

Reduction

Let $L(M)$ be the language accepted by TM M .

Theorem 1. *The language $E = \{M : L(M) = \phi\}$ is undecidable.*

Proof: Suppose TM R decides E . We use R to construct TM S that decides H .

S does the following on input $(M; w)$:

1. Using description of M and w it constructs another TM M_1 as follows.

$M_1(x)$:

if $x \neq w$ then “no”

else run M on input w and say “yes” if M accepts.

2. Simulates R on M_1 .

3. If R says “yes” then “no” else “yes”.

Rice's Theorem

Theorem 2. *Suppose C is a proper, non-empty subset of the set of all r.e. languages. Then the following problem is undecidable: Given a TM M , is $L(M) \in C$.*

“Testing any non-trivial property of the languages recognized by TMs is undecidable.”

Proof

Assume w.l.o.g. that $\phi \notin \mathcal{C}$.

Since \mathcal{C} is nonempty, we can assume that there is a language $L \in \mathcal{C}$, accepted by machine M_L .

Let TM M_H accept H .

We will construct a machine M_x whose language is either L or ϕ .

Construct the machine M_x as follows:

$M_x(y)$: if $M_H(x) = \text{yes}$ then $M_L(y)$ else \nearrow .

We show that $L(M_x) \in \mathcal{C}$ iff $x \in H$.

If $x \in H$, then M_x on input y accepts or never halts depending on whether $y \in L$. Hence the language accepted by M_x is $L \in \mathcal{C}$.

If $x \notin H$, then $M_H(x) = \nearrow$, and M_x accepts $\phi \notin \mathcal{C}$.

A Non-R.E. Language

Theorem 3. *A language L is recursive (decidable) iff both L and L^c are r.e.*

Proof: Suppose L is recursive, then L^c is also recursive and hence both are r.e.

Assume both are r.e., accepted by TM M and M^c respectively.

Then L is decided by a TM M' which on input x simulates both M and M^c in an interleaved manner.

Since either M or M^c should accept x , L will accept or reject.

Corollary 1. *H^c is not r.e. Thus the class of r.e. languages is not closed under complement.*

Recursive Inseparability

Definition 1. Two languages are called **recursively inseparable** if there is no recursive language R such that $L_1 \cap R = \phi$ and $L_2 \subset R$.

Theorem 4. Let $L_1 = \{M : M(M) = \text{yes}\}$ and $L_2 = \{M : M(M) = \text{no}\}$. Then L_1 and L_2 are recursively inseparable.

Proof: Suppose not. Let recursive language R separate them:

$$L_1 \cap R = \phi \text{ and } L_2 \subset R.$$

Consider the TM M_R deciding R .

What is $M_R(M_R)$?

If it is “yes” then $M_R \in L_1$ and this implies $M_R(M_R) = \text{no}$.

If it is “no” then $M_R(M_R) \in L_2$ and this implies $M_R(M_R) = \text{yes}$.

Contradiction.

Corollary 2. Let $L'_1 = \{M : M(\epsilon) = \text{yes}\}$ and $L'_2 = \{M : M(\epsilon) = \text{no}\}$. Then L'_1 and L'_2 are recursively inseparable.

Proof: Suppose not. Let R' separate the two. Then we will show that we can separate L_1 and L_2 of the Theorem.

For any TM M , let M' be a TM which, on any input, generates M 's description and simulates M on it.

Consider the following TM N :

On input M :

0. If M is not a TM description accept.
1. Construct M' and test whether $M' \in R'$.
2. If yes, then accept else reject.

The recursive language decided by N separates L_1 from L_2 .

Recursion Theorem

“Machines can reproduce themselves.”

Lemma 1. *There is a computable function $q : \Sigma^* \rightarrow \Sigma^*$, where, for any string w , $q(w)$ is the description of a TM machine P_w that prints out w and then halts.*

Proof: The following TM computes $q(w)$:

On input w :

1. Construct the following TM P_w :

On any input:

1. erase input
2. Write w on the tape (string).
3. Halt.

2. Output $\langle P_w \rangle$.

$$M(\mathbf{x}) = \langle M \rangle$$

We will construct a TM S that ignores its input and outputs its own description.

Print out this sentence.

Print out two copies of the following, the second one in quotes:

“Print out two copies of the following, the second one in quotes:”

Theorem 5. *There exists a TM that, on any input, outputs a copy of its own description.*

Proof

We will construct such a TM S . It has two parts A and B , i.e., $\langle S \rangle = \langle AB \rangle$, defined as follows.

1. $A = P_{\langle B \rangle}$

2. B :

On input $\langle M \rangle$, where M is a portion of a TM:

2.1 Compute $q(\langle M \rangle)$.

2.2 Combine the result with $\langle M \rangle$ to make a complete TM description.

2.3 Output this description and halt.

Recursion Theorem

Theorem 6. *Let T be a TM that computes a function $t : \Sigma^* \times \Sigma^* \rightarrow \Sigma^*$.*

There is a TM R that computes a function $r : \Sigma^ \rightarrow \Sigma^*$, where for every w ,*

$$r(w) = t(\langle R \rangle, w).$$

To make a TM R that can obtain its own description **and** then compute with it, we need only to make a machine T that takes an extra input that receives the description of the machine.

Proof

TM R has three parts A , B , and T .

1. $A = P_{\langle BT \rangle}$. (After A runs the tape contains $\langle BT \rangle$.)

2. $B = q(\langle BT \rangle)$ (Applies q to the output of A to get $\langle A \rangle$).

B then combines A , B , T into a single machine, writes its description on the tape.

3. T takes the output of B and the input w and computes with it.

Applications

Theorem 7. *H is undecidable.*

Proof: Suppose not. Then, let TM M_H decide H .

Construct the following TM B :

On input w :

1. Obtain, via the recursion theorem, own description $\langle B \rangle$.
2. Simulate M_H on input $\langle B, w \rangle$.
3. Accept if M_H rejects and reject if M_H accepts.

Running B on w does the opposite of what H says it does. Therefore H cannot decide H .

Another Non-R.E. language

Definition 2. *If M is a TM, then we say that the **length** of the description $\langle M \rangle$ of M is the number of symbols in the string describing M .*

*We say that M is **minimal** if there is no TM equivalent to M (in the language accepting sense) that has a shorter description.*

Let $MIN_{TM} = \{M \mid M \text{ is a minimal TM}\}$.

Theorem 8. *MIN_{TM} is not r.e.*

Proof

Assume that some TM E enumerates MIN_{TM} and we will obtain a contradiction.

Construct the following TM C :

On input w :

1. Obtain own description $\langle C \rangle$.
2. Run the enumerator E until a machine D appears with a longer description than that of C .
3. Simulate D on w .

Let D be a TM with a longer description than C in E 's list.

C simulates D , but is shorter than D , hence D cannot be minimal.

But D appears on E 's list. Contradiction.

A Fixed-Point Theorem

Theorem 9. *Let $t : \Sigma^* \rightarrow \Sigma^*$ be a computable function. Then there is a TM F wherein $t(\langle F \rangle)$ describes a TM equivalent to F .*

Proof: Let F be the following TM:

On input w :

1. Obtain own description $\langle F \rangle$.
 2. Compute $t(\langle F \rangle)$ to obtain the description of a TM G .
 3. Simulate G on w .
- $\langle F \rangle$ and $t(\langle F \rangle) = \langle G \rangle$ describe equivalent TMs.