CM0081 Formal Languages and Automata § 9.2 An Undecidable Problem That Is Recursively Enumerable

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 (1979) 2007].
- The natural numbers include the zero, that is,  $\mathbb{N} = \{0, 1, 2, ...\}$ .

The power set of a set A, that is, the set of its subsets, is denoted by  $\mathcal{P}A$ .

Theorem 9.3

If L is a recursive language, then  $\overline{L}$  is also a recursive language.

Proof Whiteboard. Theorem 9.3

If L is a recursive language, then  $\overline{L}$  is also a recursive language.

Proof

Whiteboard.

Theorem 9.4

If both L and  $\overline{L}$  are recursively enumerable languages, then L is recursive (and  $\overline{L}$  is recursive as well by Theorem 9.3).

Proof

Whiteboard.

Possible relations between a language L and its complement  $\overline{L}$ 

(i) Both L and  $\overline{L}$  are recursive.

Possible relations between a language L and its complement  $\overline{L}$ 

- (i) Both L and  $\overline{L}$  are recursive.
- (ii) Neither L nor  $\overline{L}$  are recursively enumerable.

Possible relations between a language L and its complement  $\overline{L}$ 

- (i) Both L and  $\overline{L}$  are recursive.
- (ii) Neither L nor  $\overline{L}$  are recursively enumerable.
- (iii) L is recursively enumerable but not recursive and  $\overline{L}$  is not recursively enumerable.

Possible relations between a language L and its complement  $\overline{L}$ 

- (i) Both L and  $\overline{L}$  are recursive.
- (ii) Neither L nor  $\overline{L}$  are recursively enumerable.
- (iii) L is recursively enumerable but not recursive and  $\overline{L}$  is not recursively enumerable.
- (iv)  $\overline{L}$  is recursively enumerable but not recursive and L is not recursively enumerable.

#### Exercise 9.2.5

Let L be recursively enumerable and let  $\overline{L}$  be non recursively enumerable. Consider the language

```
L' = \{ 0w \mid w \text{ is in } L \} \cup \{ 1w \mid w \text{ is not in } L \}.
```

Can you say for certain whether L' is recursive, recursively enumerable, or non recursively enumerable? Justify your answer.

### Solution (from Hopcroft, Motwani and Ullman [(1979) 2007])

Suppose L' were recursively enumerable. Then we could design a Turing machine M for  $\overline{L}$  as follows. Given input w, M changes its input to 1w and simulates the hypothetical Turing machine for L'. If that Turing machine accepts, then w is in  $\overline{L}$ , so M should accept. If the Turing machine for L' never accepts, then neither does M. Thus, M would accept exactly  $\overline{L}$ , which contradicts the fact that  $\overline{L}$  is not recursively enumerable. We conclude that L' is not recursively enumerable.

#### Theorems About Recursive Languages

Conventions

- 1. (M, w): Represents the Turing machine M with input w.
- 2. w is a string of 0's and 1's.

Codification of a Turing machine with an input

Let  $w_i$  be the codification of a Turing machine M. The codification of (M, w) is defined by

 $\overrightarrow{(M,w)}\coloneqq w_i111w.$ 

Definition

Let  $\Sigma = \{0, 1\}$ . The **universal language**, denoted  $L_u$ , is the set of binary strings that encode a pair (M, w) such that  $w \in L(M)$ , that is,

$$\mathcal{L}_{\mathbf{u}} \coloneqq \Big\{ \overrightarrow{(M,w)} \in \Sigma^* \; \Big| \; w \in \mathcal{L}(M) \; \Big\}.$$

#### Theorem

The language  $L_u$  is recursively enumerable.

### Idea of the proof

There exists a Turing machine U such that  $L_u=L(U).$  The machine U is called a universal Turing machine.

Theorem 9.6

The language  $\boldsymbol{L}_{\boldsymbol{u}}$  is recursively enumerable but not recursive.

Proof of  $\mathrm{L}_\mathrm{u}$  is not recursive (by contradiction (proof of negation))

Suppose  $\boldsymbol{L}_{u}$  is recursive

 $\Rightarrow \overline{L_u}$  is recursive

 $\Rightarrow L_d$  is recursive (see next slide)

 $\Rightarrow$  Contradiction because  $L_d$  is non recursively enumerable  $~~\blacksquare~~$ 

### From the recursiveness of $\overline{L_u}$ to the recursiveness of $L_d$

Given a terminating Turing machine for accepting  $\overline{L_u}$  we could use this machine for building a terminating Turing machine for accepting  $L_d$ .<sup>†</sup>



Since  $L_d$  is not recursive (because it is not recursive enumerable) then  $\overline{L_u}$  is not recursive.

The Universal Language

<sup>&</sup>lt;sup>†</sup>Figure from Hopcroft, Motwani and Ullman [(1979) 2007, Fig. 9.6].

 $\text{Code for } \mathrm{U}$ 

Since U is a Turing machine exists i (1654 digits) such that  $U = M_i$  given by (using a different codification) [Penrose 1991, pp. 56-57]:

(continued on next slide)

- Based on M-functions (subroutines with parameters) [Sicard 1997; Copeland 2004b].
- The machine is composed by 12 symbols and 4.000 instructions, approximately [Sicard Ramírez 1998].

# Small Universal Turing Machines

Notation

Let UTM(m, n) be the class of universal Turing machines with m states and n symbols.

Notation

Let UTM(m, n) be the class of universal Turing machines with m states and n symbols.

Theorem If  $UTM(m, n) \neq \emptyset$  then [Shannon 1956]:

(i)  $\operatorname{UTM}(2,n') \neq \emptyset$ , where n' is at most 4mn + n and

(ii)  $\operatorname{UTM}(m',2) \neq \emptyset$ , where  $m' = (2^l - 1)m$  and l is the smaller integer such that  $m \leq 2^l$ .

Theorem

There exists universal Turing machines in the following classes [Rogozhin 1996; Neary and Woods 2012]:

$\mathrm{UTM}(m,n)$	Author(s)
(24, 2)	Rogozhin [1996]
(19,2)	Baiocchi [2001]
(18,2)	Neary and Woods [2007]
(15,2)	Neary and Woods [2009]

(continued on next slide)

Theorem (continuation)

$\mathrm{UTM}(m,n)$	Author(s)
(10, 3)	Rogozhin [1996]
(9,3)	Neary and Woods [2009]
(7,4)	Rogozhin [1996]
(6,4)	Neary and Woods [2009]
(5,5)	Rogozhin [1996]
(4, 6)	Rogozhin [1996]
(3, 10)	Rogozhin [1996]
(2, 18)	Rogozhin [1996]

Theorem

The following classes are empty [Rogozhin 1996; Neary and Woods 2012]:

$\mathrm{UTM}(m,n)$	Author(s)
(m,1)	trivial
(3,2)	Rogozhin [1996]
(2,3)	Rogozhin [1996]
(2,2)	Rogozhin [1996]
(1,n)	Herman [1968]

## Wolfram Turing Machine



# Conway's Game of Life

Examples

(From Wikipedia)

Pulsar (Oscillator) Author: Jokey Smurf Glider (Spaceship) Author: Rodrigo Silveira Camargo

Universal Turing Machines

# Conway's Game of Life

Rules

(i) Any live cell with fewer than two live neighbours dies, as if caused by under-population.

# Conway's Game of Life

Rules

(i) Any live cell with fewer than two live neighbours dies, as if caused by under-population.(ii) Any live cell with two or three live neighbours lives on to the next generation.

Rules

(i) Any live cell with fewer than two live neighbours dies, as if caused by under-population.(ii) Any live cell with two or three live neighbours lives on to the next generation.(iii) Any live cell with more than three live neighbours dies, as if by overcrowding.

Rules

- (i) Any live cell with fewer than two live neighbours dies, as if caused by under-population.
- (ii) Any live cell with two or three live neighbours lives on to the next generation.
- (iii) Any live cell with more than three live neighbours dies, as if by overcrowding.
- (iv) Any dead cell with exactly three live neighbours becomes a live cell, as if by reproduction.

Rules

- (i) Any live cell with fewer than two live neighbours dies, as if caused by under-population.
- (ii) Any live cell with two or three live neighbours lives on to the next generation.
- (iii) Any live cell with more than three live neighbours dies, as if by overcrowding.
- (iv) Any dead cell with exactly three live neighbours becomes a live cell, as if by reproduction.

### Theorem

It is possible codified a universal Turing machine in Conway's Game of Life [Rendell 2011].

### The language of the halting problem

Let  $\Sigma = \{0, 1\}$ . The original Turing machine accepted by halting, no by final state.

 $\mathrm{H}(M) \coloneqq \{ \, w \in \Sigma^* \mid M \text{ halts given the input } w \, \}.$ 

### The language of the halting problem

Let  $\Sigma = \{0, 1\}$ . The original Turing machine accepted by halting, no by final state.

```
\mathrm{H}(M) \coloneqq \{ w \in \Sigma^* \mid M \text{ halts given the input } w \}.
```

We define the language of the halting problem by

$$\mathcal{L}_{\mathrm{hp}} \coloneqq \Big\{ \, \overrightarrow{(M,w)} \in \Sigma^* \; \Big| \; w \in \mathcal{H}(M) \, \Big\}.$$

### The language of the halting problem

Let  $\Sigma = \{0, 1\}$ . The original Turing machine accepted by halting, no by final state.

```
\mathrm{H}(M) \coloneqq \{ w \in \Sigma^* \mid M \text{ halts given the input } w \}.
```

We define the language of the halting problem by

$$\mathcal{L}_{\mathrm{hp}} \coloneqq \Big\{ \, \overrightarrow{(M,w)} \in \Sigma^* \; \Big| \; w \in \mathcal{H}(M) \, \Big\}.$$

Exercise 9.2.1 Show that  $L_{\rm hp}$  is recursively enumerable but not recursive.

# The Halting Problem: State of $Art^{\dagger}$



<sup>&</sup>lt;sup>†</sup>Figure from [Margenstern 2000].

The Halting Problem

Observation

The halting problem was introduced and named by Davis [1958, p. 70] not by Turing himself, contrary to popular belief [Copeland 2004a, p. 40].

Observation

The halting problem was introduced and named by Davis [1958, p. 70] not by Turing himself, contrary to popular belief [Copeland 2004a, p. 40].

Practical approach

'In contrast to popular belief, proving termination is not always impossible.' [Cook, Podelski and Rybalchenko 2011, p. 1]

### References

- Baiocchi, C. (2001). Three Small Universal Turing Machines. In: Machines, Computations, and Universality (MCU 2001). Ed. by Margenstern, M. and Rogozhin, Y. Vol. 2055. Lecture Notes in Computer Science. Springer, pp. 1–10. DOI: 10.1007/3-540-45132-3\_1 (cit. on p. 20).
- Cook, B., Podelski, A. and Rybalchenko, A. (2011). Proving Program Termination. Communications of the ACM 54.5, pp. 88–98. DOI: 10.1145/1941487.1941509 (cit. on pp. 34, 35).
  - Copeland, B. J. (2004a). Hypercomputation: Philosophical Issues. Theoretical Computer Science 317.1–3, pp. 251–267. DOI: 10.1016/j.tcs.2003.12.014 (cit. on pp. 34, 35).
  - Davis, M. (1958). Computability and Unsolvability. McGraw-Hill (cit. on pp. 34, 35).
  - Herman, G. T. (1968). The Uniform Halting Problem for Generalized One State Turing Machines. In: 9th Annual Symposium on Switching and Automata Theory (SWAT 1968). IEEE, pp. 368–372. DOI: 10.1109/SWAT.1968.36 (cit. on p. 22).
  - 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, 9, 14).
    - Margenstern, M. (2000). Frontier Between Decidability and Undecidability: A Survey. Theoretical Computer Science 231.2, pp. 217–251. DOI: 10.1016/S0304-3975(99)00102-4 (cit. on p. 33).

### References

- Neary, T. and Woods, D. (2007). Four Small Turing Machines. In: Machines, Computations, and Universality (MCU 2007). Ed. by Durand-Lose, J. and Margenstern, M. Vol. 4664. Lecture Notes in Computer Science. Springer, pp. 242–254. DOI: 10.1007/978-3-540-74593-8\_21 (cit. on p. 20).
- (2009). Four Small Turing Machines. Fundamenta Informaticae 91.1, pp. 123–144. DOI: 10.3233/FI-2009-0036 (cit. on pp. 20, 21).
  - (2012). The Complexity of Small Universal Turing Machines: A Survey. In: Theory and Practice of Computer Science (SOFSEM 2012). Ed. by Bieliková, M., Friedrich, G., Gottlob, G., Katzenbeisser, S. and Turán, G. Vol. 7147. Lecture Notes in Computer Science. Springer, pp. 385–405. DOI: 10.1007/978-3-642-27660-6\_32 (cit. on pp. 20, 22).

Penrose, R. (1991). The Emperor's New Mind. Penguin Books (cit. on p. 15).

- Rendell, P. (2011). A Universal Turing Machine in Conway's Game of Life. In: Proceedings of the 2011 International Conference on High Performance Computing & Simulation (HPCS 2011). Ed. by Smari, W. W. IEEE, pp. 764–772. DOI: 10.1109/HPCSim.2011.5999906 (cit. on pp. 25–29).
- Rogozhin, Y. (1996). Small Universal Turing Machines. Theoretical Computer Science 168, pp. 215–240. DOI: 10.1016/S0304-3975(96)00077-1 (cit. on pp. 20–22).

### References

- Shannon, C. E. (1956). A Universal Turing Machine with Two Internal States. In: Automata Studies. Ed. by Shannon, C. E. and McCarthy, J. Vol. 34. Annals of Mathematics Studies. Princenton University Press, pp. 157–165 (cit. on pp. 18, 19).
- Sicard, A. (1997). Máquina Universal de Turing: Algunas Indicaciones para su Construcción. Revista Universidad EAFIT 33.108, pp. 61–106 (cit. on p. 17).
- Sicard, A. and Copeland, B. J. (2004b). Appendix: Subroutines and M-functions. In: The Essential Turing. Vol. 317. 1–3, pp. 54–57. DOI: 10.1016/j.tcs.2003.12.014 (cit. on p. 17).
- Sicard Ramírez, A. (1998). Máquinas de Turing Dinámicas: Historia y Desarrollo de una Idea. MA thesis. Departamento de Informática y Sistemas. Universidad EAFIT (cit. on p. 17).