UMB CS 420
Unrecognizability
Wednesday, April 6, 2022
Announcements

• HW 8 in
  • Due Wed 4/6 11:59pm EST

• HW 9 out
  • Due Sun 4/17 11:59pm EST
Last Time: Showing Mapping Reducibility

Language $A$ is **mapping reducible** to language $B$, written $A \leq_m B$, if there is a **computable function** $f : \Sigma^* \rightarrow \Sigma^*$, where for every $w$,

$$w \in A \iff f(w) \in B.$$ 

The function $f$ is called the **reduction** from $A$ to $B$.

**Step 1:** Show there is computable fn $f$ ... by creating a TM

**Step 2:** Prove the iff is true for $f$

**Step 2a:** “forward” direction ($\Rightarrow$): if $w \in A$ then $f(w) \in B$

**Step 2b:** “reverse” direction ($\Leftarrow$): if $f(w) \in B$ then $w \in A$

**Step 2b:** Equivalent (contrapositive): if $w \notin A$ then $f(w) \notin B$
Last Time: Using Mapping Reducibility

To prove decidability ...

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

To prove undecidability ...

• If $A \leq_m B$ and $A$ is undecidable, then $B$ is undecidable.

Be careful with the direction of the reduction!
Flashback: \( \text{EQ}_{\text{TM}} \) is undecidable

\[ \text{EQ}_{\text{TM}} = \{ \langle M_1, M_2 \rangle | M_1 \text{ and } M_2 \text{ are TMs and } L(M_1) = L(M_2) \} \]

Proof by contradiction:

- **Assume** \( \text{EQ}_{\text{TM}} \) has *decider* \( R \); use to create \( \text{ET}_{\text{TM}} \) *decider*:

\[ \text{ET}_{\text{TM}} = \{ \langle M \rangle | M \text{ is a TM and } L(M) = \emptyset \} \]

\[ S = \text{“On input } \langle M \rangle, \text{ where } M \text{ is a TM:} \]

1. Run \( R \) on input \( \langle M, M_1 \rangle \), where \( M_1 \) is a TM that rejects all inputs.
2. If \( R \) accepts, *accept*; if \( R \) rejects, *reject.*"
Alternate Proof: \( EQ_{TM} \) is undecidable

\[ EQ_{TM} = \{ \langle M_1, M_2 \rangle | M_1 \text{ and } M_2 \text{ are TMs and } L(M_1) = L(M_2) \} \]

Show mapping reducibility: \( E_{TM} \leq_m EQ_{TM} \)

**Step 1:** create computable fn \( f \), computed by TM \( S \)

\( S = \) “On input \( \langle M \rangle \), where \( M \) is a TM:

1. Construct: \( \langle M, M_1 \rangle \), where \( M_1 \) is a TM that rejects all inputs.
2. Output: \( \langle M, M_1 \rangle \)

**Step 2:** show iff requirements of mapping reducibility

| \( \Rightarrow \) | If \( \langle M \rangle \in E_{TM} \), then \( \langle M, M_1 \rangle \in EQ_{TM} \) |
| \( \Leftrightarrow \) | If \( \langle M \rangle \notin E_{TM} \), then \( \langle M, M_1 \rangle \notin EQ_{TM} \) |

And use theorem ...

If \( A \leq_m B \) and \( A \) is undecidable, then \( B \) is undecidable.
Flashback: \( E_{TM} \) is undecidable

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

Proof, by contradiction:

- Assume \( E_{TM} \) has decider \( R \); use to create \( A_{TM} \) decider:

\[ S = \text{“On input } \langle M, w \rangle \text{, an encoding of a TM } M \text{ and a string } w:\]

1. Use the description of \( M \) and \( w \) to construct the TM \( M_1 \)

\[ M_1 = \text{“On input } x:\]

1. If \( x \neq w \), reject.
2. If \( x = w \), run \( M \) on input \( w \) and accept if \( M \) does.”

2. Run \( R \) on input \( \langle M_1 \rangle \).
3. If \( R \) accepts, reject; if \( R \) rejects, accept.”

If \( M \) accepts \( w \), \( M_1 \) \underline{not} in \( E_{TM} \)!
Alternate Proof: $E_{TM}$ is undecidable

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

Show mapping reducibility??: $A_{TM} \leq_{m} E_{TM}$

**Step 1:** create computable fn $f: \langle M, w \rangle \rightarrow \langle M_1 \rangle$, computed by $S$

$S = \text{“On input } \langle M, w \rangle, \text{ an encoding of a TM } M \text{ and a string } w:$

1. Use the description of $M$ and $w$ to construct the TM $M_1$

   $M_1 = \text{“On input:}$
   
   1. If $x \neq w$, reject.
   2. If $x = w$, run $M$ on input $w$ and accept if $M$ does.”

2. Output: $\langle M_1 \rangle$.

3. If $R$ accepts, reject; if $R$ rejects, accept.”

**Step 2:** show iff requirements of mapping reducibility:

? $\Rightarrow$ if $\langle M, w \rangle \in A_{TM}$, then $\langle M_1 \rangle \not\in E_{TM}$

? $\Leftarrow$ if $\langle M, w \rangle \not\in A_{TM}$, then $\langle M_1 \rangle \in E_{TM}$

- This reduces $A_{TM}$ to $E_{TM}$ !!
- It’s good enough, if: undecidable langs are closed under complement
Undecidable Langs Closed under Complement

Proof by contradiction

- **Assume** some lang $L$ is undecidable and $\overline{L}$ is decidable ...
  - Then $\overline{L}$ has a decider

- ... then we can create decider for $L$ from decider for $\overline{L}$ ...
  - Because decidable languages are closed under complement (hw8)!

Contradiction!
Turing Unrecognizable?

Is there anything out here?

$A_{TM}$

Turing-recognizable

decidable

context-free

regular

Where do these undecidable languages go?

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

$EQ_{CFG} = \{ \langle G, H \rangle | G \text{ and } H \text{ are CFGs and } L(G) = L(H) \}$

$EQ_{TM} = \{ \langle M_1, M_2 \rangle | M_1 \text{ and } M_2 \text{ are TMs and } L(M_1) = L(M_2) \}$
Thm: Some langs are not Turing-recognizable

Proof: requires 2 lemmas

• **Lemma 1**: The set of all languages is *uncountable*
  • **Proof**: Show there is a bijection with another uncountable set ... 
    • ... The set of all infinite binary sequences

• **Lemma 2**: The set of all TMs is *countable*

• Therefore, some language is not recognized by a TM (pigeonhole principle)
Mapping a Language to a Binary Sequence

\[ \Sigma^* = \{ \epsilon, 0, 1, 00, 01, 10, 11, 000, 001, \ldots \} \]

\[ A = \{ 0, 00, 01, 000, 001, \ldots \} \]

\[ \chi_A = 0 \ 1 \ 0 \ 1 \ 1 \ 0 \ 0 \ 1 \ 1 \ 1 \ \ldots \]

Each digit represents one possible string:
- 1 if lang has that string,
- 0 otherwise
**Thm:** Some langs are not Turing-recognizable

**Proof:** requires 2 lemmas

- **Lemma 1:** The set of all languages is **uncountable**
  - **Proof:** Show there is a bijection with another uncountable set ...
    - ... The set of all infinite binary sequences
      - Now just prove set of infinite binary sequences is uncountable (exercise)

- **Lemma 2:** The set of all TMs is **countable**
  - Because every TM $M$ can be encoded as a string $<M>$
  - And set of all strings is countable

- Therefore, some language is not recognized by a TM
Co-Turing-Recognizability

• A language is co-Turing-recognizable if ...
• ... it is the complement of a Turing-recognizable language.
Thm: Decidable $\Leftrightarrow$ Recognizable & co-Recognizable
Thm: Decidable $\Leftrightarrow$ Recognizable & co-Recognizable

$\Rightarrow$ If a language is **decidable**, then it is **recognizable** and **co-recognizable**

- Decidable $\Rightarrow$ Recognizable:
  - A decider is a recognizer, bc decidable langs are a subset of recognizable langs
- Decidable $\Rightarrow$ Co-Recognizable:
  - To create co-decider from a decider ... switch reject/accept of all inputs
  - A co-decider is a co-recognizer, for same reason as above

$\Leftarrow$ If a language is **recognizable** and **co-recognizable**, then it is **decidable**
**Thm:** Decidable ⇔ Recognizable & co-Recognizable

⇒ If a language is **decidable**, then it is **recognizable** and **co-recognizable**
  • Decidable ⇒ Recognizable:
    • A decider is a recognizer, bc decidable langs are a subset of recognizable langs
  • Decidable ⇒ Co-Recognizable:
    • To create co-decider from a decider ... switch reject/accept of all inputs
    • A co-decider is a co-recognizer, for same reason as above

⇐ If a language is **recognizable** and **co-recognizable**, then it is **decidable**
  • Let $M_1$ = recognizer for the language,
  • and $M_2$ = recognizer for its complement

  **Decider $M$:**
  • Run 1 step on $M_1$,
  • Run 1 step on $M_2$,
  • Repeat, until one machine accepts. If it’s $M_1$, accept. If it’s $M_2$, reject

Termination Arg: Either $M_1$ or $M_2$ must accept and halt, so $M$ halts and is a decider
A Turing-unrecognizable language

• We’ve proved:

  \( A_{TM} \) is Turing-recognizable

  \( A_{TM} \) is undecidable

• So:

  \( \overline{A_{TM}} \) is not Turing-recognizable

• Because: recognizable & co-recognizable implies decidable
Is there anything out here?

Where do these undecidable languages go?

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

\[ EQ_{CFG} = \{ \langle G, H \rangle \mid G \text{ and } H \text{ are CFGs and } L(G) = L(H) \} \]

\[ EQ_{TM} = \{ \langle M_1, M_2 \rangle \mid M_1 \text{ and } M_2 \text{ are TMs and } L(M_1) = L(M_2) \} \]
Using Mapping Reducibility to Prove ...

- Decidability

- Undecidability

- Recognizability

- Unrecognizability
More Helpful Theorems

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

If $A \leq_m B$ and $A$ is not Turing-recognizable, then $B$ is not Turing-recognizable.

• Same proofs as:
  
  If $A \leq_m B$ and $B$ is decidable, then $A$ is decidable.
  
  If $A \leq_m B$ and $A$ is undecidable, then $B$ is undecidable.
**Thm:** $EQ_{TM}$ is neither Turing-recognizable nor co-Turing-recognizable.

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

1. $EQ_{TM}$ is not Turing-recognizable

---

$A_{TM} \leq_m B_{TM}$ and $A$ is not Turing-recognizable, then $B$ is not Turing-recognizable.
Mapping Reducibility implies Mapping Red. of Complements

Language $A$ is *mapping reducible* to language $B$, written $A \leq_m B$, if there is a computable function $f : \Sigma^* \rightarrow \Sigma^*$, where for every $w$,

$$w \in A \iff f(w) \in B.$$  

The function $f$ is called the *reduction* from $A$ to $B$.  

\[
\begin{array}{c}
A \leq_m B  \\
\text{implies} \\
\overline{A} \leq_m \overline{B}
\end{array}
\]
**Thm:** $EQ_{TM}$ is neither Turing-recognizable nor co-Turing-recognizable

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

1. $EQ_{TM}$ is not Turing-recognizable

Two Choices:
- Create Computable fn: $A_{TM} \rightarrow EQ_{TM}$
- Or Computable fn: $A_{TM} \rightarrow \overline{EQ_{TM}}$

And use theorem ...

If $A \leq_m B$ and $A$ is not Turing-recognizable, then $B$ is not Turing-recognizable.
**Thm:** $\text{EQ}_{\text{TM}}$ is not Turing-recognizable

$\text{EQ}_{\text{TM}} = \{(M_1, M_2) | M_1 \text{ and } M_2 \text{ are TMs and } L(M_1) = L(M_2)\}$

- **Create Computable fn:** $A_{\text{TM}} \rightarrow \overline{\text{EQ}_{\text{TM}}}$

**Step 1**

**Computable fn**

$\langle M, w \rangle \rightarrow \langle M_1, M_2 \rangle$ $M_1$ and $M_2$ are TMs and $L(M_1) \neq L(M_2)$

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

1. Construct the following two machines, $M_1$ and $M_2$.
   $M_1 =$ “On any input:
   1. Reject.”

2. $M_2 =$ “On any input:
   1. Run $M$ on $w$. If it accepts, accept.”

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

**Step 2**, iff:

$\Rightarrow$ If $M$ accepts $w$, then $M_1 \neq M_2$

$\Leftarrow$ If $M$ does not accept $w$, then $M_1 = M_2$
**Thm:** $E_{Q TM}$ is neither Turing-recognizable nor co-Turing-recognizable

$E_{Q TM} = \{ (M_1, M_2) | M_1$ and $M_2$ are TMs and $L(M_1) = L(M_2) \}$

1. $E_{Q TM}$ is not Turing-recognizable
   - Create Computable fn: $A_{TM} \rightarrow E_{Q TM}$
   - Or Computable fn: $A_{TM} \rightarrow \overline{E_{Q TM}}$
   - **DONE!**

   If $A \leq_m B$ and $A$ is not Turing-recognizable, then $B$ is not Turing-recognizable.

2. $\overline{E_{Q TM}}$ is not co- Turing-recognizable
   - (A lang is co-Turing-recog. if it is complement of Turing-recog. lang)
Previous: $EQ_{TM}$ is not Turing-recognizable

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

- Create Computable fn: $A_{TM} \rightarrow \overline{EQ_{TM}}$

**Step 1**

$\langle M, w \rangle \rightarrow \langle M_1, M_2 \rangle$ $M_1$ and $M_2$ are TMs and $L(M_1) \neq L(M_2)$

\[
F = \text{“On input } \langle M, w \rangle \text{, where } M \text{ is a TM and } w \text{ a string:}
\]

1. Construct the following two machines, $M_1$ and $M_2$.
   \[M_1 = \text{“On any input: } 1. \text{ Reject.”} \]
   \[M_2 = \text{“On any input: } 1. \text{ Run } M \text{ on } w. \text{ If it accepts, accept.”} \]

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

- Accepts nothing
- Accepts nothing or everything
Now: $\overline{EQ_{TM}}$ is not Turing-recognizable

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

• Create Computable fn: $A_{TM} \rightarrow \overline{EQ_{TM}}$

Step 1 \( \langle M, w \rangle \rightarrow \langle M_1, M_2 \rangle \) $M_1$ and $M_2$ are TMs and $L(M_1) \neq L(M_2)$

\[
F = \text{“On input } \langle M, w \rangle, \text{ where } M \text{ is a TM and } w \text{ a string:}
\]

1. Construct the following two machines, $M_1$ and $M_2$.
   $\begin{align*}
   M_1 &= \text{“On any input:} \\
   &\quad 1. \text{Accept.”} \\
   M_2 &= \text{“On any input:} \\
   &\quad 1. \text{Run } M \text{ on } w. \text{ If it accepts, accept.”}
   \end{align*}
\]

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

Step 2, iff:
\[ \Rightarrow \text{If } M \text{ accepts } w, \text{ then } M_1 \equiv M_2 \]
\[ \Leftarrow \text{If } M \text{ does not accept } w, \text{ then } M_1 \not\equiv M_2 \]

DONE!
Unrecognizable Languages?

- \( A_{TM} \)
- Turing-recognizable
- Decidable
- Context-free
- Regular

Where do these go?

- \( E_{TM} = \{ \langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset \} \)
- \( EQ_{CFG} = \{ \langle G, H \rangle \mid G \text{ and } H \text{ are CFGs and } L(G) = L(H) \} \)
- \( EQ_{TM} = \{ \langle M_1, M_2 \rangle \mid M_1 \text{ and } M_2 \text{ are TMs and } L(M_1) = L(M_2) \} \)
Unrecognizable Languages

\[ A_{TM} \]

Turing-recognizable

decidable

context-free

regular

\[ E_{TM} = \{ \{ M \} | M \text{ is a TM and } L(M) = \emptyset \} \]

\[ EQ_{TM} = \{ \{ G, H \} | G \text{ and } H \text{ are CFGs and } L(G) = L(H) \} \]
Thm: $\mathit{EQ}_{\mathit{CFG}}$ is not Turing-recognizable

Recognizable & co-recognizable implies decidable

• We’ve proved:

  $\mathit{EQ}_{\mathit{CFG}}$ is undecidable

• We now prove:

  $\mathit{EQ}_{\mathit{CFG}}$ is co-Turing recognizable

• And conclude that:
  • $\mathit{EQ}_{\mathit{CFG}}$ is not Turing recognizable

Unrecognizability
Proof Technique #1
Thm: $EQ_{\text{CFG}}$ is co-Turing-recognizable

$EQ_{\text{CFG}} = \{(G, H) \mid G$ and $H$ are CFGs and $L(G) = L(H)\}$

Recognizer for $\overline{EQ_{\text{CFG}}}$:

- On input $(G, H)$:
  - For every possible string $w$:
    - Accept if $w \in L(G)$ and $w \notin L(H)$
    - Or accept if $w \in L(H)$ and $w \notin L(G)$
  - Else reject

This is only a recognizer because it loops for ever when $L(G) = L(H)$
Unrecognizable Languages

**Diagram:**
- $A_{TM}$: Turing-recognizable
- $E_{TM}$: $\{\{M\} | M$ is a TM and $L(M) = \emptyset\}$
- $EQ_{TM}$: $\{\{G, H\} | G$ and $H$ are CFGs and $L(G) = L(H)\}$

**Question:** Where do these go?
Unrecognizable Languages

$A_{TM}$

Turing-recognizable

decidable

context-free

regular

$E_{Q_{TM}}$ $E_{Q_{CFG}}$

Where do these go?

$E_{TM} = \{ \{ M \} \mid M \text{ is a TM and } L(M) = \emptyset \}$
Thm: $E_{TM}$ is not Turing-recognizable

Recognizable & co-recognizable implies decidable

- We’ve proved:
  - $E_{TM}$ is undecidable

- We now prove:
  $E_{TM}$ is co-Turing recognizable

- And then conclude that:
  - $E_{TM}$ is not Turing recognizable
**Thm:** $E_{TM}$ is co-Turing-recognizable

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

**Recognizer for $\overline{E_{TM}}$:** Let $s_1, s_2, \ldots$ be a list of all strings in $\Sigma^*$

"On input $\langle M \rangle$, where $M$ is a TM:

1. Repeat the following for $i = 1, 2, 3, \ldots$
2. Run $M$ for $i$ steps on each input, $s_1, s_2, \ldots, s_i$.
3. If $M$ has accepted any of these, accept. Otherwise, continue."

This is only a **recognizer** because it loops for ever when $L(M)$ is empty.
Unrecognizable Languages
Check-in Quiz 4/6

On gradescope