Tutorial 4: Undecidability, Reducibility, Rice's Theorems

Problem 1 For a language L over the alphabet $\{0,1\}$, define the language

$$\mathrm{HALF}(L) = \{x \mid x \in \Sigma^*, \text{and there exists } y \in \Sigma^* \text{ such that } |x| = |y| \text{ and } xy \in L\}$$

Prove/Disprove the following statements.
(a) If L is R.E., then $\mathrm{HALF}(L)$ must be R.E.
(b) If L is recursive, then $\mathrm{HALF}(L)$ must be recursive.

Solution (a) Let M be a DTM recognising L.

Create an NTM N for $\mathrm{HALF}(L)$:

N on input x:
- Nondeterministically choose $y \in \Sigma^*$ such that $|x| = |y|$.
- Simulate M on xy.
- If M accepts, accept x.

Thus, $\mathrm{HALF}(L)$ is R.E.

(b) Let M be a total TM deciding L.

Create a total TM N for $\mathrm{HALF}(L)$:

N on input x:
- Check for all $y \in \Sigma^*$ and $|x| = |y|$:
  - Simulate M on xy.
- If M accepts in any of the cases, accept x. Otherwise, reject.

Thus, $\mathrm{HALF}(L)$ is recursive. □

Problem 2 Prove that the following language is not recursive:

$$\mathrm{WB} = \{M\#w \mid M \text{ writes the blank symbol in some step on input } w\}$$

Solution We show a reduction $\mathrm{HP} \leq_\mathrm{m} \mathrm{WB}$:

$$N\#v \mapsto M\#w$$
$$N \text{ halts on } v \iff M \text{ writes the blank symbol in some step on } w$$

M on input w:
- Simulate N on v.
  - Use a different symbol if N tries to write the blank symbol.
- If N halts, write the (actual) blank symbol on the input.

Thus, WB is not recursive. □

Problem 3 Prove:
(a) $E_{2026} = \{M \mid M \text{ halts on exactly 2026 inputs}\}$ is not R.E.
(b) $\mathrm{AL}_{2026} = \{M \mid M \text{ halts on at least 2026 inputs}\}$ is R.E. but not recursive.

Solution (a) Show a reduction $\overline{\mathrm{HP}} \leq_\mathrm{m} E_{2026}$:

$$M\#w \mapsto N$$
$$M \text{ does not halt on } w \iff N \text{ halts on exactly 2026 inputs}$$

N on input v:
- If $v = 0^k$, $k < 2026$, accept.
- Simulate M on w.
- If M halts, accept.

Denote by $L_H(N)$, the set of all inputs on which N halts. Then,

$$L_H(N) = \begin{cases} \Sigma^* & \text{if } M \text{ halts on } w \\ \{0^k \mid k < 2026\} & \text{if } M \text{ does not halt} \end{cases}$$

which satisfies the reduction constraint.

Thus, E_{2026} is not R.E.

(b) Create a TM N recognising AL_{2026}:

N on input M:
- Simulate M on all inputs on a time sharing basis.
- If M halts on at least 2026 inputs, accept.

Thus, AL_{2026} is R.E.

Show a reduction from $\mathrm{HP} \leq_\mathrm{m} \mathrm{AL}_{2026}$:

$$M\#w \mapsto N$$
$$M \text{ halts on } w \iff N \text{ halts on at least 2026 inputs}$$

N on input v:
- Simulate M on w.
- If M halts, accept v.

Here,

$$L_H(N) = \begin{cases} \Sigma^* & \text{if } M \text{ halts on } w \\ \phi & \text{otherwise} \end{cases}$$

Thus, the reduction is correct, and AL_{2026} is not recursive. □

Problem 4 Let k be a constant positive integer. Consider the following languages:
- $\mathrm{LE}_k = \{M \mid \text{DTM } M \text{ loops on at most } k \text{ inputs}\}$
- $\mathrm{LT}_k = \{M \mid \text{DTM } M \text{ loops on less than } k \text{ inputs}\}$
- $\mathrm{GE}_k = \{M \mid \text{DTM } M \text{ loops on at least } k \text{ inputs}\}$
- $\mathrm{GT}_k = \{M \mid \text{DTM } M \text{ loops on more than } k \text{ inputs}\}$

Prove that all these languages are non-R.E. and non-co-R.E.

Solution First note that $\overline{\mathrm{LE}_k} = \mathrm{GT}_k$, $\overline{\mathrm{LT}_k} = \mathrm{GE}_k$. So we only need to show:
- $\overline{\mathrm{HP}} \leq_\mathrm{m} \mathrm{LE}_k$ and $\overline{\mathrm{HP}} \leq_\mathrm{m} \mathrm{GT}_k$ (LE_k and GT_k are non-R.E. and non-co-R.E.)
- $\overline{\mathrm{HP}} \leq_\mathrm{m} \mathrm{LT}_k$ and $\overline{\mathrm{HP}} \leq_\mathrm{m} \mathrm{GE}_k$ (LT_k and GE_k are non-R.E. and non-co-R.E.)

$\overline{\mathrm{HP}} \leq_\mathrm{m} \mathrm{LE}_k$:

$$M\#w \mapsto N$$
$$M \text{ does not halt on } w \iff N \text{ loops on at most } k \text{ inputs}$$

$$M \text{ does not halt on } w \Rightarrow N \text{ loops on at most } k \text{ inputs}$$
$$M \text{ halts on } w \Rightarrow N \text{ loops on more than } k \text{ inputs}$$

N on input v:

- Simulate M on w for $|v|$ steps.
- If M did not halt yet, accept (and halt). Otherwise, loop.

Let K' denote the set of strings on which N loops.

If M does not halt on w, then $K' = \phi$. If M halts on w in s steps, then, $K' = \{v \mid |v| \geq s\}$. So,

$$|K'| = \begin{cases} 0 & \text{if } M \text{ does not halt on } w \\ \infty & \text{if } M \text{ halts on } w \end{cases}$$

which satisfies our reduction constraint. Hence, $\overline{\mathrm{HP}} \leq_{\mathrm{m}} \mathrm{LE}_k$.

(The same construction also shows $\overline{\mathrm{HP}} \leq_{\mathrm{m}} \mathrm{LT}_k$.)

$\overline{\mathrm{HP}} \leq_{\mathrm{m}} \mathrm{GT}_k$:

$$M\#w \mapsto N$$
$$M \text{ does not halt on } w \iff N \text{ loops on more than } k \text{ inputs}$$

$$M \text{ does not halt on } w \Rightarrow N \text{ loops on more than } k \text{ inputs}$$
$$M \text{ halts on } w \Rightarrow N \text{ loops on at most } k \text{ inputs}$$

N on input v:

- Simulate M on w.
- If M halts on w, accept v.

If M halts on w, $K' = \phi$. If M does not halt, then $K' = \Sigma^*$. So

$$|K'| = \begin{cases} \infty & \text{if } M \text{ does not halt on } w \\ 0 & \text{if } M \text{ halts on } w \end{cases}$$

which satisfies our reduction constraint. Hence, $\overline{\mathrm{HP}} \leq_{\mathrm{m}} \mathrm{GT}_k$.

(The same construction also shows $\overline{\mathrm{HP}} \leq_{\mathrm{m}} \mathrm{GE}_k$.) □

Problem 5 Let $\mathrm{nsteps}(M, w)$ denote the number of steps of M on w. If M loops on w, take $\mathrm{nsteps}(M, w) = \infty$. If N also loops on v, take $\mathrm{nsteps}(M, w) = \mathrm{nsteps}(N, v)$.

Prove: recursive / R.E but not recursive / non-R.E.?

(a) $L_a = \{M\#N \mid \mathrm{nsteps}(M, \varepsilon) < \mathrm{nsteps}(N, \varepsilon)\}$
(b) $L_b = \{M\#N \mid \mathrm{nsteps}(M, \varepsilon) \leq \mathrm{nsteps}(N, \varepsilon)\}$
(c) $L_c = \{M\#N \mid \mathrm{nsteps}(M, w) < \mathrm{nsteps}(N, v) \text{ for some } w, v\}$
(d) $L_d = \{M\#N \mid \mathrm{nsteps}(M, w) < \mathrm{nsteps}(N, v) \text{ for all } w, v\}$

Solution (a) R.E. but not recursive.

To create a recogniser for L_a, simulate both M and N on ε parallelly, and if M halts earlier, then accept.

To show $\mathrm{HP} \leq_{\mathrm{m}} L_a$:

$$K\#w \mapsto M\#N$$
$$K \text{ halts on } w \iff \mathrm{nsteps}(M, \varepsilon) < \mathrm{nsteps}(N, \varepsilon)$$

$$K \text{ halts on } w \Rightarrow \text{nsteps}(M, \varepsilon) < \text{nsteps}(N, \varepsilon)$$
$$K \text{ does not halt on } w \Rightarrow \text{nsteps}(M, \varepsilon) \geq \text{nsteps}(N, \varepsilon)$$

M on input v_1:
- Simulate K on w.
- If it halts, accept.

N on input v_2:
- Loop unconditionally.

Here, we have
- If K halts on w, then $\text{nsteps}(M, \varepsilon) = \text{finite}$, $\text{nsteps}(N, \varepsilon) = \infty$; so $\text{nsteps}(M, \varepsilon) < \text{nsteps}(N, \varepsilon)$.
- If K does not halt on w, then $\text{nsteps}(M, \varepsilon) = \infty = \text{nsteps}(N, \varepsilon)$; so $\text{nsteps}(M, \varepsilon) \geq \text{nsteps}(N, \varepsilon)$.

So, L_a is non recursive.

(b) non–R.E.

Intuitively, we cannot create a recogniser for L_b, because in the case where $\text{nsteps}(M, \varepsilon) = \text{nsteps}(N, \varepsilon) = \infty$, we cannot accept $M\#N$ in finite time. (Their simulation will keep on looping and never halt, so our recogniser can't accept the input.)

To show $\overline{\text{HP}} \leq_{\text{m}} L_b \equiv \text{HP} \leq_{\text{m}} \overline{L_b}$.

Note that $\overline{L_b} = \{M\#N \mid \text{nsteps}(M, \varepsilon) > \text{nsteps}(N, \varepsilon)\}$ is essentially the same as L_a, just with the roles of M and N reversed. So the same construction as in (a) can be used (by switching M and N) to show $\text{HP} \leq_{\text{m}} \overline{L_b}$.

Thus, L_b is non-R.E.

(c) R.E. but non-recursive.

Consider an NTM which nondeterministically chooses some w and v to run M and N on (parallelly), and if M halts before N, then accepts. This is a recogniser for L_c.

$\text{HP} \leq_{\text{m}} L_c$ follows from the same construction as (a).

$$K\#x \mapsto M\#N$$
$$K \text{ halts on } x \iff \exists w, v \quad \text{nsteps}(M, w) < \text{nsteps}(N, v)$$

$$K \text{ halts on } x \Rightarrow \exists w, v \quad \text{nsteps}(M, w) < \text{nsteps}(N, v)$$
$$K \text{ does not halt on } x \Rightarrow \forall w, v \quad \text{nsteps}(M, w) \geq \text{nsteps}(N, v)$$

From the construction of M and N in (a),
- If K halts on x, then $\text{nsteps}(M, w) = \text{finite}$, $\text{nsteps}(N, v) = \infty$; so $\text{nsteps}(M, w) < \text{nsteps}(N, v)$, for any w, v.
- If K does not halt on x, then $\text{nsteps}(M, w) = \infty = \text{nsteps}(N, v)$; so $\text{nsteps}(M, w) \geq \text{nsteps}(N, v)$, for all w, v.

So, L_c is non recursive.

(d) non-R.E.

To show $\overline{\text{HP}} \leq_{\text{m}} L_d$:

$$K\#x \mapsto M\#N$$

$$K \text{ does not halt on } x \iff \text{nsteps}(M, w) < \text{nsteps}(N, v) \quad \forall w, v$$

$$K \text{ does not halt on } x \Rightarrow \forall w, v \quad \text{nsteps}(M, w) < \text{nsteps}(N, v)$$
$$K \text{ halts on } x \Rightarrow \exists w, v \quad \text{nsteps}(M, w) \geq \text{nsteps}(N, v)$$

M on input w:
- Simulate K on x for $|w|$ steps.
- If K did not halt yet, accept w (and halt). Otherwise, loop.

N on input v:
- Simulate K on x.
- If it halts, accept v.

If K does not halt on x, then
- M will halt on all strings in finite steps.
- N will loop on every string.
- So, we have $\forall w, v \quad \text{nsteps}(M, w) < \text{nsteps}(N, v)$

If K halts on x, say in s steps:
- M loops on strings w, with $|w| \geq s$.
- N will halts on all strings in finite steps.
- So, we have $\exists w, v \quad \text{nsteps}(M, w) \geq \text{nsteps}(N, v)$

Thus, the reduction is correct.

Therefore, L_d is non-R.E. □

Problem 6 Prove that the following languages are not recursive.

(a) $L_1 = \{M\#N \mid L(M) = L(N)\}$

(b) $L_2 = \{M\#N \mid L(M) \subseteq L(N)\}$

(c) $L_3 = \{M\#N \mid L(M) \cap L(N) = \phi\}$

(d) $L_4 = \{M\#N\#P \mid L(M) \cap L(N) = L(P)\}$

Solution (a) To show $\text{HP} \leq_{\text{m}} L_1$:

$$K\#w \mapsto M\#N$$
$$K \text{ halts on } w \iff L(M) = L(N)$$

M on input v_1:
- Accept v_1.

N on input v_2:
- Simulate K on w.
- If K halts, then accept v_2.

$$L(M) = \Sigma^*$$
$$L(N) = \begin{cases} \Sigma^* & \text{if } K \text{ halts on } w \\ \phi & \text{if } K \text{ does not halt on } w \end{cases}$$

So, if K halts on w, $L(M) = L(N)$ and if K does not halt on w, then $L(M) \neq L(N)$, completing the reduction. Thus, L_1 is non-recursive.

(b) To show $\text{HP} \leq_{\text{m}} L_2$.

Use the same construction as in (a).

If K halts on w, $L(M) \subseteq L(N)$, and if K does not halt on w, then $L(M) \not\subseteq L(N)$, completing the reduction. Thus, L_2 is non-recursive.

(c) Here we will show $\overline{\mathrm{HP}} \leq_{\mathrm{m}} L_3$.

$$K\#w \mapsto M\#N$$
$$K \text{ does not halt on } w \iff L(M) \cap L(N) = \phi$$

$$K \text{ does not halt on } w \Rightarrow L(M) \cap L(N) = \phi$$
$$K \text{ halts on } w \Rightarrow L(M) \cap L(N) \neq \phi$$

Again use the same construction as in (a).

If K halts on w, $L(M) \cap L(N) = \Sigma^* \neq \phi$. If K does not halt on w, then $L(M) \cap L(N) = \phi$.

Thus, L_3 is non-R.E., and consequently also non-recursive.

(d) To show $\overline{\mathrm{HP}} \leq_{\mathrm{m}} L_4$.

$$K\#w \mapsto M\#N\#P$$
$$K \text{ does not halt on } w \iff L(M) \cap L(N) = L(P)$$

$$K \text{ does not halt on } w \Rightarrow L(M) \cap L(N) = L(P)$$
$$K \text{ halts on } w \Rightarrow L(M) \cap L(N) \neq L(P)$$

M on input v_1:

- Simulate K on w.
- If K halts, accept v_1.

N on input v_2:

- Accept v_2

P on input v_3:

- Reject v_3.

Here,

$$L(M) = \begin{cases} \Sigma^* & \text{if } K \text{ halts on } w \\ \phi & \text{if } K \text{ does not halt on } w \end{cases}$$
$$L(N) = \Sigma^*$$
$$L(P) = \phi$$

So, if K halts on w, $L(M) \cap L(N) = \Sigma^* \neq \phi = L(P)$. If K does not halt on w, then $L(M) \cap L(N) = \phi = L(P)$.

Thus, L_4 is non-R.E. and thus consequently non-recursive. □

Problem 7 Prove that the following problems on a TM M are decidable.

(a) Decide whether M halts on some input within 2026 steps.

(b) Decide whether M on a given input w moves left at least 2026 times.

Solution (a) Simulate M on all possible inputs w, $|w| \leq 2026$, parallelly, for 2026 steps. If it halts on any one of them (within 2026 steps), then accept M. Otherwise, reject.

Why does this work?

- **Case 1:** M does not halt on any input within 2026 steps. Then our decider correctly rejects M (since it won't halt in 2026 steps on our chosen inputs).
- **Case 2:** M halts on some input w^* within 2026 steps.
  - If $|w^*| \leq 2026$, then again our decider correctly accepts M since its simulation of w^* will halt within 2026 steps.
  - If $|w^*| > 2026$, and we know that M halts on w^* within 2026 steps. This must mean that M only ever sees at most 2026 characters of w^* from the start (to see more characters, it would require more steps). Then, consider the prefix α of w^* consisting of its first 2026 characters. The behaviour of M on α should be identical to that on w^*, since the part of the input tape which is reachable by the machine is identical. Therefore, M halts on α within 2026 steps. Hence, our decider correctly accepts M, since its simulation of α will halt within 2026 steps.

(b) Simulate M on w for $2026 \times (|w| + |Q|)$ steps. During the simulation, if M moves left at least 2026 times, accept. Otherwise, reject.

Why does this work? Between any two L moves that M makes, what is the maximum number of R moves it can take? If it is at the start of the tape (at the $\vdash$), it can go right for $|w|$ steps, reaching the end of the string. Then, it can go further right, finding a blank symbol, say at state q_1. It may move right again, going to some state q_2. If it keeps moving right, at some point it will repeat states and be stuck in a loop, forever moving right (because it keeps seeing $\sqcup$ symbols). So, we can bound the number of consecutive right moves M can take before taking a left move, to be $|w| + |Q|$. So between two L moves, there can be at most $|w| + |Q|$ R moves. Thus, it follows that if M has at least k left moves, they must happen within $k(|w| + |Q|)$ steps. □

Problem 8 Use Rice's theorems to prove that neither the following languages nor their complements are R.E.

(a) FIN $= \{M \mid L(M) \text{ is finite}\}$.

(b) REG $= \{M \mid L(M) \text{ is regular}\}$.

Solution (a) FIN is non monotone, since $\phi \subseteq \Sigma^*$, but $P_{\text{FIN}}(\phi) = T$, and $P_{\text{FIN}}(\Sigma^*) = F$. Therefore by Rice's theorem, FIN is non-R.E.

$\overline{\text{FIN}}$ is monotone, so we cannot comment on it by Rice theorem. We must show $\overline{\text{HP}} \leq_{\text{m}} \overline{\text{FIN}}$ to prove it is non-R.E.

(b) Let $A = \{a^p \mid p \text{ is prime}\}$.

REG is non monotone, since $\phi \subseteq A$, but $P_{\text{REG}}(\phi) = T$ and $P_{\text{REG}}(A) = F$. Therefore by Rice's theorem, REG is non-R.E.

$\overline{\text{REG}}$ is non monotone, since $A \subseteq \Sigma^*$, but $P_{\overline{\text{REG}}}(A) = T$ and $P_{\overline{\text{REG}}}(\Sigma^*) = F$. Therefore by Rice's theorem, $\overline{\text{REG}}$ is non-R.E. □

Problem 9 Prove/disprove: No non-trivial property of R.E. languages is semidecidable.

Solution Counter example: Non-emptiness.

($\overline{E_{\text{TM}}} = \{M \mid L(M) \neq \phi\}$ is a non trivial property, which is semidecidable.) □