CSE 105
Theory of Computation

- Professor Jeanne Ferrante
Today’s Agenda

- Undecidability
  - Review and More Problems
  - A Non-TR Language

Reminders and announcements:
- HW 7 (Last!!) due Fri May 27 by 11:59 pm
- Reading Quiz 9 (Last!!) due Mon May 30 by 11:59 pm
- REVIEW SESSIONS (Nathan Speidel):
  - Wed May 25 8 pm - 9:20 pm in WLH 2001
  - Thurs Jun 2 7 pm - 8:50 pm in Peterson 108
- Final Exam: If you need a LEFT handed seat, make a private post to instructors on Piazza
Status: Language Hierarchy

- **Regular**
- **Context-Free**
- **Decidable**
- **Turing-Recognizable**

**CF but not Regular:**
\( \{0^n1^n | n > 0\} \) By pumping lemma (Regular)

**Decidable but not CF:**
\( \{ a^n b^n c^n | n > 0\} \) By pumping lemma (CF)

**Turing-Recognizable but Not Decidable:**
\( A_{TM} \)
\( \text{HALT}_{TM}, E_{TM} \)
\( \text{REGULAR}_{TM} \) By Diagonalization

**Not Turing Recognizable:**
\( ? \) By counting
Decidable Or Undecidable?

Decidable languages:
Construct a Decider!

- $A_{DFA}$
- $E_{DFA}$
- $EQ_{DFA}$
- $ALL_{DFA}$
- $A_{CFG}$
- $E_{CFG}$
- Any CFL

Undecidable languages:
By Diag. (D) or Reduction

- $A_{TM}$ (D)
- $E_{TM}$
- $EQ_{TM}$
- $HALT_{TM}$
- $REGULAR_{TM}$
- $T = \{ <M> \mid M \text{ is a TM and if } M \text{ accepts } w \text{ then it also accepts } w^R \}$
Ex: \( T = \{ <M> \mid M \text{ is a TM and if } M \text{ accepts } w \text{ then it also accepts } w^R \} \) is undecidable

We show that \( A_{TM} \) reduces to \( T \).

Proof: By Contradiction. Assume that \( T \) decidable, with TM \( R_T \). We show that \( A_{TM} \) is decidable, a contradiction.

• Construct a TM \( M_{ATM} \) that decides \( A_{TM} \):

\[
M_{ATM} = \begin{cases} 
\text{On input } <M, w>:\n1. \text{Construct } X = \text{On input } w: \n2. \text{Run } R_T \text{ with input } <X>; \text{ If } R_T \text{ accepts, then accept. If } R_T \text{ rejects, reject.} \end{cases}
\]

Correctness: \( M_{ATM} \) is a decider since \( R_T \) is, and accepts \( <M, w> \) iff \( R_T \) accepts \( <X> \) iff \( M \) accepts \( w \). So \( L(M_{ATM}) = A_{TM} \).

• But \( A_{TM} \) is undecidable, a contradiction. So the assumption is false and \( E_{TM} \) is undecidable.
How do we construct $X$?

Choose very simple $X$!

We construct $X$ so that

- $L(X) = \{01, 10\}$ if $M$ accepts $w$ OR $L(X) = \{01\}$ if $M$ does not accept $w$

- Note the *subset* relationship between these 2

- If $M$ accepts $w$, can add more strings to $L(X)$!

- $X =$ “On input $y$:
  1. If $y \neq 01$ or 10, reject.
  2. If $y = 01$, accept
  3. If $y = 10$, run $M$ on $w$. If it accepts, accept”

- Then within $M_{ATM}$ run $M_T$ with input $<X>$ to distinguish between these 2, and decide $A_{TM}$
Ex: $T = \{<M> \mid M$ is a TM and if $M$ accepts $w$ then it also accepts $w^R\}$ is undecidable

We show that $A_{TM}$ reduces to $T$.

Proof: By Contradiction. Assume that $T$ decidable, with TM $R_T$. We show that $A_{TM}$ is decidable, a contradiction.

• Construct a TM $M_{ATM}$ that decides $A_{TM}$:

$M_{ATM} = "$ On input $<M,w>$:
1. Construct $X = "$ On input $w$: // $L(X) = \{01, 10\}$ iff $M$ accepts $w$
   1. If $w \neq 01$ or $10$, reject.
   2. If $w = 01$, accept
   3. If $w = 10$, run $M$ on $w$. If it accepts, accept"

2. Run $R_T$ with input $<X>$; If $R_T$ accepts, then accept. If $R_T$ rejects, then reject. ”

Correctness: $M_{ATM}$ is a decider since $R_T$ is, and accepts $<M,w>$ iff $R_T$ accepts $<X>$ iff $L(X) = \{01, 10\}$ iff $M$ accepts $w$ iff $L(M_{ATM}) = A_{TM}$.

But $A_{TM}$ is undecidable, a contradiction. So the assumption is false and $E_{TM}$ is undecidable.
We just did a Reduction

• For 2 problems $P_1$ and $P_2$, $P_1$ reduces to $P_2$ if any solution for $P_2$ can be used to solve $P_1$.

Suppose $P_1$ reduces to $P_2$, then which of the following are true?

- **A.** If $P_2$ is decidable, then $P_1$ is also decidable.
- **X.** If $P_1$ is decidable, then $P_2$ is also decidable.
- **✓.** If $P_1$ is undecidable, then $P_2$ is also undecidable.
- **✓.** None or more than one of the above are true.
Example. \( \text{INF}_{TM} = \{<M> \mid M \text{ a TM and } L(M) \text{ is infinite}\} \) is undecidable

We show that \( A_{TM} \) reduces to \( \text{INF}_{TM} \).

Proof: By contradiction. Assume that \( \text{INF}_{TM} \) is decidable, with TM \( R \). We show then that \( A_{TM} \) is decidable, a contradiction.

Construct TM \( M_{ATM} \) that decides \( A_{TM} \):

\( M_{ATM} = " \text{On input } <M,w>: \)  

1. Construct TM \( X \):

2. Run \( R \) on \( <X> \). If \( R \) accepts, accept. If \( R \) rejects, reject. "

- Correctness: \( M_{ATM} \) is a decider since \( R \) is, and accepts \( <M,w> \) iff \( R \) accepts \( <X> \) iff ........... iff \( M \) accepts \( w \).
- But \( A_{TM} \) is undecidable, a contradiction. So the assumption is false and \( \text{INF}_{TM} \) is undecidable.
What should X do?

• If M accepts w, then \( L(X) \) should be infinite
  – Pick one! Let’s make it easy and choose \{0,1\}*
• If M rejects w, then \( L(X) \) must be finite
  – Pick one! \{0\}

\( X(z) = \) “1. If \( z = 0 \), accept.
  2. If \( z \neq 0 \), then run M on w. If M accepts w, then accept z”

Qu: What is \( L(X) \) if M accepts w?

A. \( L(X) = \{0\} \)
B. \( L(X) = \{z\} \)
C. \( L(X) = \{0,1\}^* \)
D. \( L(X) = \{ w \in \{0,1\}^* \mid w \neq 0 \} \)
E. None of the above
What should X do?

• If M accepts w, then \( L(X) \) should be infinite
  – Pick one! Let’s make it easy and choose \( \{0,1\}^* \)

• If M rejects w, then \( L(X) \) must be finite
  – Pick one! \( \{0\} \)

\( X(z) = \) “1. If \( z = 0 \), accept.

2. If \( z \) is \( \neq 0 \), then run M on w. If M accepts w, then accept z”

Qu: What is \( L(X) \) if M does not accept w?

A. \( L(X) = \{0\} \)
B. \( L(X) = \{z\} \)
C. \( L(X) = \{0,1\}^* \)
D. \( L(X) = \{ w \in \{0, 1\}^* \mid w \neq 0 \} \)
E. None of the above
Ex. \( \text{INF}_{\text{TM}} = \{<M> \mid M \text{ a TM and } L(M) \text{ is infinite}\} \) is undecidable

We show that \( A_{\text{TM}} \) reduces to \( \text{INF}_{\text{TM}} \).

**Proof:** By contradiction. Assume that \( \text{INF}_{\text{TM}} \) is decidable, with TM R. We show then that \( A_{\text{TM}} \) is decidable, a contradiction.

Construct TM \( M_{\text{ATM}} \) that decides \( A_{\text{TM}} \):

\[
M_{\text{ATM}} = \text{"On input } <M,w>:\n\]

1. Construct TM X: X (z) = “
   a) If z = 0, accept. // X accepts only string 0 if M does not accept w
   b) If z \neq 0, then run M on w. If M accepts w, then accept z
      // X accepts all strings if M accepts w”

2. Run R on <X>. If R accepts, accept. If R rejects, reject. “
   
   • Correctness: \( M_{\text{ATM}} \) is a decider since R is, and accepts <M,w> iff R accepts <X> iff \( L(X) \) is infinite iff M accepts w.
   
   • But \( A_{\text{TM}} \) is undecidable, a contradiction. So the assumption is false and \( \text{INF}_{\text{TM}} \) is undecidable.
Many Undecidability proofs follow a common pattern:

- Always a proof by contradiction
  - Assume $T$ is decidable by TM $M_T$
    - $T$ checks for condition $P$, and always halts with accept or reject
- Use $M_T$ to construct TM $M_{ATM}(<M,w>)$ to decide $A_{TM}$
- **Within** $M_{ATM}$, construct special TM $X$ such that
  1. If $M$ accepts $w$, then $L(X)$ has property $P$
  2. If $M$ does not accept $w$, then $L(X)$ has **property not $P$**

Run $M_T$ with input $<X>$ to distinguish between $P$ or not $P$ for $L(X)$, to decide if $M$ accepts $w$

- Show that $M_{ATM}$ decides $A_{TM}$ for the contradiction

Note: sometimes easier to build $X$ so that $X$ has $P$ iff $w$ not in $L(M)$
$\text{ODD}_T = \{<M> \mid M \text{ is a TM and } L(M) \text{ consists of strings of odd length}\}$ is undecidable

We show $A_T$ reduces to $\text{ODD}_T$

Proof: Assume that $\text{ODD}_T$ is decidable, with TM $R$. We show that $A_T$ is decidable, a contradiction.

- Construct a TM $M_{ATM}$ that decides $A_T$:

  $M_{ATM} = \text{“On input } <M,w>:\text{”}$

  1. Construct TM $X$ as follows:
     
     $X = \text{“On input } y:\text{”}$
     
     If $y = 0$ then run $M$ on $w$. If $M$ accepts, accept

  2. Run $R$ on $<X>$. If $R$ accepts, accept. If $R$ rejects, reject. “

- Correctness: $M_{ATM}$ is a decider since $R$ is, and $R$ accepts $<X>$ iff $L(X) = \{0\}$ iff $M$ accepts $w$.

- But $A_T$ is undecidable, a contradiction. So the assumption is false and $\text{ODD}_T$ is undecidable.

What is $L(X)$ if $M$ does not accept $w$?

A. $A_T$
B. $\{0\}$
\[\text{C. } \phi\]
D. $0$
E. None of the above
Why this $X$??

Want $X$ to be such that

- $L(X)$ consists of strings of odd length if $M$ accepts $w$
- $L(X)$ does not consist of strings of odd length if $M$ does not accept $w$

$X = \text{“On input } y:\n1. \text{If } y = 0 \text{ then run } M \text{ on } w. \text{ If } M \text{ accepts, accept”}$
Recall: There are exceptions!

No “special X” construction needed for:

• $A_{TM}$ \textit{directly} reduces to $HALT_{TM}$

Define $M_{ATM}(<M,w>):$

1. Run $M_{HALT}(<M,w>)$ If $M_{HALT}$ rejects, then reject.
2. Else, run $M(w)$; If it accepts, then accept.
   If it rejects, then reject.

• And we don’t need to use only $A_{TM}$ to get our contradiction!
Proof $\text{INF}_{\text{TM}}$ undecidable using $M_{\text{HALT}}$

Let $R$ decide
$\text{INF}_{\text{TM}} = \{ <M> \mid M \text{ is TM and } L(M) \text{ is infinite}\}$

Show there is a decider for $M_{\text{HALT}}$

$M_{\text{HALT}}$: "On input $<M,w>$
1. Build $X$ such that $L(X)$ is infinite iff $M$ halts on $w$.

2. Run $R$ on $<X>$. If $R$ accepts, accept; if $R$ rejects, reject."

$X$: "On input $y$
1. Run $M$ on $w$. If $M$ halts on $w$."

Review: Thm 5.4: $\text{EQ}_{\text{TM}} = \{ \langle M_1, M_2 \rangle \mid M_1, M_2 \text{ are TMs and } L(M_1) = L(M_2) \}$ is undecidable.

We show $E_{\text{TM}}$ directly reduces to $\text{EQ}_{\text{TM}}$

Proof: By contradiction.

- Assume that $\text{EQ}_{\text{TM}}$ is decidable, and some TM $M_{\text{EQ}}$ decides it.

- Construct a TM $M_{\text{E}}$ that decides $E_{\text{TM}}$:
  - $M_{\text{E}}(\langle M \rangle)$: “
    1. Run $M_{\text{EQ}}(\langle M, \Phi \rangle)$ // Where $\Phi$ is the TM that accepts empty set
    2. If $M_{\text{EQ}}$ accepts, then accept. If it rejects, then reject. “

- But $E_{\text{TM}}$ is undecidable, a contradiction. So the assumption is false and $\text{EQ}_{\text{TM}}$ is undecidable.
We have examples of languages that are not decidable...

ARE THERE LANGUAGES THAT ARE NOT TURING-RECOGNIZABLE?
Could $A_{TM}$ be not Turing Recognizable?

$A_{TM} = \{<M,w> \mid M \text{ is a TM, } M \text{ accepts } w\}$

- No. We proved it was TR earlier by making this TM:
  - $M_{ATM-Rec}(<M,w>):$
    - Run $M(w)$ and accept if it accepts, reject if it rejects
    - (If it loops that’s ok since this isn’t supposed to be a decider)

- But...what about the complement of $A_{TM}$?
  - Lets assume (for contradiction) that it is also TR, and has a TM $M_{Co-ATM-Rec}$ that recognizes it
  - Could we use $M_{ATM-Rec}$ and $M_{Co-ATM-Rec}$ to build a decider(!) for $A_{TM}$?
Thm.: $\overline{A_{TM}}$ is not Turing-recognizable

Proof by contradiction

- Assume $\overline{A_{TM}}$ is TR, and so some TM $M_{Co-ATM-Rec}$ recognizes it.

- We know that $A_{TM}$ is TR, so let $M_{ATM-Rec}$ be a TM that recognizes it.

- We define $M_{ATM-Decider} (<M, w>):$
  1. Run $M_{ATM-Rec} (<M, w>)$ and $M_{Co-ATM-Rec} (<M, w>)$ in parallel.
  2. If $M_{ATM-Rec}$ accepts then accept. If $M_{Co-ATM-Rec}$ accepts, then reject.

Is $M_{ATM-Decider}$ a decider?

A. No

B. Yes, because $M_{ATM-Rec}$ and $M_{Co-ATM-Rec}$ are deciders

C. Yes, since $\overline{A_{TM}}$ is the complement of $A_{TM}$
Thm.: $\overline{A_{TM}}$ is not Turing-recognizable
Proof by contradiction

• Assume $\overline{A_{TM}}$ is TR, and so some TM $M_{\text{Co-ATM-Rec}}$ recognizes it.

• We know that $A_{TM}$ is TR, so let $M_{\text{ATM-Rec}}$ be a TM that recognizes it.

• We define $M_{\text{ATM-Decider}}(\langle M, w \rangle)$:
  1. Run $M_{\text{ATM-Rec}}(\langle M, w \rangle)$ and $M_{\text{Co-ATM-Rec}}(\langle M, w \rangle)$ in parallel.
  2. If $M_{\text{ATM-Rec}}$ accepts then accept. If $M_{\text{Co-ATM-Rec}}$ accepts, then reject.

• One of the subroutines is guaranteed to accept, by definition of complement, so this machine is a decider for $A_{TM}$.

• But $A_{TM}$ is undecidable, a contradiction. So the assumption is false and $\overline{A_{TM}}$ is not TR.
Theorem 4.22: A language \( L \) is decidable iff \( L \) is both TR and co-TR

- **L Decidable -> TR and co-TR**
  - Easy proof! Just use the decider to recognize \( L \), so it is TR. Then use the decider again to build a recognizer for \( \overline{L} \), by flipping the accept/reject result.

- **L TR and co-TR -> Decidable**
  - We can use the same run-in-parallel method we did for \( A_{TM} \) and \( \overline{A}_{TM} \) to build a decider for \( L \):
    - Run recognizer TM’s for \( L \) and \( \overline{L} \) in parallel
    - If the TM for \( L \) accepts, accept. If the TM for \( \overline{L} \) accepts, reject.
    - One of these must accept, so we can conclude this is a decider, and accepts \( A_{TM} \) .
Status: Language Hierarchy

- **Regular**
- **Context-Free Decidable**
- **Turing-Recognizable**

**CF but not Regular:** \( \{0^n1^n \mid n > 0\} \)
   - By pumping lemma (Regular)

**Decidable but not CF:** \( \{ a^n b^n c^n \mid n > 0\} \)
   - By pumping lemma (CF)

**Turing-Recognizable but Not Decidable:** \( A_{TM} \)
   - By Diagonalization
   - By Reduction

**Not Turing Recognizable:** \( \overline{A_{TM}} \)
   - TH 4.22