BU CS 332 – Theory of Computation

Lecture 18:

- More Mapping Reductions
- Computation History Method

Reading:

Sipser Ch 5.3, 5.1

Mark Bun

March 24, 2021

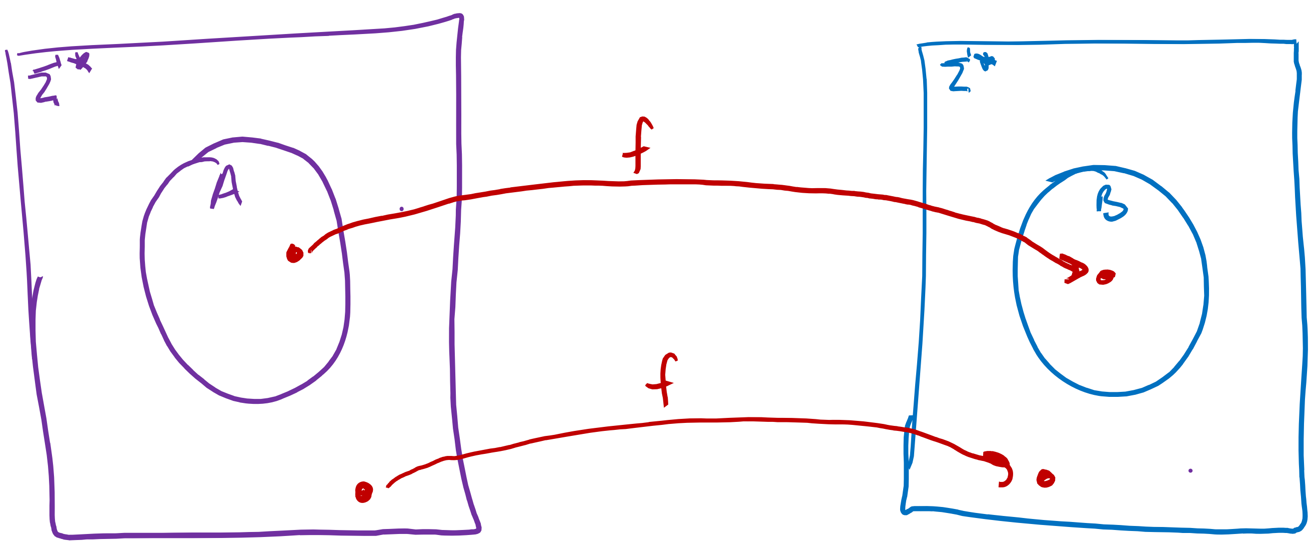
Mapping Reductions

Definition:

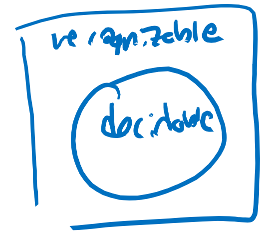
Language A is mapping reducible to language B, written

$$A \leq_{\mathrm{m}} B$$

if there is a computable function $f : \Sigma^* \rightarrow \Sigma^*$ such that for all strings $w \in \Sigma^*$, we have $w \in A \Leftrightarrow f(w) \in B$



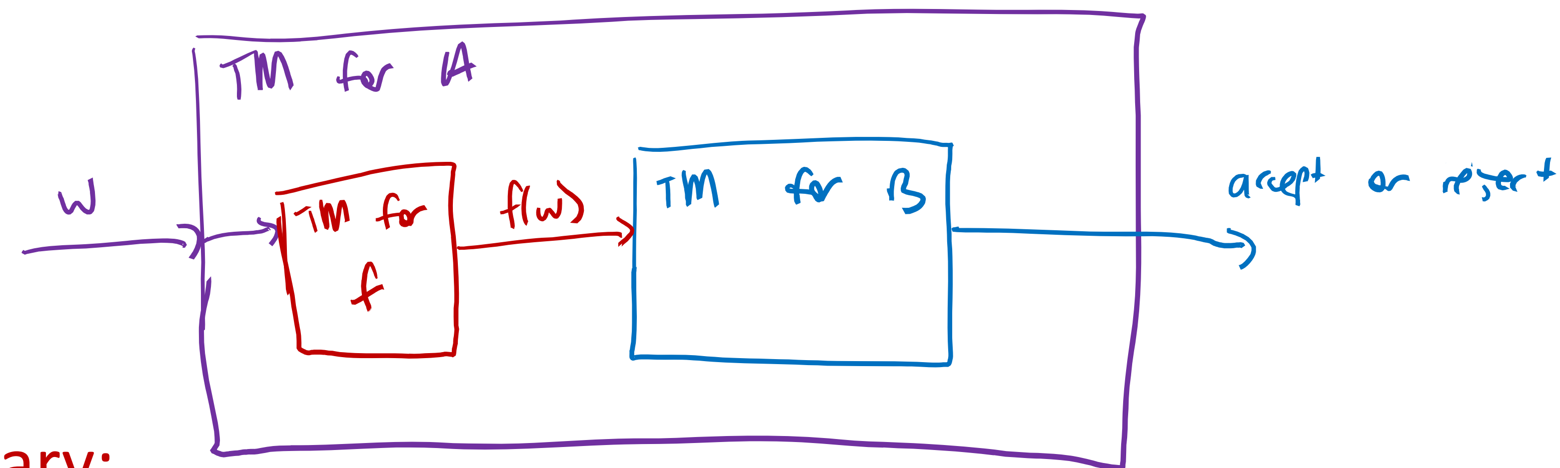
Mapping Reductions: Implications



Theorem:

"respectively"

If $A \leq_{\mathrm{m}} B$ and B is decidable (resp. recognizable), then A is also decidable (resp. recognizable)



Corollary:

If $A \leq_{\mathrm{m}} B$ and A is undecidable (resp. unrecognizable), then B is also undecidable (resp. unrecognizable)

Example: Another reduction to EQ_{TM}

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

Theorem: $A_{\mathrm{TM}} \leq_{\mathrm{m}} EQ_{\mathrm{TM}}$

Proof: The following TM N computes the reduction f:

What should the inputs and outputs to f be?

a) f should take as input a pair $\langle M_1, M_2\rangle$ and output a pair $\langle M, w\rangle$
b) f should take as input a pair $\langle M, w\rangle$ and output a pair $\langle M_1, M_2\rangle$
c) f should take as input a pair $\langle M_1, M_2\rangle$ and either accept or reject
d) f should take as input a pair $\langle M, w\rangle$ and either accept or reject

input $\in A_{\mathrm{TM}} \iff f(\text{input}) \in EQ_{\mathrm{TM}}$

$\langle M, w\rangle \in A_{\mathrm{TM}} \iff \langle M_1, M_2\rangle = f(\langle M, w\rangle) \in EQ_{\mathrm{TM}}$

Example: Another reduction to EQ_{TM}

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

Theorem: $A_{\text{TM}} \leq_{\text{m}} EQ_{\text{TM}}$

Proof: The following TM N computes the reduction f:

Want: $\langle M, w\rangle \in A_{TM} \Leftrightarrow \langle M_1, M_2\rangle \in EQ_{TM}$

On input $\langle M, w\rangle$: (instance of A_{TM})

1. Construct TMs M_1, M_2 as follows:

M_1 = "On input x,
1. Ignore x
2. Run M on w. Return its output"

M_2 = "On input x, Accept."

$L(M_2) = \Sigma^*$

$L(M_1) = \begin{cases} \Sigma^* & \text{if } M \text{ accepts } w \\ \emptyset & \text{if } M \text{ does not accept } w \end{cases}$

$\Downarrow$

$L(M_1) = L(M_2)$ iff M accepts input w

$\Downarrow$

$\langle M, w\rangle \in A_{TM} \Leftrightarrow f(\langle M, w\rangle) = \langle M_1, M_2\rangle \in EQ_{TM}$

2. Output $\langle M_1, M_2\rangle$ (instance of EQ_{TM})

Consequences of $A_{\mathrm{TM}} \leq_{\mathrm{m}} EQ_{\mathrm{TM}}$

1. Since A_{TM} is undecidable, EQ_{TM} is also undecidable

$A \leq_m B \Rightarrow \overline{A} \leq_m \overline{B}$

2. $A_{\mathrm{TM}} \leq_{\mathrm{m}} EQ_{\mathrm{TM}}$ implies $\overline{A_{\mathrm{TM}}} \leq_{\mathrm{m}} \overline{EQ_{\mathrm{TM}}}$
   Since $\overline{A_{\mathrm{TM}}}$ is unrecognizable, $\overline{EQ_{\mathrm{TM}}}$ is unrecognizable

unrecognizability not closed under complement:

- $\overline{A_{TM}}$ is unrecog.
- $\overline{\overline{A_{TM}}}$: A_{TM} is recognizable

EQ_{TM} itself is also unrecognizable

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

Theorem: $\overline{A_{\text{TM}}} \leq_{\text{m}} EQ_{\text{TM}}$ hence EQ_{TM} is unrecognizable

Proof: The following TM computes the reduction:

$L(M_1) = \begin{cases} \Sigma^* & \text{if } M \text{ accepts } w \\ \phi & \text{if } M \text{ does not acc. } w \end{cases}$

On input $\langle M, w\rangle$:

1. Construct TMs M_1, M_2 as follows:

$L(M_2) = \phi$

$\Rightarrow L(M_1) = L(M_2)$ iff M does not accept w

M_1 = "On input x,
1. Ignore x
2. Run M on input w
3. If M accepts, accept.
   Otherwise, reject."

M_2 = "On input x,
1. Ignore x and reject"

$\Rightarrow \langle M, w\rangle \in \overline{A_{TM}}$ iff $\langle M_1, M_2\rangle \in EQ_{TM}$

$\Rightarrow \langle M, w\rangle \in \overline{A_{TM}}$ iff $f(\langle M, w\rangle) \in EQ_{TM}$.

2. Output $\langle M_1, M_2\rangle$

Computation History Method

Problems in Language Theory

Apparent dichotomy:

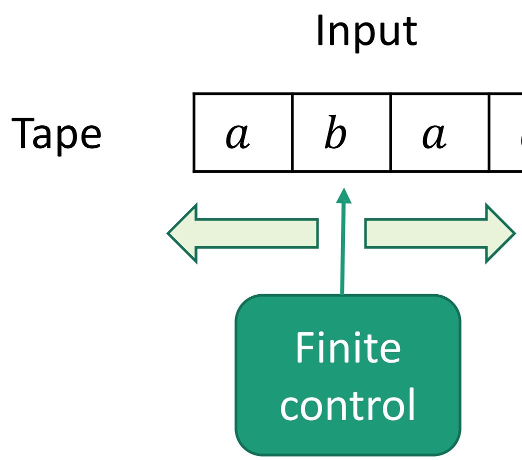
- TMs seem to be able to solve problems about the power of weaker computational models (e.g., DFAs)
- TMs can't solve problems about the power of TMs themselves

Question: Are there undecidable problems that do not involve TM descriptions?

$A_{\mathbf{DFA}}$ **decidable**	$A_{\mathbf{TM}}$ **undecidable**
$E_{\mathbf{DFA}}$ **decidable**	$E_{\mathbf{TM}}$ **undecidable**
$EQ_{\mathbf{DFA}}$ **decidable**	$EQ_{\mathbf{TM}}$ **undecidable**

Linear Bounded Automata (LBA)

A linear bounded automaton (LBA) is a TM variant with a bounded tape. The number of tape cells is the length of the input.



Intermediate in power between DFAs and TMs:

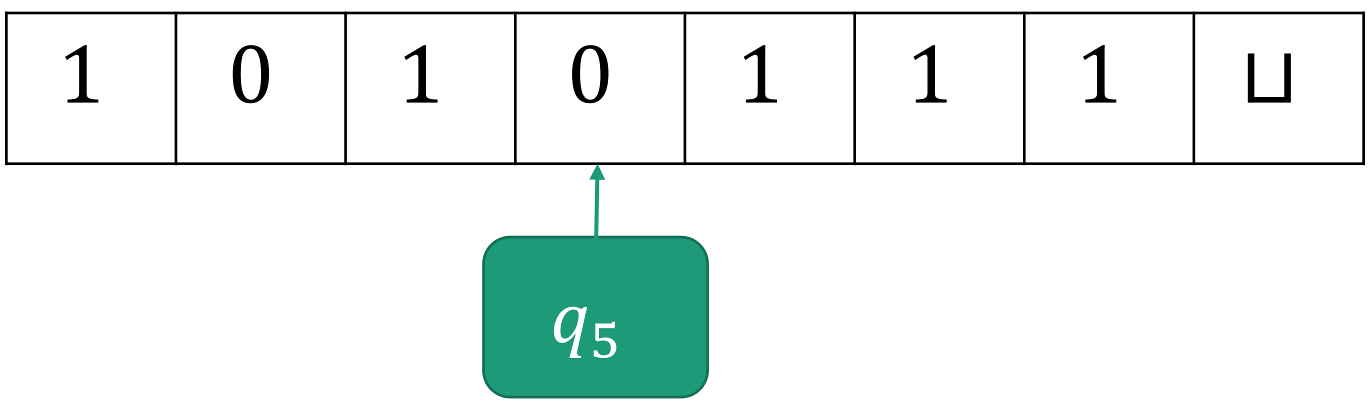
Regular langs. $\subsetneq$ $\underbrace{\mathrm{SPACE}(n)}_{\text{languages rec. by LBAs}}$ $\subsetneq$ Turing-recognizable langs.

Configurations

A configuration is a string uqv where $q \in Q$ and $u, v \in \Gamma^*$

- Tape contents = uv
- Current state = q
- Tape head on first symbol of v

Ex. $101q_5 0111 \sqcup$



Computing with Configurations

A sequence of configurations $C_0, \dots, C_\ell$ is an accepting computation history for TM (or LBA) M on input w if

1. C_0 is the start configuration $q_0 w_1 \dots w_n$
2. Every C_{i+1} legally follows from C_i
3. C_ℓ is an accepting configuration ... q_{accept} ...

Rejecting computation history: Same thing, but C_ℓ is a rejecting configuration ... q_{reject} ...

If M loops on w, there is no accepting or rejecting computation history

Counting Configurations

How many distinct configurations are possible for an LBA with k states, a symbols in its tape alphabet, and a tape of length n?

How many possible strings of the form:

$$\boxed{x_1 x_2 \dots x_i\, q\, x_{i+1} \dots x_n}$$

are there where:

- $x_1, \dots, x_n \in \Gamma$ $\quad (|\Gamma| = a)$
- $q \in Q$ $\quad (|Q| = k)$

a. kan

b. $k + a + n$

c. ka^n

d. kna^n

1. How many $x_1 \dots x_n$'s are there? a^n
2. How many choices for q? k
3. How many locations are there for TM head? n

Total # of configs $= a^n \cdot k \cdot n$

LBA Halting

Theorem: Let B be an LBA with k states and a symbols in its tape alphabet. Then B halts on input w if and only if B halts on input w within kna^n steps.

Proof: $\Leftarrow$) If B halts on w w/in kna^n steps, then B halts on w

$\Rightarrow$) Contraposition. Assume B does not halt on w w/in $\underbrace{kna^n}_{\ell}$ steps
Show B does not halt on w

- B run on w enters configurations $\underbrace{C_0 \to C_1 \to \dots \to C_\ell}_{\ell+1 \text{ configurations}}$
- $\le \ell$ possible config. for B on w
- By pigeonhole, $\exists\, 0 \le i < j \le \ell$ s.t. $C_i = C_j$ $\qquad C_0 \to C_1 \to \dots \to C_i \to \dots$, $C_i = C_j$

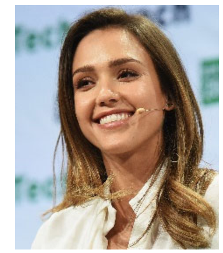
$\Rightarrow$ B enters an infinite loop, repeatedly returning to C_i

Deciding A_{LBA}

$A_{\text{LBA}} = \{\langle B, w\rangle \mid B \text{ is an LBA that accepts input } w\}$

Theorem: A_{LBA} is decidable

Proof: The following TM decides A_{LBA}:



On input $\langle B, w\rangle$:

1. Simulate B on input w **for $\boldsymbol{kna^n}$ steps**
2. If simulation accepts, accept.

   If simulation rejects or has not yet halted, reject.

Analysis: • If $\langle B,w\rangle \in A_{LBA}$, then B accepts w. By LBA Halting Thm
B accepts w w/in kna^n steps. $\Rightarrow$ TM accepts $\langle B,w\rangle$

• If $\langle B,w\rangle \notin A_{LBA}$, B either rejects or loops. In either case, TM rejects

LBAs can "check" TMs

LBAs are not powerful enough to perform general TM computations themselves.

But they can *check* the computation of a general TM M on input w

$B_{M,w}$ = "On input $x = \langle C_0, C_1, \dots, C_\ell \rangle$ a sequence of configs.:

Accept if all of the following hold, and reject otherwise:

1. C_0 is the starting configuration of M on w,
2. Every C_{i+1} legally follows from C_i, and
3. C_ℓ is an accepting configuration"

What is the language of $B_{M,w}$?

$L(B_{M,w}) = \{ x = \langle C_0, \dots, C_\ell \rangle \mid$ x is an accepting computation history for M on $w \}$

Computation History Method

Reduction from the undecidable language A_{TM} to a language L using the following idea:

Given an input $\langle M, w \rangle$ to A_{TM}, the ability to solve L enables checking the existence of an accepting computation history for M on w

Can be used to prove undecidability of E_{LBA}, ALL_{CFG}, Post Correspondence Problem, first-order logic …

E_{LBA} is unrecognizable

$E_{\text{LBA}} = \{\langle B\rangle \mid B \text{ is an LBA recognizing } \emptyset\}$

Theorem: $\overline{A_{\text{TM}}} \leq_{\text{m}} E_{\text{LBA}}$ hence E_{LBA} is unrecognizable

Proof: The following TM computes the reduction:

On input $\langle M, w\rangle$: (TM M, string w, input to $\overline{A_{TM}}$)

1. Construct LBA $B_{M,w}$ =as follows:

   $B_{M,w}$ = "On input $x = \langle C_0, C_1, \dots, C_\ell\rangle$ a sequence of configs.:

   Accept if x is an accepting computation history of

   M on w. Otherwise, reject.

   $L(B_{M,w}) = \{x \mid x$ is an accepting computation hist of M on input $w\}$

2. Output $\langle B_{M,w}\rangle$. (input to E_{LBA})

Analysis:
- $\langle M,w\rangle \in \overline{A_{TM}} \Rightarrow M$ does not accept on w
  $\Rightarrow \nexists\, x$ s.t. x is an accepting comp. hist. of M on w
  $\Rightarrow L(B_{M,w}) = \emptyset \Rightarrow \langle B_{M,w}\rangle \in E_{LBA}$
- $\langle M,w\rangle \notin \overline{A_{TM}} \Rightarrow M$ accepts on input $w \Rightarrow \exists\, x$ an accepting comp. hist. $\Rightarrow L(B_{M,w}) \neq \emptyset$
  $\Rightarrow \langle B_{M,w}\rangle \notin E_{LBA}$

Recap of LBAs

LBAs are simple:

- Can determine whether an LBA halts on a given input by checking if it repeats a configuration
- Implies A_{LBA} is decidable

LBAs are powerful:

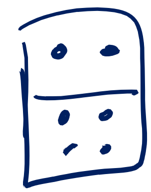
- An LBA can check the computation of a general TM on a given input
- Implies E_{LBA} is undecidable

Problems in Language Theory

A_{DFA} **decidable**	A_{LBA} **decidable**	A_{TM} **undecidable**
E_{DFA} **decidable**	E_{LBA} **undecidable**	E_{TM} **undecidable**
EQ_{DFA} **decidable**	EQ_{LBA} **undecidable**	EQ_{TM} **undecidable**

Undecidable problems outside language theory

<u>Post Correspondence Problem (PCP):</u>



Domino: $\left[\frac{a}{ab}\right]$. Top and bottom are strings.

Input: Collection of dominos.

$$\left[\frac{aa}{aba}\right], \left[\frac{ab}{aba}\right], \left[\frac{ba}{aa}\right], \left[\frac{abab}{b}\right]$$

Match: List of some of the input dominos (repetitions allowed) where top = bottom

$$\left[\frac{ab}{aba}\right], \left[\frac{aa}{aba}\right], \left[\frac{ba}{aa}\right], \left[\frac{aa}{aba}\right], \left[\frac{abab}{b}\right]$$

Problem: Does a match exist? This is undecidable