CSCI 361 Lecture 22: Wrap Up and Evals

Shikha Singh

Announcements & Logistics

- Grab feedback on **1-page paper drafts**
- **Survey paper deadlines:**
  - 10 min presentation + Q&A on Dec 5
  - Final paper due Dec 6 on Gradescope
- End early today to allow time for student evaluations
- **Self-scheduled final exam** between **Dec 7-15**
  - 2.5 hr exam
  - **Practice exam** will release soon
- Office hours **during reading period (Mon/Tues)**: **2-3.30 pm**

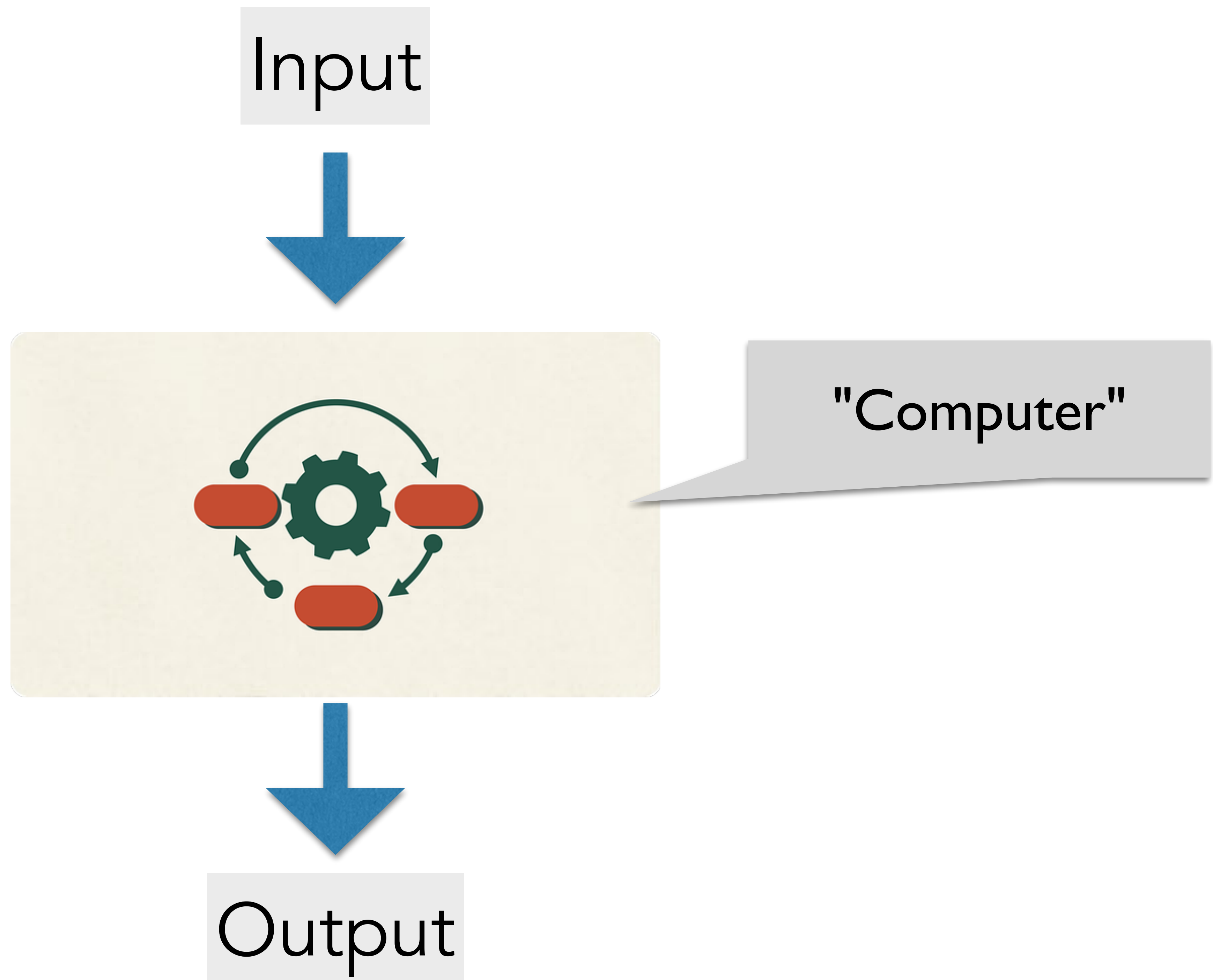
Last Time

- Wrapped up complexity theory

Today

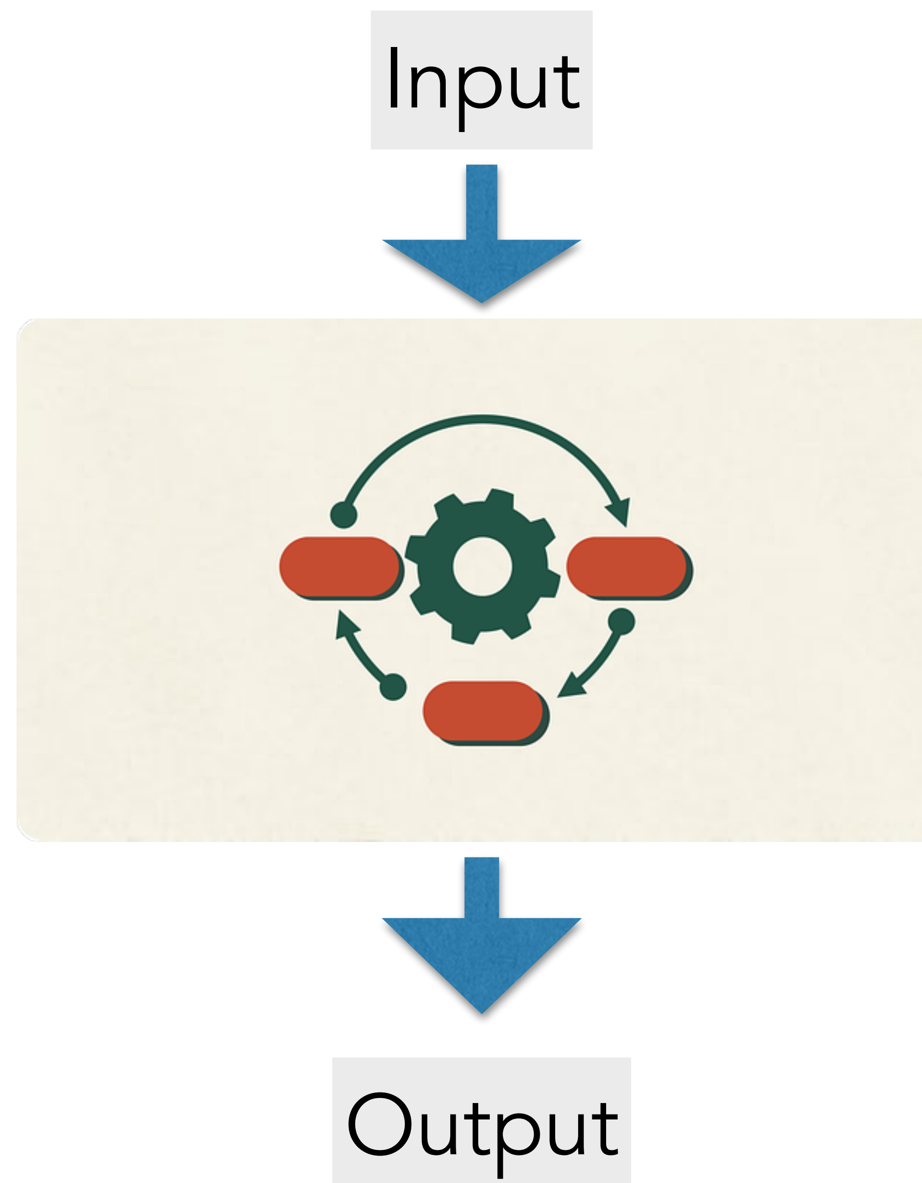
- Wrap up and review course
- Course evals

What is Computation



Defining Computation

- **Computation**: manipulation of information/data to solve a problem
- **Computational problem**: the input/output pairs
- **Algorithm**: description of how this data is manipulated



Theory of Computation

- Need a formal model of what it means to solve a problem
- Theory of Computation:
    - Building a mathematical model for computation
    - Using the model to understand the power and limits of computation
    - Gain insights that inform applications

Topics and Theme

What are the fundamental capabilities and limitations of computers?

- **Automata Theory**
  - Finite automaton and Regular Languages
  - PDAs and CFLs
- **Computability Theory**:
  - Model a modern computer as a Turing machine
  - Identify what problems can and cannot be solved by it
- **Complexity Theory**:
  - Classify problems based on efficiency of solving it

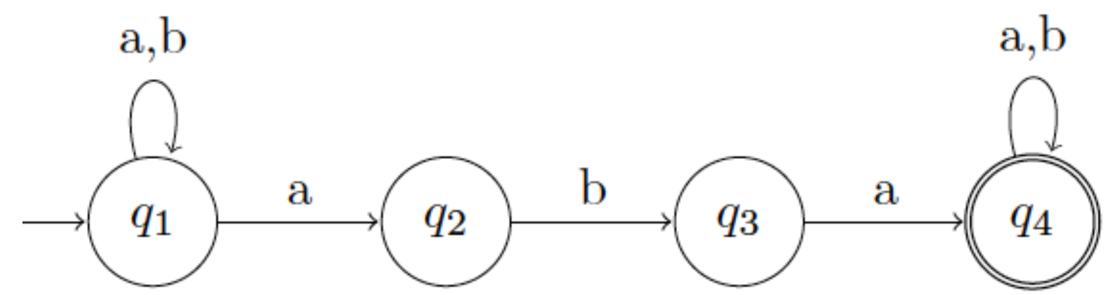
Regular Languages and DFA

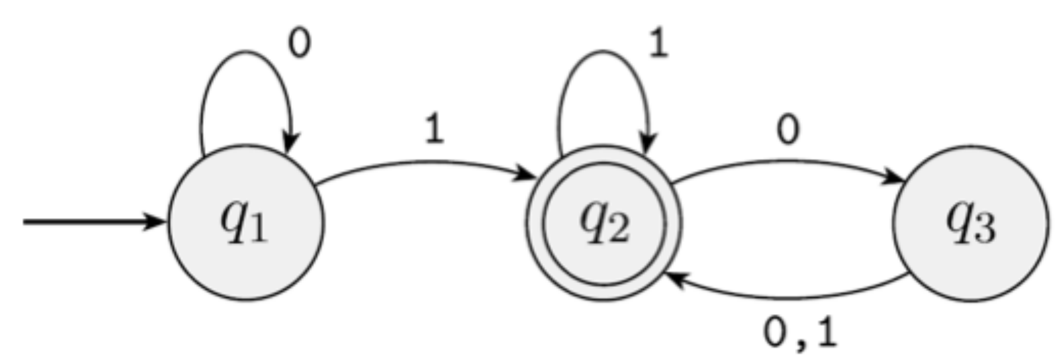
- **Question.** What did we learn about finite automaton and regular languages?

Regular Languages and DFA

- Equivalence of models:
    - DFA ⇔ NFA ⇔ Regular Expressions
- Closure properties
    - Closed under intersection, union, complement, concatenation, star, set difference, reverse, etc.
- Minimal DFAs and Equivalence classes
- How to prove a language is not regular:
    - Pumping lemma
    - Myhill Nerode

$$(ab \cup a)^*$$





Power and Limitations

- DFAs are good at simple repetitive or sequencing problems
- Do not have enough memory to count to an arbitrary number
- Examples of languages for which no DFA exists?
  - $\{0^n 1^n \mid n \geq 0\}$
  - $\{w \in \{0,1\}^* \mid$ number of 0s same as number of 1s$\}$
  - $\{ww \mid w \in \{0,1\}^*\}$
  - $\{ww^R \mid w \in \{0,1\}^*\}$
- To add more computation power, let's add some memory: a stack

Context-Free Languages

- **Question.** What did we learn about PDAs and context-free languages?

Context-Free Languages

- Context-free grammars for generating CFLs
- Push-down automaton for recognizing CFLs
- Equivalence: (Non-deterministic) PDA $\Longleftrightarrow$ CFGs
- Closure properties:
  - Closed under union, Kleene star, reverse, concatenation
  - Not closed under intersection, complement
  - Intersection of a CFL and regular language is context-free
- How to prove a language is not context-free:
  - Pumping lemma: uses the fact that to generate arbitrarily long strings, must reuse a variable (recursion)

Power and Limitations

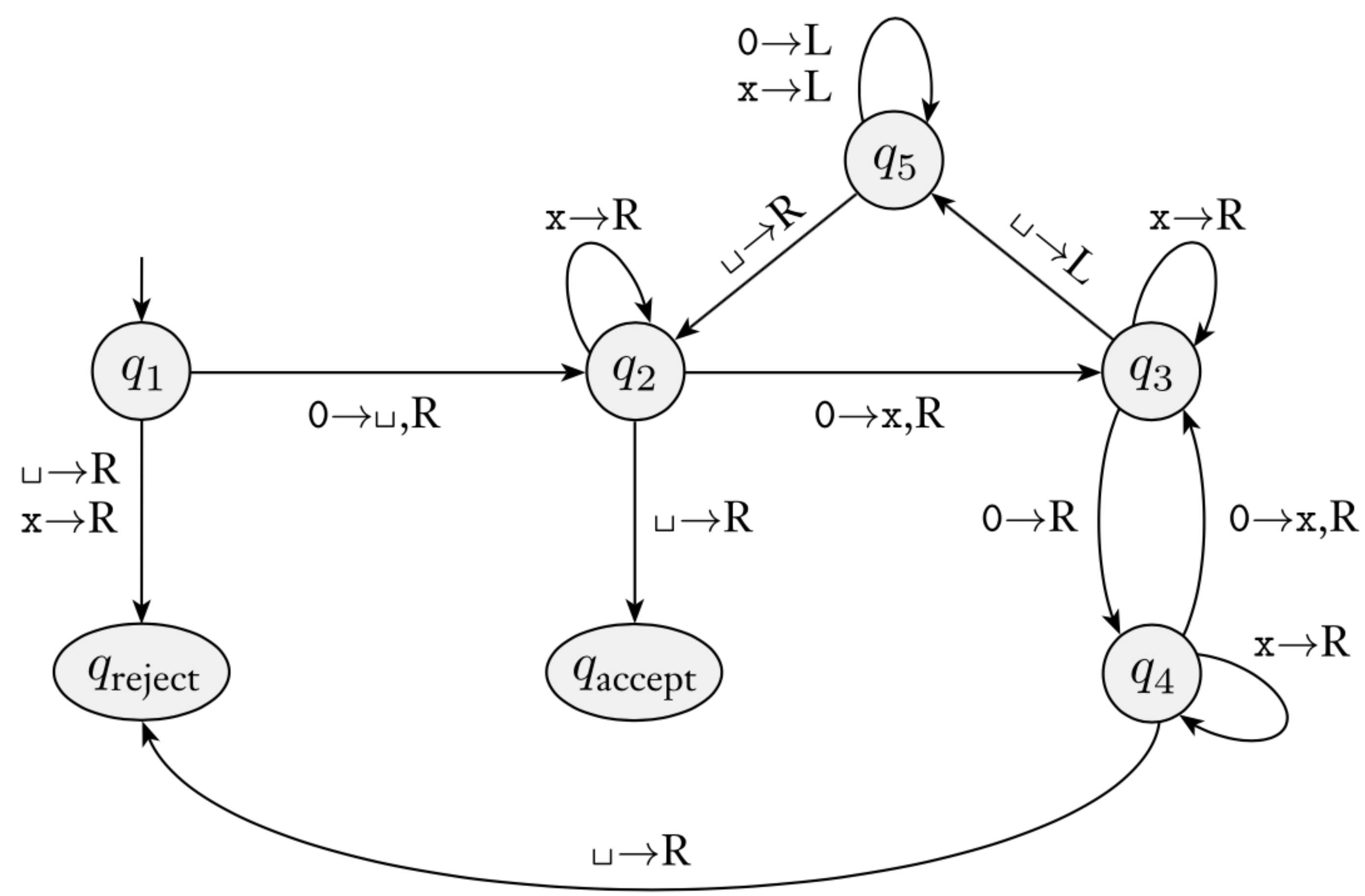
- Access to a stack (recursion), add considerable power over a DFA
- Examples of CFLs that are **not regular**?
  - $\{0^n 1^n \mid n \geq 0\}$
  - $\{w \in \{0,1\}^* \mid$ number of 0s same as number of 1s in $w\}$
  - $\{ww^R \mid w \in \{0,1\}^*\}$
  - $\{w \mid w \in \{0,1\}^*$ and $w = w^R\}$ (Palindromes)
  - $\{a^i b^j c^k \mid i, j, k \geq 0$ and $i = j$ or $j = k\}$
- Still has limitations: examples of languages that are not context-free?

Non-Context-Free Languages

- Pairing/Counting examples we have seen:
  - $\{a^n b^n c^n \mid n \geq 0\}$, $\{a^n b^n a^n\}$, $\{ww \mid w \in \{a,b\}^*\}$
  - HW: language of palindromes with equal # of 1s and 0s
  - Strings over $\{a,b,c\}$ with equal # of a's, b's and c's
  - $\{a^n b^m a^n b^m \mid n,m \geq 0\}$
  - $\{w\ a^n\ w^R\ b^n \mid w \in \{a,b\}^*, n \geq 0\}$
- Non-linear counting examples:
  - $\{a^{2^n} \mid n \geq 0\}$, $\{a^p \mid \text{p is a prime}\}$, $\{a^{n^2} \mid n \geq 0\}$
  - Intuition: structure is too rigid to be able to be "pumped"

Moving Up: Turing Machines

- A finite automaton with infinite memory
- **Question.** What did we learn about Turing machines and languages decided by TMs?



TM and TM Decidable Languages

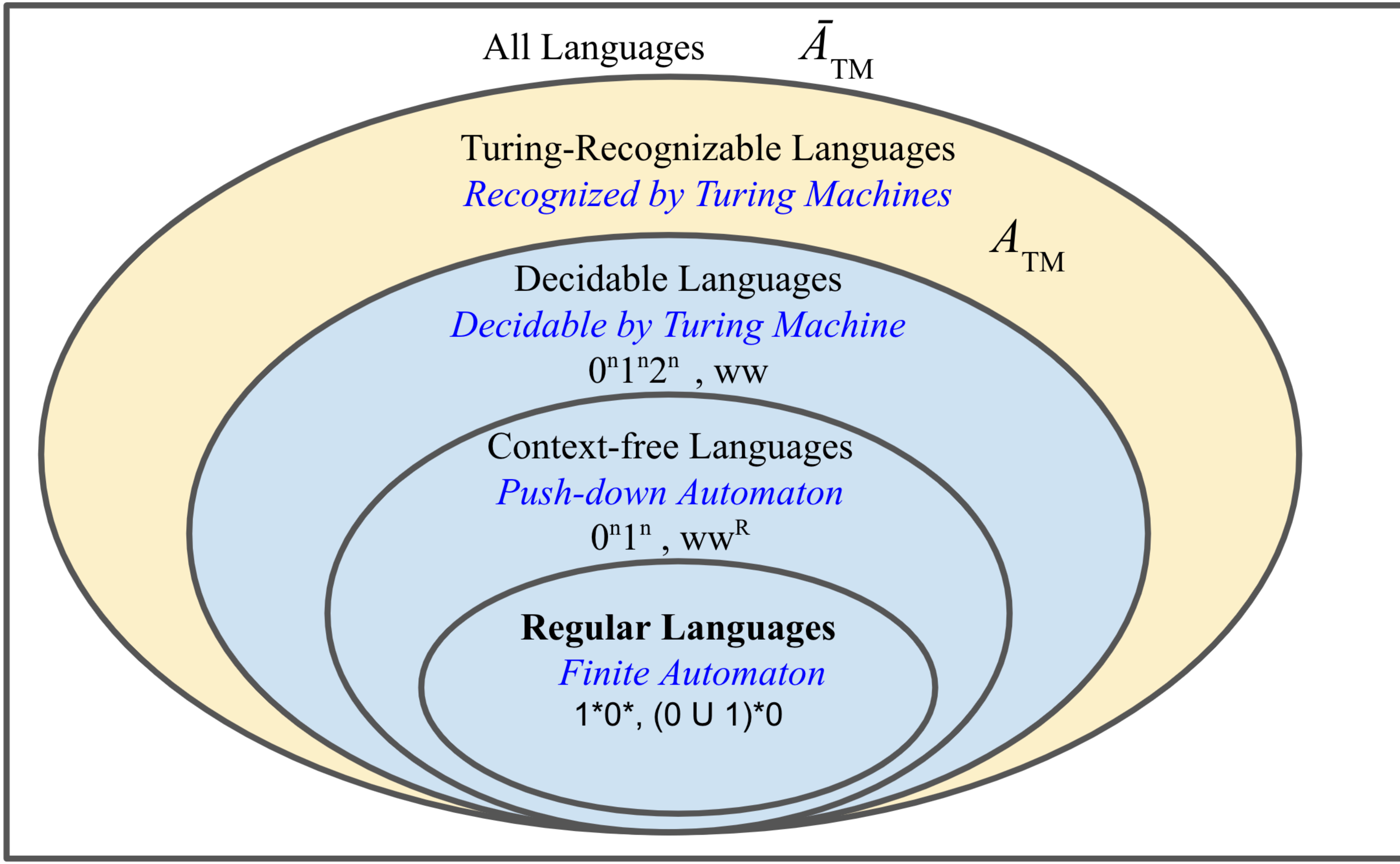
- Church-Turing Thesis
  - Anything that be computed by algorithms can be done on a TM
- Models of Turing machines:
  - Multi-tape, non-deterministic
- Properties of decidable languages:
  - Closed under union, intersection and complement
  - L is decidable iff L and $\overline{L}$ are TM recognizable
- Decidable languages about semantic properties of DFAs/CFGs?
  - A_{DFA}, A_{CFG}, E_{DFA}, E_{CFG}, EQ_{DFA}, etc.

Limits of Computation: Undecidability

- There are infinitely many decision problems that cannot be solved by any TM (counting argument)
- What do these problems look like?
  - Diagonalization to prove A_{TM} is undecidable
- Reductions to prove a bunch of other problems are undecidable
  - Halting problem, E_{TM}, $\mathsf{EQ}_{\mathsf{TM}}$, $\mathsf{REGULAR}_{\mathsf{TM}}$
  - Rice's theorem says any non-trivial property of language of TM is undecidable
- Introduced mapping reductions (useful for TM recognizability)

Undecidability and Unrecognizability

- CFG related problems that are undecidable?
    - EQ_{CFG}, $\cap_{CFG}$
- How did we prove these were undecidable?
    - Using PCP
    - How did we prove PCP is undecidable?
        - Using computation-history method
- How did we find Turing unrecognizable problems?
    - If an undecidable language is Turing recognizable, its complement must be **not Turing recognizable**
- Is there a language that is neither Turing recognizable not TM co-recognizable?


All Languages
$\bar{A}_{TM}$
Turing-Recognizable Languages
Recognized by Turing Machines
A_{TM}
Decidable Languages
Decidable by Turing Machine
$0^n1^n2^n$, ww
Context-free Languages
Push-down Automaton
0^n1^n , ww^R
Regular Languages
Finite Automaton
1*0*, (0 U 1)*0

Complexity Theory

- **Question.** What did we learn about the different complexity classes?

Complexity Theory

- Time complexity classes: **P**, **NP**, **NP**-complete, **NP**-hard, **EXPTIME**
- **(Cook-Levin Theorem)** SAT is NP complete.
- Implications of P vs NP
- Reductions to prove other problems are NP complete:
  - Vertex Cover
  - Clique
  - 3Color
  - Hamiltonian Cycle, etc
- Final picture: $\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} = \mathrm{NPSPACE} \subseteq \mathrm{EXPTIME}.$

> We know **P ≠ EXPTIME**, so one of these containments is proper but we don't know which one

Takeaways

- Computational laws:
  - The simpler the model, the easier it is to verify its properties
  - The more power we add, the less we can verify
  - Computers can't do everything: many problems are not solvable
  - Many problems are not solvable in a reasonable amount of time
- TCS is a young field: there is a lot we still don't know

Thank you!

- You all should be proud of how much you've learned
- Good luck on the presentations & final exam and have great winter break!

Course Evaluations

Course Evals Logistics

- Two parts: **(1) SCS form** , **(2) Blue sheets** (both on GLOW)
- Your responses are **confidential** and we will only receive a report of your anonymized comments after we have submitted all grades for this course
- **SCS forms** are used for tenure/promotion & seen by CAP etc, **blue sheets are open-ended** comments directed only to your instructor

*To access the online evaluations, log into **Glow** (glow.williams.edu) using your regular Williams username and password (the same ones you use for your Williams email account). On your Glow dashboard you'll see a course called "**Course Evaluations**." Click on this and then follow the instructions you see on the screen. If you have trouble finding the evaluation, you can ask a neighbor for help or reach out to ir@williams.edu.*