# Basic Background

**Def**: A DFA is a tuple $M=(Q, A, \circ, q_0, F)$

where:

- $Q$ = set of states
- $A$ = finite alphabet
- $\circ : Q \times A \to Q$ = transition function
- $q_0$ = start state
- $F$ = accept states

We extend $\circ$ to a function $Q \times A^* \to Q$

by induction on the length of a word

$$q \,\hat{\circ}\, \varepsilon = q \quad , \quad q \,\hat{\circ}\, \sigma = q \circ \sigma$$

$$q \,\hat{\circ}\, w\sigma = (q \,\hat{\circ}\, w) \circ \sigma$$

Then:

$$\boxed{\mathcal{L}(M) = \{ w \in A^* : q_0 \,\hat{\circ}\, w \in F \}}$$

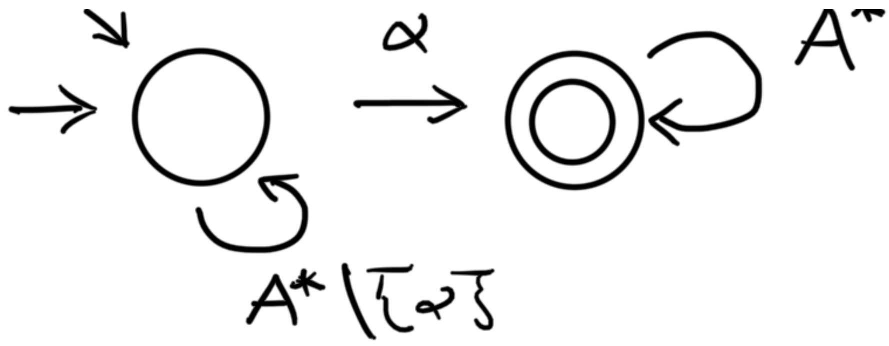
**Def**: A language $\mathcal{L} \subseteq A^*$ is regular

if there is a DFA $M$ s.t. $\mathcal{L} = \mathcal{L}(M)$.

**Example**: Fix $\alpha \in A$. Then

$$\mathcal{L} = \{ w \in A^* : w \text{ contains } \alpha \}$$

is regular.




**Recall:** Regular = Accepted by DFA

= Accepted by NFA

= Defined by regular expression

Non-regularity is shown by the

Pumping Lemma, Mynill-Nerode theorem,

....

**Recall:** Every regular language $\mathcal{L}$ has

a unique minimal DFA $\mathcal{A}(\mathcal{L})$ that accepts it.

This can be algorithmically obtained

starting from any DFA for $\mathcal{L}$

## §4.2: Automata and monoids

Let $\mathcal{A} = (Q, A, \circ, q_0, F)$ be a DFA.

For each $v \in A^*$ the map

$$q \longmapsto q \circ v$$

defines a function $\underline{f_v} : Q \to Q$.

Clearly, these functions compose as:

$$f_v f_w = f_{vw}$$

and so the set

$$M(\mathcal{A}) = \{ f_v : v \in A^* \}$$

is a monoid with composition, while the map

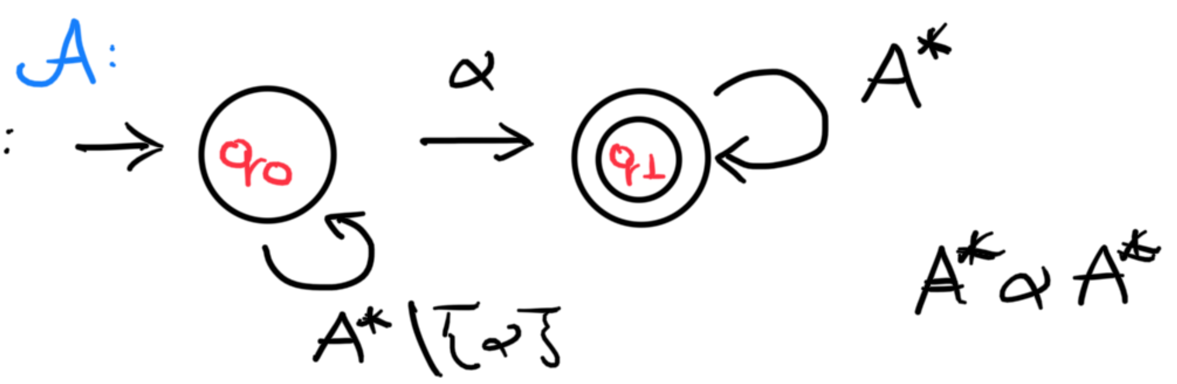
$$\eta_A : A^* \to M(\mathcal{A})$$
$$v \longmapsto f_v$$

is a morphism from $\boxed{A^*}$ onto $M(\mathcal{A})$.

We call $M(\mathcal{A})$ the <u>transition monoid</u> of $\mathcal{A}$. Clearly:

$$|M(\mathcal{A})| \leq |Q|^{|Q|} < \infty$$

<u>Example</u>:




$M(\mathcal{A})$:

$M(\mathcal{A})$: $f_a : q_0 \mapsto q_1, \; q_1 \mapsto q_1$ | $f_b = \mathrm{id} \; (b \neq a)$

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

Observe:

If $X = \{ f_v : f_v(q_0) \in F \} \subseteq M(\mathcal{A})$

then

$$\boxed{\mathcal{L}(A) = \eta_A^{-1}(X)}$$

Def: We say that a monoid $M$ recognises a language $\mathcal{L} \subseteq A^*$ if there is a morphism

$$\varphi : A^* \to M$$

and a set $X \subseteq M$ such that

$$\mathcal{L} = \varphi^{-1}(X).$$

We also say that $\varphi$ recognises $\mathcal{L}$

NB: $\mathcal{L}$ is regular iff it is recognised by some finite monoid

## §4.3: Syntactic monoid and syntactic morphism

**Def:**

Let $\mathcal{L} \subseteq A^*$ be regular, and $\mathcal{A}(\mathcal{L})$ its minimal automaton. Then

$$\mathrm{Synt}(\mathcal{L}) := M(\mathcal{A}(\mathcal{L}))$$

$$\eta_{\mathcal{L}} := \eta_{\mathcal{A}(\mathcal{L})}$$

are called the syntactic monoid and the syntactic morphism of $\mathcal{L}$.

**Example:** $U_1 = \{0,1\}$ is the syntactic monoid of $\mathcal{L} = A^* a A^*$.

**Def:** Let $M, N$ be monoids. We say that $M$ divides $N$, and write

$$\underline{M \prec N},$$

if there is a submonoid $N'$ of $N$ and a surjective morphism $N' \to M$

$\boxed{\text{Fact:}}$ $\prec$ is the smallest transitive relation that includes both the submonoid and quotient relations.

p.21

**Proposition 4.5**: Let $\mathcal{L}$ be regular and $M$ a monoid. Then:

$$M \text{ recognises } \mathcal{L} \iff \mathrm{Synt}(\mathcal{L}) \prec M$$

<u>Pf</u>: Basically just checking definitions

(So if $M_1 < M_2$ and $M_1$ recognises $\mathcal{L}$, then so does $M_2$)

**Proposition 4.6**: $\mathrm{Synt}(\mathcal{L})$ can also be obtained as a quotient of $A^*$ by the congruence

$$u \sim_{\mathcal{L}} v \iff \forall x, y \in A^* \, (xuy \in \mathcal{L} \Leftrightarrow xvy \in \mathcal{L})$$

# Chapter 16: Varieties

## §16.1: Motivation and examples

$A^* \alpha A^*$

Q: What are the regular languages recognised by $U_1 = \{0,1\}$?

If $\varphi : A^* \to U_1$ is a morphism then any set $\varphi^{-1}(X)$ for $X \subseteq U_1$ has the form

$\rightarrow \boxed{B^*} \leftarrow$ or $\rightarrow \boxed{A^* \mid B^*}$

if $1 \in X$ or $X = \emptyset$ — if $X = \{0\}$

for some $B \subseteq A$.

$A^*$ $\quad$ $\varphi(w) = 1 \Rightarrow$ all letters in $w$ are mapped to $1$

So: membership of $w$ in $\lambda$ depends only on $\alpha(w) = \{u \in A : u \text{ appears in } w\}$

- Call this property $\boxed{P(\lambda)}$.

$$\boxed{\text{Task: Given } \lambda, \text{ decide if } P(\lambda)}$$

- Clearly $P(\lambda)$ is preserved by union and complement, so

$$P = \{\lambda \subseteq A^* : P(\lambda)\}$$

forms a Boolean algebra.

- Not every $\lambda$ in $P$ is recognised by

$\mathcal{U}_1$, e.g. $a^* \cup b^*$

- However, it follows by "basic properties of the syntactic monoid" that
  $\mathcal{P}$ = languages recognised by finite direct powers of $\mathcal{U}_1$.

Letting:

$J_1$ = finite monoids that divide a finite direct power of $\mathcal{U}_1$

then:

$$\rightarrow \boxed{\mathcal{P}(\mathcal{L}) \Longleftrightarrow \mathrm{Synt}(\mathcal{L}) \in J_1}$$

$J_1$ is an example of a pseudovariety

| |
|---|
| We have managed to reduce a syntactic property of $\mathcal{L}$ to an algebraic property of $\mathrm{Synt}(\mathcal{L})$ |

Q: Does this really help in checking

efficiently if $\mathcal{L}$ has $\mu$!

It gives decidability:

→ $\boxed{\begin{array}{l}\text{Fact: If } \varphi : A^* \to M^r \text{ is a} \\ \text{morphism, then } \varphi(A^*) \text{ embeds} \\ \text{into } \underline{M^s}, \text{ where } s = |M|^{|A|}.\end{array}}$

So, we can check all the divisors of $(U_\perp)^{2^{|A|}}$ and see if $\mathrm{Synt}(\mathcal{L})$ is isomorphic to any of them

There's a better approach!

Evidently, $U_\perp$ is commutative & idempotent (ie all its elements are idempotent).

$0 \times 0 = 0$
$1 \times 1 = 1$

These properties are preserved by products + division, so:

$$J_\perp \subseteq \mathbf{B.S.}\ell$$

semilattices

bounded

= commutative & idempotent monoid

Conversely:

Let $\varphi : A^* \to M$ be a morphism.
where $M$ is comm. & idempotent
Then $\forall w_1, w_2 \in A^*$

$$\alpha(w_1) = \alpha(w_2) \Rightarrow \varphi(w_1) = \varphi(w_2)$$

For any $m \in M$:

$\therefore$ $\varphi^{-1}(m)$ has $P$

$$\therefore \quad Synt(\varphi^{-1}(m)) \in J_1$$

But

Fact: For any finite monoid $M$ and morphism $\varphi : A^* \to M$:

$$M \prec \underbrace{\prod_{m \in M} \underbrace{Synt(\varphi^{-1}(m))}_{\in J_1}}_{J_1}$$

$$\therefore \quad M \in J_1$$

So: $J_1 = BSe$

So $M \in J_1 \iff M$ satisfies the

so $M \in J_1 \iff M$ satisfies the identities $xy = yx$ & $x^2 = x$

Given the mult. table of $M$ we can check these in poly time

$\mathrm{Synt}(\mathcal{L}) =$

§ 1.2 : Piecewise-testable language

The identities here are "profinite", ie they are of the form

$$x^{\omega} = x x^{\omega}$$

the unique idempotent power of $x$

$$x, x^2, x^3, \ldots$$

Gr : $x^{\omega} = 1$

§

§ 1.3:

Fix some pseudovariety $\mathbb{V}$ of finite monoids. Given some finite A,

$$A^*\mathcal{V} = \{ L \subseteq A^* : \mathrm{Synt}(L) \in \mathbb{V} \}$$

We can think of $\mathbb{V}$ as an operator taking a finite set, and returning a family of ~~formal~~ regular languages.

variety of languages

<u>Task</u>: $L \subseteq A^*$, check if $L \in \mathcal{V}$.

This can be decided if there is some effective criterion to determine if a finite monoid belongs to the pseudovariety $\mathbb{V}$.

<u>Theorem (Reiterman)</u>
Pseudovarieties are precisely the families of finite monoids defined by profinite identities.

Theorem (Eilenberg)

Let $V$ assign to each alphabet $A$ a family $A^*V$ of regular languages. Then $V$ is a var. of languages

$\iff$ 1. $A^*V$ is closed under Boolean operations

2. if $L \in A^*V$ and $w \in A$ then $w^{-1}L \in A^*V$ and $Lw^{-1} \in A^*V$

3. If $L \in A^*V$ and $\varphi : B^* \to A^*$ is a morphism of fin gen free monoids then $\varphi^{-1}(L) \in B^*V$.