- Quick review
- Enumerators (Section 3.2)
- Church-Turing thesis (Section 3.3)
- High-level Turing machine descriptions (Section 3.3)
- Decidable languages (Section 4.1)

Some images from slides by Michael Sipser
Announcements

- Homework 1 grades will be released today – Look for email from Gradescope
  - Solutions will be posted in Submitty Course Materials
  - Please submit regrade requests within 1 week on Gradescope. I’m also happy to discuss in office hours.
  - Median: 93, Mean: 89.18, Std Dev: 13:08
- Homework 2
  - Due Saturday, Feb 20th by 11:59 pm (in Gradescope)
Announcements

- Exam 1 – Thursday, Feb 25th
  - Will be given through Gradescope.
  - Available 12pm – 8pm NY Time. You will have 2 hours.
  - More details next lecture.

- Practice Exam available in Gradescope now (optional)
  - 10 minute timed exam.
  - So you can see how Exam 1 will work.

- Review Problems
  - Exam review problems posted in Submitty Course Materials
  - Updated after each class
  - Solutions at end of chapters
Quick Review – Languages and Turing Machines

- On processing an input string $w$, a Turing machine can:
  - Enter the accept state (and halt).
  - Enter the reject state (and halt).
  - Loop forever (not an accept or a reject).

- The **language** of a Turing machine $M$ is $L(M) = \{w \mid M$ accepts $w\}$.

- A language $C$ is **Turing-recognizable** if $L(M) = A$ for some Turing machine $M$.
  - Also called **recursively enumerable**.

- A Turing machine $M$ is a **decider** if $M$ halts on all inputs.

- A language $C$ is **Turing-decidable** if $C = L(M)$ for some Turing machine decider $M$. 
Quick Review – Describing a Turing Machine
Give an implementation-level description of a Turing machine that decides the language

\[ C = \{a^ib^jc^k \mid i \times j = k \text{ and } i,j,k \geq 1\}. \]
Give an implementation-level description of a Turing machine that decides the language

\[ C = \{a^ib^jc^k \mid i \times j = k \text{ and } i,j,k \geq 1\}. \]
Quick Review – Turing Machine Variants

- We discussed:
  - Turing machines that allow read-write head to “Stay Put”
  - Multitape (deterministic) Turing machines
  - Nondeterministic Turing machines

- Showed that all these variants have the same “power” as the standard single tape deterministic Turing machine:
  - New features are optional – every standard Turing machine is trivially equivalent to each variant.
  - For each variant, showed how to construct equivalent standard Turing machine.
    - Adding these new features does not increase power – can’t recognize/decide more languages

- Some variants remove features from the standard Turing machine model.
  - How to prove/disprove equivalence with standard model?
Another Variant - Enumerators

- An **enumerator** is a Turing machine with an attached printer.
  - Initialized with blank tape.
  - It prints strings on the tape, one at a time, possibly forever.
- The language **enumerated** by E is the set of strings that are eventually printed out.
  - Strings can be printed in any order and can be repeated.
Turing Enumerators

- **Theorem:** A language is Turing-recognizable if and only if some enumerator enumerates it.
Theorem: A language is Turing-recognizable if and only if some enumerator enumerates it.
Church - Turing Thesis (1936)

- All reasonable computing models have the same power as Turing machines.
  - Shared feature: unrestricted access to unlimited memory.
  - Reasonable = ability to perform only a finite amount of work in a single step.
  - Examples: \(\lambda\)-calculus, \(<\text{insert\_favorite\_programming\_language\_here}>\)
Hilbert’s 10th Problem

- In 1900, mathematician David Hilbert posed 23 significant math problems as a challenge to the coming century.
- **Hilbert’s 10th problem:** Devise an algorithm that tests whether a polynomial has an integral root.
  - A polynomial is a sum of terms, where each term is a product of variables and a constant (a coefficient).
    - E.g., $6x^3yz^2 + 3xy^2 - x^3 - 10$
  - A root of a polynomial is an assignment of values to its variables that makes the polynomial equal 0.
  - A root is an integral root if the variables’ values are all integers.

- In 1970, Yuri Matijasevic proved that no such algorithm exists.
  - Using notions laid out in the Church-Turing thesis.
Hilbert’s 10th Problem
Ways to describe a Turing machine:
- Formal description, implementation-level description, high-level description
- High-level description – use prose to describe an algorithm

Input to Turing machine is always a string
- Need to specify how to encode an object \( O \) as a string, denoted \( \langle O \rangle \).
- Can also encode several objects into single input string, denoted \( \langle O_1, O_2, \ldots, O_k \rangle \).
- Up to you to decide on encoding.
Describe a Turing Machine $M$ that decides the language $A=\{\langle G \rangle \mid G$ is a connected undirected graph\}. 
Decidability (Chapter 4)
Decision problems are problems that have a YES or NO answer.

Equivalent formulation:
- Let $A$ be a language such that $w \in A$ iff the answer to the decision problem with input $w$ is YES.

A problem is **decidable** if there is a Turing machine that decides $A$.
- There is an algorithm that decides $A$.
- This problem can be solved.

A decision problem (language) is **undecidable** if no Turing machine decides it.
- This problem cannot be solved.
Acceptance Problem for DFAs

- Does a given DFA $B$ accept an input string $w$? $A = \{ (B,w) \mid B \text{ is a DFA that accepts input string } w \}$

- **Theorem:** $A$ is a decidable language.