CM0081 Automata and Formal Languages
§ 9.3 Undecidable Problems About Turing Machines

Andrés Sicard-Ramírez

Universidad EAFIT

Semester 2024-1

Preliminaries

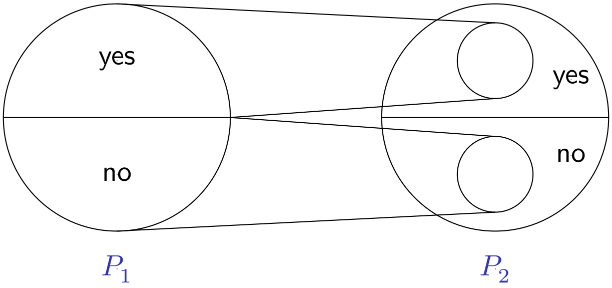
Conventions

- The number and page numbers assigned to chapters, examples, exercises, figures, quotes, sections and theorems on these slides correspond to the numbers assigned in the textbook [Hopcroft, Motwani and Ullman 2007].
- The natural numbers include the zero, that is, $\mathbb{N} = \{0, 1, 2, \dots\}$.
- The power set of a set A, that is, the set of its subsets, is denoted by $\mathcal{P}\, A$.

Reductions

Definition

Let P_1 and P_2 be two problems. A **reduction** from P_1 to P_2 is a Turing machine that takes an instance of P_1 written on its tape and halts with an instance of P_2 that have the same answer (i.e. a reduction is an algorithm).



Reductions

Theorem 9.7

If there is a reduction from P_1 to P_2 then:

(i) if P_1 is undecidable then so P_2,

Reductions

Theorem 9.7

If there is a reduction from P_1 to P_2 then:

(i) if P_1 is undecidable then so P_2,

(ii) if P_1 is not recursively enumerable then so P_2.

Reductions

Theorem 9.7

If there is a reduction from P_1 to P_2 then:

(i) if P_1 is undecidable then so P_2,

(ii) if P_1 is not recursively enumerable then so P_2.

Proof

Hint: Suppose the P_2 is decidable/recursively enumerable and find a contradiction.

Turing Machines that Accept the Empty Language

Notation

Henceforth, we'll regard strings as the Turing machines they represent.

Two languages

Let $\Sigma = \{0, 1\}$. Then

$$\mathrm{L_e} := \{\, M \in \Sigma^* \mid \mathrm{L}(M) = \emptyset \,\},$$
$$\mathrm{L_{ne}} := \{\, M \in \Sigma^* \mid \mathrm{L}(M) \neq \emptyset \,\}.$$

Turing Machines that Accept the Empty Language

Theorem 9.8

The language L_{ne} is recursively enumerable.

† Figure from Hopcroft, Motwani and Ullman [2007, Fig. 9.8].

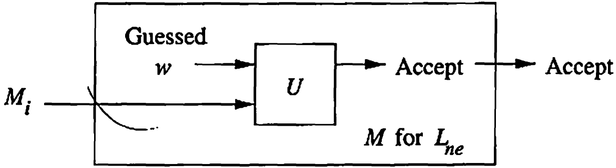
Turing Machines that Accept the Empty Language

Theorem 9.8

The language L_{ne} is recursively enumerable.

Proof

Construction of a non-determinist Turing machine to accept L_{ne}:[†]



■

[†]Figure from Hopcroft, Motwani and Ullman [2007, Fig. 9.8].

Turing Machines that Accept the Empty Language

Theorem 9.9

The language L_{ne} is not recursive.

Turing Machines that Accept the Empty Language

Proof

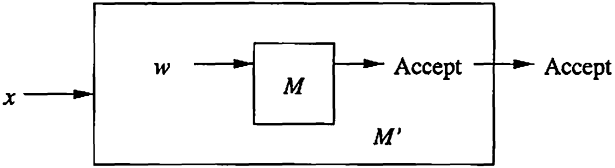
1. Reduction from $\mathrm{L_u}$ to $\mathrm{L_{ne}}$ where the pair (M, w) is converted in M', such that $w \in \mathrm{L}(M)$ iff $\mathrm{L}(M') \neq \emptyset$.

† Figure from Hopcroft, Motwani and Ullman [2007, Fig. 9.9].

Turing Machines that Accept the Empty Language

Proof

1. Reduction from $\mathrm{L_u}$ to $\mathrm{L_{ne}}$ where the pair (M, w) is converted in M', such that $w \in \mathrm{L}(M)$ iff $\mathrm{L}(M') \neq \emptyset$.
2. The key is that M' ignores its input.[†]

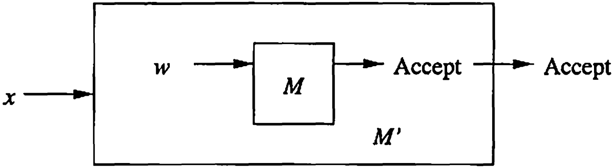


[†]Figure from Hopcroft, Motwani and Ullman [2007, Fig. 9.9].

Turing Machines that Accept the Empty Language

Proof

1. Reduction from $\mathrm{L_u}$ to $\mathrm{L_{ne}}$ where the pair (M, w) is converted in M', such that $w \in \mathrm{L}(M)$ iff $\mathrm{L}(M') \neq \emptyset$.
2. The key is that M' ignores its input.[†]



3. $\mathrm{L_{ne}}$ is not recursive by Theorem 9.7. ∎

[†]Figure from Hopcroft, Motwani and Ullman [2007, Fig. 9.9].

Turing Machines that Accept the Empty Language

Theorem 9.10

The language $\mathrm{L_e}$ is not recursively enumerable.

Turing Machines that Accept the Empty Language

Theorem 9.10

The language $\mathrm{L_e}$ is not recursively enumerable.

Proof

Hint: The language $\mathrm{L_e}$ is the complement of the language $\mathrm{L_{ne}}$.

Rice's Theorem

Set of the recursively enumerable languages

$$\mathcal{RE} := \{\, L \subseteq \Sigma^* \mid L \text{ is a recursively enumerable language} \,\}.$$

Rice's Theorem

Set of the recursively enumerable languages

$$\mathcal{RE} := \{\, L \subseteq \Sigma^* \mid L \text{ is a recursively enumerable language} \,\}.$$

Properties (subsets) of the recursively enumerable languages

Property P of $\mathcal{RE}$: $P \subseteq \mathcal{RE}$.

Rice's Theorem

Set of the recursively enumerable languages

$$\mathcal{RE} := \{\, L \subseteq \Sigma^* \mid L \text{ is a recursively enumerable language} \,\}.$$

Properties (subsets) of the recursively enumerable languages

Property P of $\mathcal{RE}$: $P \subseteq \mathcal{RE}$.

Trivial properties

$P(L) = \emptyset$ or $P(L) = \mathcal{RE}$.

Rice's Theorem

Set of the recursively enumerable languages

$$\mathcal{RE} := \{\, L \subseteq \Sigma^* \mid L \text{ is a recursively enumerable language} \,\}.$$

Properties (subsets) of the recursively enumerable languages

Property P of $\mathcal{RE}$: $P \subseteq \mathcal{RE}$.

Trivial properties

$P(L) = \emptyset$ or $P(L) = \mathcal{RE}$.

Example

$P(L)$: L is a language regular.

Rice's Theorem

Theorem 9.11 (Rice's theorem, first version)

Every non-trivial property of $\mathcal{RE}$ is undecidable [Rice 1953].

Rice's Theorem

Theorem 9.11 (Rice's theorem, first version)

Every non-trivial property of $\mathcal{RE}$ is undecidable [Rice 1953].

How to prove Rice's theorem?

We identify a property P by the Turing machines M such that $\mathrm{L}(M) \in P$.

Rice's Theorem

Theorem 9.11 (Rice's theorem, first version)

Every non-trivial property of $\mathcal{RE}$ is undecidable [Rice 1953].

How to prove Rice's theorem?

We identify a property P by the Turing machines M such that $\mathrm{L}(M) \in P$.

Theorem (Rice's theorem, second version)

If $P \subseteq \mathcal{RE}$ is a non-trivial property then

$$L_P := \{ M \in \Sigma^* \mid \mathrm{L}(M) \in P \}$$

is undecidable.

Rice's Theorem

Proof

Case $\emptyset \notin P$.

1. Let L be a language and M_L be a Turing machine such $L \neq \emptyset$, $L \in P$ and $L = \mathrm{L}(M_L)$.
   Reduction from $\mathrm{L_u}$ to L_P where the pair (M, w) is converted in M' such that:[†]
   (i) $\mathrm{L}(M') = \emptyset$ (i.e. $M' \notin L_P$) if $w \notin \mathrm{L}(M)$ and
   (ii) $\mathrm{L}(M') = L$ (i.e. $M' \in L_P$) if $w \in \mathrm{L}(M)$.

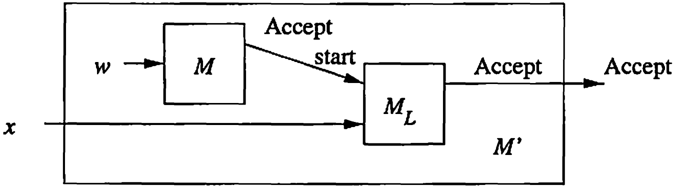
[†]Figure from Hopcroft, Motwani and Ullman [2007, Fig. 9.10].

Rice's Theorem

Proof

Case $\emptyset \notin P$.

1. Let L be a language and M_L be a Turing machine such $L \neq \emptyset$, $L \in P$ and $L = \mathrm{L}(M_L)$. Reduction from $\mathrm{L_u}$ to L_P where the pair (M, w) is converted in M' such that:[†]
   (i) $\mathrm{L}(M') = \emptyset$ (i.e. $M' \notin L_P$) if $w \notin \mathrm{L}(M)$ and
   (ii) $\mathrm{L}(M') = L$ (i.e. $M' \in L_P$) if $w \in \mathrm{L}(M)$.



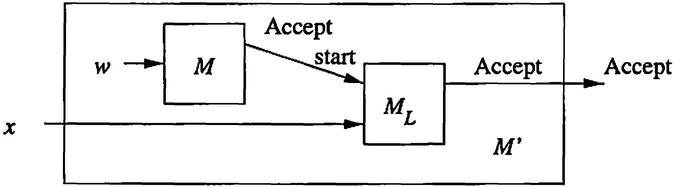
[†]Figure from Hopcroft, Motwani and Ullman [2007, Fig. 9.10].

Rice's Theorem

Proof

Case $\emptyset \notin P$.

1. Let L be a language and M_L be a Turing machine such $L \neq \emptyset$, $L \in P$ and $L = \mathrm{L}(M_L)$. Reduction from $\mathrm{L_u}$ to L_P where the pair (M, w) is converted in M' such that:[†]
   (i) $\mathrm{L}(M') = \emptyset$ (i.e. $M' \notin L_P$) if $w \notin \mathrm{L}(M)$ and
   (ii) $\mathrm{L}(M') = L$ (i.e. $M' \in L_P$) if $w \in \mathrm{L}(M)$.



2. L_P is not recursive by Theorem 9.7.a.

[†]Figure from Hopcroft, Motwani and Ullman [2007, Fig. 9.10].

Rice's Theorem

Proof (continuation)

Case $\emptyset \in P$.

1. By the previous case, $\overline{P}$ is undecidable, i.e. $L_{\overline{P}}$ is undecidable.
2. $\overline{L_P} = L_{\overline{P}}$.
3. Suppose L_P is decidable then $\overline{L_P}$ would be also decidable (contradiction).
4. Therefore, L_P is undecidable.

■

Rice's Theorem

Observation

All problems about Turing machines that involve only the language that the Turing machine accepts are undecidable.

Rice's Theorem

Observation

All problems about Turing machines that involve only the language that the Turing machine accepts are undecidable.

Examples

- Is the language accepted by the Turing machine empty? Is it finite? Is it regular? Is it context-free?
- Does the language accepted by the Turing machine contain the string 'hello world'? Does it contain all the even numbers?

Rice's Theorem

Observation

Rice's theorem does not imply that everything about Turing machines is undecidable.

Rice's Theorem

Observation

Rice's theorem does not imply that everything about Turing machines is undecidable.

Example

It is decidable if a Turing machine has five states.

References

Hopcroft, J. E., Motwani, R. and Ullman, J. D. [1979] (2007). Introduction to Automata Theory, Languages, and Computation. 3rd ed. Pearson Education (cit. on pp. 2, 8, 9, 11–13, 23–25).

Rice, H. G. (1953). Classes of Recursively Enumerable Sets and Their Decision Problems. Transactions of the American Mathematical Society 74.2, pp. 358–366. DOI: 10.1090/S0002-9947-1953-0053041-6 (cit. on pp. 20–22).