Today’s lecture

• The diagonal language (Diag) is undecidable
• Identifying and proving other undecidable languages
• Mapping reducibility
• Acceptance problem (A_{TM}) is undecidable
Summary

- RE U coRE is countable
- $P(\{0,1\}^*)$ is uncountable
- There is a language $L$ in $P(\{0,1\}^*) \setminus (\text{RE U coRE})$!
Undecidable Languages

• There are uncountably many undecidable languages!

• In fact, there are uncountably many languages that are not even in RE (or coRE)!

• Questions:
  - Can we find a specific language not in RE or coRE?
  - Can we find interesting languages not in RE or coRE?
  - Is HALT_{TM} undecidable?
Diagonalization

- \([0,1)\) is uncountable because given any list of \(r\) in \([0,1)\)
  - \(0.\overline{0}0100010\ldots\)
  - \(0.\overline{1}1001000\ldots\)
  - \(0.01\overline{1}10001\ldots\)
  - \(0.10\overline{1}10101\ldots\)
  - \(0.1001\overline{1}000\ldots\)

  we can build an \(r=0.\overline{1}0011\ldots\) that is not in the list

- We can make a list of all \(L\) in \(RE: L(M_1), L(M_2), L(M_3)\ldots\)

- Can we build a language not in this list?
A language not in RE

- We want a language $L$ that is
  - different from $L(M_1)$
  - different from $L(M_2)$
  - different from $L(M_3)$
  - ..... 
  - different from $L(M_k)$
  - .....
A language not in RE

- We want a language $L$ that is different from $L(M_1)$
- We want $L$ to be different from $L(M_2)$
- We want $L$ to be different from $L(M_3)$
- We want $L$ to be different from $L(M_k)$
- We want $L$ to be different from $\ldots$

Two numbers are different if they differ at some digit.

Two languages are different if they differ at some string $w$:

Either $w$ in $L(M)$ but not in $L$,
Or $w$ in $L$ but not in $L(M)$.
A language not in RE

- We want a language $L$ that is
  - different from $L(M_1)$ at $<M_1>$
  - different from $L(M_2)$ at $<M_2>$
  - different from $L(M_3)$ at $<M_3>$
  - .....  
  - different from $L(M_k)$ at $<M_k>$
  - .....  
- $L = \{ <M> \mid M \text{ it a TM such that } <M> \text{ is not in } L(M) \}$
A “diagonal” language

- \( \text{Diag=} \{<M>| M \text{ is a TM s.t. } <M> \text{ is not in } L(M) \} \)

- Why is Diag different from \( L(M_k) \)?
  - We need a string \( w \) that belongs to one but not the other
  - Let \( w = <M_k> \)
  - If \( <M_k> \) is in \( L(M_k) \), then
    - Not (Not “\( <M_k> \) is in \( L(M_k) \)"
    - “\( <M_k> \) is not in \( L(M_k) \)” is false
    - \( <M_k> \) is not in Diag
  - If \( w \) is not in Diag, then \( w \) is in \( L(M_k) \)
A “diagonal” language

- \text{Diag}=\{<M>| M \text{ is a TM s.t. } <M> \text{ is not in } L(M) \}\}

Why is Diag different from \(L(M_k)\)?

- We need a string \(w\) that belongs to one but not the other
- Let \(w = <M_k>\)
- If \(<M_k>\) is in \(L(M_k)\), then
  - Not (Not "<M_k> is in \(L(M_k)\)"")
  - "<M_k> is not in \(L(M_k)\)" is false
  - \(<M_k>\) is not in \text{Diag}
- If \(w\) is not in \text{Diag}, then \(w\) is in \(L(M_k)\)

Question: What can you tell about \text{Diag}?

A) Diag is in RE, but not in coRE
B) Diag is coRE, but not in RE
C) Diag is neither in RE nor in coRE
D) Diag is decidable
E) I don’t know
Diag is in coRE

- $\text{Diag}=\{<M> \mid M \text{ is a TM s.t. } <M> \text{ is not in } L(M) \}$

- Here is a recognizer for $\text{Diag}$

  $M_{\text{diag}}(w) =$

  1) Check if $w = <M>$ for some TM $M$. If not, accept.
  2) Parse $w$ as $<M>$ for some TM $M$
  3) Run $M$ on input $w$
  4) If $M(w)$ accepts, then accept, else reject.
Diag is in coRE

- Diag = \{ <M> \mid M \text{ is a TM s.t. } <M> \text{ is not in } L(M) \}
- Here is a recognizer for \textbf{Diag}:

  \begin{align*}
  M_{\text{diag}}(w) = \\
  1) \text{ Check if } w = <M> \text{ for some TM } M. \text{ If not, accept.} \\
  2) \text{ Parse } w \text{ as } <M> \text{ for some TM } M \\
  3) \text{ Run } M \text{ on input } w \\
  4) \text{ If } M(w) \text{ accepts, then accept, else reject.}
  \end{align*}

Is \( M_{\text{diag}} \) a decider?

A) Yes, because \( L(M_{\text{diag}}) = \text{Diag} \)

B) No, because it can loop at step 1

C) No, because it can loop at step 2

D) No, because it can loop at step 3

E) I don’t know
Diag is in coRE

- Diag = \{ <M> \mid M \text{ is a TM s.t. } <M> \notin L(M) \}
- Here is a recognizer for Diag:

  \[ M_{\text{diag}}(w) = \]

  1) Check if \( w = <M> \) for some TM \( M \). If not, accept.
  2) Parse \( w \) as \( <M> \) for some TM \( M \)
  3) Run \( M \) on input \( w \)
  4) If \( M(w) \) accepts, then accept, else reject.

What can you say about Diag:

A) Diag is decidable
B) Diag is in RE, but not coRE
C) Diag is in coRE, but not RE
D) Diag is neither in RE nor coRE
E) I don’t know
Summary

- Diag is in coRE because $L(M_{\text{diag}}) = \text{Diag}$
- Diag is not in RE because $\text{Diag} \neq L(M_k)$ for all TM $M_k$
- Diag is undecidable
- Diag is in RE, but not coRE
- So far: We have found specific languages not in RE or coRE.
- Can we find more interesting examples?
  - What about $\text{HALT}_{\text{TM}} = \{<M,w> | M(w) \text{ terminates}\}$?
  - What about $\text{A}_{\text{TM}} = \{<M,w> | M(w) \text{ accepts}\}$?
Tool: Reducibility

- Reading: Chapter 5
- We will cover the material in a different order
  - Focus on what the text calls “mapping reducibility”
  - See textbook Ch 5.3
- Why?
  - More intuitive and easier to use than general reductions
  - Most commonly used in computer science
Idea

• Goal: identify and prove undecidable languages
• Let U be the set of undecidable languages
• Define a “reduction” operation such that U is closed under reduction
• Show that $\text{HALT}_{TM}$, $A_{TM}$, can be obtained from Diag by reduction
• Conclusion: $\text{HALT}_{TM}$, $A_{TM}$ are undecidable!
• Bonus: mapping reductions can be used also to study languages that are not in RE or coRE
Computable Functions

• Let $f: \Sigma^* \rightarrow \Sigma^*$ be a function from strings to strings

• Function $f$ is (Turing-)computable if there is a (deterministic) TM $M$ such that for every input string $w$,
  - $M(w)$ terminates
  - Upon termination, the tape contains the string $f(w)$

• Remarks:
  - We don’t care about accepting or rejecting the input
  - Tape content is used to describe computations with output in $\Sigma^*$
Mapping reducibility

• Let $A, B \subseteq \Sigma^*$ be any two languages

• Definition: $A$ is map-reducible to $B$ (written “$A \leq_m B$”) if there is a function $f$ such
  1) The function $f$ is computable (by a TM $M$)
  2) For all $w \in A$, we have $f(w) \in B$
  3) For all $w$ not in $A$, we have that $f(w)$ is not in $B$

• Equivalently: $(w \in A) \iff (f(w) \in B)$
Reductions and Undecidability

• Assume
  – $A \leq_m B$, i.e., there is a map reduction from $A$ to $B$
  – $A$ is undecidable

• Claim: $B$ is also undecidable

• Proof:
  – Let $F$ be a TM computing a map reduction from $A$ to $B$
  – WTS: if $A$ is undecidable, then $B$ is undecidable
  – Equivalently: if $B$ is decidable, then $A$ is decidable
Reductions and undecidability

- Let $F$ be a TM computing a map reduction from $A$ to $B$
- **WTS:** if $B$ is decidable, then $A$ is decidable
  - Let $M$ is a decider for $B$
  - Goal: build a decider $M'$ for $A$
- $M'(w)$:
  1) Compute $w' = F(w)$
  2) Run $M(w')$
  3) If $M(w')$ accepts, then accept $w$, otherwise reject $w$
Reductions and undecidability

- Let F be a TM computing a map reduction from A to B
- \textbf{WTS:} if B is decidable, then A is decidable
  - Let M is a decider for B
  - Goal: build a decider M' for A
- M'(w):
  1) Compute \( w' = F(w) \)
  2) Run M(w')
  3) If M(w') accepts, then accept w, otherwise reject w

\begin{enumerate}
\item M' is a decider
\item A) Yes
\item B) No, because it may loop in step 1
\item C) No, because it may loop in step 2
\item D) It depends on w
\item E) I don't know
\end{enumerate}
Reductions and undecidability

- Let $F$ be a TM computing a map reduction from $A$ to $B$
- WTS: if $B$ is decidable, then $A$ is decidable
  - Let $M$ is a decider for $B$
  - Goal: build a decider $M'$ for $A$

- $M'(w)$:
  1) Compute $w' = F(w)$
  2) Run $M(w')$
  3) If $M(w')$ accept, then accept $w$, otherwise reject $w$

What can you say about $L(M')$?
- A) $L(M') = A$
- B) $L(M') \subset A$
- C) $L(M') \supset A$
- D) None of the above
- E) I don't know
Proof of $A \subseteq L(M')$

- Let $F: A \leq_m B$, and $L(M) = B$

- $M'(w)$:
  1) Compute $w' = F(w)$
  2) Run $M(w')$, and accept iff $M(w')$ accept

- Assume $w \in A$, then
  - $F(w) \in B$ (by definition of reduction)
  - $M(w')$ accepts
  - $M'(w)$ accepts
Proof of $A \supset L(M')$

- Let $F: A \leq_m B$, and $L(M) = B$

- $M'(w)$:
  1) Compute $w' = F(w)$
  2) Run $M(w')$, and accept iff $M(w')$ accept

- Assume $w$ is not in $A$, then
  - $F(w)$ is not in $B$ (by definition of reduction)
  - $M(w')$ rejects
  - $M'(w)$ rejects
A_{TM} is undecidable

- We give a reduction F from Diag to A_{TM}

- F(w):
  - Parse w as <M> for some TM M
  - Let w' = <M,w>
  - Output w'

- Assume w = <M> is in Diag
- Then w is not in L(M)
- Then <M,w> is not in A_{TM}.

- So, w' is in A_{TM}
$A_{TM}$ is undecidable

- We give a reduction $F$ from $A=\text{Diag}$ to $B=A_{TM}$

- $F(w)$:
  - Parse $w$ as $<M>$ for some TM $M$
  - Let $w' = <M, w>$
  - Output $w'$

- Assume $w = <M>$ is in $\text{Diag}$
- Then $w$ is not in $L(M)$
- Then $<M, w>$ is not in $A_{TM}$.
- So, $w'$ is in $A_{TM}$

Claim: $F$ is a reduction from $A$ to $B$

A) Yes, because if $w \in A$, then $F(w) \in B$
B) Yes, but the proof is not complete
C) No, $F$ is not a valid reduction
D) I don't know
\( A_{TM} \) is undecidable (cont.)

- We give a reduction \( F \) from \( A = \text{Diag} \) to \( B = A_{TM} \)

- \( F(w) \):
  - Parse \( w \) as \(<M>\) for some TM \( M \)
  - Let \( w' = <M, w> \)
  - Output \( w' \)

- Assume \( w \) is not in Diag. There are two cases:
  - \( W = <M> \) for some TM such that \( w \) is in \( L(M) \).
    - Then \( <M, w> \) is in \( A_{TM} \), i.e., \( w' \) is not in \( A_{TM} \)
  - \( W \) does not parse as \(<M>\). What is the output of \( F \) if parsing fails?
$A_{\text{TM}}$ is undecidable (fixed)

- We give a reduction $F$ from $A=\text{Diag}$ to $B=A_{\text{TM}}$

- $F(w)$:
  - Parse $w$ as $<M>$ for some TM $M$
    - If parsing fails, then output $<M_a, \text{“I love CSE105”}>$, where $M_a(x) = \text{accept}$
  - Let $w' = <M,w>$
  - Output $w'$

- Assume $w$ is not in Diag. There are two cases:
  - $W=<M>$ for some TM such that $w$ is in $L(M)$.
    - Then $<M,w>$ is in $A_{\text{TM}}$, i.e. $w'$ is not in $A_{\text{TM}}$
  - $W$ does not parse as $<M>$. What is the output of $F$ if parsing fails?
For next Time

- Try to prove that $\text{HALT}_{\text{TM}}$ is undecidable
- Reading: Sipser Chapter 5
- HW6 due tomorrow night!