Recitation 04: Undecidability, Unrecognizability, and Reducibility

In this recitation we’ll practice working with reductions to show that a language is undecidable or unrecognizable. We begin by providing a diagram summarizing the languages we’ve seen in lecture, and whether they are decidable, recognizable, or unrecognizable. Then, we review general and mapping reducibility. Finally, we work an example using general reducibility to show that a particular language is undecidable, and another example using mapping reducibility to show that a particular language is both unrecognizable and co-unrecognizable.

Undecidability and unrecognizability

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

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

**Theorem 2.** $A_{TM}^{\overline{}}$ 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, $A_{TM}^{\overline{}}$ 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.

In other words, this means B is at least as hard as A. We often use general reductions 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.

We have a second, related definition of reducibility.

**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).

![Diagram](https://via.placeholder.com/150)

**Theorem 6.** If \( A \leq^m B \), then \( \overline{A} \leq^m \overline{B} \).

**Proof.** Suppose \( A \leq^m B \). Then there exists a computable function \( f \) such that for all \( w \in \Sigma^* \), we have \( w \in A \) iff \( f(w) \in B \). This means that

1. If \( w \in A \), then \( f(w) \in B \).
2. If \( w \notin A \), then \( f(w) \notin B \).

By definition of complement, this can be rewritten in terms of \( \overline{A} \) and \( \overline{B} \) as follows. For all \( w \in \Sigma^* \),

1. If \( w \notin \overline{A} \), then \( f(w) \notin \overline{B} \).
2. If \( w \in \overline{A} \), then \( f(w) \in \overline{B} \).
Thus we have \( w \in A \) iff \( f(w) \in B \). We conclude that \( A \leq_m B \).

We often use mapping reductions to show that a language is unrecognizable. We can also use mapping reductions to show that a language is undecidable (however, it is more common to prove undecidability using general reductions). The following theorem summarizes the ways in which we use mapping reductions.

**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.

Don’t be confused about the direction in which mapping reductions take place. Given \( A \leq_m B \), the direction of the notation \( \leq_m \) is meant to compare the difficulty of \( A \) and \( B \). That is, solving \( A \) is not more difficult than solving \( B \). Remembering the intuition behind the notation helps us to understand the content of Thm. 7.

Notice that complements of a language are always generally reducible to the original language, i. e., \( \overline{A} \) is reducible to \( A \), because if we have a solver to \( A \) we may always reverse its answer to solve \( \overline{A} \). However, the same is not true for mapping reductions! Notice that \( \overline{A}_{TM} \not\leq_m A_{TM} \) since \( \overline{A}_{TM} \) is unrecognizable but \( A_{TM} \) is recognizable. That means we can’t use general reducibility to show that a language is unrecognizable. Instead we use general reductions to show undecidability, and mapping reductions to show unrecognizability.

This also tells us that mapping reducibility is a stronger condition than general reducibility. If \( A \leq_m B \), then \( A \) is also reducible to \( B \) in the general sense. However, if \( A \) is reducible to \( B \) in the general sense, it is not guaranteed that \( A \leq_m B \). For example, \( \overline{A}_{TM} \) is reducible to \( A_{TM} \) in the general sense, but \( \overline{A}_{TM} \) is not mapping reducible to \( A_{TM} \).

**Problem solving strategies: undecidability and unrecognizability**

1. To show that language \( B \) is undecidable, find a general reduction from some undecidable language \( A \) to \( B \). Our proof outline is generally as follows:

   - 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 \( A = A_{TM} \).
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 $A = \overline{A_{TM}}$. Describing $f$ serves as a shorthand for the following proof outline:

- Assume for contradiction that there exists a TM $R$ that recognizes $B$.
- Then construct a TM $S$ that takes an input $w$, computes $f(w)$, runs $R$ on $f(w)$, and outputs the result of $R$ on $f(w)$. $S$ will decide $A$.

**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

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

is undecidable.

**Solution 1.** We find a general reduction from $A_{TM}$ to $TT_{TM}$. Assume for contradiction that $R$ is a decider for $TT_{TM}$. We construct the following decider $S$ for $A_{TM}$.

**$S$:** On input $\langle M, w \rangle$:

1. Construct the following 2-tape Turing machine $T_M$:

   **$T_M$:** On input $x$:

   (a) Simulate $M$ on $x$ using the first tape of $T_M$.
   (b) If $M$ 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 when $T_M$ is run on $w$, it writes a nonblank symbol on its second tape. So $\langle T_M, w \rangle \in TT_{TM}$. Then $R$ accepts $\langle T_M, w \rangle$, so $S$ also accepts $\langle M, w \rangle$.

If $M$ rejects $w$, by halting or looping (or if the string $\langle M, w \rangle$ is garbage), then when $T_M$ is run on $w$ it does not use its second tape. So $\langle T_M, w \rangle \notin TT_{TM}$. Then, $R$ halts and rejects $\langle T_M, w \rangle$, so $S$ also halts and rejects $\langle M, w \rangle$. 
Hence $S$ decides $A_{TM}$. This is a contradiction because we know $A_{TM}$ is undecidable. We conclude that $TT_{TM}$ is undecidable.

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

Example 2. Show that

$$REG_{TM} = \{ \langle M \rangle \mid M \text{ is a TM and } L(M) \text{ is a regular language} \}$$

is unrecognizable.

Solution 2. We will show that

$$\overline{A_{TM}} \leq_m REG_{TM}.$$ 

That is, we need to find a computable function $f$ such that $\langle M, w \rangle \in \overline{A_{TM}}$ iff $f(\langle M, w \rangle) = T_{M,w} \in REG_{TM}$.

In other words: If $TM$ $M$ accepts $w$, then we want the language of $T_{M,w}$ to be nonregular. Otherwise, we want the language of $T_{M,w}$ to be regular.

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.

$T_{M,w}$: On input $x$:

1. Run $M$ on $w$.
2. If $M$ accepts, check if $x = 0^k1^k$ for some $k \geq 0$. If so, accept.
3. Otherwise, reject.

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 $HALT_{TM}$ is undecidable.

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.

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

This proves that $\overline{A_{TM}} \leq_m REG_{TM}$. We know $\overline{A_{TM}}$ is unrecognizable, so $REG_{TM}$ is also unrecognizable.

Example 3. Show that $REG_{TM}$ is co-T-unrecognizable.

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

$$\overline{A_{TM}} \leq_m \overline{REG_{TM}}.$$ 

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

In other words: If $M$ accepts $w$, then we want the language of $T_{M,w}$ to be regular. If $M$ does not accept $w$, then we want the language of $T_{M,w}$ to be nonregular.

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

If \( M \) rejects \( w \), by halting or looping (or if the string \( \langle M, w \rangle \) is garbage), 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.

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