Intro to Theory of Computation

LECTURE 17

CS 464

Last time

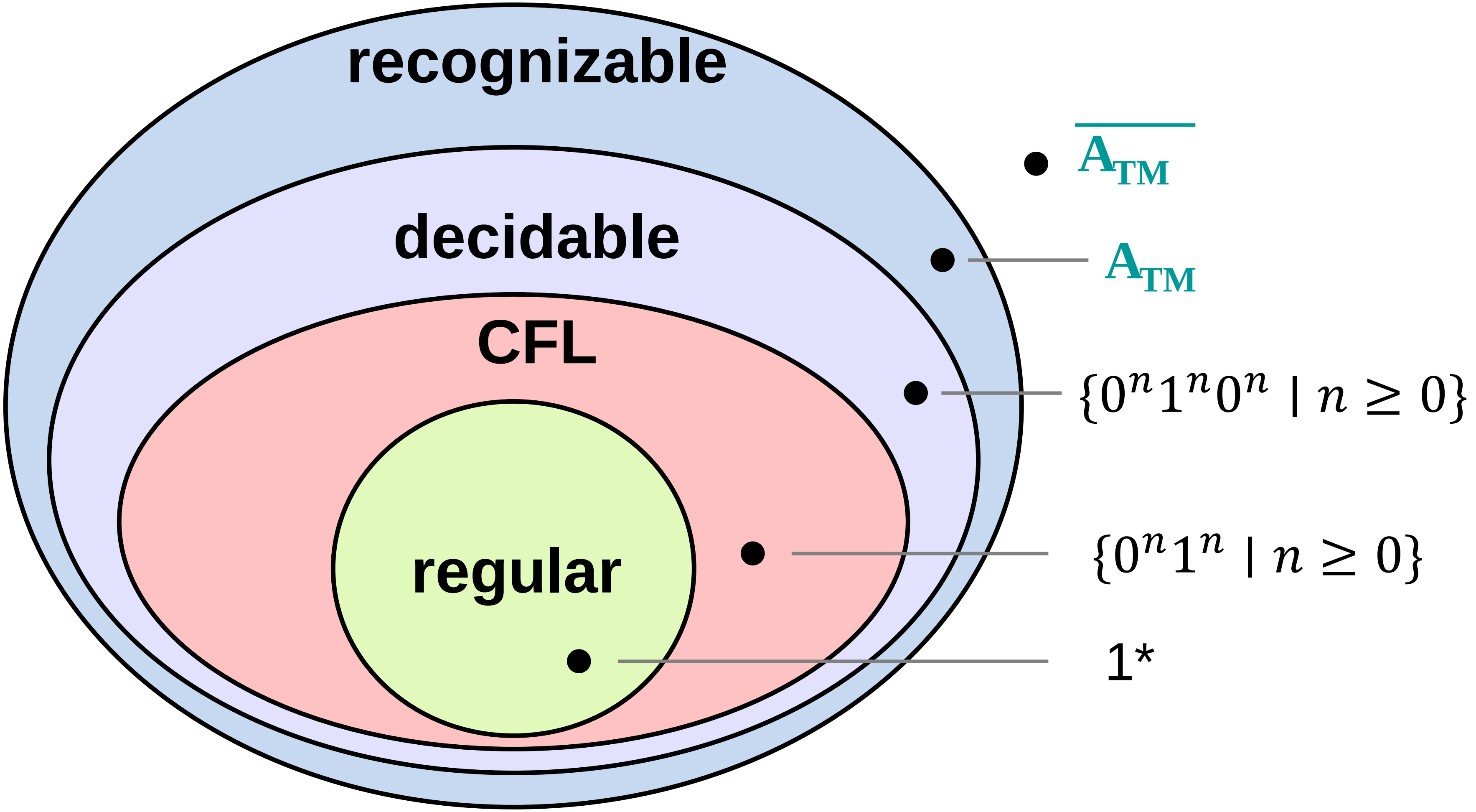
- Countable/uncountable sets.
- Diagonalization
- Undecidable/unrecognizable languages (A_{TM} is undecidable)

Today

- Reductions

Sofya Raskhodnikova

Classes of languages



A_{TM} = { ⟨M,w⟩ | M is a TM, w is a string, and M accepts w }

CS 464

I-clicker question (frequency: AC)

The fact that A_{TM} is undecidable means that

A. if we are given input <M,w>, then M is not a decider

B. there is no TM S such that L(S)= A_{TM}

C. there is no TM S that accepts on strings in A_{TM} and halts and rejects on strings on strings not in A_{TM} .

D. Both B and C are correct.

E. None of the above.

Theorem. Language L is decidable iff L and $\overline{L}$ are Turing-recognizable

Proof:

1) L is decidable ⇒ L and $\overline{L}$ are Turing-recognizable.

(on the board)

2) L and $\overline{L}$ are Turing-recognizable ⇒ L is decidable.

(on the board)

Prove that the following languages are Turing-recognizable

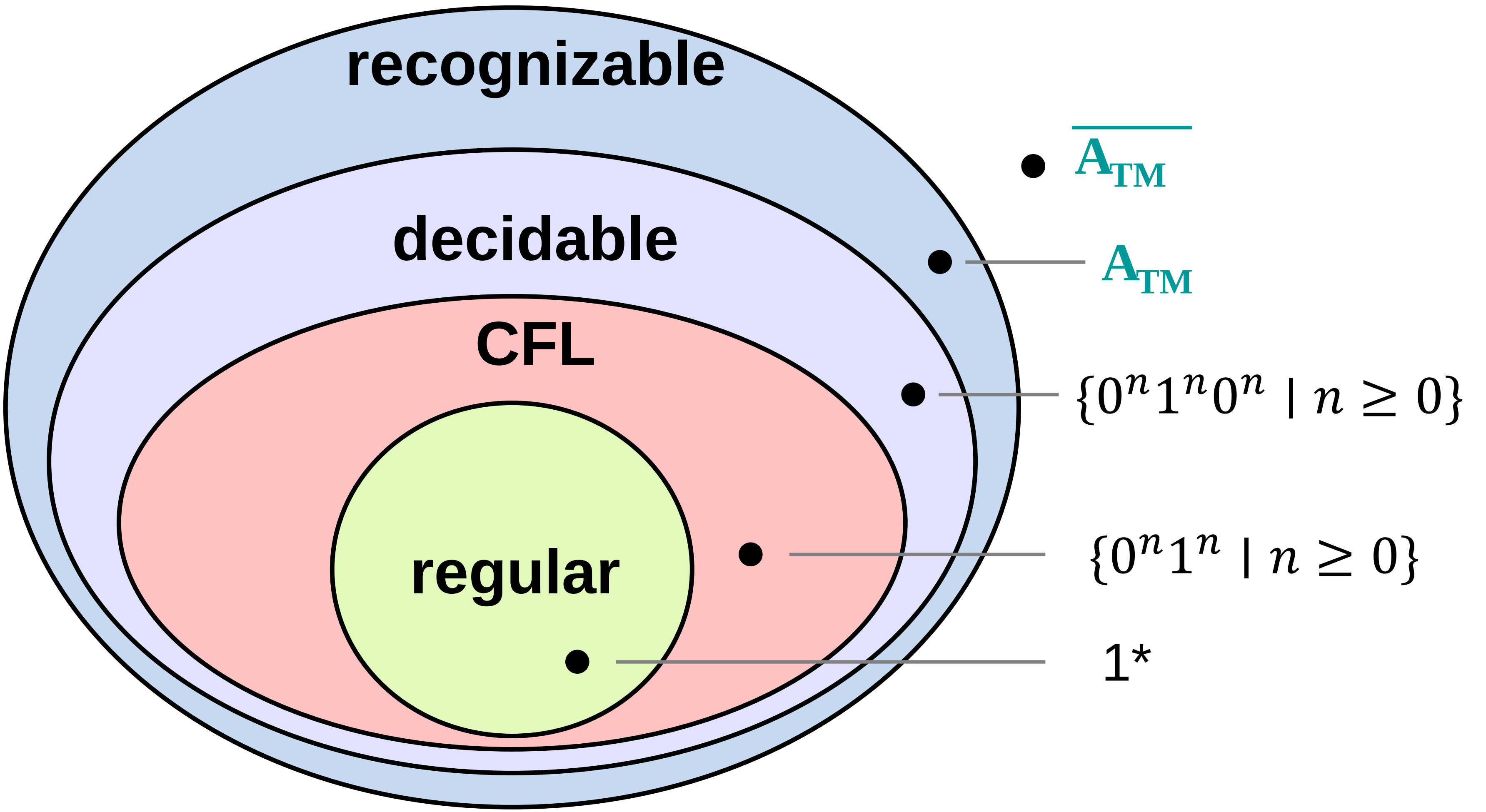
$A_{TM} = \{\langle M, w\rangle \mid M \text{ is a TM that accepts string } w\}$

$HALT_{TM} = \{\langle M, w\rangle \mid M \text{ is a TM that halts on string } w\}$

Corollary. $\overline{A_{TM}}$ is not Turing-recognizable.

(on the board)

Classes of languages



A_{TM} = { ⟨M,w⟩ | M is a TM, w is a string, and M accepts w }

Problems in language theory

A_{DFA} decidable	A_{CFG} decidable	A_{TM} undecidable
E_{DFA} decidable	E_{CFG} decidable	E_{TM} ?
EQ_{DFA} decidable	EQ_{CFG} ?	EQ_{TM} ?

Now that we have an undecidable language, can we get more?

Reductions

What is a reduction?

A reduction from problem A to problem B is an algorithm for problem A that uses a subroutine for problem B.

Many of the deciders we constructed use reductions to one of these problems: A_{DFA}, E_{DFA}, EQ_{DFA}, A_{CFG}, E_{CFG}.

Old Theorem. EQ_{DFA} is decidable.

$EQ_{DFA} = \{ \langle D_1, D_2 \rangle \mid D_1, D_2$ are DFAs & $L(D_1) = L(D_2)\}$

Proof: The following TM M decides EQ_{DFA}.

M = `` On input $\langle D_1, D_2 \rangle$, where D_1, D_2 are DFAs:

1. Construct a DFA D that recognizes the set difference of $L(D_1)$ and $L(D_2)$. (on the board)
2. Run the decider for E_{DFA} on <D>.
3. If it accepts, accept. O.w. reject."

That's a reduction from EQ_{DFA} *to* E_{DFA}.

How to tell a difference between a CS student and a CE student?

1. Put an empty kettle in the middle of the kitchen floor and ask your subjects to boil some water.

- Both subjects will fill the kettle with water, turn on the stove and turn the flame on.

2. Put the kettle full of water on the stove and ask the subjects to boil the water.

- CE student will turn the flame on.
- CS student will empty the kettle and put it in the middle of the kitchen floor… thereby reducing the problem to the one that has already been solved!

Using reductions to prove undecidability

We want to prove that language L is undecidable.

Idea: Use a proof by contradiction.

1. Suppose to the contrary that L is decidable.
2. Use a decider for L as a subroutine to construct a decider for A_{TM}.
3. But A_{TM} is undecidable. Contradiction!

CS 464

Prove that $HALT_{TM}$ is undecidable

$HALT_{TM} = \{\langle M, w\rangle \mid M$ is a TM that halts on string $w\}$

Proof: Suppose to the contrary that $HALT_{TM}$ is decidable, and let R be a TM that decides it.

We construct TM S that decides A_{TM}.

S = `` On input $\langle M, w\rangle$, where M is a TM and w is a string:

1. Run TM *R* on input <M,w>.
2. If it rejects, reject.
3. If R accepts, simulate M on w.
4. If it accepts, accept. O.w. reject.”

M must halt on w, since R accepted.

That’s a reduction from A_{TM} to $HALT_{TM}$.

Prove that E_{TM} is undecidable

$E_{TM} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset\}$

Proof: Suppose to the contrary that E_{TM} is decidable, and let R be a TM that decides it.

We construct TM S that decides A_{TM}.

S = `` On input $\langle M, w\rangle$, where M is a TM and w is a string:

1. Run TM *R* on input ???

CS 464

Prove that E_{TM} is undecidable

$E_{TM} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset\}$

Proof: Suppose to the contrary that E_{TM} is decidable, and let R be a TM that decides it.

We construct TM S that decides A_{TM}.

S = `` On input $\langle M, w\rangle$, where M is a TM and *w* is a string:

1. Construct TM M'.

   M' = `` On input *x*,
   1. Ignore the input.
   2. Run TM *M* on input w.
   3. If it accepts, accept.”

2. Run TM *R* on input $<M'>$.
3. If it rejects, accept. O.w. reject.”