Last time:
- The Reducibility Method for proving undecidability and T-unrecognizability
- General reducibility
- Mapping reducibility

Today:
- The Computation History Method for proving undecidability
- The Post Correspondence Problem is undecidable
- Linearly bounded automata
- Undecidable problems about LBAs and CFGs
Remember

To prove some language $B$ is undecidable, show that $A_{\text{TM}}$ (or any known undecidable language) is reducible to $B$. 
Revisit Hilbert’s 10th Problem

Recall $D = \{ (p) | \text{polynomial } p(x_1, x_2, \ldots, x_k) = 0 \text{ has integer solution} \}$

Hilbert’s 10th problem (1900): Is $D$ decidable?

Theorem (1971): No

Proof: Show $A_{TM}$ is reducible to $D$. [would take entire semester]

Do toy problem instead which has a similar proof method.

Toy problem: The Post Correspondence Problem.

Method: The Computation History Method.
Post Correspondence Problem

Given a collection of pairs of strings as dominoes:

\[ P = \left\{ \begin{bmatrix} t_1 \\ b_1 \end{bmatrix}, \begin{bmatrix} t_2 \\ b_2 \end{bmatrix}, \ldots, \begin{bmatrix} t_k \\ b_k \end{bmatrix} \right\} \]

A match is a finite sequence of dominoes in \( P \) (repeats allowed) where the concatenation of the \( t \)'s = the concatenation of the \( b \)'s.

\[
\text{Match} = \begin{bmatrix} t_{i_1} \\ b_{i_1} \end{bmatrix} \begin{bmatrix} t_{i_2} \\ b_{i_2} \end{bmatrix} \ldots \begin{bmatrix} t_{i_l} \\ b_{i_l} \end{bmatrix} \text{ where } t_{i_1} t_{i_2} \ldots t_{i_l} = b_{i_1} b_{i_2} \ldots b_{i_l}
\]

Example: \( P = \left\{ \begin{bmatrix} ab \\ aba \end{bmatrix}, \begin{bmatrix} aa \\ aba \end{bmatrix}, \begin{bmatrix} ba \end{bmatrix}, \begin{bmatrix} abab \\ b \end{bmatrix} \right\} \)

Check-in 10.1

Let \( P_1 = \left\{ \begin{bmatrix} aa \\ aaba \end{bmatrix}, \begin{bmatrix} ba \\ ab \end{bmatrix}, \begin{bmatrix} ba \end{bmatrix} \right\} \)

Does \( P_1 \) have a match?

(a) Yes.
(b) No.
**Defn:** A configuration of a TM is a triple \((q, p, t)\) where
- \(q\) = the state,
- \(p\) = the head position,
- \(t\) = tape contents

representing a snapshot of the TM at a point in time.

Encode configuration \((q, p, t)\) as the string \(t_1q\) where \(t = t_1t_2\) and the head position is on the first symbol of \(t_2\).

Configuration: \((q_3, 6, aaaaaabbbbbb)\)

Encoding as a string: \(aaaaaq_3abbbbbb\)
**TM Computation Histories**

**Defn:** An (accepting) computation history for TM $M$ on input $w$ is a sequence of configurations $C_1, C_2, ..., C_{\text{accept}}$ that $M$ enters until it accepts.

Encode a computation history $C_1, C_2, ..., C_{\text{accept}}$ as the string $C_1 \# C_2 \# \cdots \# C_{\text{accept}}$ where each configuration $C_i$ is encoded as a string.

A computation history for $M$ on $w = w_1 w_2 \cdots w_n$. Here say $\delta(q_0, w_1) = (q_7, a, R)$ and $\delta(q_7, w_2) = (q_8, c, R)$. 

\[
\begin{array}{c}
C_1 \\
\begin{array}{c}
q_0 w_1 w_2 \cdots w_n \\
\# \\
\end{array}
\end{array} \\
\begin{array}{c}
C_2 \\
\begin{array}{c}
aq_7 w_2 \cdots w_n \\
\# \\
\end{array}
\end{array} \\
\begin{array}{c}
C_3 \\
\begin{array}{c}
acq_8 w_3 \cdots w_n \\
\# \\
\end{array}
\end{array} \\
\vdots \\
\begin{array}{c}
C_{\text{accept}} \\
\begin{array}{c}
\cdots q_{\text{accept}} \cdots \\
\# \\
\end{array}
\end{array}
\end{array}
\]
**Defn:** A linearly bounded automaton (LBA) is a 1-tape TM that cannot move its head off the input portion of the tape.

Let \( A_{\text{LBA}} = \{ \langle B, w \rangle \mid \text{LBA } B \text{ accepts } w \} \)

**Theorem:** \( A_{\text{LBA}} \) is decidable

Proof: (idea) If \( B \) on \( w \) runs for long, it must be cycling.

**Claim:** For inputs of length \( n \), an LBA can have only \( |Q| \times n \times |\Gamma|^n \) different configurations.

Therefore, if an LBA runs for longer, it must repeat some configuration and thus will never halt.

Decider for \( A_{\text{LBA}} \):

\[ D_{A_{\text{LBA}}} = \text{“On input } \langle B, w \rangle \text{“} \]

1. Let \( n = |w| \).
2. Run \( B \) on \( w \) for \( |Q| \times n \times |\Gamma|^n \) steps.
3. If has accepted, accept.
4. If it has rejected or is still running, reject.”

must be looping
Let $E_{LBA} = \{ \langle B \rangle | \text{ } B \text{ is an LBA and } L(B) = \emptyset \}$

**Theorem:** $E_{LBA}$ is undecidable

**Proof:** Show $A_{TM}$ is reducible to $E_{LBA}$. Uses the computation history method.

Assume that TM $R$ decides $E_{LBA}$

Construct TM $S$ deciding $A_{TM}$

$S =$ “on input $\langle M, w \rangle$

1. Construct LBA $B_{M,w}$ which tests whether its input $x$ is an accepting computation history for $M$ on $w$, and only accepts $x$ if it is.

2. Use $R$ to determine whether $L(B_{M,w}) = \emptyset$.

3. Accept if no. Reject if yes.”

---

Check-in 10.2

What do you think of the Computation History Method? Check all that apply.

(a) Cool !

(b) Just another theorem.

(c) I’m baffled.

(d) I wish I was in 6.046.
Coffee Break
Recall $PCP = \{ \langle P \rangle | P \text{ has a match} \}$

Theorem: $PCP$ is undecidable

Proof: Show $A_{TM}$ is reducible to $PCP$. Uses the computation history method.

Technical assumption: Match must start with $[t_1 \ b_1]$. Can fix this assumption.

Assume that TM $R$ decides $PCP$

Construct TM $S$ deciding $A_{TM}$

$S =$ “on input $\langle M, w \rangle$

1. Construct PCP instance $P_{M,w}$ where a match corresponds to a computation history for $M$ on $w$.
2. Use $R$ to determine whether $P_{M,w}$ has a match.
3. Accept if yes. Reject if no.”
Constructing $P_{M,w}$

Make $P_{M,w}$ where a match is a computation history for $M$ on $w$.

$[u_1] = \begin{bmatrix} # \\ #q_0 w_1 \cdots w_n # \end{bmatrix}$ (starting domino)

For each $a, b \in \Gamma$ and $q, r \in Q$ where $\delta(q, a) = (r, b, R)$ put $[q \ a \ b \ r]$ in $P_{M,w}$

(Handles right moves. Similar for left moves.)

Ending dominos to allow a match if $M$ accepts:

$\begin{bmatrix} a & q_{\text{accept}} \\ q_{\text{accept}} & q_{\text{accept}} \end{bmatrix}$

Check-in 10.3

What else can we now conclude? Choose all that apply.

(a) $PCP$ is T-unrecognizable.
(b) $\overline{PCP}$ is T-unrecognizable.
(c) Neither of the above.

Illustration:

$w = 223$

$\delta(q_0, 2) = (q_7, 4, R)$

Match completed!

... one detail needed.
ALL_{CFG} is undecidable

Let \( ALL_{CFG} = \{ \langle G \rangle | G \text{ is a CFG and } L(G) = \Sigma^* \} \)

Theorem: \( ALL_{CFG} \) is undecidable

Proof: Show \( A_{TM} \) is reducible to \( ALL_{PDA} \) via the computation history method.

Assume TM \( R \) decides \( ALL_{PDA} \) and construct TM \( S \) deciding \( A_{TM} \).

\( S = \text{"On input } \langle M, w \rangle \text{"} \)

1. Construct PDA \( B_{M,w} \) which tests whether its input \( x \) is an accepting computation history for \( M \) on \( w \), and only accepts \( x \) if it is NOT.

2. Use \( R \) to determine whether \( L(B_{M,w}) = \Sigma^* \).

3. Accept if no. Reject if yes.”

\( B_{M,w} \) operation:

- Nondeterministically push some \( C_i \) and pop to compare with \( C_{i+1} \).
- \( Accept \) if invalid step of \( M \), or if start wrong, or if end isn’t accepting.

Reverse even-numbered \( C_i \) to allow comparing with \( C_{i+1} \) via stack.
Computation History Method is useful for showing the undecidability of problems involving testing for the existence of some object.

\[ D \] Is there an integral solution (to the polynomial equation)?

\[ E_{LBA} \] Is there some accepted string (for the LBA)?

\[ PCP \] Is there a match (for the given dominos)?

\[ ALL_{CFG} \] Is there some rejected string (for the CFG)?

In each case, the object is the computation history in some form.
Quick review of today

1. Defined configurations and computation histories.
2. Gave The Computation History Method to prove undecidability.
3. $A_{LBA}$ is decidable.
4. $E_{LBA}$ is undecidable.
5. $PCP$ is undecidable.
6. $ALL_{CFG}$ is undecidable.
Eliminating the technical assumption

Technical assumption: Match must start with $[t_1/b_1]$.

Fix this assumption as follows.

Let $P = \{ [t_1/b_1], [t_2/b_2], \ldots, [t_k/b_k] \}$ where we require match to start with $[t_1/b_1]$.

Create new $P' = \{ [\tilde{t}_1/\tilde{b}_1], [\tilde{t}_1/\tilde{b}_1], [\tilde{t}_2/\tilde{b}_2], \ldots, [\tilde{t}_k/\tilde{b}_k] \}$

For any string $u = u_1, \ldots, u_k$, let
\[
\star u = \star u_1 \star u_2 \cdots \star u_k \\
u \star = u_1 \star u_2 \cdots \star u_k \\
\star u \star = \star u_1 \star u_2 \cdots \star u_k \star \\
\]

Then let $P' = \{ [*t_1/b_1*], [*t_1/b_1*], [*t_2/b_2*], \ldots, [*t_k/b_k*], [\star$] \}