for sorting functions and runs them until it returns the correct answer. So, I made it. If you like running arbitrary code in your browser, try it out.

Like (or hate) it? Comment on HackerNews

stackoverflow_sort(
Try a list of numbers, a string, a list of words or json.

[8,6,7,5,3,0,9]

);

Sort

var output =
Output from the function.

;

output console

Coding jobs will be done by AI

MIT News
ON CAMPUS AND AROUND THE WORLD

Browse or Search

FULL SCREEN

Researchers have developed a flexible way of combining deep learning and symbolic reasoning to teach computers to write short computer programs. Here, Armando Solar-Lezama (left), a professor at CSAIL, speaks with graduate student Maxwell Nye.

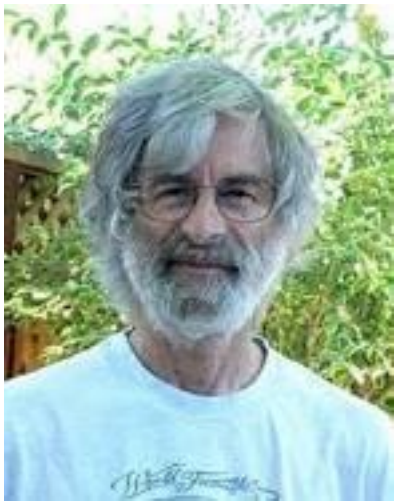
Photo: Kim Martineau

Toward artificial intelligence that learns to write code

Researchers combine deep learning and symbolic reasoning for a more flexible way

So am I doomed?

There will still be room for high level *algorithmic* thinking!



Today, programming is generally equated with coding. It's hard to convince students who want to write code that they should learn to think mathematically, above the code level, about what they're doing. Perhaps the following observation will give them pause. ***It's quite likely that during their lifetime, machine learning will completely change the nature of programming. The programming languages they are now using will seem as quaint as Cobol, and the coding skills they are learning will be of little use. But mathematics will remain the queen of science, and the ability to think mathematically will always be useful.***

Questions/Comments?

Proof Idea vs. Proof Details

Questions 1 and 2

For Q1 and Q2, think of the algorithm and proof ideas as things that go inside a header (`.h`) file. They are the high level overview of how you are approaching the problem; you don't have to be very technical here. For example, listing out all the steps in your algorithm, what proof technique are you using, what property of the algorithm are you induction on, etc.

Algorithm and proof details are the implementation inside the source (`.cc`) file. They are simply the detailed technical algorithm/ proof of the idea that was outlined.

More on the idea vs details divide

Always start off with the ideas. Just smashing random keys on the keyboards won't get you anywhere with writing code and certainly would not help with proofs. In the real world, a user of your library doesn't care about the details; just wants to know how to use it. Similarly, in your proof and algorithm ideas, briefly explain what you're doing, how it works and why it should work. For example, if you're using contradiction in the proof details; just state that you use contradiction on a specific property (but do specify which property).

In the algorithm and proof details, be as detailed as you can be and try to avoid loopholes (more explained below).

Questions/Comments?