1 Hierarchy of Languages

Are there Turing-acceptable (recursively enumerable) languages that are not Turing-decidable (recursive)? Are there Turing-undecidable (Turing-unrecognizable) languages?
Decidable Languages.

Definition 1 A TM $M$ accepts a language $L \subseteq \Sigma^*$, if
\[ L = \{ w : M(w) = \text{accept} \}. \]

Definition 2 A TM $M$ decides a language $L \subseteq \Sigma^*$, if $\forall w \in \Sigma^*$,
\[ M(w) = \begin{cases} 
\text{accept} & \text{if } w \in L; \\
\text{reject} & \text{if } w \notin L;
\end{cases} \]

Definition 3 A language $L$ is Turing-acceptable, or recursively enumerable if there is a TM $M$ which accepts $L$.

Definition 4 A language $L$ is Turing-decidable, or recursive if there is a TM $M$ which decides $L$.
Examples of decidable Languages

Proposition 1 $A_{\text{DFA}} = \{\langle B, w \rangle : B \text{ is a DFA that accepts } w \}$ is a decidable language.

Proof. A description of a TM $M$ which decides $A_{\text{DFA}}$.

for input $\langle B, w \rangle$

run $B$ on $w$;

if $B$ accepts $w$, $M$ accepts $\langle B, w \rangle$;
else $M$ rejects $\langle B, w \rangle$. \(\Box\)

Proposition 2 $A_{\text{NFA}} = \{\langle B, w \rangle : B \text{ is an NFA that accepts } w \}$ is a decidable language.

Proof. A description of a TM $M$ which decides $A_{\text{NFA}}$.

for input $\langle B, w \rangle$

construct a DFA $C$ which is equivalent to $B$;
run $C$ on $w$;
if $C$ accepts $w$, $M$ accepts $\langle B, w \rangle$;
else $M$ rejects $\langle B, w \rangle$. \(\Box\)
**Proposition 3** \( A_{\text{REX}} = \{ \langle R, w \rangle : R \text{ is a regular expression that generates } w \} \) is a decidable language.

**Proof.** A description of a TM \( M \) which decides \( A_{\text{REX}} \).

for input \( \langle R, w \rangle \)

construct a DFA \( C \) which accepts the language generated by \( R \);
run \( C \) on \( w \);
if \( C \) accepts \( w \), \( M \) accepts \( \langle B, w \rangle \);
else \( M \) rejects \( \langle R, w \rangle \).  \( \square \)

**Proposition 4** \( E_{\text{DFA}} = \{ \langle B \rangle : B \text{ is a DFA and } L(B) = \emptyset \} \) is a decidable language.

**Proposition 5** \( EQ_{\text{DFA}} = \{ \langle A, B \rangle : A \text{ and } B \text{ are DFAs s.t. } L(A) = L(B) \} \) is a decidable language.
Proposition 6 \( E_{\text{CFG}} = \{ \langle G \rangle : G \text{ is a CFG and } L(G) = \emptyset \} \) is a decidable language.

**Proof.** If \( L(G) \neq \emptyset \), then there is a parse tree whose yield is a string of terminals and whose height \( \leq |V| \). This is because every parse tree of height \( \geq |V| \) has a path with two nodes labeled with the same variable. Then, a smaller parse tree can be constructed by excising the part of the tree between the nodes with the same label. The process can be repeated until a parse tree of height \( \leq |V| \) is constructed with the terminal yield. Thus, to learn if \( L(G) \) is empty, it is sufficient to search through all parse trees of height \( \leq |V| \). If none of them yields a terminal string, the language is empty. \( \Box \)

Proposition 7 \( A_{\text{CFG}} = \{ \langle G, w \rangle : G \text{ is a CFG that generates string } w \} \) is a decidable language.

**Proof Idea.** Use the Chomsky normal form (all rules are of the form \( A \rightarrow BC, A \rightarrow a, \text{ and } A \rightarrow \epsilon \), where \( B, C \) are not start variables). Convert \( G \) into \( G' \) with \( L(G) = L(G') \).

Generate all derivations of length at most \( 2|w| + 1 \). If \( w \) is generated, accept, else reject. \( \Box \)
3 Existence of Turing-unrecognizable Languages.

Chuck Norris counts to infinity ... twice

The main claim of this section is that there are languages that are not accepted by any Turing machine. The proof to this claim is obtained by the method invented by Georg Ferdinand Ludwig Philipp Cantor. The method is used for counting, or measuring infinite sets. It turns out there are different kinds of infinite sets: although there are infinite number of TMs and an infinite number of languages, still there are more languages than Turing machines.

Definition 5 Given two sets $A$ and $B$, a function

$$f : A \rightarrow B$$

is called one-to-one, if

$$\forall x, y \in A, x \neq y, f(x) \neq f(y).$$

Definition 6 Given two sets $A$ and $B$, a function

$$f : A \rightarrow B$$

is called onto, if

$$\forall y \in B, \exists x \in A, f(x) = y.$$

Definition 7 Two sets $A$ and $B$ are called of the same size, or of the same cardinality, if there exists a one-to-one, onto function $f : A \rightarrow B$. 
Proposition 8

Let $\mathcal{N}$ be the set of all non-negative integers, and let $\mathcal{N}_e$ be the set of all even non-negative integers. Then $\mathcal{N}$ and $\mathcal{N}_e$ are of the same cardinality.

**Proof.** Consider function $f : \mathcal{N} \rightarrow \mathcal{N}_e$ defined by

$$f(i) = 2i, \quad i \in A.$$  

Definition 8 *If a set $A$ is of the same cardinality as $\mathcal{N}$, it is called countable; otherwise, the set is called uncountable.*

Theorem 1 *Let $\mathcal{Q}$ be the set of positive rational numbers. Then $\mathcal{Q}$ is countable.*

**Proof.** We consider a set $\mathcal{Q}'$, comprised of all rationales $p/q$, where $p, q = 1, 2, 3, \ldots$. Clearly, every positive rational number $r$ is repeated in $\mathcal{Q}'$ an infinite number of times. We prove that even $\mathcal{Q}'$ is
countable. The correspondence \( f \) is defined by the formula

\[
f\left( \frac{p}{q} \right) = \frac{(p + q - 1)(p + q - 2)}{2} + p
\]

and is illustrated in the figure below.

![Figure](image_url)

**Theorem 2** Let \( \text{SUB}_f \) be the set of all finite subsets of \( \mathbb{N} \). Then \( \text{SUB}_f \) is countable.

**Theorem 3** The set \( \mathcal{R} \) of all infinite 0,1-sequences is not countable.

**Proof.** Suppose that contrary to the claim, \( \mathcal{R} \) is countable and let \( f : \mathbb{N} \rightarrow \mathcal{R} \) be a one-to-one correspondence between these sets. Denote \( f_i(k) \) to be the \( i \)th entry of the sequence \( f(k) \) (\( i, k > 0 \)). Now, consider the sequence

\[
s = \{1 - f_1(1), 1 - f_2(2), 1 - f_3(3), \ldots, 1 - f_k(k), \ldots\}.
\]

By construction, the \( k \)th entry of \( s \) is different from \( f_k(k) \), thus \( s \) is different from every \( f(k) \) (\( k = 1, 2, \ldots \)). This proves that \( f \) is not a mapping onto; consequently, \( \mathcal{R} \) is not countable. \( \Box \)
Remark 1 The argument used for the proof of the previous theorem is called the diagonalization argument.

Application of Cantor’s set theory to Turing computations.

Theorem 4 There are Turing-unrecognizable languages.

Proof. We may assume that all states of any TM are taken from a countable set of states

\[ \{q_1, q_2, q_3, \ldots \} \].

We may also assume that all letters of the tape alphabet of every TM are taken from a countable set of letters

\[ \{\gamma_1, \gamma_2, \gamma_3, \ldots \} \].

Since every Turing machine can be encoded as a finite string over a finite alphabet

\[ Q \cup \Sigma \cup \Gamma \cup \{R, L\} \],

the set of all Turing machines is countable.

On the other hand, every language over \( \Sigma \) is uniquely described as an infinite 0,1-sequence. Indeed, let the countable set of all finite strings over \( \Sigma \) be

\[ w_1, w_2, w_3, \ldots \].

Then, a language \( L \subseteq \Sigma^* \) is given by a sequence

\((\delta_1, \delta_2, \ldots)\), where \( \delta_i = 1 \) (resp. \( \delta_i = 0 \)) if \( w_i \in L \) (res. \( w_i \notin L \)).

Since the set of 0,1-sequences is uncountable and the set of Turing machines is countable, we obtain the result. \( \square \)
4 Power sets

**Definition 9** For a set $A$, $2^A$ denotes the set of all subsets of $A$.

**Theorem 5** For any set $A$, no one-to-one mapping $f : A \rightarrow 2^A$ maps $A$ onto $2^A$.

**Proof.** Suppose the claim of the theorem is incorrect, and a mapping $f : A \rightarrow 2^A$ is a one-to-one mapping from $A$ onto $2^A$. For every element $a \in A$, we say that

- $a$ is friendly if $a \in f(a)$; and
- $a$ is hostile if $a \notin f(a)$.

Denote $T$ the set of all hostile elements of $A$ with respect to the mapping $f$. Since $f$ is a mapping from $A$ onto $2^A$, there is an element $t \in A$ for which $f(t) = T$. Such an element is unique, since $f$ is a one-to-one mapping. (*Is there any difference if $T = \emptyset$?*)

If $t$ were a friendly element, then $t \in f(t) = T$. This contradicts the definition of $T$ as the set of all hostile elements in $A$. Thus, $t$ cannot be friendly.

On the other hand, if $t$ were hostile, then $t \notin f(t) = T$, which contradicts to the fact that $T$ is the set of all hostile elements.

The resulting contradiction shows that the assumption that there exists a one-to-one mapping $f$ from $A$ onto $2^A$ is incorrect, which proves the Theorem. \qed