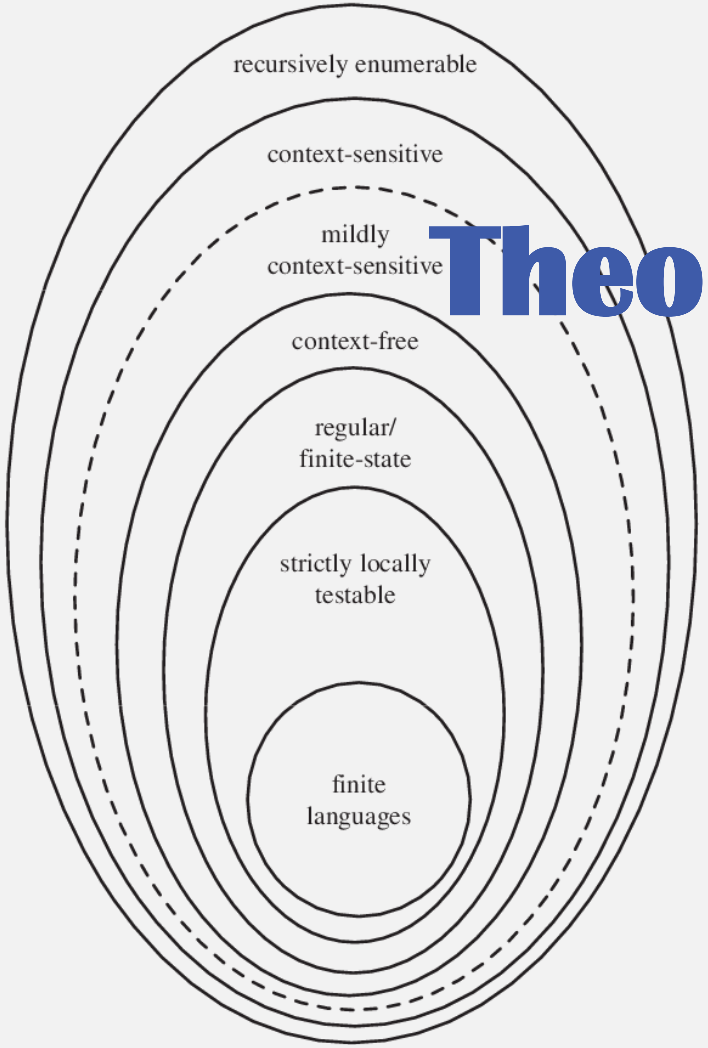


Welcome to CS622!

Theory of Formal Languages

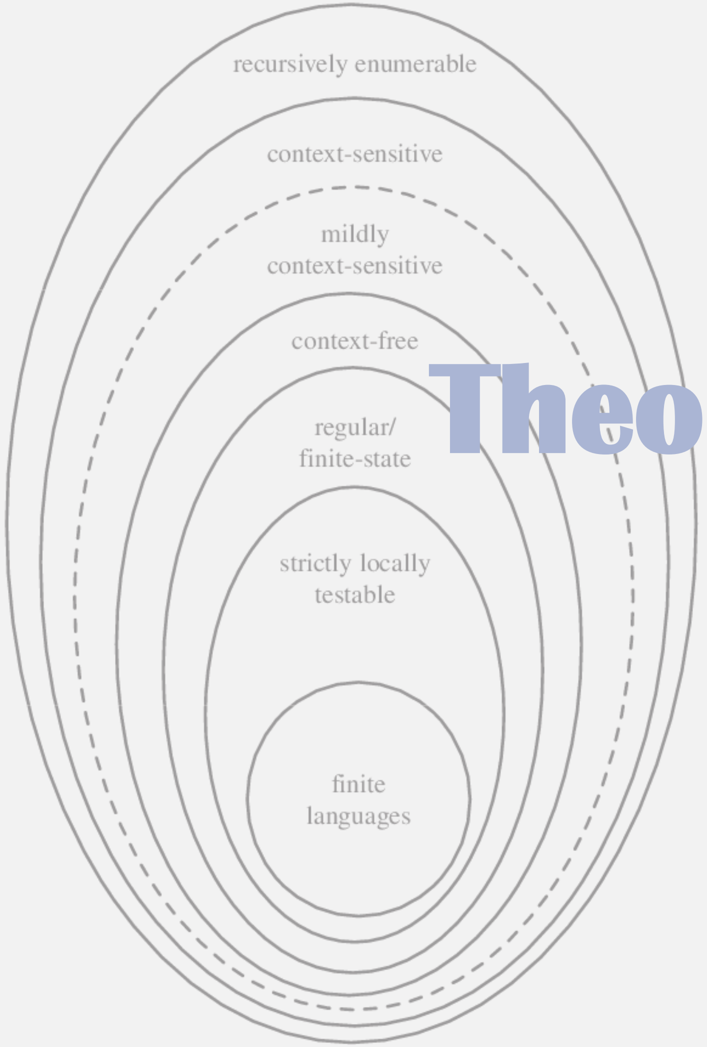
UMass Boston Computer Science

Instructor: Stephen Chang

Fall 2021

Lecture Logistics

- This is a remote class!
  - At least until Sept 30th
- Lectures will be recorded and posted to Blackboard/Echo360
  - Slides will typically be posted to the course web page before class
- Type questions into Zoom's chat
  - Don't use the hand raise feature
  - Please be patient since I may only monitor occasionally
- Keep audio and video off normally
- I may call on students randomly during lecture
  - Turn on audio and video when speaking
  - Please be presentable
- Quiz (5min) at end of every lecture (on gradescope)



Welcome to CS 622!

Theory of Formal Languages

UMass Boston Computer Science

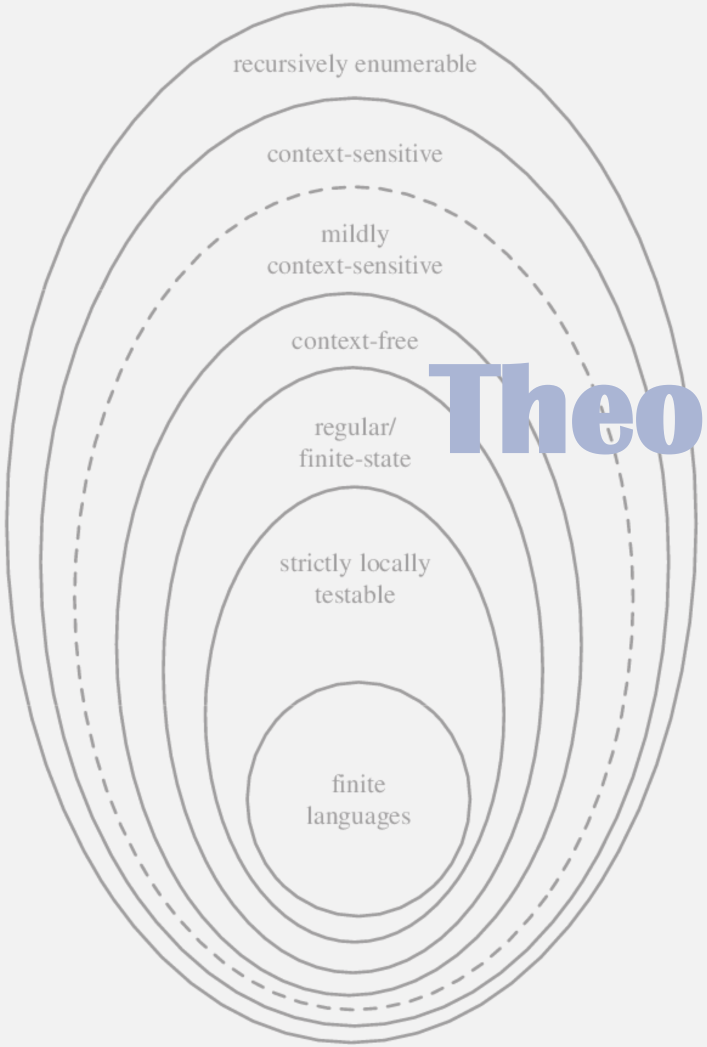
Instructor: Stephen Chang

Fall 2021

What's a "language"???

What’s a “language”?

What’s a “language”?



Welcome to CS 622!

Theory of Formal Languages

UMass Boston Computer Science

Instructor: Stephen Chang

Fall 2021

"Defined mathematically"

The Formal Definition of a Language

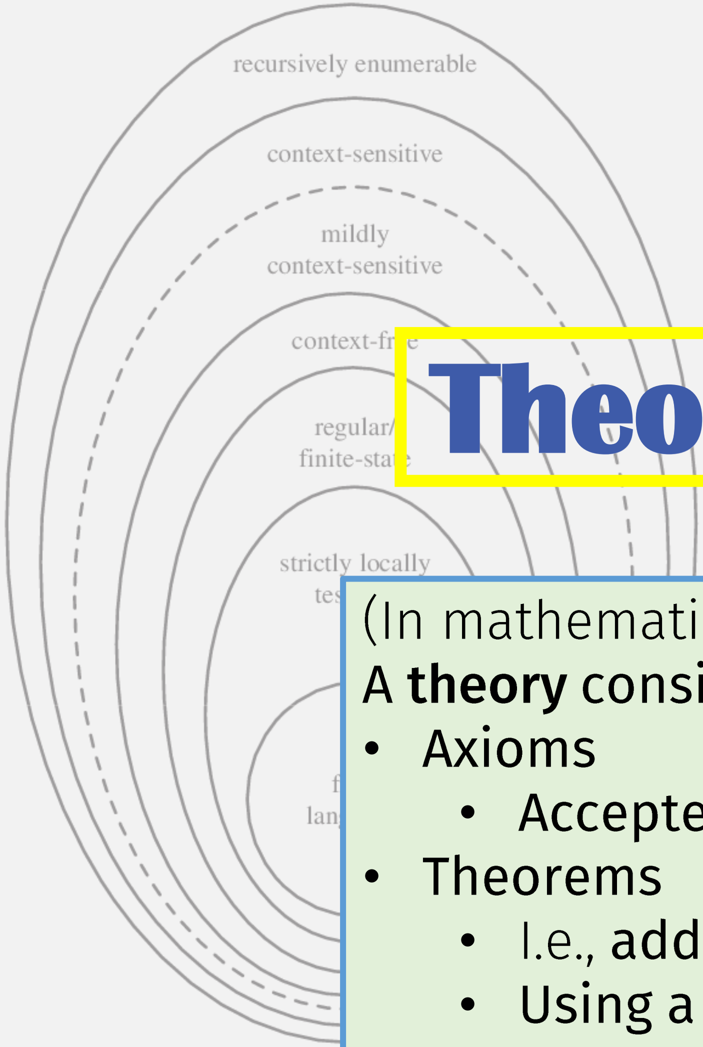
- A **language** is a (possibly infinite) set of *strings*

- A **string**/word is a (finite) sequence of chars from an *alphabet*

- An **alphabet** is a (finite, non-empty) set of chars/symbols

The Formal Definition of a Language

- A **language** is a (possibly infinite) set of *strings*
  - E.g., the set of all binary numbers
  - all Python programs
  - all words in English dictionary
  - $\Sigma^* =$ language of all possible strings over alphabet Σ
    - For all languages L over alphabet Σ, $L \subseteq \Sigma^*$
- A **string**/word is a (finite) sequence of chars from an *alphabet*
  - E.g., `010101`
  - `hello`
  - ε (sometimes λ) is the empty string (length zero string)
- An **alphabet** is a (finite, non-empty) set of chars/symbols
  - E.g., {`0`, `1`} (binary digits, the alphabet of computers)
  - {a, b, …, z} (lowercase letters)
  - set of ASCII chars
  - Alphabets are often denoted with the Σ symbol



Welcome to CS 622!

Theory of Formal Languages

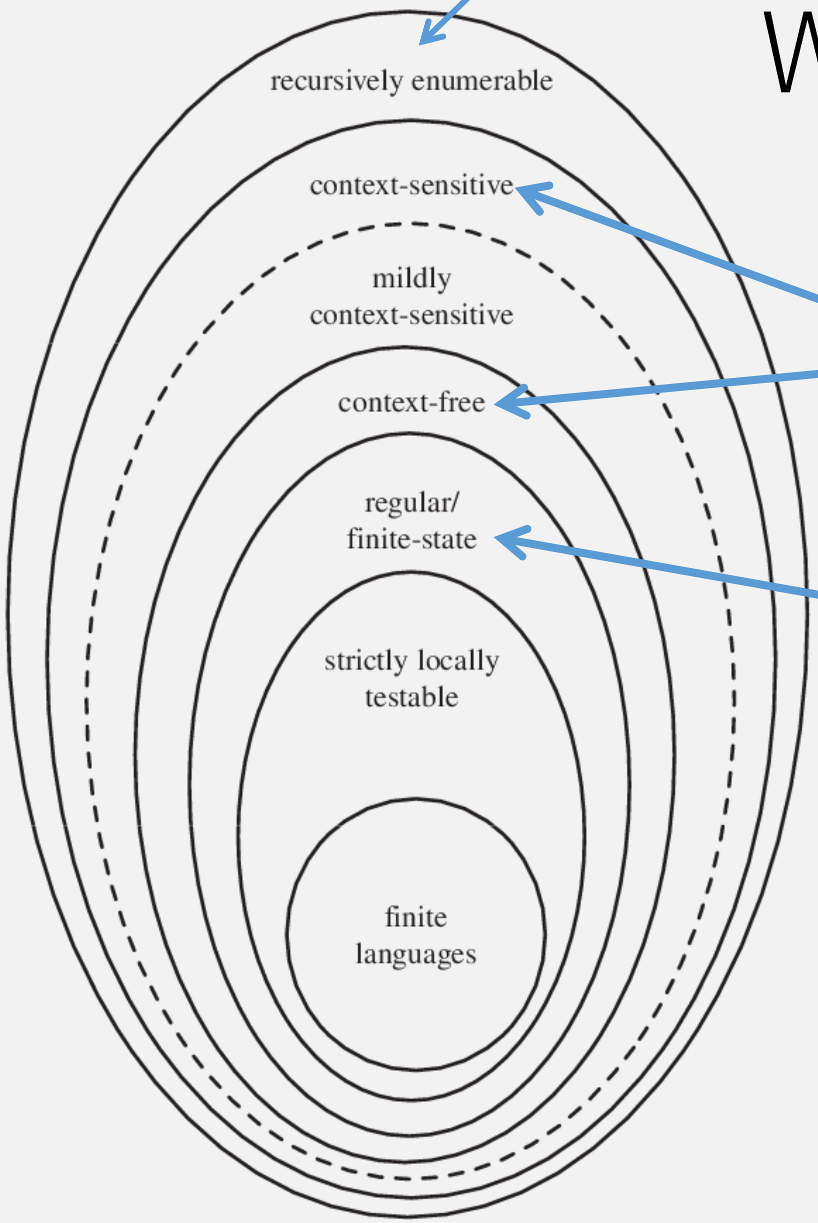
(In mathematical logic)

A **theory** consists of:

- Axioms
    - Accepted facts and definitions
- Theorems
    - I.e., additional facts derived from axioms and previous theorems
    - Using a deductive system
        - I.e., “if p then q” and “p”, then “q” (modus ponens)

Each of these language classes corresponds to a different kind of computer!

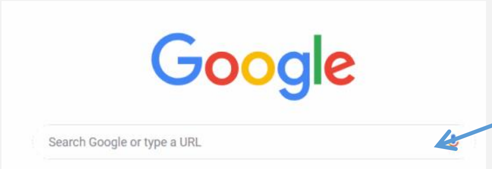
Why study languages formally?



"Chomsky" hierarchy

1. To communicate with computers!
   - We need to know what "languages" they can understand
   - E.g., Python, C++, Java programs?

- Simpler languages are often more convenient
  - E.g., text search, arithmetic



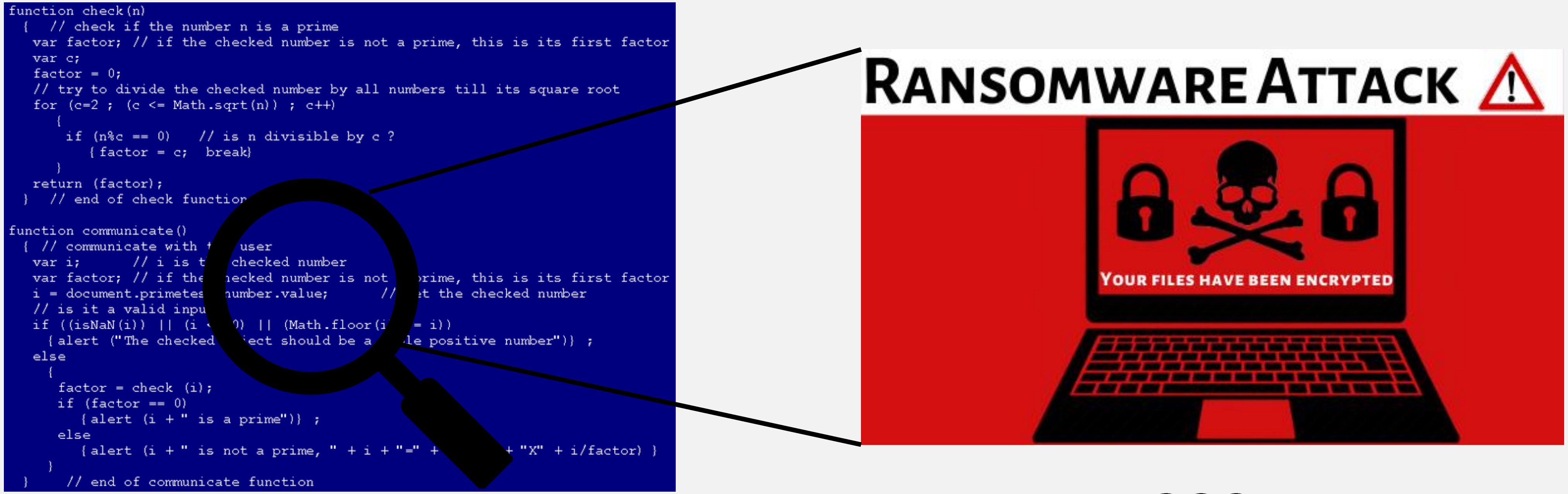
Don't want to write Python here

- Different languages require different computing power to understand/recognize

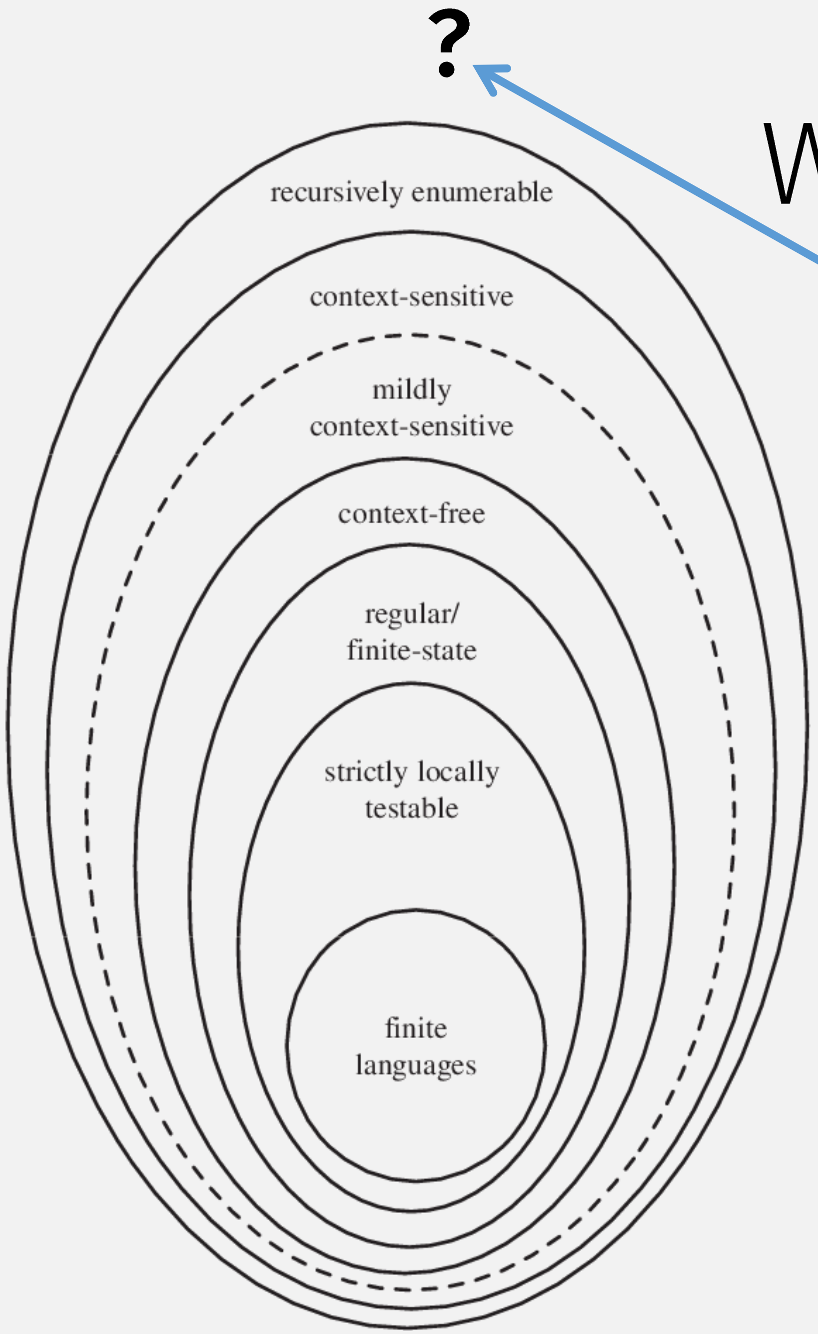
So the formal study of **languages** is also the formal study of **computation**!

Why study computers formally?

2. **To predict what programs will do**
   - (without running them!)



???



Why study computers formally?

3. To know the <u>limits</u> of computers
   - <u>I.e.</u>, what they can't do

- More practically, resource-limited computers
  - <u>I.e.</u>, what can I compute with …
    - … a certain amount of time?

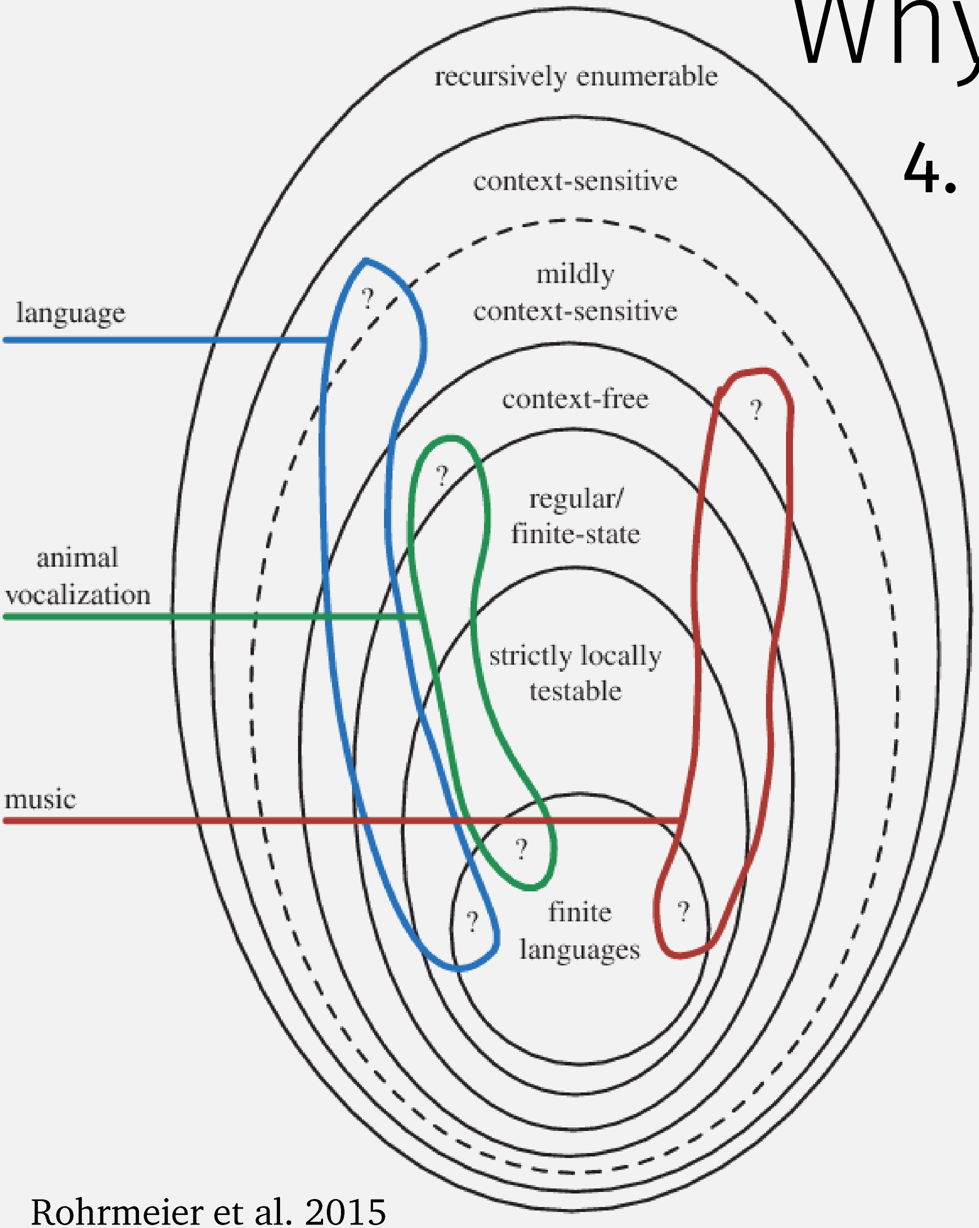
    - … a certain amount of memory space?

    - … a certain probability?

    - … a limited circuit size?

Why study languages formally?

4. Many, many <u>practical</u> applications
   - E.g., Can we formally model …
     - … human spoken language?
     - … animal communication?
     - … music?



Rohrmeier et al. 2015

More Practical Applications

Writing secure software:

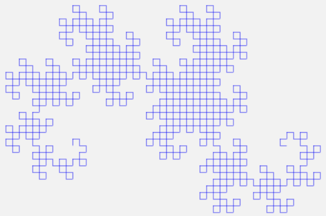
> “The **LANGSEC** (Language-Theoretic Security) community posits that <u>the only path to trustworthy software</u> …
>
> … is treating all valid inputs as a **formal language** …
>
> … where input-parsing is handled by **automata** with the required computation power.
>
> ----- `langsec.org`

Applications of Formal Langs: Beyond CS

- Lindenmeyer grammars model plant growth

variables : X F
constants : + − [ ]
start : X
rules : (X → F+[[X]-X]-F[-FX]+X), (F → FF)
angle : 25°

- Many fractal patterns in nature are CFGs

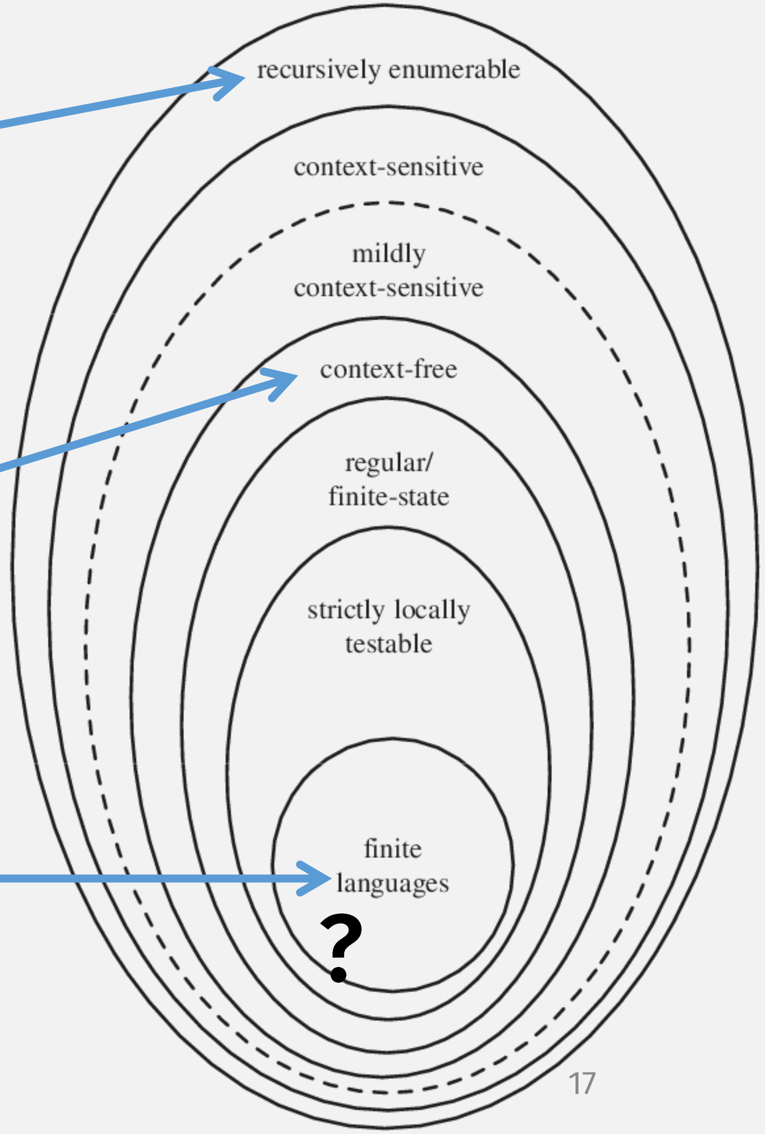


- DNA has its own formal language

Bulletin of Mathematical Biology
Volume 49, Issue 6, 1987, Pages 737-759

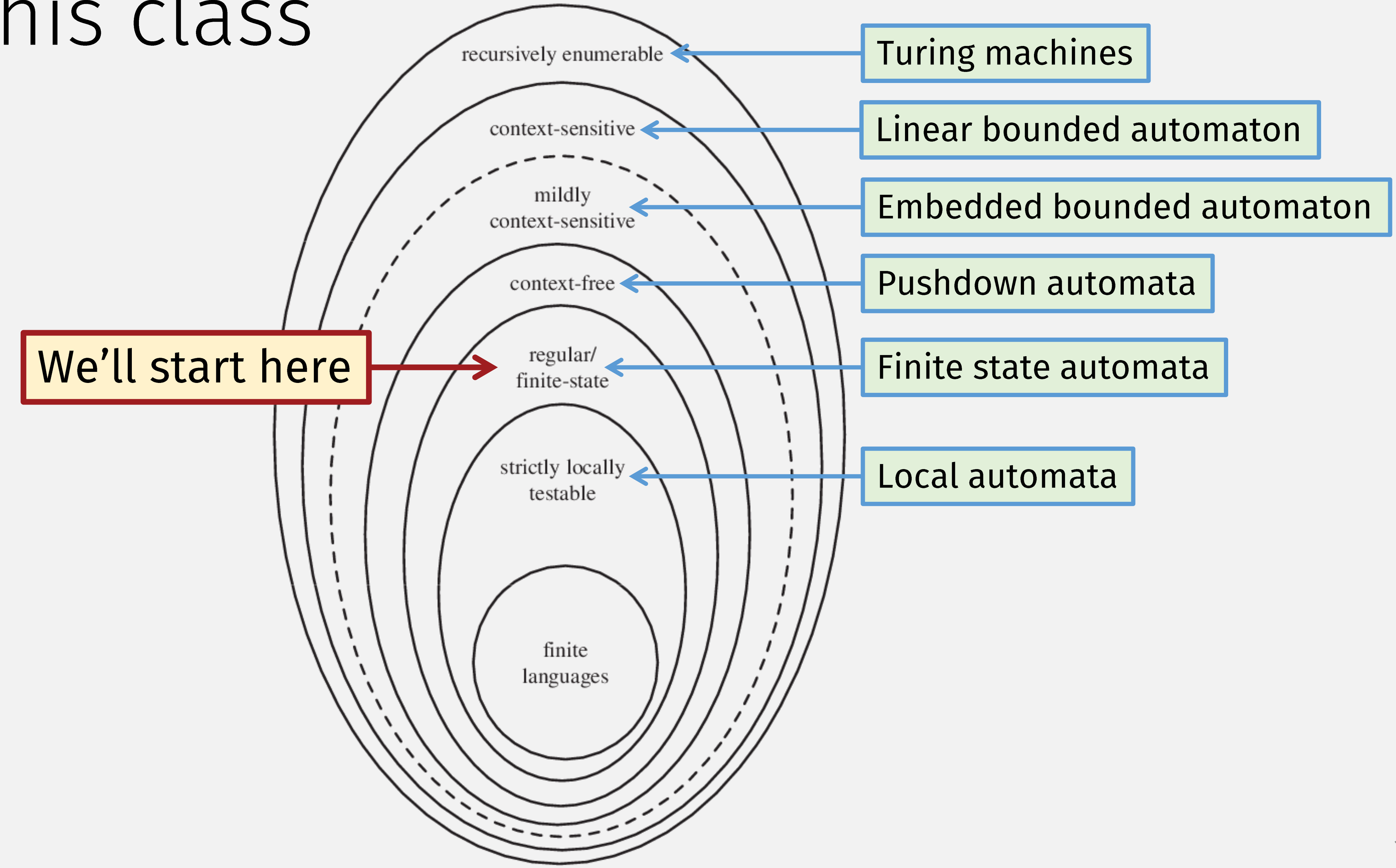
Formal language theory and DNA: An analysis of the generative capacity of specific recombinant behaviors

Tom Head



In this class

Languages **Computation Models**



CS420 vs CS622

CS622

- Deeper topics, faster paced
- More proofs (so brush up on CS220/CS320)

Course Logistics

Course website:

https://www.cs.umb.edu/~stchang/cs622/f21/

Quiz 9/8

See gradescope.com