# Strings and Languages

Languages of things
- numbers
- graphs
- polynomials
- ...

- Each thing should have a finite representation
  - Each such representation can be encoded as a string
- Only the two symbols 0 and 1 suffice for all finite encodings as strings

## Alphabet

A finite set of symbols.
letters

**Examples**: Binary alphabet $\{0, 1\}$

Alphabet of decimal digits $\{0, 1, 2, 3, \ldots, 9\}$

Add + and − to this alphabet

Add . to represent floating-point numbers

ASCII alphabet — printable characters

Roman alphabet — $\{a, b, \ldots, z\} \cup \{A, B, \ldots, Z\}$

Alphabet for English Language

$\{\sqcup\} \cup \{a, b, \ldots, z\} \cup \{A, B, \ldots, Z\} \cup \{0, 1, 2, \ldots, 9\} \cup \{\text{punctuation symbols}\}$

$\sqcup \longrightarrow$ space / blank $\quad \sqcup \quad \not b$

Use $\Sigma, \Gamma, \Delta$ to stand for alphabets $\qquad \{0, 1\}$ suffices

Symbols: $\underline{a, b, c}$, $0, 1, 2, \ldots$

<u>String</u>: (over an alphabet $\Sigma$)

a <u>finite</u> <u>sequence</u> of symbols from $\Sigma$

$\longrightarrow$ ordered — first element

2nd —

3rd —

<u>Example</u>: $011000110001 \longrightarrow$ binary string

abracadabra $\longrightarrow$ string over the Roman alphabet

Notation: $\underline{u, v, w, x, y, z}$ to represent strings

$\alpha, \beta, \gamma$

<u>length</u> of a string is the number of symbols in it.

$|abracadabra| = 11$ length of $w$ is denoted as $|w|$

<u>String of length 0</u> "null"

"" ($\epsilon$) $\epsilon$ — the <u>empty</u> string $|\epsilon| = 0$.

Given $\Sigma$, we denote by $\Sigma^*$ the set of all strings over $\Sigma$.

$\Sigma = \{a, b, c\}$ $\Sigma^* = \{\epsilon, a, b, c, aa, ab, ac, ba, bb, bc, ca, cb, cc, aaa, aab, aac, \ldots\}$

$a \in \Sigma$ $a^n = \underbrace{aaa\ldots a}_{n \text{ times}}$ $a^0 = \epsilon$

$\Sigma^*$ supports an operation called <u>concatenation</u>

$x = a_1a_2\cdots a_m$

$y = b_1b_2\cdots b_n$

$xy = a_1a_2\cdots a_m b_1b_2\cdots b_n$

Concatenation on $\Sigma^*$

- closed
- associative
- ✗ not necessarily commutative $\qquad (ab)(ba) \neq (ba)(ab)$
- $\epsilon$ acts as the identity $\qquad w\epsilon = \epsilon w = w$
- ✗ No inverse

monoid $\qquad \Sigma^*$ is a monoid under concatenation

Languages $\qquad$ A language over an alphabet $\Sigma$ is any subset of $\Sigma^*$.

$L, L_1, L_2, \ldots A, B \longrightarrow$ represent languages

sets of strings $\longrightarrow$ a lot of things

# Countability vs Uncountability

$\Sigma$ is a finite alphabet $(\Sigma \neq \emptyset)$

$|\Sigma| = 1$
unary alphabet

$\Sigma^*$ is countable.

There are $|\Sigma|^n$ strings of length $n$

$n = 0, 1, 2, 3, \ldots$

a countable union of countable (finite) sets

$L \subseteq \Sigma^*$

The set of all languages over $\Sigma$

$$= \mathcal{P}(\Sigma^*) = 2^{\Sigma^*} \xrightarrow[\text{theorem}]{\text{power-set}} \underline{\text{uncountable.}}$$

Representation

Strings $\rightarrow$ finite sequences (no problem)

Languages $\longrightarrow$ finite $\longrightarrow$ explicit enumeration

$\{ w_1, w_2, w_3, \ldots, w_n \}$

$\longrightarrow$ infinite

$\longrightarrow$ explicit enumeration not possible

$\downarrow$

Require finite description

– English language [the set of all strings of even length]

– Mathematical description [ $\{ w \in \{0,1\}^* \mid w$ is a multiple of $3 \}$ ]

– Recursive description [ palindromes over $\{a, b\}$

core $\longrightarrow$ $\epsilon, a, b$

$symb_n \cdots symb_2 \; symb_1 \; core \; symb_1 \; symb_2 \cdots symb_n$ ]

(Grammars)

– Procedures / Machines

Languages over $\Sigma$

Description of a language should be "finite"

$\hookrightarrow$ uses some alphabet — English/Roman alphabet

— alphabet of math symbols

$\downarrow$

$\Gamma$ $\qquad |\Gamma|$ is finite (and non-empty)

$\longrightarrow$ a string over $\Gamma$

Descriptions are elements of $\Gamma^*$. $\longrightarrow$ countable

cf $\mathcal{P}(\Sigma^*)$ is uncountable

- Only countably many languages can have finite descriptions
- Most of the languages cannot be described (finitely).

## Computational Problems

Given $L \subseteq \Sigma^*$ and $w \in \Sigma^*$, decide whether $w \in L$ or not. [Language-membership problem]

**Decision problem** — This definition is good for all theoretical developments

— Stood the test of time

Problems $\equiv$ Languages

- There are uncountably many problems.
- Only countably many machines (automata) to solve problems.
- There are unsolvable problems — Difficult to find
  — Proving unsolvability

## Strings and Languages

- Operations on strings
  - concatenation (defined)
  - Powers of a string $w^n$, $n \geq 0$

$$w^n = \begin{cases} \epsilon & \text{if } n = 0 \\ ww^{n-1} & \text{if } n \geq 1 \end{cases}$$

$(01)^3 = 010101$

$01^3 = 0111$

$0^3 1^3 = 000111$

  - Prefix (Suffix): $w \in \Sigma^*$. $u \in \Sigma^*$ is called a prefix (suffix) of $w$ if $w = uv$ $(vu)$ for some $v \in \Sigma^*$.

All prefixes of $abbca$ are $\underline{\epsilon, a, ab, abb, abbc,}$ $abbca$ → proper

All suffixes of $abbca$ are $\underline{\epsilon, a, ca, bca, bbca,}$ $abbca$

proper

- <u>Operations on languages</u>

Languages are subsets of $\Sigma^*$ $\longrightarrow$ universal set

$A, B \subseteq \Sigma^*$

$A \cup B$

$A \cap B$

$\sim A = \overline{A} = \Sigma^* \setminus A$

Set-theoretic identities hold
- De Morgan
- Distributivity

$A = \{a, ab\}$

$B = \{\epsilon, b\}$

$AB = \{a, b, ab, abb\}$

<u>Concatenation</u> $A, B \subseteq \Sigma^*$

$AB = \{ uv \mid u \in A \text{ and } v \in B \}$

**Powers:** $A \subseteq \Sigma^*$

$\Sigma = \{0, 1\}$

$$A^n = \begin{cases} \{\epsilon\} & \text{if } n = 0 \\ A A^{n-1} & \text{if } n \geq 1 \end{cases}$$

$$\Sigma^* = \{\epsilon\} \cup \underline{\{0, 1\}} \cup \underline{\{0, 1\}}^2 \cup \underline{\{0, 1\}}^3 \cup \cdots$$

$A = \{a, ab\}$

$A^0 = \{\epsilon\}$

$A^1 = \{a, ab\}$

$A^2 = \{aa, aab, aba, abab\}$

$\cdots$

$$\boxed{A^* = \bigcup_{n \geq 0} A^n}$$

(asterate of $A$) (kleene star)

$$\boxed{A^+ = \bigcup_{n \geq 1} A^n}$$

Identities : [Easy proofs]

$$A^* A^* = A^*$$

$$(A^*)^* = A^*$$

$$\emptyset^* = \{\epsilon\}$$

$[\emptyset^0 = \{\epsilon\},\ \emptyset = \emptyset\{\epsilon\} = \emptyset$

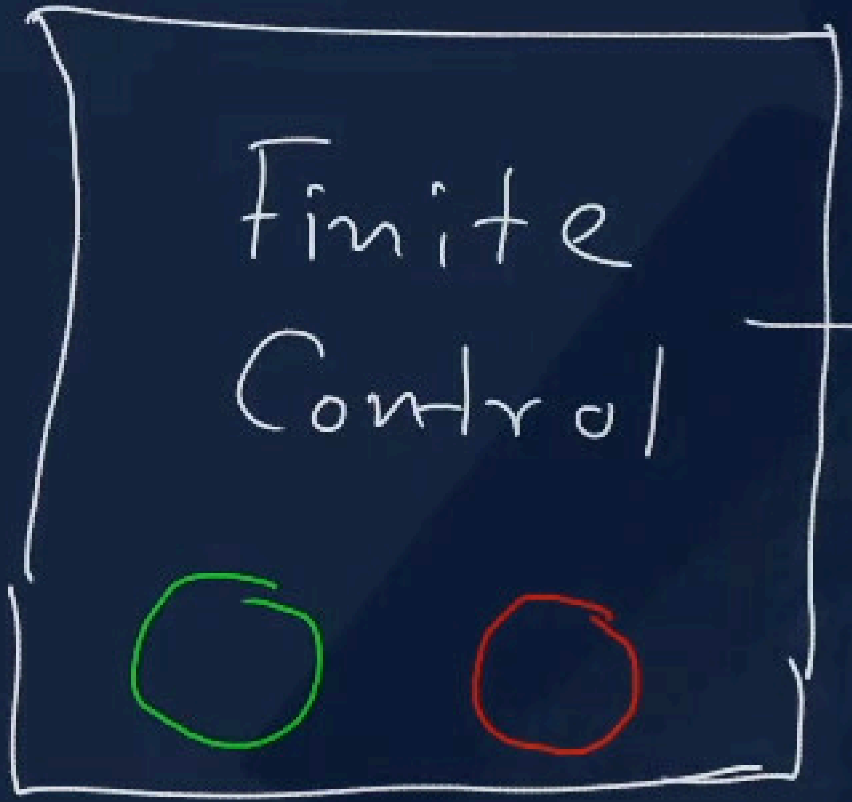
$\emptyset^2, \emptyset^3, \ldots = \emptyset]$

$$A^* = \{\epsilon\} \cup AA^*$$

$$= \{\epsilon\} \cup A^* A$$

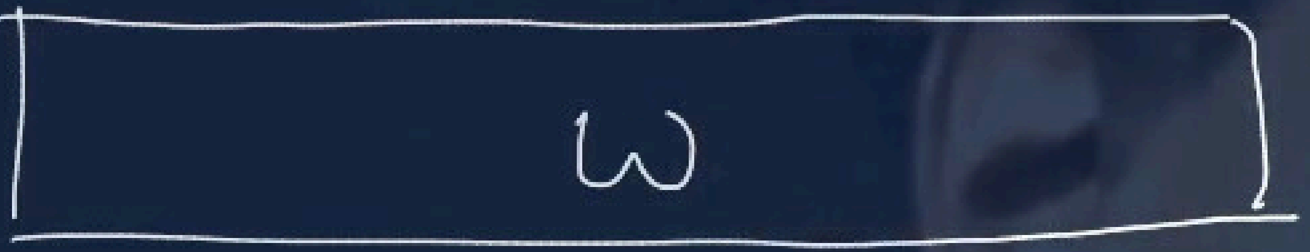
# Description of Languages by automata [plural of automaton]

Represents $L$




$\rightarrow$ consists of a finite set $Q$ of states

Input string




(Externally)

$L \subseteq \Sigma^*$

Input: $w \in \Sigma^*$

Output: Accept or Reject

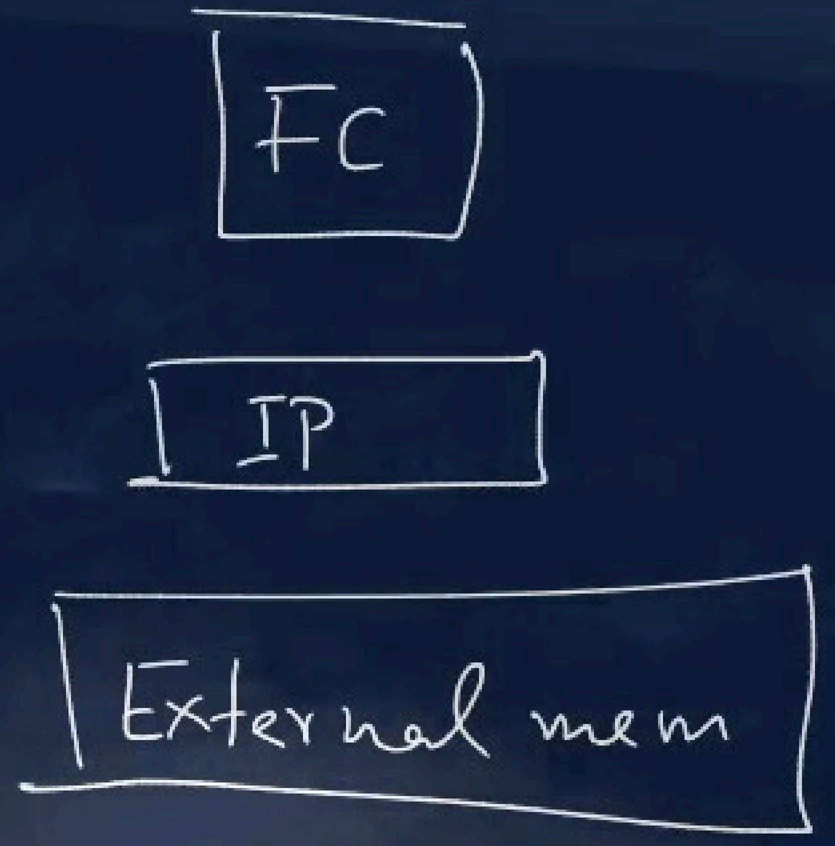
Do computation on $w$.

If $w \in L$, the green lamp glows
If $w \notin L$, the red lamp glows.

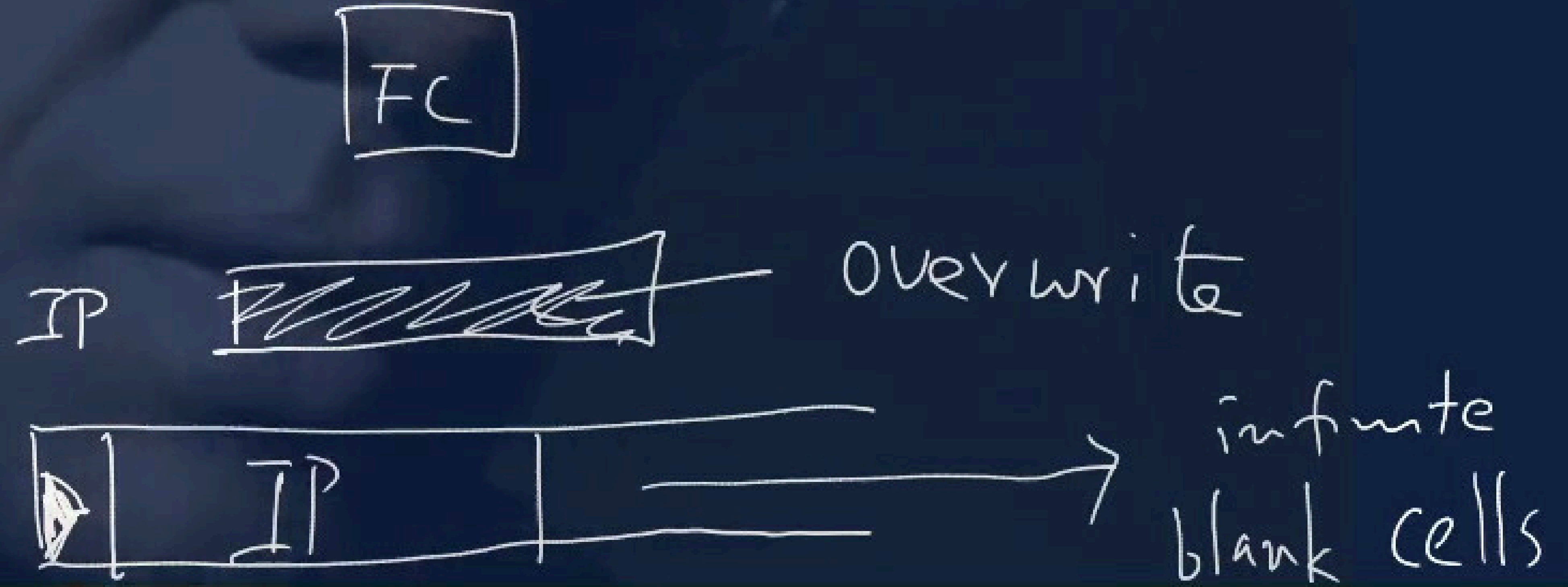
M/c stops.

# Different types of automatons

External memory




- Finite automata
  (no external memory)
  Only finite memory (FC)
- Pushdown automata
  Infinite external memory organized and accessed as a stack
- Linear-bounded automata
  IP tape is rewritable.
- Turing machine
  Infinite read/write tape



# Chomsky Hierarchy    Grammars

Regular — Finite automata —— Linear grammars (3)

CFL — Pushdown automata —— Context-free grammars (2)

CSL — Linear bounded automata —— Context-sensitive grammars (1)

Turing machines —— Unrestricted grammars (Type 0)