Intro to Theory of Computation

CS 332

LECTURE 16

Last time

- $\overline{A_{TM}}$ is unrecognizable
- Reductions

Today

- Reductions
- Mapping reductions

Sofya Raskhodnikova

CS 332

Problems in language theory

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

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 332

Exercise

To prove that E_{TM} is undecidable

- **A.** **we assumed E_{TM} had a decider and used it to construct a decider for A_{TM}**
- **B.** **we assumed A_{TM} had a decider and used it to construct a decider for E_{TM}**
- **C.** **we constructed a TM *S* that on input <*M*, *w*> decides whether *M* accepts *w*, assuming the existence of a TM *R* that decides on input <*M′*> whether the language of <*M′*> is empty**
- **D.** **There is more than one correct answer.**
- **E.** **None of the above.**

CS 332

Prove that EQ_{TM} is undecidable

$EQ_{TM} = \{\langle M_1, M_2 \rangle \mid M_1, M_2 \text{ are TMs}, L(M_1) = L(M_2)\}$

Proof: Suppose to the contrary that EQ_{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 TMs M', M''.

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

M'' = ``Accept."

2. Run TM *R* on input <M', M''>.
3. If $\boxed{\text{it accepts,}}$ accept. O.w. reject."

CS 332

Proof 2 that EQ_{TM} is undecidable

$EQ_{TM} = \{\langle M_1, M_2 \rangle \mid M_1, M_2 \text{ are TMs}, L(M_1) = L(M_2)\}$

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

We construct TM S that decides E_{TM}. **What do we change?**

S = `` On input ⟨$\boxed{M}$⟩, where M is a TM ~~and w is a string~~:

1. Construct $\boxed{\text{TM } M'}$.

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

   M'' = ``Accept.”

2. Run TM R on input $\boxed{<M, M'>}$
3. If $\boxed{\text{it accepts,}}$ accept. O.w. reject.”

Problems in language theory

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

Mapping Reductions

Computable functions

A function $f: \Sigma^* \to \Sigma^*$ is computable if some TM M, on every input w, halts with only $f(w)$ on its tape.

Example 1: $f(\langle x, y \rangle) = x + y$.

Example 2: $f(\langle M, w \rangle) = \langle M' \rangle$, where M is a TM and w is a string, and M' is a TM that ignores its input and runs M on w.

Mapping reductions

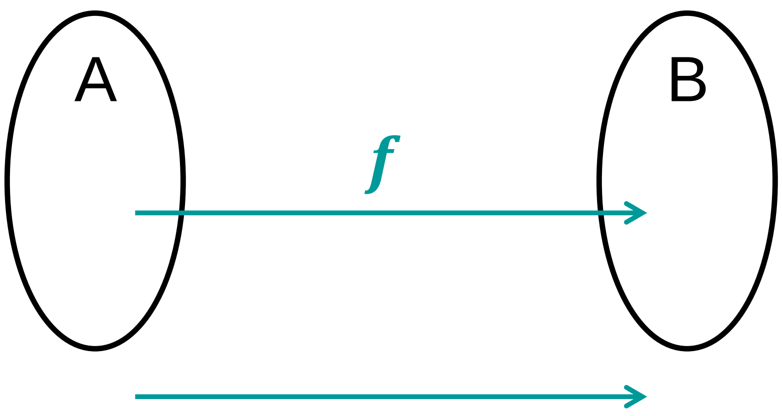
Given languages A and B,

$$\mathbf{A \leq_m B}$$

if there is a computable function f,

such that for all strings w,

$$w \in A \text{ iff } f(w) \in B.$$



CS 332

Exercise

If $\overline{\mathrm{A}} \leq_m \overline{B}$, we can conclude that

A. $A \leq_m B$

B. $B \leq_m \mathrm{A}$

C. $\overline{\mathrm{A}} \leq_m B$

D. $\overline{\mathrm{B}} \leq_m A$

E. **None of the above.**

Mapping reductions: decidability

Theorem. If $A \leq_m B$ and B is decidable, then A is decidable.

Proof: Let M be a decider for B and f be a mapping reduction from A to B.
Construct a decider for A:
``On input w:
1. Compute f(w).
2. Run M on f(w).
3. If it accepts, accept. O.w. reject.”

CS 332

Using mapping reductions to prove undecidability

Theorem. If $A \le_m B$ and B is decidable, then A is decidable.

Corollary. If $A \le_m B$ and A is undecidable, then B is undecidable.

Example: If $A_{TM} \le_m B$, then B is undecidable.

Mapping reductions: recognizability

Theorem. If $A \leq_m B$ and B is Turing-recognizable, then A is Turing-recognizable.

Proof: Let M be a TM that recognizes B and f be a mapping reduction from A to B.
Construct a TM that recognizes A:
``On input w:

1. Compute f(w).
2. Run M on f(w).
3. If it accepts, accept. O.w. reject."

Using mapping reductions to prove **un**recognizability

Theorem. If $A \le_m B$ and B is Turing-recognizable, then A is Turing-recognizable.

Corollary. If $A \le_m B$ and A is **un**recognizable, then B is **un**recognizable.

Example: If $\overline{A_{TM}} \le_m B$, then B is **un**recognizable.

Old proof that EQ_{TM} is undecidable

$EQ_{TM} = \{\langle M_1, M_2 \rangle \mid M_1, M_2$ are TMs, $L(M_1) = L(M_2)\}$

Proof: Suppose to the contrary that EQ_{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 TMs M', M''.

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

   M'' = ``Accept."

2. Run TM *R* on input <M', M''>.
3. If $\boxed{\text{it accepts,}}$ accept. O.w. reject."

CS 332

$A_{TM} \leq_m EQ_{TM}$

Proof: The following TM computes the reduction:

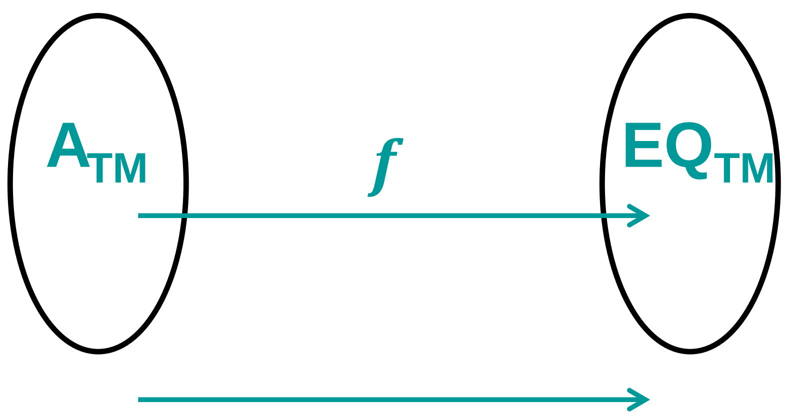
F = `` On input $\langle M, w \rangle$, where M is a TM and w is a string:

1. Construct TMs M', M''.

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

   M'' = ``Accept."

2. Output $\langle M', M'' \rangle$."



Conclusions from $A_{TM} \leq_m EQ_{TM}$

1. **Since A_{TM} is undecidable, so is EQ_{TM}**

2. $\overline{A_{TM}} \leq_m \overline{EQ_{TM}}$

   Since $\overline{A_{TM}}$ is unrecognizable, so is $\overline{EQ_{TM}}$

CS 332

Prove that EQ_{TM} is unrecognizable

Proof: We give a mapping reduction $\overline{A_{TM}} \leq_m EQ_{TM}$

The following TM computes the reduction:

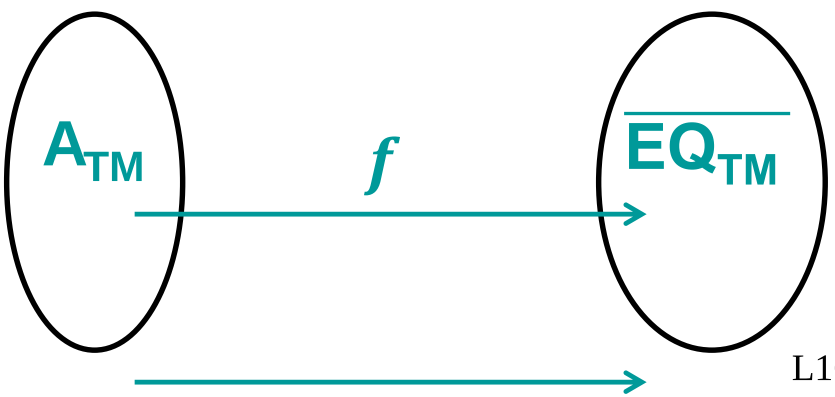
F = `` On input $\langle M, w \rangle$, where M is a TM and w is a string:

1. Construct TMs M', M''.

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

M'' = ``Reject.''

2. Output <M', M''>.''



Problems in language theory

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