

# Finite Automata

## In the notion of symbolic dynamics

Stanisław Ferchmin

Instytut Matematyki  
Uniwersytet Jagielloński

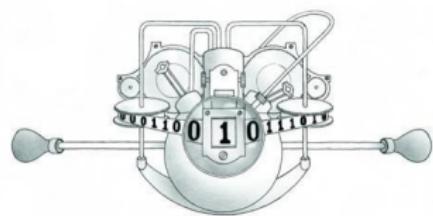
January 22, 2026

# Turing machine

Turing machine is a formal description of a simple computer. Turing used it in theoretic papers to explore the limits of what is computable and what is not.

Components of Turing's machine:

- $w \in \mathcal{A}^n = \{0, 1\}^n$
- Reading device that can read, erase, replace, move to adjacent position.
- Finite collection of states  $S_1, \dots, S_N$  where  $N$  is the size of the machine. Each state comes with set of instructions:
  - read the symbol
  - replace the symbol or not
  - move to the left or right position
  - move to another or stay at the same state



## Definition 1 (Shift of finite type)

A shift space  $X$  is a shift of finite type (SFT) if:

$\exists \mathcal{F} \subseteq \mathcal{A}^+$  finite, such that  $X = X_{\mathcal{F}}$ .

If  $M + 1$  is the length of the longest forbidden word, then we say  $X$  is an SFT with memory  $M$ .

## Theorem 1

Every SFT  $X$  over a finite alphabet  $\mathcal{A}$  can be recoded such that the list of forbidden words consists of 2-words only.

## Proof.

Sketch: Create new alphabet  $\{b_1, b_2, \dots, b_n\} = \mathcal{B} = \mathcal{A}^M$ , where  $X$  has memory  $M$ . Then by sliding block code  $\pi(x)_i = b_j$  and fact that  $\mathcal{A}$  is finite, proof is over. (Blackboard?) □

# Trasition graph and matrix

## Theorem 2

*Every SFT over finite alphabet can be represented by finite graph  $\mathcal{G}$  with vertices labeled by letters of  $\mathcal{B}$  and edges  $b_i \rightarrow b_j$  only if  $\pi^{-1}(b_i b_j)$  contains no forbidden word of  $X$*

## Definition 2

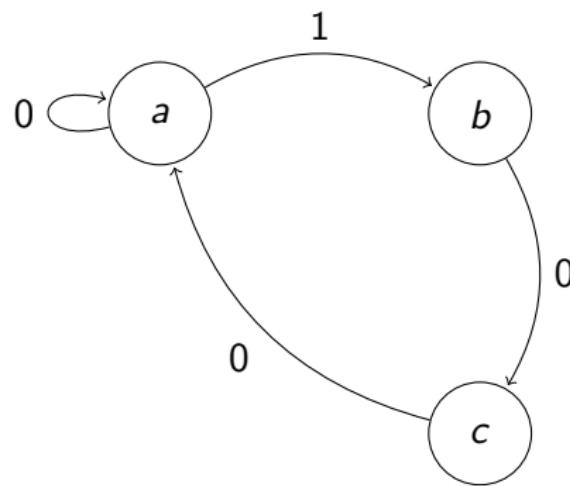
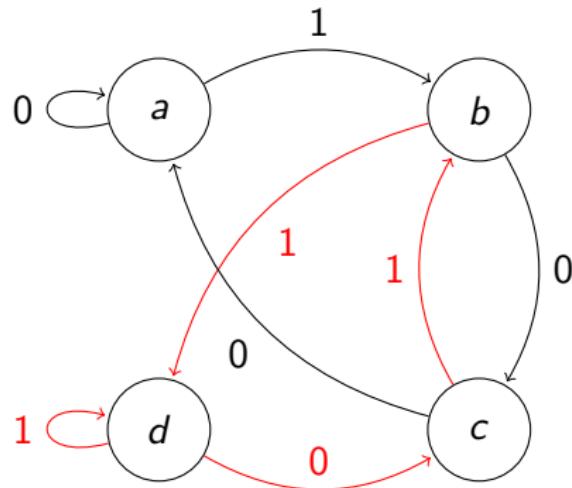
*The graph  $\mathcal{G}$  is called the transition graph of the SFT. The Matrix  $A = (a_{ij})_{i,j \in \mathcal{B}}$  is the transition matrix  $a_{ij} = 1 \iff$  edge  $i \rightarrow j$  exists in  $\mathcal{G}$ , else:  $(a_{ij}) = 0$*

## Definition 3

*A shift space  $X$  is coded if  $X = X_{\mathcal{G}}$  for some countable graph.  
Simple conclusion is that every SFT is coded.*

## Example 1.

Let  $X_{\mathcal{F}}$  be the SFT with  $\mathcal{F} = \{11, 101\}$  over  $\mathcal{A} = \{0, 1\}$ . We recode the alphabet to  $\mathcal{B} = \{a = 00, b = 01, c = 10, d = 11\}$ .



# Transition matrix of SFT

## Definition 4

A non-negative matrix  $A = (a_{ij})$  is called irreducible if for every  $i, j$  there is  $k$  such that  $a_{ij}^{(k)} > 0$ . For index  $i$ , set  $\text{per}(i) = \gcd(k > 1 : a_{ij}^{(k)} > 0)$ . If  $A$  is irreducible, then  $\text{per}(i)$  is the same for every  $i$ , and we call it the period of  $A$ . We call  $A$  aperiodic if its period is 1.

## Definition 5

The function  $p : \mathbb{N} \rightarrow \mathbb{N}$  defined by:

$p(n) = \#\{n\text{-words in } \mathcal{L}(X)\}$  is called the word complexity of  $X$ .

# Topological entropy

## Theorem 3 (Perron-Frobenius)

Let  $A$  be irreducible, aperiodic, non-negative matrix. Then:

- ① There is a real positive eigenvalue  $\lambda$  (called leading eigenvalue), of algebraic multiplicity one, such that  $\lambda > |\mu|$  for every other eigenvalue  $\mu$  of matrix  $A$ .
- ② The eigenvector associated to  $\lambda$  can be chosen strictly positive.

## Definition 6

The topological entropy of a subshift is:

$$h_{top}(X, \sigma) = \limsup_{n \rightarrow \infty} \frac{1}{n} \log(p(n))$$

## Theorem 4

The entropy of irreducible SFT equals  $\ln(\lambda)$  where  $\lambda$  is the leading eigenvalue of the transition matrix.

## Example 1. continuation

Transition matrix of  $\mathcal{X}_{\mathcal{G}}$

$$A = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

## Properties and Entropy

This matrix is clearly non-negative, irreducible and aperiodic.

**Eigenvalues:**  $\{1.4656, -0.2328 \pm 0.7926i\}$

**Corresponding norms:**  $\{1.4656, 0.8260, 0.8260\}$

Note that entropy doesn't change under conjugation:

$$h_{top}(X, \sigma) = \log(1.4656) \approx 0.5515$$

# Finite automata

## Definition 7

A finite automaton is a simplified type of Turing machine that can only read a tape from left to right, and not write on it. The components are  $M = \{Q, \mathcal{A}, q_0, F, \delta\}$  where:

- ①  $Q$  = collection of states the machine can be in.
- ②  $\mathcal{A}$  = the alphabet in which the tape is written.
- ③  $q_0$  = the initial state in  $Q$
- ④  $F$  = collection of all finite states in  $Q$ , the FA halts when it reaches one.
- ⑤  $\delta$  = is the rule how to go from one state to the next when reading a symbol  $a \in \mathcal{A}$  on the tape. Formally  $\delta : Q \times \mathcal{A} \rightarrow Q$

## Definition 8

Language  $\mathcal{L}$  is regular if it can be recognized by a finite automaton.

## Example 1. continuation

FA

Let:  $M = \{Q, \mathcal{A}, q_0, F, \delta\}$  be our FA with corresponding components:

①  $Q = \{00, 01, 10, t\}$

②  $\mathcal{A} = \{0, 1\}$

③  $q_0 = \{00\}$

④  $F = \{a, b, c\}$

⑤  $\delta(q_i, \sigma) = \begin{cases} 00 & \text{if } (q_i, \sigma) \in \{(00, 0), (10, 0)\} \\ 01 & \text{if } (q_i, \sigma) = (00, 1) \\ 10 & \text{if } (q_i, \sigma) = (01, 0) \\ t & \text{otherwise} \end{cases}$

# Python implementation

## Example 1. continuation

For  $M = (\{00, 01, 10, t\}, \{0, 1\}, \delta, 00, \{00, 01, 10\})$

$\delta$  as defined above

```
Topological Entropy: 0.5515
Adjency matrix of our system: [[1. 1. 0.]
 [0. 0. 1.]
 [1. 0. 0.]]
Legal words of length 4: ['0000', '0001', '0010', '0100', '1000', '1001']
Is '010010' accepted? True
```

## Example 2.

$$M = (\{a, b, t\}, \{0, 1\}, \delta, a, \{a, b\})$$
$$\delta = \{((a, 0), a), ((a, 1), b), ((b, 0), a), ((b, 1), t), ((t, 0), t), ((t, 1), t)\}$$

```
Topological Entropy: 0.6942
Adjency matrix of our system: [[1. 1.]
 [1. 0.]]
Legal words of length 4: ['0000', '0001', '0010', '0100', '0101', '1000', '1001', '1010']
Is '010110' accepted? False
```

# References and Sources

- Bruin H. (2017). *Notes on Symbolic Dynamics*. University of Vienna.
- Roman A. *Języki formalne i automaty*. Institute of Computer Science, Jagiellonian University.