Recitation 05: Undecidability, Unrecognizability, and Reducibility

In today’s recitation, we will gain practice with using reductions to show that a language is undecidable or unrecognizable. First, we’ll review general and mapping reducibility. Then, we will work an example using general reducibility to prove a language is undecidable, and another example using mapping reducibility to prove a language is unrecognizable and co-unrecognizable.

Undecidability and unrecognizability

We proved the following facts about $A_{TM}$ in lecture.

**Theorem 1.** $A_{TM}$ is T-recognizable and undecidable.

**Theorem 2.** $\overline{A_{TM}}$ is unrecognizable.

We define the class of co-Turing-recognizable languages as the class of complements of recognizable languages.

**Definition 1.** Language $B$ is co-Turing-recognizable if $\overline{B}$ is Turing-recognizable.

For example, $\overline{A_{TM}}$ is co-Turing-recognizable because $A_{TM}$ is Turing-recognizable.

We proved the following theorem in lecture.

**Theorem 3.** Language $B$ is decidable iff both $B$ and $\overline{B}$ are T-recognizable.

In other words, language $B$ is decidable iff $B$ is both Turing-recognizable and co-Turing-recognizable. Figure 1 shows this visually, and lists several more examples of undecidable and/or unrecognizable languages.

We proved that $A_{TM}$ is undecidable using a diagonalization argument. In the following theorem, we show that there are uncountably many unrecognizable languages, using diagonalization as an intermediate step.

**Theorem 4.** The set of T-unrecognizable languages is uncountably infinite.
Proof. We want to show that

1. The set of all TMs is countable, and
2. The set of all languages is uncountable.

We prove that the set of all TMs is countable by encoding each TM as a string \( \langle M \rangle \in \Sigma^* \). We know \( \Sigma^* \) is countable because there are only finitely many strings of each length.

We prove that the set of all languages is uncountable as follows. Each language \( L \) is a subset of the strings \( s_1, s_2, s_3, \ldots \) in \( \Sigma^* \). We can show a correspondence between each language and an infinite binary string, with the \( i \)-th bit of the binary string indicating whether string \( s_i \) is in \( L \). Then we use a diagonalization argument (similar to the proof that \( \mathbb{R} \) is uncountable) to show that the set of infinite binary strings is uncountable.

Since each TM recognizes exactly one language, we conclude that uncountably many languages do not have a corresponding TM. So the set of unrecognizable languages is uncountable. \( \square \)

Reducibility

We have seen two types of reducibility in lecture: "general" reducibility and mapping reducibility.

Definition 2 (General reducibility). Language \( A \) is reducible to language \( B \) if the following holds: if we can solve \( B \), then we can solve \( A \) using the solver for \( B \) as a subroutine.

Informally, "\( A \) is reducible to \( B \)" implies that...
1. If $A$ is hard, then $B$ is also hard, and
2. If $B$ is easy, then $A$ is also easy.

We often use general reduction to show that a language is undecidable. We can formalize the notion that hardness of $A$ implies hardness of $B$ as follows.

**Theorem 5.** Suppose $A$ is reducible to $B$ in the general sense. Then

1. If $A$ is undecidable, then $B$ is undecidable.
2. If $B$ is decidable, then $A$ is decidable.

However, we can’t use general reducibility to show that a language is unrecognizable. For example, the unrecognizable language $\overline{A}_{TM}$ is reducible to $A_{TM}$, but $A_{TM}$ is recognizable.

**Definition 3 (Mapping reducibility).** Language $A$ is mapping reducible to language $B$ if there exists a computable function $f : \Sigma^* \rightarrow \Sigma^*$ such that $w \in A$ iff $f(w) \in B$. We write this as $A \leq_m B$.

In other words, we want $f$ to map strings in $A$ to strings in $B$, and map strings not in $A$ to strings not in $B$ (as shown in Figure 2).

![Figure 2: Mapping reducibility. We write $A \leq_m B$ if some computable function $f$ maps strings in language $A$ to strings in language $B$, and maps strings not in $A$ to strings not in $B.\)
1. If \( w \not\in A \), then \( f(w) \not\in B \).
2. If \( w \in A \), then \( f(w) \in B \).

Thus we have \( w \in A \) iff \( f(w) \in B \). We conclude that \( A \leq_m B \). \( \square \)

We often use mapping reduction to show that a language is unrecognizable. We can also use mapping reduction to show that a language is undecidable (however, it is more common to prove undecidability using general reductions, as mentioned earlier).

**Theorem 7.** Suppose \( A \leq_m B \). Then

1. If \( A \) is unrecognizable, then \( B \) is unrecognizable.
2. If \( B \) is recognizable, then \( A \) is recognizable.
3. If \( A \) is undecidable, then \( B \) is undecidable.
4. If \( B \) is decidable, then \( A \) is decidable.

**Problem solving strategies: undecidability and unrecognizability**

1. To show that language \( B \) is undecidable, find a general reduction from some undecidable language \( A \) to \( B \). Assume for contradiction that there is a TM \( R \) that decides \( B \). Then construct a TM \( S \) that decides \( A \), using \( R \) for a subroutine. Often, we use \( A = \overline{A_{TM}} \).

2. To show that language \( B \) is unrecognizable, find a mapping reduction from some unrecognizable language \( A \) to \( B \) (written as \( A \leq_m B \)). Describe a computable function \( f \) such that \( w \in A \) iff \( f(w) \in B \). Often, we use \( A = \overline{A_{TM}} \).

**Example problems**

In the following example (Problem 5.10 in the textbook), we prove that a language is undecidable using a general reduction from \( A_{TM} \).

**Example 1.** Show that

\[
B = \{ \langle M, w \rangle \mid M \text{ is a 2-tape TM that writes a nonblank symbol on its second tape when run on } w \}
\]

is undecidable.

**Solution 1.** We find a general reduction from \( A_{TM} \) to \( B \). Assume for contradiction that Turing decider \( R \) decides \( B \). We construct the following decider \( S \) that decides \( A_{TM} \).
\(S\): On input \(\langle M, w \rangle\):

1. Construct the following 2-tape Turing machine \(T_M\):

   \[
   T_M: \text{On input } x:\ \\
   (a) \text{Simulate } M \text{ on } x \text{ using the first tape of } T_M. \\
   (b) \text{If } M \text{ accepts, write a nonblank symbol on the second tape.}
   \]

2. Run \(R\) on \(\langle T_M, w \rangle\).

3. Accept if \(R\) accepts.
   Reject if \(R\) rejects.

If \(M\) accepts \(w\), then \(T_M\) run on \(w\) would write a nonblank symbol on the second tape. So \(\langle T_M, w \rangle \in B\). Then \(R\) accepts \(\langle T_M, w \rangle\), so \(S\) also accepts \(\langle M, w \rangle\).

If \(M\) rejects \(w\) (either by halting or by looping forever), then \(T_M\) run on \(w\) would not not use the second tape. So \(\langle T_M, w \rangle \notin B\). Then, \(R\) halts and rejects \(\langle T_M, w \rangle\), so \(S\) also halts and rejects \(\langle M, w \rangle\).

Thus, \(S\) decides \(A_{TM}\). This is a contradiction because we know \(A_{TM}\) is undecidable. We conclude that \(B\) is undecidable.

In the next examples, we prove that a language is unrecognizable and co-unrecognizable using mapping reductions from \(\overline{A_{TM}}\).

**Example 2.** Show that

\[
\text{REG}_{TM} = \{ \langle M \rangle \mid \text{M is a TM and } L(M) \text{ is a regular language} \}
\]

is \(T\)-unrecognizable.

**Solution 2.** We will show that

\[
\overline{A_{TM}} \leq_m \text{REG}_{TM}.
\]

We find a computable function \(f\) such that \(\langle M, w \rangle \in A_{TM}\) if and only if \(f(\langle M, w \rangle) = T_{M,w} \in \text{REG}_{TM}\).

In other words: If TM \(M\) rejects \(w\), then we want the language of \(T_{M,w}\) to be regular. If \(M\) accepts \(w\), then we want the language of \(T_{M,w}\) to be nonregular. We can design \(T_{M,w}\) such that if \(M\) accepts \(w\), then \(L(T_{M,w})\) is a specific nonregular language, such as \(\{0^k1^k \mid k \geq 0\}\).

We construct \(T_{M,w}\) as follows.

\[
T_{M,w}: \text{On input } x:\ \\
1. \text{Run } M \text{ on } w. \\
2. \text{If } M \text{ accepts, check if } x = 0^k1^k \text{ for some } k \geq 0. \text{ If so, accept. If not, reject.}
\]
If \( M \) rejects, **reject**.

If \( M \) rejects \( w \) (either by halting or looping), then \( T_{M,w} \) rejects all inputs (either by halting or looping). So \( L(T_{M,w}) = \emptyset \) which is a regular language.

If \( M \) accepts \( w \), then \( T_{M,w} \) accepts precisely the strings of the form \( 0^k1^k \). So \( L(T_{M,w}) = \{0^k1^k \mid k \geq 0\} \) which is a nonregular language.

This proves that \( \overline{A_{TM}} \leq_m \overline{REG_{TM}} \). We know \( \overline{A_{TM}} \) is \( T \)-unrecognizable, so \( \overline{REG_{TM}} \) is also \( T \)-unrecognizable.

**Example 3.** Show that \( \overline{REG_{TM}} \) is co-\( T \)-unrecognizable.

**Solution 3.** The proof is similar to the last problem. We will show that

\[ A_{TM} \leq_m \overline{REG_{TM}}. \]

We find a computable function \( f \) such that \( \langle M, w \rangle \in A_{TM} \) if and only if \( f(\langle M, w \rangle) = T_{M,w} \in \overline{REG_{TM}} \).

In other words: If \( M \) rejects \( w \), then we want the language of \( T_{M,w} \) to be nonregular. If \( M \) accepts \( w \), then we want the language of \( T_{M,w} \) to be regular.

We construct \( T_{M,w} \) as follows.

\( T_{M,w} \): On input \( x \):

1. **Accept** if \( x = 0^k1^k \) for some \( k \geq 0 \).
2. Otherwise, run \( M \) on \( w \).
3. **Accept** if \( M \) accepts. **Reject** if \( M \) rejects.

If \( M \) rejects \( w \) (either by halting or looping), then \( T_{M,w} \) only accepts inputs of the form \( x = 0^k1^k \) and rejects all other inputs (either by halting or looping). So \( L(T_{M,w}) = \{0^k1^k \mid k \geq 0\} \) which is a nonregular language.

If \( M \) accepts \( w \), then \( T_{M,w} \) accepts all strings. So \( L(T_{M,w}) = \Sigma^* \) which is a regular language.

We have shown that \( A_{TM} \leq_m \overline{REG_{TM}} \). We know \( A_{TM} \) is \( T \)-unrecognizable, so \( \overline{REG_{TM}} \) is also \( T \)-unrecognizable. This proves that \( \overline{REG_{TM}} \) is co-\( T \)-unrecognizable.

It’s possible that \( M \) rejects \( w \) by looping forever. In that case, \( T_{M,w} \) also rejects by looping forever (on all inputs). However, we cannot write “if \( M \) loops forever, then loop” as part of the program for \( T_{M,w} \). This is because \( T_{M,w} \) can’t determine that \( M \) will loop forever, since the halting problem \( \text{HALT}_{TM} \) is undecidable.