Recursive languages

- A language $L \subseteq \Sigma^*$ is recursively enumerable if $L = L(M)$, for some Turing machine $M$.

$$w \rightarrow M \rightarrow \begin{cases} 
\text{yes,} & \text{if } w \in L \\
\text{no,} & \text{if } w \notin L \\
M \text{ does not halt,} & \text{if } w \notin L
\end{cases}$$

- A language $L \subseteq \Sigma^*$ is recursive if $L = L(M)$ for some Turing machine $M$ that halts on all inputs $w \in \Sigma^*$.

$$w \rightarrow M \rightarrow \begin{cases} 
\text{yes,} & \text{if } w \in L \\
\text{no,} & \text{if } w \notin L
\end{cases}$$

- **Lemma.** $L$ is recursive iff both $L$ and $\bar{L} = \Sigma^* \setminus L$ are recursively enumerable.

Enumerating languages

- An enumerator is a Turing machine $M$ with extra output tape $T$, where symbols, once written, are never changed.
- $M$ writes to $T$ words from $\Sigma^*$, separated by $\$.
- Let $G(M) = \{ w \in \Sigma^* \mid w \text{ is written to } T \}$.

Some results

- **Lemma.** For any finite alphabet $\Sigma$, there exists a Turing machine that generates the words $w \in \Sigma^*$ in canonical ordering (i.e., $w < w' \iff |w| < |w|$ or $|w| = |w|$ and $w \prec_{\text{lex}} w'$).
- **Lemma.** There exists a Turing machine that generates all pairs of natural numbers (in binary encoding).
  \begin{proof}
  Use the ordering $(0,0)$, $(1,0)$, $(0,1)$, $(2,0)$, $(1,1)$, $(0,2)$, \ldots
  \end{proof}
- **Proposition.** $L$ is recursively enumerable iff $L = G(M)$, for some Turing machine $M$.

Computing functions

- **Unary encoding of natural numbers:** $i \in \mathbb{N} \mapsto ||...||_i = |i|$
  (binary encoding would also be possible)
- $M$ computes $f : \mathbb{N}^k \rightarrow \mathbb{N}$ with $f(i_1, \ldots, i_k) = m$:
  - Start: $|i_1 |0|^k 0 ... |^k$
  - End: $|m$
- **$f$ partially recursive:**
  $$i_1, \ldots, i_k \rightarrow M \rightarrow \begin{cases} 
\text{halts with } f(i_1, \ldots, i_k) = m, \\
\text{does not halt, i.e., } f \text{ undefined.}
\end{cases}$$
- **$f$ recursive:**
  $$i_1, \ldots, i_k \rightarrow M \rightarrow \text{halts with } f(i_1, \ldots, i_k) = m.$$
Turing machines codes

- May assume

\[ M = (Q, \{0, 1\}, \{0, 1, \#\}, \delta, q_1, \#, \{q_2\}) \]

- Unary encoding

\[ 0 \mapsto 0, 1 \mapsto 00, \# \mapsto 000, L \mapsto 0, R \mapsto 00 \]

- \( \delta(q_i, X) = (q_j, Y, R) \) encoded by

\[ 0^{i}10...010^{i}10...0 \]

- \( \delta \) encoded by

\[ 111 \text{ code}_1, 11 \text{ code}_2 11...11 \text{ code}_r, 111 \]

- Encoding of Turing machine \( M \) denoted by \( \langle M \rangle \).

Numbering of Turing machines

- Lemma. There exists a Turing machine that generates the natural numbers in binary encoding.

- Lemma. The language of Turing machine codes is recursive.

- Proposition. There exists a Turing machine \( \text{Gen} \) that generates the binary encodings of all Turing machines.

- Theorem. There exist a bijection between the set of natural numbers, Turing machine codes and Turing machines.

\[
\text{Gen} \quad \rightarrow \quad \langle M \rangle \quad \rightarrow \quad \text{Equality test} \quad + \text{counter} \quad \rightarrow \quad \text{number } n
\]

\[
\text{Gen} \quad \rightarrow \quad \text{Count up to } n \quad \rightarrow \quad \langle M \rangle \quad \rightarrow \quad M
\]

Diagonalization

- Let \( w_i \) be the \( i \)-th word in \( \{0, 1\}^* \) and \( M_j \) the \( j \)-th Turing machine.

- Table \( T \) with \( t_{ij} = \begin{cases} 1, & \text{if } w_i \in L(M_j) \\ 0, & \text{if } w_i \notin L(M_j) \end{cases} \)

\[
\begin{array}{cccccc}
  j & 1 & 2 & 3 & 4 & \ldots \\
  i & 1 & 0 & 1 & 1 & 0 & \ldots \\
  & 2 & 1 & 1 & 0 & 1 & \ldots \\
  & 3 & 0 & 0 & 1 & 0 & \ldots \\
  \vdots & \vdots & \vdots & \vdots & \vdots & \ddots
\end{array}
\]

- Diagonal language \( L_d = \{ w_i \in \{0, 1\}^* \mid w_i \notin L(M_i) \} \).

- Theorem. \( L_d \) is not recursively enumerable.

- Proof: Suppose \( L_d = L(M_k) \), for some \( k \in \mathbb{N} \). Then

\[ w_k \in L_d \iff w_k \notin L(M_k), \]

contradicting \( L_d = L(M_k) \).
Universal language

- $\langle M, w \rangle$: encoding $\langle M \rangle$ of $M$ concatenated with $w \in \{0, 1\}^*$.

- Universal language
  \[ L_u = \{ \langle M, w \rangle \mid M \text{ accepts } w \} \]

- Theorem. $L_u$ is recursively enumerable.
- A Turing machine $U$ accepting $L_u$ is called universal Turing machine.
- Theorem (Turing 1936). $L_u$ is not recursive.

Decision problems

- Decision problems are problems with answer either yes or no.
- Associate with a language $L \subseteq \Sigma^*$ the decision problem $D_L$
  
  Input: $w \in \Sigma^*$
  
  Output: \[
  \begin{cases}
  \text{yes,} & \text{if } w \in L \\
  \text{no,} & \text{if } w \notin L
  \end{cases}
  \]
  and vice versa.

- $D_L$ is decidable (resp. semi-decidable) if $L$ is recursive (resp. recursively enumerable).
- $D_L$ is undecidable if $L$ is not recursive.

Reductions

- A many-one reduction of $L_1 \subseteq \Sigma_1^*$ to $L_2 \subseteq \Sigma_2^*$ is a computable function $f : \Sigma_1^* \rightarrow \Sigma_2^*$ with $w \in L_1 \iff f(w) \in L_2$.

- Proposition. If $L_1$ is many-one reducible to $L_2$, then
  
  1. $L_1$ is decidable if $L_2$ is decidable.
  2. $L_2$ is undecidable if $L_1$ is undecidable.

Post’s correspondence problem

- Given pairs of words
  \[(v_1, w_1), (v_2, w_2), \ldots, (v_k, w_k)\]
  over an alphabet $\Sigma$, does there exist a sequence of integers $i_1, \ldots, i_m, m \geq 1$, such that
  \[v_{i_1} \ldots v_{i_m} = w_{i_1} \ldots w_{i_m}.\]

- Example
  \[
  \begin{array}{|c|c|c|}
  \hline
  i & v_i & w_i \\
  \hline
  1 & 11 & 111 \\
  2 & 1011 & 10 \\
  3 & 10 & 0 \\
  \hline
  \end{array}
  \Rightarrow v_2 v_1 v_3 = w_2 w_1 w_3 = 101111110
  \]

- Theorem (Post 1946). Post’s correspondence problem is undecidable.