BU CS 332 – Theory of Computation

https://forms.gle/G5t6TLBUsBoA56cL6

Lecture 16:

- More Examples of Reductions
- Mapping Reductions

Reading:

Sipser Ch 5.1, 5.3

Mark Bun

March 26, 2025

Last Time: Reductions

A reduction from problem A to problem B is an algorithm for problem A which uses an algorithm for problem B as a subroutine

If such a reduction exists, we say "A reduces to B"

Positive uses: If A reduces to B and B is decidable, then A is also decidable

Ex. E_{DFA} is decidable $\Rightarrow$ EQ_{DFA} is decidable

Negative uses: If A reduces to B and A is undecidable, then B is also undecidable

Ex. UD is undecidable $\Rightarrow$ A_{TM} is undecidable

Halting Problem

Reduce from A_{TM} to $HALT_{TM}$

known this is undecidable

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

Theorem: $HALT_{\text{TM}}$ is undecidable

Proof: Suppose for contradiction that there exists a decider H for $HALT_{\text{TM}}$. We construct a decider for V for A_{TM} as follows:

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

1. Run H on input $\langle M, w\rangle$
2. If H rejects, reject
3. If H accepts, run M on w
4. If M accepts, accept

   Otherwise, reject.

Use H to test whether M will make a decision on w or not

Run M on w

- $\langle M, w\rangle \in A_{TM} \Rightarrow M$ accepts input w
  $\Rightarrow H$ accepts input $\langle M, w\rangle$ ✓
  $\Rightarrow$ proceed to line 3, V accepts in line 4
- $\langle M, w\rangle \notin A_{TM}$ either:
  a) M loops on $w \Rightarrow$
  H rejects input $\langle M, w\rangle$
  $\Rightarrow V$ rejects in line 2 ✓
  b) M rejects on $w \Rightarrow$
  H accepts $\langle M, w\rangle$
  $\Rightarrow$ proceed to line 3, V rejects in line 4 ✓

This is a reduction from A_{TM} to $HALT_{\text{TM}}$

Halting Problem

Computational problem: Given a program (TM) and input w, does that program halt on input w?

- A central problem in formal verification
- Dealing with undecidability in practice:
  - Use heuristics that are correct on most real instances, but may be wrong or loop forever on others
  - Restrict to a "non-Turing-complete" subclass of programs for which halting is decidable
  - Use a programming language that lets a programmer specify hints (e.g., loop invariants) that can be compiled into a formal proof of halting

$A_{TM} = \{\langle M, w\rangle \mid$ TM M accepts input $w\}$

Computational problem:
Given TM M, does M recognize the empty language?

Emptiness testing for TMs

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

Theorem: E_{TM} is undecidable

Proof: Suppose for contradiction that there exists a decider R for E_{TM}. We construct a decider V for A_{TM} as follows:

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

1. Run R on input ??? $\langle M\rangle$

2. If R accepts $\langle M\rangle$: reject
   If R rejects $\langle M\rangle$: accept.

$\langle M, w\rangle \in A_{TM}$
$\Rightarrow M$ accepts w
$\Rightarrow L(M) \neq \emptyset \Rightarrow \langle M\rangle \notin E_{TM}$
$\Rightarrow R$ rejects $\langle M\rangle$
$\Rightarrow V$ accepts ✓

$\langle M, w\rangle \notin A_{TM}$
$\Rightarrow M$ does not accept w
$\Rightarrow L(M)$ might or might not be empty
So behavior of R inconclusive

This is a reduction from A_{TM} to E_{TM}

Emptiness testing for TMs

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

Theorem: E_{TM} is undecidable

Proof: Suppose for contradiction that there exists a decider R for E_{TM}. We construct a decider V for A_{TM} as follows:

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

1. Construct a TM N as follows:

$$\left.\begin{aligned} M \text{ accepts } w &\Leftrightarrow R \text{ rejects} \\ &\Leftrightarrow \langle N \rangle \notin E_{\mathrm{TM}} \\ &\Leftrightarrow L(N) \neq \emptyset \end{aligned}\right\} \text{Need this for correctness}$$

2. Run R on input $\langle N \rangle$
3. If R rejects, accept. Otherwise, reject

What do we want out of machine N?

a) $L(N)$ is empty iff M accepts w

b) $L(N)$ is non-empty iff M accepts w

c) $L(M)$ is empty iff N accepts w

d) $L(M)$ is non-empty iff N accepts w

This is a reduction from A_{TM} to E_{TM}

Emptiness testing for TMs

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

Theorem: E_{TM} is undecidable

Proof: Suppose for contradiction that there exists a decider R for E_{TM}. We construct a decider V for A_{TM} as follows:

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

1. Construct a TM N as follows:

   "On input x: Ignore x

   Run M on w and output the result."

2. Run R on input $\langle N\rangle$
3. If R rejects, accept. Otherwise, reject

Proof that V decides A_{TM}

$\langle M,w\rangle \in A_{\mathrm{TM}}$
$\Rightarrow M$ accepts w
$\Rightarrow \forall x$ N accepts x
$\Rightarrow \boxed{L(N) = \Sigma^* \neq \emptyset}$
$\Rightarrow R$ rejects $\Rightarrow V$ accepts ✓

$\langle M,w\rangle \notin A_{\mathrm{TM}}$
$\Rightarrow M$ does not accept w
$\Rightarrow \forall x$ N does not accept x
$\Rightarrow \boxed{L(N) = \emptyset}$
$\Rightarrow R$ accepts $\Rightarrow V$ rejects ✓

V decides A_{TM}, but know A_{TM} undecidable ✗
$\Rightarrow V$ must not have been a decider
$\Rightarrow R$ must not have been a decider

This is a reduction from A_{TM} to E_{TM}

Equality Testing for TMs

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

Theorem: EQ_{TM} is undecidable

Proof: Suppose for contradiction that there exists a decider R for EQ_{TM}. We construct a decider for E_{TM} as follows:

On input $\langle M\rangle$:

1. Construct TMs N_1, N_2 as follows:

   N_1 = N_2 =

2. Run R on input $\langle N_1, N_2\rangle$
3. If R accepts, accept. Otherwise, reject.

This is a reduction from E_{TM} to EQ_{TM}

Equality Testing for TMs

want V accept $\iff$ R accepts $\iff$ $L(N_1) = L(N_2)$

What do we want out of the machines N_1, N_2?

a) $L(M) = \emptyset$ iff $N_1 = N_2$ $\boxed{\text{b)}}$ $L(M) = \emptyset$ iff $L(N_1) = L(N_2)$

c) $L(M) = \emptyset$ iff $N_1 \neq N_2$ d) $L(M) = \emptyset$ iff $L(N_1) \neq L(N_2)$

TM V:

On input $\langle M \rangle$: // want: $V(\langle M \rangle)$ accepts $\iff$ $L(M) = \emptyset$

1. Construct TMs N_1, N_2 as follows:

N_1 = $L(N_1) = \emptyset$

N_2 = $L(N_2) = L(M)$

2. Run R on input $\langle N_1, N_2 \rangle$ // $L(N_1) = L(N_2) \iff R$ accepts input $\langle N_1, N_2 \rangle$

3. If R accepts, accept. Otherwise, reject.

This is a reduction from E_{TM} to EQ_{TM}

Equality Testing for TMs

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

Theorem: EQ_{TM} is undecidable

Proof: Suppose for contradiction that there exists a decider R for EQ_{TM}. We construct a decider for E_{TM} as follows:

On input $\langle M\rangle$: TM V

1. Construct TMs N_1, N_2 as follows:

   N_1 = "On input x:
   reject"

   $L(N_1) = \emptyset$

   $N_2 = M$

   $L(N_2) = L(M)$

2. Run R on input $\langle N_1, N_2\rangle$
3. If R accepts, accept. Otherwise, reject.

$\langle M\rangle \in E_{TM}$
$\Rightarrow L(M) = \emptyset$
$\Rightarrow L(N_1) = \emptyset = L(N_2)$
$\Rightarrow R$ accepts
$\Rightarrow V$ accepts ✓

$\langle M\rangle \notin E_{TM}$
$\Rightarrow L(M) \neq \emptyset$
$\Rightarrow L(N_1) = \emptyset \neq L(N_2)$
$\Rightarrow R$ rejects
$\Rightarrow V$ rejects. ✓

However we know E_{TM} undecidable $\Rightarrow R$ could not have been a decider $\Rightarrow EQ_{TM}$ undecidable

This is a reduction from E_{TM} to EQ_{TM}

Regular language testing for TMs

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

Theorem: REG_{TM} is undecidable

Proof: Suppose for contradiction that there exists a decider R for REG_{TM}. We construct a decider for A_{TM} as follows:

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

M accepts input w $\Leftrightarrow$ L(N) is regular

1. Construct a TM N as follows:

2. Run R on input $\langle N \rangle$

3. If R accepts, accept. Otherwise, reject

This is a reduction from A_{TM} to REG_{TM}

Regular language testing for TMs

$$REG_{\mathrm{TM}} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) \text{ is regular}\}$$

Theorem: REG_{TM} is undecidable

Proof: Suppose for contradiction that there exists a decider R for REG_{TM}. We construct a decider for A_{TM} as follows:

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

1. Construct a TM N as follows:

   N = "On input x,

   1. If $x \in \{0^n 1^n \mid n \geq 0\}$, accept
   2. Run TM M on input w
   3. If M accepts, accept. Otherwise, reject."

2. Run R on input $\langle N \rangle$
3. If R accepts, accept. Otherwise, reject

M accepts w
$\Rightarrow \forall x$ N accepts x
$\Rightarrow L(N) = \{0,1\}^*$

M does not accept w
$\Rightarrow$ N accepts $x \Leftrightarrow x \in \{0^n 1^n \mid n \geq 0\}$
$\Rightarrow L(N) = \{0^n 1^n \mid n \geq 0\}$ which is not regular

This is a reduction from A_{TM} to REG_{TM}

Mapping Reductions

🚩 🚩 Warning 🚩 🚩

What's wrong with the following "proof"?

Bogus "Theorem": A_{TM} is not Turing-recognizable

Bogus "Proof": Let R be an alleged recognizer for A_{TM}. We construct a recognizer S for unrecognizable language $\overline{A_{\mathrm{TM}}}$:

TM S

On input $\langle M, w\rangle$: (input to $\overline{A_{TM}}$)
1. Run R on input $\langle M, w\rangle$
2. If R accepts, reject. If R rejects, accept.

Bug:
If M loops on w,
R might loop
$\Rightarrow$ S might loop

This sure looks like a reduction from $\overline{A_{\mathrm{TM}}}$ to A_{TM}

Mapping Reductions: Motivation

1. How do we formalize the notion of a reduction?
2. How do we use reductions to show that languages are unrecognizable?
3. How do we protect ourselves from accidentally "proving" bogus statements about recognizability?

Computable Functions

Definition:

A function $f: \Sigma^* \rightarrow \Sigma^*$ is computable if there is a TM M which, given as input any $w \in \Sigma^*$, halts with only $f(w)$ on its tape. ("Outputs $f(w)$")

Computable Functions

Definition:

A function $f: \Sigma^* \to \Sigma^*$ is computable if there is a TM M which, given as input any $w \in \Sigma^*$, halts with only $f(w)$ on its tape. ("Outputs $f(w)$")

Example 1: $f(w) = \text{sort}(w)$

Example 2: $f(\langle x, y \rangle) = x + y$

Computable Functions

Definition:

A function $f: \Sigma^* \to \Sigma^*$ is computable if there is a TM M which, given as input any $w \in \Sigma^*$, halts with only $f(w)$ on its tape. ("Outputs $f(w)$")

Example 3: $f(\langle M, w \rangle) = \langle M' \rangle$ where M is a TM, w is a string, and M' is a TM that ignores its input and simulates running M on w

TM computing f:

On input $\langle M, w \rangle$:
1. Construct TM M':
   "On input x: (Ignore x)
   Run M on w. If accepts, accept. If rejects, reject."
2. Output $\langle M' \rangle$.

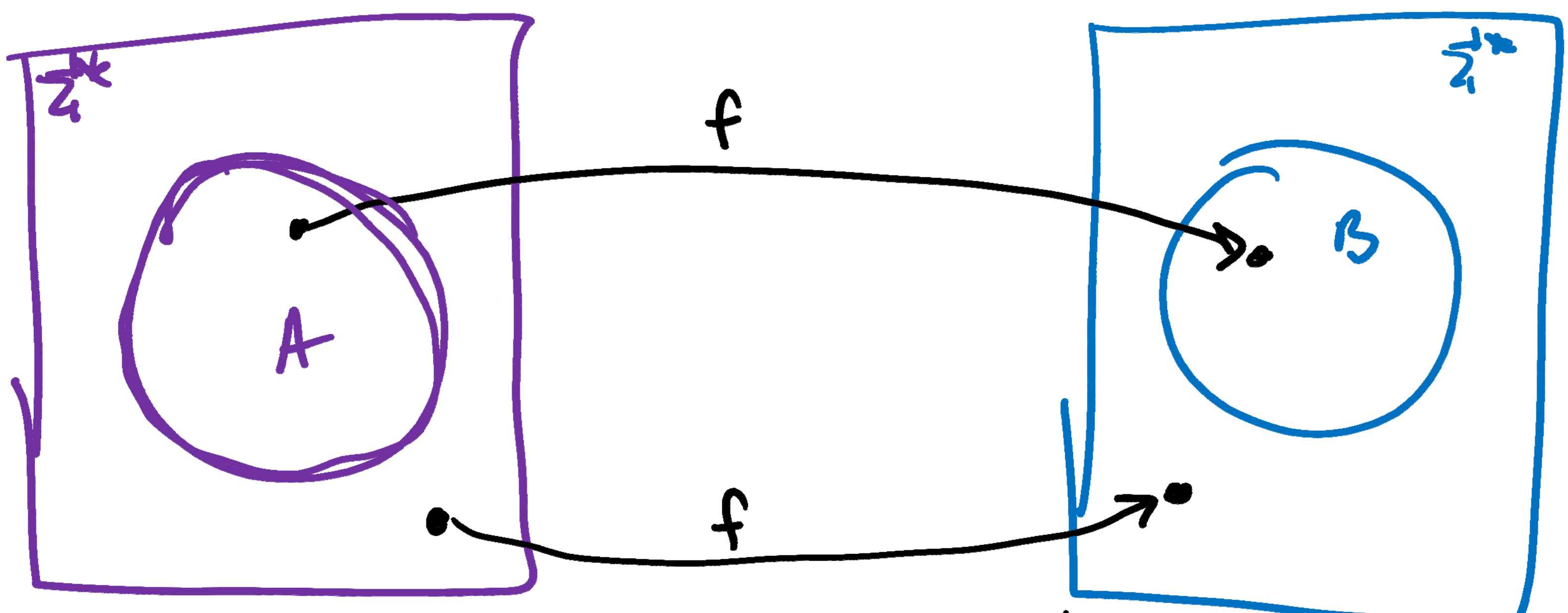
Mapping Reductions

Definition:

Let $A, B \subseteq \Sigma^*$ be languages. We say A is mapping reducible to B, written

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

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



Mapping Reductions

<u>Definition:</u>

Language A is mapping reducible to language B, written

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

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

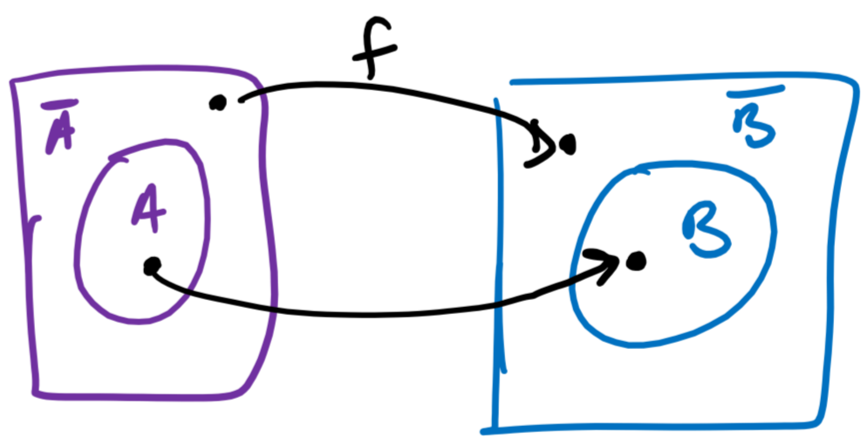
If $A \leq_{\mathrm{m}} B$, which of the following is true?

a) $\bar{A} \leq_{\mathrm{m}} B$

b) $A \leq_{\mathrm{m}} \bar{B}$

c) $\bar{A} \leq_{\mathrm{m}} \bar{B}$

d) $\bar{B} \leq_{\mathrm{m}} \bar{A}$



Decidability

Correctness proof:

$w \in A \Rightarrow f(w) \in B$ (by def. of mapping reduction)
$\Rightarrow M$ accepts $f(w)$ (since M decides B)
$\Rightarrow N$ accepts

Theorem: If $A \leq_{\mathrm{m}} B$ and B is decidable, then A is also decidable

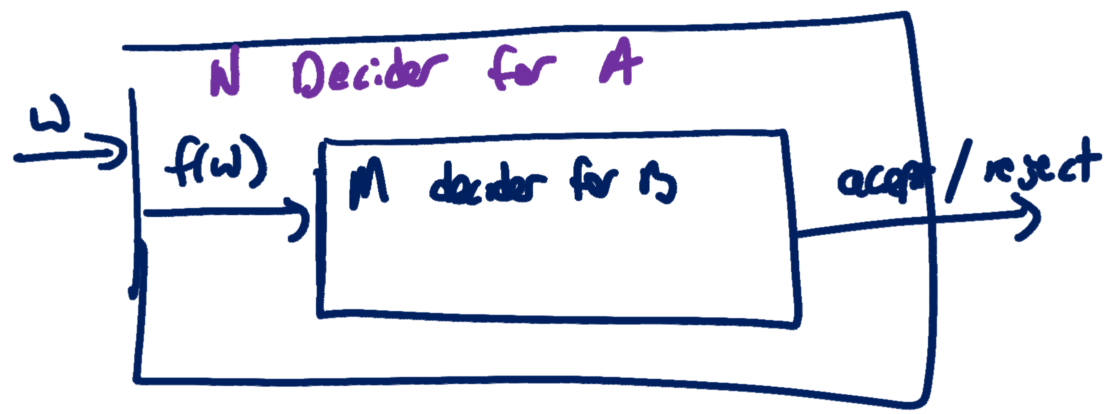
$w \notin A \Rightarrow f(w) \notin B$
$\Rightarrow M$ rejects $f(w) \Rightarrow N$ rejects

Proof: Let M be a decider for B and let $f: \Sigma^* \rightarrow \Sigma^*$ be a mapping reduction from A to B. We can construct a decider N for A as follows:

On input w:

1. Compute $f(w)$
2. Run M on input $f(w)$
3. If M accepts, accept.

   If it rejects, reject.



Undecidability

Theorem: If $A \leq_{\mathrm{m}} B$ and B is decidable, then A is also decidable

Corollary: If $A \leq_{\mathrm{m}} B$ and A is undecidable, then B is also undecidable

Old Proof: Equality Testing for TMs

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

Theorem: EQ_{TM} is undecidable

Proof: Suppose for contradiction that there exists a decider R for EQ_{TM}. We construct a decider for E_{TM} as follows:

On input $\langle M\rangle$:

1. Construct TMs M_1, M_2 as follows:

$M_1 = M$

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

2. Run R on input $\langle M_1, M_2\rangle$
3. If R accepts, accept. Otherwise, reject.

This is a reduction from E_{TM} to EQ_{TM}

New Proof: Equality Testing for TMs

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

Theorem: $E_{\mathrm{TM}} \leq_{\mathrm{m}} EQ_{\mathrm{TM}}$ (Hence EQ_{TM} is undecidable)

Proof: The following TM N computes the reduction f:

On input $\langle M\rangle$:

1. Construct TMs M_1, M_2 as follows:

$M_1 = M$

$M_2 =$ "On input x,

1. Ignore x and reject"

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

Mapping Reductions: Recognizability

Theorem: If $A \leq_{\mathrm{m}} B$ and B is recognizable, then A is also recognizable

Proof: Let M be a recognizer for B and let $f: \Sigma^* \to \Sigma^*$ be a mapping reduction from A to B. Construct a recognizer N for A as follows:

Proof of correctness (same as before):

$w \in A \Rightarrow f(w) \in B$ (correctness of mapping red.)
$\Rightarrow M$ accepts $f(w)$ (correctness of M)
$\Rightarrow N$ accepts

$w \notin A \Rightarrow f(w) \notin B$
$\Rightarrow M$ does not accept $f(w)$
$\Rightarrow N$ does not accept

On input w:

1. Compute $f(w)$
2. Run M on input $f(w)$
3. If M accepts, accept.

   If it rejects, reject.

Unrecognizability

Theorem: If $A \leq_{\mathrm{m}} B$ and B is recognizable, then A is also recognizable

Corollary: If $A \leq_{\mathrm{m}} B$ and A is unrecognizable, then B is also unrecognizable

Corollary: If $\overline{A_{\mathrm{TM}}} \leq_{\mathrm{m}} B$, then B is unrecognizable