Theorem \( L_{\text{Diag}} \) is Turing-recognizable, but not decidable.

Proof It is easily seen that the following TM recognizes \( L_{\text{Diag}} \).

\[
M_{\text{Diag}} = \begin{cases} 
\text{on input } w, \\
\text{Determine index } i \text{ s.t. } w = w_i; \\
\text{Determine (the description of) } \\
TM M_i; \\
\text{Simulate } M_i \text{ on } w_i \text{ and } \\
\text{Accept or Reject or Run forever accordingly.}
\end{cases}
\]

We note that it is possible for \( M_{\text{Diag}} \) to determine the index \( i \) and \( m/c \ M_i \) because \( \Sigma^* \) and \( L_{TM} = \{ \langle M_1 \rangle, \langle M_2 \rangle, \ldots \} \) are both effectively enumerable. Clearly, by design,

\[
L(M_{\text{Diag}}) = \{ w \mid M_{\text{Diag}} \text{ accepts } w \} \\
= \{ w_i \mid M_i \text{ accepts } w_i \} \\
= L_{\text{Diag}}.
\]
Now we show that $\overline{L_{\text{Diag}}}$ is undecidable. It is more convenient to show that

$$\overline{L_{\text{Diag}}} = \{ w_i \mid i \geq 1, \text{ } M_i \text{ does not accept } w_i \}$$

is undecidable and note the easy fact:

**Fact** A language $L$ is decidable iff $\overline{L}$ is.

**Proof** If a TM $M$ decides $L$ then a TM that decides $\overline{L}$ can be designed so as to simulate $M$ and in the end, switch the Accept or Reject decision of $M$.

**Claim** $\overline{L_{\text{Diag}}}$ is undecidable. In fact $\overline{L_{\text{Diag}}}$ is not even Turing recognizable.

**Proof** Suppose on the contrary that a TM $M$ recognizes $\overline{L_{\text{Diag}}}$, let $k \geq 1$ be the index s.t. $\langle M \rangle = \langle M_k \rangle$ in the effective enumeration of TM-descriptions. Let $W = W_k$ be the $k^{th}$ string in effective
enumeration of $\Sigma^*$. We show that the TM $M$ and the language $\overline{L_{\text{Diag}}}$ "disagree" on the input $w$, giving a contradiction ($M$ is supposed to recognize $\overline{L_{\text{Diag}}}$). Indeed,

$$
M \text{ accepts } w \iff M_k \text{ accepts } w_k \iff w_k \notin L_{\text{Diag}} \iff w \notin L_{\text{Diag}}.
$$

We note that
- $L_{\text{Diag}}$ is T.R. but not decidable.
- $\overline{L_{\text{Diag}}}$ is not even T.R.

This is an example of the following general fact.

**Fact** A language $L$ is decidable

$$
\iff \text{ Both } L, \overline{L} \text{ are T.R.}
$$

In particular, if $L$ is T.R. but not decidable, then $\overline{L}$ is not even T.R.
Proof of $\Rightarrow$: This is easy. If $L$ is decidable, then so is $\overline{L}$, and hence both are T.R. as well.

Proof of $\Leftarrow$: Suppose $L, \overline{L}$ both are T.R.

Let $M, M'$ be TMs that recognize $L, \overline{L}$ respectively. That is, $\forall x \in \Sigma^*$,

$x \in L \Rightarrow M$ accepts $x$ (eventually).

$x \in \overline{L} \Rightarrow M'$ accepts $x$ (""").

Now the TM $\tilde{M}$ that decides $L$ can, on input $x$, simulate both $M$ and $M'$ on $x$, alternately for one more step each, and Accepts if $M$ accepts. Rejects if $M'$ accepts.

$\tilde{M}$ decides $L$ because:

\[ x \in L \Rightarrow M \text{ accepts } x \]

\[ \Rightarrow \tilde{M} \text{ accepts } x. \]

\[ x \notin L \Rightarrow x \in \overline{L} \Rightarrow M' \text{ accepts } x \]

\[ \Rightarrow \tilde{M} \text{ Rejects } x. \]
Now that we know that $L_{\text{Diag}}$ is undecidable, we can show that several problems concerning TMs are also undecidable, by reducing $L_{\text{Diag}}$ to these problems.

**Theorem** Acceptance Problem for TMs:

$$A_{TM} = \{ \langle M, w \rangle \mid M \text{ is a } TM \text{ that accepts } w \}$$

is TR. but undecidable.

**Proof** It is easily seen that the following TM recognizes $A_{TM}$.

$$M^* := \text{ on input } \langle M, w \rangle,$$

Simulate $M$ on $w$.

Accept or Reject or Runforever according.

$M^*$ accepts $\langle M, w \rangle \iff M$ accepts $w$

$$\iff \langle M, w \rangle \in A_{TM}.$$

Now we show that $A_{TM}$ is undecidable by reducing $L_{\text{Diag}}$ to $A_{TM}$. Specifically, we show that if there were a (hypothetical) TM $R$
that decides $A_{TM}$, then $R$ can be used to design a $TM$ $\tilde{M}$ that decides $L_{Diag}$. Since $L_{Diag}$ is already known to be undecidable, it follows that $A_{TM}$ is actually undecidable.

Towards this end, suppose (on contrary) that $R$ decides $A_{TM}$. I.e.

$$R(<M, w>) = \begin{cases} 
\text{Accept if } M \text{ accepts } w, \\
\text{Reject if } M \text{ rejects } w \text{ or runs forever.}
\end{cases}$$

It is easily seen now that $\tilde{M}$ as below decides $L_{Diag}$.

$\tilde{M} := "$On input $w$,
Determine index $i$ s.t. $w = w_i$.
Determine the $TM$ $<M_i>$.
Use $R$ to decide whether $M_i$ accepts $w_i$.
If $M_i$ accepts $w_i$, accept.
If $M_i$ does not accept $w_i$, reject."
\( \tilde{M} \) accepts \( w \) if \( w = w_i \), \( M_i \) accepts \( w_i \):

\[
\text{i.e. if } w_i \in L_{Diag}.
\]

\( \tilde{M} \) rejects \( w \) if \( w = w_i \), \( M_i \) does not accept \( w_i \):

\[
\text{i.e. if } w = w_i \& L_{Diag}.
\]

Thus \( \tilde{M} \) decides \( L_{Diag} \) (a contradiction).

Corollary \( \overline{A}_{TM} \) is not even T.R.

**Theorem** Halting Problem for TMs:

\[
\text{HALT}_{TM} = \{ \langle M, w \rangle \mid M \text{ is a TM that halts on } w \}\]

is T.R. but undecidable.

Proof: Clearly, the following TM recognizes

\[
\text{HALT}_{TM}, \quad M^* := " \text{On input } \langle M, w \rangle \\
\text{Simulate } M \text{ on } w, \\
\text{If } M \text{ halts (i.e. accepts or rejects), accept. }"
\]

\[
\langle M, w \rangle \in \text{HALT}_{TM} \quad \Rightarrow \quad M \text{ halts on } w \\
\quad \Rightarrow \quad M^* \text{ accepts } \langle M, w \rangle.
\]
\(<M,w> \notin \text{HALT}_{\text{TM}} \Rightarrow M \text{ runs forever on } w\)
\Rightarrow M^* \text{ runs forever on } <M,w> \text{ (} \because M^* \text{ just simulates } M)\).

This shows that indeed \(M^*\) recognizes \(\text{HALT}_{\text{TM}}\).

Now \(\text{HALT}_{\text{TM}}\) is undecidable because if there were a (hypothetical) decider \(R\) for \(\text{HALT}_{\text{TM}}\), one can design a decider \(\tilde{M}\) for \(A_{\text{TM}}\). The latter is not possible since \(A_{\text{TM}}\) is already known to be undecidable.

Indeed \(\tilde{M}\), on input \(<M,w>\) for \(A_{\text{TM}}\), uses \(R\) to decide whether \(M\) halts on \(w\).

If \(M\) does not halt on \(w\), \(\tilde{M}\) rejects.
If \(M\) halts on \(w\), \(\tilde{M}\) simulates \(M\) on \(w\) and accepts or rejects accordingly.

Clearly \(\tilde{M}\) accepts \(<M,w>\) if \(M\) accepts \(w\) and rejects otherwise, and hence decides \(A_{\text{TM}}\).
Corollary: $\text{HALT}_\text{TM}$ is not even T.R.

Theorem: (Non-)Emptiness Problem for TMs.

$E_\text{TM} = \{ \langle M \rangle \mid M \text{ is a TM s.t. } L(M) \neq \emptyset \}$

is T.R. but undecidable.

Proof: Showing that $E_\text{TM}$ is T.R. is left as a nice exercise :).

We show that $E_\text{TM}$ is undecidable by reducing $A_\text{TM}$ to it. Suppose (on contrary) that $E_\text{TM}$ is decidable and a TM $R$ decides it.

We will design a TM $\widetilde{M}$ that decides $A_\text{TM}$ (reaching a contradiction).

$\widetilde{M} := \text{"On input } \langle M, w \rangle \text{,}

\text{construct a TM } D \text{ that behaves as follows:}

D := \text{"On input } x, \\
\text{If } x \neq w, \text{ reject.} \\
\text{If } x = w, \text{ simulate} $
M on w and accept/reject/run forever accordingly."

By running R on <D>, determine whether L(D) ≠ ∅.

If L(D) ≠ ∅, Accept.
If L(D) = ∅, Reject."

We note that \( \tilde{M} \) indeed decides \( A_{TM} \).

\[ \langle M, w \rangle \in A_{TM} \Rightarrow M \text{ accepts } w \]
\[ \Rightarrow L(D) = \{w\} \neq ∅ \]
\[ \Rightarrow \tilde{M} \text{ accepts } \langle M, w \rangle. \]

\[ \langle M, w \rangle \notin A_{TM} \Rightarrow M \text{ rejects/runs forever on } w \]
\[ \Rightarrow L(D) = ∅ \]
\[ \Rightarrow \tilde{M} \text{ rejects } \langle M, w \rangle. \]

The main point is that from the perspective of the m/c D that is designed, all inputs other than w are rejected outright. Thus

\[ L(D) = \{w\} \text{ or } L(D) = ∅ \]
depending on whether \( M \) accepts \( w \) or not.

\[
\text{Corollary: } \quad E_{\text{TM}} = \{ \langle M \rangle \mid M \text{ is a TM s.t. } L(M) = \emptyset \}
\]
is not even T.R.

Exercise Using a proof similar as above show that:

- \( \text{All}_{\text{TM}} \) is \underline{decidable}
- \( \text{All}_{\text{TM}} \) is not \underline{even T.R.}. Here

\[
\text{All}_{\text{TM}} = \{ \langle M \rangle \mid M \text{ is a TM s.t. } L(M) = \Sigma^* \}
\]

Exercise Show that:

\[
\text{EQ}_{\text{TM}} = \{ \langle M, M' \rangle \mid M, M' \text{ are TMs, } L(M) = L(M') \}
\]
is not T.R.

Finally, we prove a rather general result known as Rice's Theorem. It states that any non-trivial "property" of the language \( L(M) \) recognized by a TM \( M \), given the description \( \langle M \rangle \), is undecidable.
Rice's Theorem

Let $\mathcal{P}$ be a subclass of the class of T.R. languages. $\mathcal{P}$ is non-trivial if
- there is a lang. $A \in \mathcal{P}$ (i.e. $\mathcal{P}$ is non-empty)
- $\forall B \notin \mathcal{P}$ (i.e. $\mathcal{P}$ is not all the T.R languages).

Then the following lang. is undecidable.

$$L_\mathcal{P} = \{ <M> | \text{M is a TM s.t. } L(M) \in \mathcal{P} \}.$$  

Note: The theorem, in one sweep, shows that all these languages are undecidable.

$\{<M>| L(M) \neq \emptyset \}$ \quad $\mathcal{P} =$ All T.R. languages except $\emptyset$.

$\{<M>| L(M) = \Sigma^* \}$ \quad $\mathcal{P} = \{ \Sigma^* \}$.

$\{<M>| L(M) \text{ is regular} \}$ \quad $\mathcal{P} =$ Class of regular langs.

$\{<M>| L(M) \text{ is context-free} \}$ \quad $\mathcal{P} =$ \_$c.f.$ languages.
Proof. W.l.o.g. we can assume that (the empty language) \( \emptyset \notin \mathcal{P} \). Otherwise, we could consider the property \( \overline{\emptyset} \), observe that \( L_{\overline{\emptyset}} = L_\emptyset \), and \( L_{\overline{\emptyset}} \) is undecidable iff \( L_\emptyset \) is.

So let's assume \( \emptyset \notin \mathcal{P} \). Since \( \mathcal{P} \) is non-trivial, there is some T.R. language \( A \in \mathcal{P} \). Suppose a TM \( M_A \) recognizes the language \( A \).

We now show that \( L_\emptyset \) is undecidable by reducing \( A_{TM} \) to it. Specifically, given input \( \langle M, w \rangle \) "we" design a TM \( R \) such that for lang. \( A_{TM} \)

\[
\langle M, w \rangle \in A_{TM} \Rightarrow L(R) = A
\]

In particular \( L(R) \in \mathcal{P} \).

\[
\langle M, w \rangle \notin A_{TM} \Rightarrow L(R) = \emptyset.
\]

In particular \( L(R) \notin \mathcal{P} \).

Thus, if \( L_\emptyset \) were decidable, "one" could decide
whether \( L(R) \in P \) or not

i.e. \( L(R) = A \) or \( L(R) = \emptyset \)

i.e. \( \langle M, w \rangle \in A_{TM} \) or \( \langle M, w \rangle \notin A_{TM} \),

and thus decide \( A_{TM} \), reaching a contradiction.

The m/c \( R \) is designed as follows:

\[
R := \text{On input } x = (q_0, \omega, G_0, M)
\]

Ignore \( x \) for now and first simulate \( M \) on \( w \).

If \( M \) rejects \( w \), reject.

If \( M \) runs forever on \( w \), run forever.

Else \( M \) accepts \( w \). In this case simulate \( M_A \) on \( x \) and Accept or Reject or Run forever accordingly.

Clearly if \( M \) rejects \( w \) or runs forever on \( w \) then \( R \) rejects or runs forever on every input \( x \).

In either case \( L(R) = \emptyset \). Thus

\[
\langle M, w \rangle \notin A_{TM} \Rightarrow L(R) = \emptyset \quad \text{as needed.}
\]
On the other hand

if $M$ accepts $w$ then

Behavior of $R$ on every input $x$ is same as $M_A$ on that input.

since $R$ just simulates $M_A$ on $x$,

Hence $L(R) = A$, the lang. accepted by the machine $M_A$.

Thus $\langle M, w \rangle \in A_{TM} \implies L(R) = A$ as needed.

Remark: In the proof "we design" or "one could decide" are to be interpreted as "a TM can design" or "a TM could decide" respectively.