Notes for Wednesday, June 2nd

Recall: \( A_{TM} = \{ \langle M, w \rangle | M \text{ is a TM and } M \text{ accepts } w \} \). \( A_{TM} \) is Turing-recognizable (via Universal TM) but not decidable (via diagonalization technique).

Now we ask the question: is there a language that is not even Turing-recognizable.

Suppose \( A_{TM} \) is also Turing-recognizable.

Theorem: \( L \) is decidable iff \( L \) and \( \overline{L} \) are Turing recognizable

Proof:

\((\Rightarrow)\) All decidable languages are Turing-recognizable, so \( L \) is Turing-recognizable. If \( L \) is decidable, that automatically implies that \( L \) is Turing-recognizable. If \( L \) is decidable, \( \overline{L} \) is also decidable (decidable languages are closed under complement), so \( \overline{L} \) is also Turing-recognizable.

\((\Leftarrow)\) If \( L \) and \( \overline{L} \) are Turing-recognizable, then there exist \( M_1 \) and \( M_2 \) such that \( L(M_1) = L \) and \( L(M_2) = \overline{L} \). We can construct a decider TM for \( L \):

“on input \( \langle M, w \rangle \):
run \( M_1 \) and \( M_2 \) on \( w \) by alternating one step at a time
If \( M_1 \) accepts, \( M \) accepts
If \( M_2 \) accepts, \( M \) rejects”

This way, \( M \) is guaranteed to halt on all inputs (because the string is either in \( L \) or \( \overline{L} \), and because \( M_1 \) and \( M_2 \) are run in parallel, it doesn’t matter if one of them goes into an infinite loop). Thus, \( L \) is decidable.

Corollary: \( \overline{A_{TM}} \) is not Turing-recognizable.

(If it were, \( A_{TM} \) itself would be decidable by the theorem, which is a contradiction)

This is the Chomsky hierarchy of problems:

\[
\begin{align*}
\text{TURING-REC (} A_{TM} \text{)} \\
\text{DECIDABLE (} 0^n1^n0^n \text{)} \\
\text{CFL (} 0^n1^n \text{)} \\
\text{REG (} 0^*1^* \text{)}
\end{align*}
\]

\( \overline{A_{TM}} \) is undecidable; are there more such problems?

Suppose you want to show that \( B \) is undecidable, and you know that \( A \) is undecidable. If you can use \( B \) to solve \( A \) (\( B \) is a decider for \( A \)), then \( A \) is decidable and this is a contradiction.

In this way, you can reduce an undecidable problem \( A \) to another problem \( B \). If \( B \) is decidable, then there is a contradiction.

The notion is to use the new problem \( B \) to solve the original problem \( A \)

Notation: \( A \) is reducible to \( B \) if you can use \( B \) to solve \( A \). We write \( A \leq B \).

Suppose \( B \leq C \), and \( C \leq D \). Then we can write \( A \leq B \leq C \leq D \).

Let \( E_{TM} = \{ \langle M \rangle | M \text{ is a TM and } L(M) = \emptyset \} \).

Theorem: \( A_{TM} \leq E_{TM} \) (this \( E_{TM} \) is undecidable, by reduction)

Proof: Assume \( E_{TM} \) is decidable. Then, there exists a decider TM \( M_E \) such that \( L(M_E) = E_{TM} \).

Construct a decided for \( A_{TM} \) as follows:

\( \text{on input } \langle M, w \rangle, \)
1. Build TM $M_1$ on input $x$:

   (a) If $x \neq w$, reject
   (b) If $x = w$, then simulate $M$ on $w$, accept if $M$ accepts

   (then $L(M_1) = \{\{w\} \text{ if } M \text{ accepts } w, \emptyset \text{ otherwise}\}$)

2. Feed $M_1$ to $M_E$

3. Accept $\langle M, w \rangle$ if $M_E$ rejects $\langle M_1 \rangle$; Reject $\langle M, w \rangle$ if $M_E$ accepts $\langle M_2 \rangle$.

This is a contradiction, so $E_{TM}$ is undecidable.