Decidability

- Recall: Random Access Machine (RAM)
  - program is a finite sequence of instructions
  - input and output is a bit string, written on special tapes
  - random access to an unbounded number of memory cells
- We say that a RAM $M$ halts on input $x \in \{0, 1\}^*$ if given $x$ as input, $M$ halts after a finite number of steps
  - no restrictions are made on the running time, the number of memory cells used, or the sizes of the numbers stored in memory
- We say that a RAM $M$ halts on all inputs if it halts on all inputs $x \in \{0, 1\}^*$
• We say a function $f : \{0, 1\}^* \to \{0, 1\}^*$ is *computable* if there is a RAM that computes $f$ (in particular, $M$ must halt on all inputs).

• If a RAM $M$ computes the characteristic function of a language $L$, then we say $M$ *decides* $L$.

• We say that a language $L \subseteq \{0, 1\}^*$ is *decidable* if some RAM decides it.
  – Analogous to P

• Simple fact: $L$ is decidable $\iff \overline{L}$ is decidable.
We say that a language $L$ is \emph{recursively enumerable} if there is a decidable language $L'$ such that

$$\forall x \in \{0, 1\}^* : x \in L \iff \exists w \in \{0, 1\}^* : \langle x, w \rangle \in L'$$

- Sipser: Turing recognizable
- Analogous to \textbf{NP}
Existence of undecidable languages:

- There are only countably many RAM’s
- There are uncountably many languages
- \( \therefore \) undecidable languages exist

A specific undecidable language:

- We say RAM \( M \) accepts a string \( x \) if \( M \) halts and outputs 1 on input \( x \)

\[ A_{RAM} := \left\{ \langle M, x \rangle : M \text{ is a RAM}, \quad x \in \{0, 1\}^*, \quad M \text{ accepts } x \right\} \]
Theorem: $A_{\text{RAM}}$ is undecidable

Proof:

- Suppose it were decidable
- Let $H$ be a RAM that decides it:
  
  \[
  H(\langle M, x \rangle) = \begin{cases} 
  1 & \text{if } M \text{ accepts } x \\
  0 & \text{otherwise} 
  \end{cases}
  \]

- Construct a new RAM $D$ as follows:
  
  on input $\langle M \rangle$:
  
  output $1 - H(\langle M, \langle M \rangle \rangle)$

- $D$ always halts and always outputs 0 or 1
- $D(\langle D \rangle) = 1 - H(\langle D, \langle D \rangle \rangle) = 1 - D(\langle D \rangle)$
Theorem: $A_{\text{RAM}}$ is recursively enumerable

Proof:

- A “witness” $w$ for $\langle M, x \rangle$ is a bound on the running time of $M$ on input $x$
- To verify a witness $w$
  - just run $M$ on input $x$ for up to $w$ steps
  - if $M$ halts and outputs 1 within $w$ steps, then output 1, and output 0 otherwise
Theorem:

• a language $L$ is decidable $\iff$ both $L$ and $\overline{L}$ are recursively enumerable

Proof:

• $\Rightarrow$: clear

• $\Leftarrow$:
  - let $W_1$ be the set of witnesses for $x \in L$, and let $W_2$ be the set of witnesses for $x \in \overline{L}$
  - enumerate all strings $\{0, 1\}^*$, testing for membership in $W_1$ and $W_2$
  - eventually, one string will be in either $W_1$ or $W_2$

Corollary: $\overline{A_{RAM}}$ is not recursively enumerable
Recursive enumerability: other characterizations

Some terminology:

- We say $M$ recognizes $L$ if:
  - $x \in L \implies M$ accepts $x$
  - $x \notin L \implies M$ does not accept $x$ (it may halt and output something $\neq 1$, or it may go into an infinite loop)

Notation: $L(M) = \{x : M$ accepts $x\}$
Theorem:

- $L$ is recursively enumerable $\iff$ some RAM recognizes $L$

Proof:

- $\Rightarrow$ build a RAM that enumerates all possible witness, testing each
- $\Leftarrow$: the witness is a bound on the running time
Enumerators:

- Let $c : \{0, 1\}^* \rightarrow \{0, 1\}^*$ be some simple, prefix-free encoding function
- We say $M$ enumerates $L$, if the following holds:
  - if we allow $M$ to run (with no input) forever, it writes to its output tape
    
    $$c(x_1)c(x_2)c(x_3)\cdots,$$

    and $L = \{c(x_i) : i = 1, 2, 3, \ldots \}$
Theorem:

- $L$ is recursively enumerable $\iff$ some RAM enumerates $L$

Proof:

- $\Rightarrow$: build a RAM that enumerates all pairs $(x, w)$, and outputs $c(x)$ if $w$ is a witness for $x$
- $\Leftarrow$: the witness is a bound on the running time needed to generate $c(x)$
Reducibility

Reductions:

- Suppose $L_1, L_2$ are languages
- We say $L_1$ is reducible to $L_2$, if there is a computable function $f : \{0, 1\}^* \rightarrow \{0, 1\}^*$, such that $x \in L_1 \iff f(x) \in L_2$
- Notation: $L_1 \leq L_2$

Simple facts:

- If $L_1 \leq L_2$ and $L_2$ is decidable, then $L_1$ is decidable
- If $L_1 \leq L_2$ and $L_1$ is undecidable, then $L_2$ is undecidable
More undecidable problems

Theorem:

- The language

\[ \text{HALT}_{\text{RAM}} := \{ \langle M, x \rangle : \text{RAM } M \text{ halts on input } x \} \]

is undecidable

Proof:

- Reduction: \( A_{\text{RAM}} \leq \text{HALT}_{\text{RAM}} \)
- Map \( \langle M, x \rangle \) to \( \langle M', x \rangle \), where \( M' \) is the RAM:
  - on input \( x' \):
    - run \( M \) on input \( x' \) until it halts
    - if and when \( M \) halts with an output \( y \):
      - if \( y = 1 \) then halt
      - else go into an infinite loop
Theorem:

- The language 
  \[ E_{\text{RAM}} := \{ \langle M \rangle : M \text{ is a RAM and } L(M) = \emptyset \} \]
  is undecidable

Proof:

- Reduction: \( A_{\text{RAM}} \leq \overline{E_{\text{RAM}}} \)
- Map \( \langle M, x \rangle \) to \( \langle M' \rangle \), where \( M' \) is the RAM:
  on input \( x' \):
  
  if \( x' = x \) // \( x \) is “hardwired” into \( M' \)
  
  then run \( M \) on input \( x \)
  
  else output 0 and halt

- Verify: \( M \) accepts \( x \) \( \iff \) \( L(M') \neq \emptyset \)
Theorem:

- The language

\[
EQ_{\text{RAM}} := \{ \langle M_1, M_2 \rangle : M_1, M_2 \text{ are RAM’s and } L(M_1) = L(M_2) \}
\]

is undecidable

Proof:

- Reduction: \( E_{\text{RAM}} \leq EQ_{\text{RAM}} \)
- Map \( \langle M \rangle \) to \( \langle M, M_0 \rangle \), where \( M_0 \) is the RAM:
  
  on input \( x \):
  
  output 0 and halt
Theorem:

- The language

\[ REG_{\text{RAM}} := \{ \langle M \rangle : L(M) \text{ is regular} \} \]

is undecidable

Proof:

- Reduction: \( A_{\text{RAM}} \leq REG_{\text{RAM}} \)
- Map \( \langle M, x \rangle \) to \( \langle M' \rangle \), where \( M' \) is the RAM:
  
  on input \( x' \):
  
  - if \( x' \in \{0^n1^n : n \geq 0 \} \)
    
    then output 1 and halt
  
  - else run \( M \) on input \( x \)

- Observe: if \( M \) accepts \( x \), then \( L(M') = \{0, 1\}^* \)
  
  otherwise, \( L(M') = \{0^n1^n \} \)
Theorem (Rice’s Theorem):

- Any non-trivial property of the language accepted by a RAM is undecidable
- More precisely: let \( P \) be a language consisting of RAM descriptions \( \langle M \rangle \), such that
  - \( P \) is non-trivial: \( P \) contains some, but not all descriptions
  - membership in \( P \) depends only on the language accepted by the RAM:
    \[
    L(M_1) = L(M_2) \Rightarrow (\langle M_1 \rangle \in P \iff \langle M_2 \rangle \in P)
    \]
- Then \( P \) is undecidable
Proof:

- Let $M_0$ be a RAM with $L(M_0) = \emptyset$
- We may assume that $\langle M_0 \rangle \not\in P$ (otherwise, use $\overline{P}$ in place of $P$)
- Let $M_1$ be any RAM with $\langle M_1 \rangle \in P$
- Reduction: $A_{\text{RAM}} \leq P$
Proof (cont’d):

- Map \( \langle M, x \rangle \) to \( M' \), where \( M' \) is the RAM:
  
  on input \( x' \):
  
  run \( M \) on input \( x \)
  
  if and when \( M \) halts with an output \( y \):
    
    if \( y = 1 \) then
      
      run \( M_1 \) on input \( x' \)
    
    else
      
      output 0 and halt

- if \( M \) accepts \( x \) then \( L(M') = L(M_1) \), and \( \langle M_1 \rangle \in P \Rightarrow \langle M' \rangle \in P \)

- if \( M \) does not accept \( x \), then \( L(M') = \emptyset = L(M_0) \), and \( \langle M_0 \rangle \notin P \Rightarrow \langle M' \rangle \notin P \)