Chapter 7

Regular languages

In this chapter we study *regular* languages, characterized by deterministic finite automata (see Definitions 3.3.1), as the class of most simple languages in Chomsky's hierarchy. It turns out that a regular language is essentially a pack of words whose syntax can be *uniformly*[1] verified using a constant amount of memory (i.e. space) in real time[2].

On the other hand, being too simple implies that the computational model admit an algebraic interpretation, in addition to its coalgebraic structure as a dynamical system. Also, this simplicity of structure gives rise to the fact that all four fundamental classes coincide, making the subject ideal for a first encounter with a computational model.

→ Sec. 5.4

7.1 On most simple computers

There may be many different approaches to categorize languages in terms of *complexity* of their descriptions[3], while one of the most basic approaches is to measure the resources used by the best verification algorithm for the membership problem[4]. This approach, although deep as a theory, is quite simple when one is going to impose strong conditions on the algorithms to simplify them. Choosing the most natural resources as *time* and *work space* (i.e. *used memory*), and forcing these to their extremal limits as being *real time* and using *constant* memory, raises the following question,

→ Exr. 7.12.1

> What are the most simple computers/deciders that can operate in real time and using constant memory (i.e. to process any given input)?

* Will be completed when students hand in Exercise 7.12.1!

7.2 Automata and algebras

Let us start by defining a simple algebraic structure that turn out to be the core structure of one of the simplest computational models called *abstract automata*.

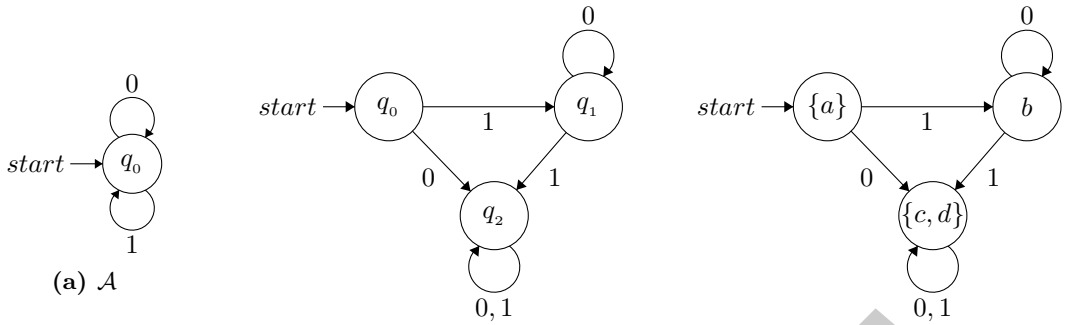
→ Def. 1.7.5
→ Sec. 5.4

Definition 1.7.2 Preautomata

A *preautomaton* $\mathcal{A}(Q, q_0 \in Q, \{\tau_i : Q \bullet\!\!\longrightarrow Q\}_{i\in\Sigma})$ (or sometimes a Σ-*preautomaton* to emphasize the set of input symbols), on the *state space* Q, and the set of input symbols Σ of size n, is an algebra having a nullary operation $q_0 \in Q$ and n unary operations τ_i. Each unary operation τ_i is called the *transition map* of symbol $i \in \Sigma$, where q_0 is said to be the *initial state*. Sometimes, we may refer to such a preautomaton as a Σ-*preautomaton* if we want to explicitly refer to the set of input symbols Σ.

→ Sec. 20.1.6

[1] i.e. the verification algorithm does not depend on the word being verified.
[2] i.e. in time equal to the length of the word.
[3] This is essentially the main theme of the theory of *computational complexity*, containing many different approaches as *structural complexity*, *Kolmogorov complexity*, etc.
[4] Also see abstract Blum-Shub complexity theory (e.g. see []).



(a) $\mathcal{A}$ **(b)** $\mathcal{B}$ **(c)** $\mathcal{C}$

Figure 7.1 – See Example 2.7.2.

Throughout this chapter, by an algebra we mean a preautomaton as an algebra of the type defined above. ▶ → Assumptions

Example 2.7.2 Graphical description of preautomata

The data expressing a preautomaton $\mathcal{A}(Q, q_0 \in Q, \{\tau_i : Q \bullet\!\!\longrightarrow Q\}_{i\in\Sigma})$ may also be presented in an schematic way, as a labeled multigraph $\mathcal{A}(Q, E \subseteq Q^2, +, -, \ell)$, such that → Sec. 20.1.8

$$[e \stackrel{\text{def}}{=} (q_s, q_t) \in E,\ e^- = q_s,\ e^+ = q_t \text{ and } \ell(e) = i] \quad \Leftrightarrow \quad (q_s, q_t) \in \tau_i,$$

where on the right hand side we are treating τ_i as a relation. We also, add an arrow pointing to the state q_0, distinguishing this state from the other ones (as an extra structure). → Sec. 20.1.2

Note that, within this setup, for each state $q \in Q$ and $i \in \Sigma$ there is exactly one directed edge leaving q whose label is i, since τ_i is a function. This, clearly, implies that any preautomaton works deterministically, and that the corresponding graph is a *directed* labeled graph. → Sec. 3.2

Figure 7.1a shows the trivial preautomaton on the one element state space $\{q_0\}$. Also, the preautomata $\mathcal{B}(Q, q_0, \{\tau_i : Q \bullet\!\!\longrightarrow Q\}_{i\in\Sigma})$ depicted in Figure 7.1b and the preautomata $\mathcal{C}(P, \{a\}, \{\eta_i : P \bullet\!\!\longrightarrow P\}_{i\in\Sigma})$ depicted in Figure 7.1c are essentially the same but they have different state spaces with the following transition maps, → Def. 3.7.2 → Thm. 5.7.3

τ_i	$i=0$	$i=1$
q_0	q_2	q_1
q_1	q_1	q_2
q_2	q_2	q_2

η_i	$i=0$	$i=1$
$\{a\}$	$\{c,d\}$	b
b	b	$\{c,d\}$
$\{c,d\}$	$\{c,d\}$	$\{c,d\}$

We will formalize this equality concept in what follows. ◇

The concepts of a homomorphism and an isomorphism are defined to make it possible to compare algebraic structures regardless of the nature of their objects. For the case of preautomata we have the following definition. → Sec. 20.2

Definition 3.7.2 Homomorphisms and isomorphisms

Let $\mathcal{A}(Q, q_0 \in Q, \{\tau_i : Q \bullet\!\!\longrightarrow Q\}_{i\in\Sigma})$ and $\mathcal{B}(P, p_0 \in P, \{\eta_i : P \bullet\!\!\longrightarrow P\}_{i\in\Sigma})$ be two preautomata with the same set of input symbols Σ. A *homomorphism* from $\mathcal{A}$ to $\mathcal{B}$ is a mapping $\sigma : Q \longrightarrow P$ that preserves the algebraic structures (see Figure 7.2), i.e.

$$\begin{aligned} \forall i \quad & \eta_i \circ \sigma = \sigma \circ \tau_i, \\ & \sigma(q_0) = p_0. \end{aligned}$$

Also, $\mathcal{A}$ and $\mathcal{B}$ are said to be *isomorphic* , denoted by $\mathcal{A} \cong \mathcal{B}$, if there exists a bijective homomorphism $\sigma : \mathcal{A} \bullet\!\!\longrightarrow \mathcal{B}$ such that the inverse, $\sigma^{-1} : \mathcal{B} \bullet\!\!\longrightarrow \mathcal{A}$, is also a homomorphism. ▶ → Exr. 7.12.2

Definition 4.7.2 Closed subsets and reduced preautomata

Let $\mathcal{A} = (A, O)$ be an algebra and let $\tau \in O$ be an n-ary operation of A. A subset B of A is τ-*closed* if for every choice of n elements $a_1, \ldots, a_n$ in B we have $\tau(a_1, \ldots, a_n) \in B$. Also, a

subset $B \subseteq A$ is said to be a *closed subset* of $\mathcal{A}$ if B is τ-closed for each operation τ of $\mathcal{A}$. A *subalgebra* of $\mathcal{A}$ is any algebra $\mathcal{B} = (B, N)$ which is isomorphic to a closed subset C of $\mathcal{A}$ along with the restrictions of all operations in O to C. Hence, note that, the concept of a subalgebra is just defined up to isomorphisms of algebras of the same type. → Sec. 20.2

Within this context, in a preautomaton $\mathcal{A}(Q, q_0 \in Q, \{\tau_i : Q \bullet\!\!\longrightarrow Q\}_{i\in\Sigma})$, a state q_t is said to be *accessible* (or *reachable*) from q_s if there exists $i_1, i_2, \ldots, i_m$ such that

$$q_t = \tau_{i_m} \circ \cdots \circ \tau_{i_2} \circ \tau_{i_1}(q_s).$$

A *reduced preautomaton* $\mathcal{A}(Q, q_0 \in Q, \{\tau_i : Q \bullet\!\!\longrightarrow Q\}_{i\in\Sigma})$ is a preautomaton whose only subalgebra is $\mathcal{A}$ itself. Note that, equivalently, $\mathcal{A}$ is reduced if each state $q \in Q$ is accessible from q_0. → Exr. 7.12.3

▶

7.3 Automata and congruence relations

Recall that an *equivalence relation* on a set A is a binary relation $\sim$ on A that is *reflexive*, *symmetric* and *transitive*. Also, the equivalence class of an element a in A with respect to the equivalence relation $\sim$, denoted by $[a]_\sim$, is the set of all elements of A which are equivalent to x, i.e. $[a]_\sim = \{b \in A \ : \ a \sim b\}$. The set of all equivalence classes of A with respect to $\sim$, denoted by ${}^A/_\sim \ \in \Pi(A)$, is called the *quotient set* of A by $\sim$ and is a partition of A. On the other hand, recall that for any partition ζ of a set A, one may define an equivalence relation $\sim$ on A where $a \sim b$ if and only if a and b belong to the same element of ζ. Hence, talking about equivalence relations on a set is essentially the same as talking about partitions of the set itself. → Sec. 20.1.2

Recall that, given any onto map $\sigma : A \longrightarrow\!\!\bullet\ B$ gives rise to an equivalence relation $\sim_\sigma$ on A according to which → Not. $\sim_\sigma$

$$a_1 \sim_\sigma a_2 \Leftrightarrow \sigma(a_1) = \sigma(a_2).$$

Hence, the collection of all $\sigma-$inverse images of the elements of B forms a partition of A.

Definition 1.7.3 Congruence relations

Let $\mathcal{A} = (A, O)$ be an algebra. A *congruence relation* is an equivalence relation $\sim$ on A which is compatible with the algebra operations. In the case of a preautomaton, where each operation is either nullary or unary, this means that for every a and b in A and each unary operation τ_i the following equality hold regardless of choosing the representatives a and b,

$$[a]_\sim = [b]_\sim \ \Leftrightarrow \ [\tau_i(a)]_\sim = [\tau_i(b)]_\sim. \tag{7.1}$$

▶

Equation 7.1 as the main property of a congruence relation guarantees that one may define a well-defined quotient algebra as follows.

Definition 2.7.3 The quotient preautomaton

Let $\mathcal{A}(Q, q_0 \in Q, \{\tau_i : Q \bullet\!\!\longrightarrow Q\}_{i\in\Sigma})$ be a preautomaton and $\sim$ be a congruence relation on

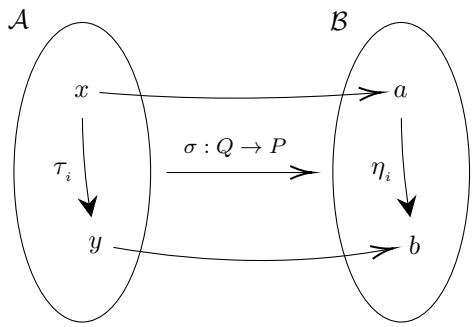


Figure 7.2 – A homomorphism from $\mathcal{A}$ to $\mathcal{B}$.

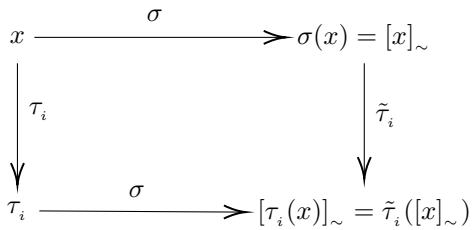


Figure 7.3 – The canonical map commutative diagram (see Definition 4.7.3).

$\mathcal{A}$. The *quotient preautomaton* of $\mathcal{A}$ by $\sim$ is defined as

$$\mathcal{A}/\sim \stackrel{\text{def}}{=} (Q/\sim,\ [q_0]_\sim,\ \{\tilde{\tau}_i : Q/\sim \bullet\!\!\longrightarrow\ Q/\sim\}_{i\in\Sigma})$$

where $\tilde{\tau}_i([a]_\sim) \stackrel{\text{def}}{=} [\tau_i(a)]_\sim$. Note that by Equation 7.1 each $\tilde{\tau}_i([a]_\sim)$ is well-defined as above where (a) can be any one of the representatives of the equivalence class $[a]_\sim$. ▶

Example 3.7.3 Trivial congruence relations

Given a preautomaton $\mathcal{A}(Q, q_0 \in Q, \{\tau_i : Q \bullet\!\!\longrightarrow Q\}_{i\in\Sigma})$, considering Id_Q as a relation, one may verify that Id_Q is a congruence relation for which $[q]_{Id_Q} = \{q\}$ for any $q \in Q$. Clearly, in this setting, $\mathcal{A}/_{Id_Q}$ is isomorphic to $\mathcal{A}$ itself.

On the other hand, consider $\mathbf{1}_Q \stackrel{\text{def}}{=} Q^2$ as a relation and note that it has only one equivalence class which is Q itself. Hence, again, trivially, $\mathbf{1}_Q$ is a congruence relation and $\mathcal{A}/\mathbf{1}_Q$ is isomorphic to the preautomaton of Figure 7.1a with only one state. ◇

→ Not. $\mathbf{1}_A$
→ Exm. 2.7.2

Example 4.7.3 The canonical map

Let $\mathcal{A}(Q, q_0 \in Q, \{\tau_i : Q \bullet\!\!\longrightarrow Q\}_{i\in\Sigma})$ be a preautomaton and $\sim$ be a congruence relation on $\mathcal{A}$. The *natural map* or the *canonical map*, $\sigma_\sim : \mathcal{A} \bullet\!\!\longrightarrow \mathcal{A}/\sim$, is defined as $\sigma_\sim(x) \stackrel{\text{def}}{=} [x]_\sim$. Note that by definition, $\sigma_\sim$ is a homomorphism (see Figure 7.3). In other words,

$$\sigma_\sim(\tau_i(x)) = [\tau_\sim(x)]_\sim = \tilde{\tau}_\sim(\sigma_\sim(x)),$$

and

$$\sigma_\sim(q_0) = [q_0]_\sim.$$

◇

Using what we have developed so far, we are ready to prove one of the main isomorphism theorems that will be our basic tool to classify preautomata in what follows.

Theorem 5.7.3 *If σ is a homomorphism from a preautomaton $\mathcal{A}$ to a reduced preautomaton $\mathcal{B}$, then $\mathcal{A}/_{\sim_\sigma}$ and $\mathcal{B}$ are isomorphic.*

Proof. Given $\mathcal{A}(Q, q_0 \in Q, \{\tau_i : Q \bullet\!\!\longrightarrow Q\}_{i\in\Sigma})$ and $\mathcal{B}(P, p_0 \in P, \{\eta_i : P \bullet\!\!\longrightarrow P\}_{i\in\Sigma})$ where $\mathcal{B}$ is reduced, note that, since σ is a homomorphism, for any choice of representatives x and y, where $x \sim y$, we have

$$\begin{aligned}
[x]_{\sim_\sigma} = [y]_{\sim_\sigma} &\Leftrightarrow \sigma(x) = \sigma(y) \\
&\Leftrightarrow \eta_i(\sigma(x)) = \eta_i(\sigma(y)) \\
&\Leftrightarrow \sigma(\tau_i(x)) = \sigma(\tau_i(y)) \\
&\Leftrightarrow [\tau_i(x)]_{\sim_\sigma} = [\tau_i(y)]_{\sim_\sigma}.
\end{aligned}$$

Hence, $\sim_\sigma$ is a congruence relation on $\mathcal{A}$ and we can construct the quotient preautomaton,

$$\mathcal{A}/_{\sim_\sigma} \stackrel{\text{def}}{=} (Q/_{\sim_\sigma},\ [q_0]_{\sim_\sigma},\ \{\tilde{\tau}_i : Q/_{\sim_\sigma} \ \to\ Q/_{\sim_\sigma}\}_{i\in\Sigma}),$$

where $\tilde{\tau}_i([x]_{\sim_\sigma}) = [\tau_i(x)]_{\sim_\sigma}$. Consequently, we define the map $\tilde{\sigma} : \mathcal{A}/_{\sim_\sigma} \bullet\!\!\longrightarrow \mathcal{B}$ as $[x]_{\sim_\sigma} \mapsto \sigma(x)$.

Next, note that, $\tilde{\sigma}$ is a one to one map, since

$$\begin{aligned} [x]_{\sim_\sigma} = [y]_{\sim_\sigma} &\Leftrightarrow \sigma(x) = \sigma(y) \\ &\Leftrightarrow \tilde{\sigma}(x) = \tilde{\sigma}(y). \end{aligned}$$

Also, $\tilde{\sigma}$ is an onto map since $\mathcal{B}$ is reduced and σ is a homomorphism. On the other hand, $\tilde{\sigma}$ is a homomorphism too, since → Exr. 7.12.4

$$\tilde{\sigma}([q_0]_{\sim_\sigma}) = \sigma(q_0) = p_0,$$

and for any $i \in \Sigma$ we have,

$$\begin{aligned} \tilde{\sigma}(\tilde{\tau}_i([x]_{\sim_\sigma})) &= \tilde{\sigma}([\tau_i(x)]_{\sim_\sigma}) \\ &= \sigma(\tau_i(x)) \\ &= \eta_i(\sigma(x)) \\ &= \eta_i(\tilde{\sigma}([x]_{\sim_\sigma})). \end{aligned}$$

Hence, $\tilde{\sigma}$ is an isomorphism, and consequently, $\mathcal{A}/_{\sim_\sigma} \cong \mathcal{B}$. □ → Def. 3.7.2

7.4 Universal preautomata

Intuitively, a *universal preautomaton* is a preautomaton whose quotients essentially generate all existing preautomata up to isomorphism. Here comes the technical definition.

Definition 1.7.4 Universal preautomata

The *right universal preautomaton*, $\mathcal{U}^r_\Sigma$, is the *infinite* preautomaton → Exm. 25.20.1

$$\mathcal{U}^r_\Sigma(\Sigma^*, \epsilon \in \Sigma^*, \{r_i : \Sigma^* \bullet\!\!\longrightarrow \Sigma^*\}_{i\in\Sigma})$$

with the infinite state space Σ^*, whose transition maps are *right concatenations*, i.e.,

$$\forall\, i \in \Sigma, \quad r_i(w) \stackrel{\text{def}}{=} wi.$$

Similarly, the *left universal preautomaton*, $\mathcal{U}^l_\Sigma$, is the *infinite* preautomaton

$$\mathcal{U}^l_\Sigma(\Sigma^*, \epsilon \in \Sigma^*, \{l_i : \Sigma^* \bullet\!\!\longrightarrow \Sigma^*\}_{i\in\Sigma})$$

with the infinite state space Σ^*, whose transition maps are *left concatenations*, i.e.,

$$\forall\, i \in \Sigma, \quad l_i(w) \stackrel{\text{def}}{=} iw.$$

▶

An important property of $\mathcal{U}^r_\Sigma$ is the fact that for any given Σ-preautomaton

$$\mathcal{B}(P, p_0 \in P, \{\eta_i : P \bullet\!\!\longrightarrow P\}_{i\in\Sigma}),$$

there exists a canonical homomorphism $\sigma_{\mathcal{B}} : \mathcal{U}^r_\Sigma \bullet\!\!\longrightarrow \mathcal{B}$. This fact, along with the fundamental isomorphism theorem (i.e. Theorem 5.7.3), enables one to think of any reduced Σ-preautomaton → Sec. 2.1

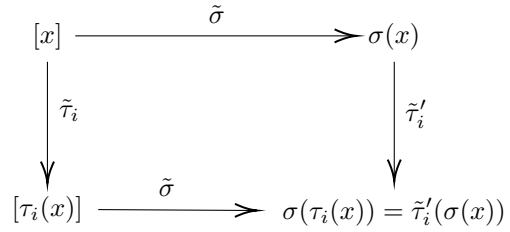


Figure 7.4 – A homomorphism from $\mathcal{A}/_{\sim_\sigma}$ to $\mathcal{B}$.

as a quotient of $\mathcal{U}^r_\Sigma$. This is our second trick to control and confine the space of species we are going to study. To set notation, given $w \stackrel{\text{def}}{=} w_1 w_2 \dots w_n$, let us define → Thm. 2.7.4

$$\mathcal{B}_r(w) \stackrel{\text{def}}{=} \eta_{w_n} \circ \cdots \circ \eta_{w_2} \circ \eta_{w_1}(p_0),$$

where q_0 and η_i are nullary and unary operations of $\mathcal{B}$.

Theorem 2.7.4 *For any reduced preautomaton $\mathcal{B}(P, p_0 \in P, \{\eta_i : P \bullet\!\!\longrightarrow P\}_{i\in\Sigma})$ there exists a canonical homomorphism $\sigma_B : \mathcal{U}^r_\Sigma \twoheadrightarrow \mathcal{B}$ defined as $\sigma_B(w) \stackrel{\text{def}}{=} \mathcal{B}_r(w)$.*

Proof. The mapping σ_B is a homomorphism (see Figure 7.5), since

$$\sigma_B(\epsilon) \quad = \quad p_0$$

and for any $i \in \Sigma$ we have,

$$\begin{aligned} \sigma_B(r_i(w)) &= \sigma_B(wi) \\ &= \eta_i \circ \eta_{w_n} \circ \cdots \circ \eta_{w_1}(p_0) \\ &= \eta_i(\sigma_B(w)), \end{aligned}$$

Hence, σ_B is an onto map since $\mathcal{B}$ is reduced. □ → Exr. 7.12.4

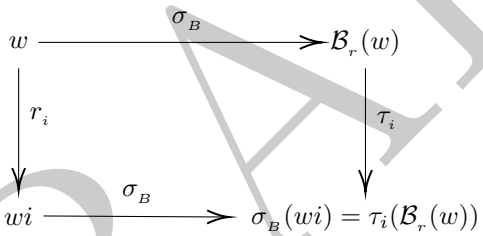


Figure 7.5 – The canonical onto homomorphism σ_B from $\mathcal{U}^r_\Sigma$ to $\mathcal{B}$.

Let us explain how Theorem 2.7.4 can be used efficiently to study Σ-preautomata. Think of strings w in Σ^* as tiny germs and assume that you have a special type of eyeglasses (designed for the Σ-preautomaton $\mathcal{B}$) that lower the resolution and wearing them, you can only distinguish equivalence classes $[a]_\sim$ as a collection of germs, where $\sim$ is the congruence relation induced by σ_B. Then Theorem 2.7.4 implies that for any given preautomaton $\mathcal{B}$, there is a pair of eyeglasses that wearing them you can see $\mathcal{B}$ as a preautomaton with states as subsets of $\mathcal{U}^r_\Sigma$ and transitions as some restrictions of *right concatenation*. This in a sense is telling that any preautomaton with any transition map can be redesigned as a quotient of $\mathcal{U}^r_\Sigma$ for which the transition maps are coming from right concatenation.

Corollary 3.7.4 *Any reduced preautomaton $\mathcal{B}(P, p_0 \in P, \{\eta_i : P \bullet\!\!\longrightarrow P\}_{i\in\Sigma})$ is isomorphic to $\mathcal{U}^r_\Sigma/\sim_{\sigma_B}$ where $\sim_{\sigma_B}$ is the congruence relation which is induced by the canonical homomorphism $\sigma_B : \mathcal{U}^r_\Sigma \twoheadrightarrow \mathcal{B}$.* → Thm. 5.7.3

It is instructive to add a note on the usage of word "right" used in Definition 1.7.4, emphasizing that, actually, the word essentially refers to the *end of input*, in the sense that the transition map always add the bit to the *rightmost* end of the input. For this, we may sometimes use the word *right congruence* relation for a congruence relation on $\mathcal{U}^r_\Sigma$ (resp. a *left congruence* relation is similarly defined for $\mathcal{U}^l_\Sigma$). It is also interesting to note that, following this line of thought, one may prove "left" counterparts of Theorem 2.7.4 and Corollary 3.7.4, *mutatis mutandis*, as follows. → $\mathcal{B}_r(w)$ → Sec. 7.6

Theorem 4.7.4 *For any reduced preautomaton $\mathcal{B}(P, p_0 \in P, \{\eta_i : P \bullet\!\!\longrightarrow P\}_{i\in\Sigma})$ there exists a canonical homomorphism $\sigma_B : \mathcal{U}^l_\Sigma \twoheadrightarrow \mathcal{B}$ defined as $\sigma_B(w) \stackrel{\text{def}}{=} \mathcal{B}_l(w)$.*

Corollary 5.7.4 *Any reduced preautomaton $\mathcal{B}(P, p_0 \in P, \{\eta_i : P \bullet\!\!\longrightarrow P\}_{i\in\Sigma})$ is isomorphic to* → Thm. 5.7.3

$\mathcal{U}^l_\Sigma/\sim_{\sigma_B}$ *where* $\sim_{\sigma_B}$ *is the congruence relation which is induced by the canonical homomorphism* $\sigma_B : \mathcal{U}^l_\Sigma \longrightarrow\!\!\bullet\, \mathcal{B}$.

We will talk more about this duality in the next forthcoming sections (see Section 7.6).

Example 6.7.4

Consider the $\{0,1\}$-preautomaton $\mathcal{B}$ depicted in Figure 7.6a, and note that applying Corollary 3.7.4 we find a (right) congruence relation $\sim$ with the equivalence classes → Exr. 7.12.5

$$[\epsilon]_\sim = \{\epsilon\}, \quad [1]_\sim = \{10^n \ : \ n \in \mathbb{N}\}, \quad \text{and} \quad [0]_\sim = \{0,1\}^* - ([1]_\sim \cup [\epsilon]_\sim).$$

Hence, we have $\mathcal{B} \cong \mathcal{U}^r/\sim$ while the canonical homomorphism is depicted in Figure 7.6b. ◇

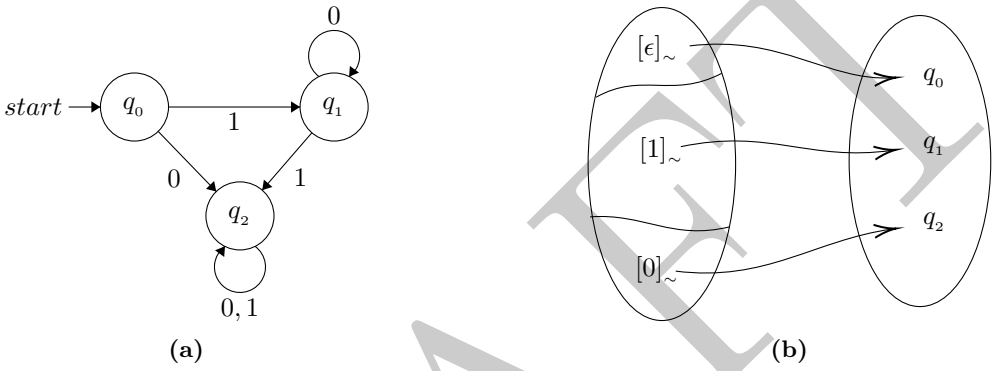


Figure 7.6 – See Example 6.7.4.

7.5 The minimal automaton

So far we have studied some basic properties of the structure *preautomaton* as an algebra, however, this is still far from the structure of a *computer*. As our first step toward filling this gap, we should somehow add the concept of an *output* to the structure that gives rise to the following definition.

Definition 1.7.5 Abstract automata

An *abstract automaton* $\mathcal{A}(Q, q_0 \in Q, \{\tau_i : Q \bullet\!\!\longrightarrow Q\}_{i\in\Sigma}, \gamma_\mathcal{A})$, is a preautomaton

$$\mathcal{A}(Q, q_0 \in Q, \{\tau_i : Q \bullet\!\!\longrightarrow Q\}_{i\in\Sigma})$$

along with an *output map* $\gamma_\mathcal{A} : Q \bullet\!\!\longrightarrow \{0,1\}$. Note that the output map naturally induces a *right output profile* by

$$\forall\ w \in \Sigma^*, \quad \chi_{L(\mathcal{A})}(w) \stackrel{\text{def}}{=} \gamma_\mathcal{A}(\mathcal{A}_r(w)),$$

which defines the (right) *language of* $\mathcal{A}$, denoted by $L(\mathcal{A})$, through its characteristic function. Note that, by default, *the language* of an abstract automaton is defined to be the language determined by the right output profile of the abstract automaton, where for the language determined by the left output profile, we explicitly use the word *left*. ▶

Hence, we will be facing three fundamental problems as follows,

- As the design problem, given a language $L \subseteq \Sigma^*$, when and how can we design an abstract automaton whose right output profile determines L?
- As an optimization problem, if such an abstract automaton exists, then how can we characterize and construct the *minimal abstract automaton*, $\mathcal{M}_r(L)$, whose right output profile determines L and has the minimum number of states?
- Given a language $L \subseteq \Sigma^*$, is it true that $\mathcal{M}_r(L) \cong \mathcal{M}_l(L)$ or they are different.

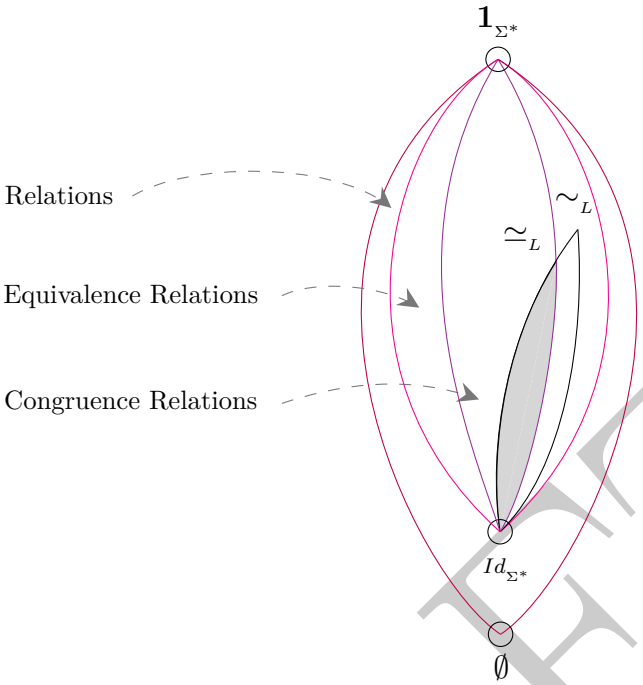


Figure 7.7 – See Theorem 3.7.5.

These are the main questions that we are going to answer hereafter. To begin, note that any language $L \subseteq \Sigma^*$ determines an equivalence relation $\sim_L$ with two equivalence classes L and L^c. In other words, for any pair or words x and y in Σ^*,

$$x \sim_L y \;\Leftrightarrow\; [\{x, y\} \subseteq L \text{ or } \{x, y\} \subseteq L^c]\,.$$

Also, note that, for any equivalence relation $R \subseteq \Sigma^* \times \Sigma^*$, the inclusion $R \subseteq \sim_L$ implies that each equivalence class of $\sim_L$ is partitioned by some equivalence classes of R, i.e. the partition consisting of equivalence classes of R is a *refinement* of the partition $\{L, L^c\}$ of Σ^*.

On the other hand, since the set of Σ-preautomata is in one to one correspondence with that of congruence relations on Σ^*, compatibility with the output map essentially means that we should look for congruence relations $R \subseteq \sim_L$ and then consider the Σ-preautomaton which is determined by R. As a trivial observation the inclusion $Id_{\Sigma^*} \subseteq \sim_L$ proves the following.

Proposition 2.7.5 The trivial abstract automata
Any language $L \subseteq \Sigma^*$ is the language of the abstract automaton $(\mathcal{U}^r_\Sigma, \chi_L)$.

Unfortunately, this trivial solution is not interesting at all, at least as far as the theory of computation is concerned, since the proposed abstract automaton has an *infinite* number of states.

Hence, as our next try, one may seek for the largest possible congruence relation $R \subseteq \sim_L$. Note that, this is not a trivial question since the property of *compatibility with respect to right concatenation* is not necessarily preserved by the union operation and consequently, union of two congruence relations may not be a congruence relation (see Figure 7.7). However, one may try to study the closure properties of *compatibility with respect to right concatenation* and try to characterize the closure of union of all congruence relations smaller than $\sim_L$. In what follows we will show that the following set

→ Sec.20.1.7

$$\mathcal{C}^r_L \stackrel{\text{def}}{=} \{R \subseteq \Sigma^* \times \Sigma^* \;:\; R \subseteq \sim_L \text{ and } R \text{ is a right congruence relation}\},$$

has a maximum element, and we will provide a characterization of this maximum element.

Theorem 3.7.5 Minimal abstract automaton (algebraic version)

Given a language $L \subseteq \Sigma^*$ the largest right congruence relation smaller than $\sim_{_L}$, which is denoted by $\simeq^r_{_L}$, exists and is defined as follows,

$$\forall\ \{x, y\} \subseteq \Sigma^*, \quad x \simeq^r_{_L} y \ \Leftrightarrow\ x\backslash L = y\backslash L.$$

In other words, $x \simeq^r_{_L} y$ if and only if for any $u \in \Sigma^*$, the words xu and yu are either both in L or both are not in L.

Proof. It is easy to verify that $\simeq^r_{_L}$ is an equivalence relation. First, let us verify that $\simeq^r_{_L}$ is indeed a congruence relation. For any choice of representatives x and y, we have

$$\begin{aligned}
x \simeq^r_{_L} y \quad &\Leftrightarrow \quad x\backslash L = y\backslash L \\
&\Leftrightarrow \quad \forall u \in \Sigma^* \ \ (\{xu, yu\} \subseteq L \text{ or } \{xu, yu\} \subseteq L^{^c}) \\
&\Rightarrow \quad \forall u \in \Sigma^* \ \ \forall i \in \Sigma \ \ (\{xiu, yiu\} \subseteq L \text{ or } \{xiu, yiu\} \subseteq L^{^c}) \\
&\Leftrightarrow \quad xi\backslash L = yi\backslash L \\
&\Leftrightarrow \quad xi \simeq^r_{_L} yi \\
&\Leftrightarrow \quad r_i(x) \simeq^r_{_L} r_i(y).
\end{aligned}$$

Then, we prove that $\simeq^r_{_L} \subseteq \sim_{_L}$. For this note that,

$$\begin{aligned}
x \simeq^r_{_L} y \quad &\Leftrightarrow \quad x\backslash L = y\backslash L \\
&\Leftrightarrow \quad \forall u \in \Sigma^* \ \ (\{xu, yu\} \subseteq L \text{ or } \{xu, yu\} \subseteq L^{^c}) \\
&\Rightarrow \quad (\{x\epsilon, y\epsilon\} \subseteq L \text{ or } \{x\epsilon, y\epsilon\} \subseteq L^{^c}) \\
&\Leftrightarrow \quad (\{x, y\} \subseteq L \text{ or } \{x, y\} \subseteq L^{^c}) \\
&\Leftrightarrow \quad x \sim_{_L} y.
\end{aligned}$$

Finally, let us prove that for any given congruence relation ρ, we have

$$\rho \ \subseteq \sim_{_L} \implies \rho \ \subseteq \simeq^r_{_L},$$

showing that $\simeq^r_{_L}$ is the maximum element of $\mathcal{C}_{_L}$. To do so, first, by induction on the length of u we prove

$$(x, y) \in \rho \quad \Rightarrow \quad \forall u \in \Sigma^* \ \ (xu, yu) \in \rho.$$

Note that the claim is true when $u = \epsilon$. Also, if we know that the claim is true for all u with $|u| \leq k$, then $(x, y) \in \rho$ and $(xu, yu) \in \rho$ for some $|u| \leq k$ implies that

$$\forall\ i \in \Sigma \quad (r_i(xu), r_i(yu)) \in \rho \quad \Rightarrow \quad \forall\ i \in \Sigma \quad (xui, yui) \in \rho.$$

Consequently, we have,

$$\begin{aligned}
(x, y) \in \rho \quad &\Leftrightarrow \quad \forall u \in \Sigma^* \ \ (xu, yu) \in \rho \\
&\Rightarrow \quad \forall u \in \Sigma^* \ \ xu \sim_{_L} yu \\
&\Leftrightarrow \quad \forall u \in \Sigma^* \ \ (\{xu, yu\} \subseteq L \text{ or } \{xu, yu\} \subseteq L^{^c}) \\
&\Leftrightarrow \quad x\backslash L = y\backslash L \\
&\Leftrightarrow \quad x \simeq^r_{_L} y.
\end{aligned}$$

□

Theorem 3.7.5 explicitly describes the largest right congruence relation smaller that $\sim_{_L}$, giving rise to the minimal abstract automaton $\mathcal{M}_{_r}(L) \stackrel{\text{def}}{=} \mathcal{U}^r_\Sigma / \simeq^r_{_L}$. In the sequel, we are going to provide another explicit equivalent description of this abstract automaton as follows.

Definition 4.7.5 The minimal (right) abstract automaton of $L \subseteq \Sigma^*$

Given a language $L \subseteq \Sigma^*$, we define the abstract automaton $\mathcal{M}_{_r}(L)$ as the abstract au-

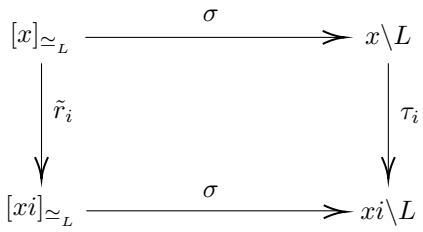


Figure 7.8 – See Theorem 5.7.5.

tomaton with the set of states

$$Q^r_{\mathcal{M}} \stackrel{\text{def}}{=} \{x\backslash L \ : \ x \in \Sigma^*\},$$

and the starting state (i.e. the nullary operation) $x\backslash L = L \in Q^r_{\mathcal{M}}$. Also, for any $i \in \Sigma$, the transition maps $\tau_{_i}$ is defined as *left quotient by* i, i.e.

$$\tau_{_i}(x\backslash L) \stackrel{\text{def}}{=} i\backslash(x\backslash L) = (xi)\backslash L \in Q^r_{\mathcal{M}}.$$

Moreover, the output map $\gamma_{\mathcal{M}} : Q^r_{\mathcal{M}} \longmapsto \{0,1\}$ is defined as,

$$\gamma_{\mathcal{M}}(x\backslash L) = 1 \quad \Leftrightarrow \quad \epsilon \in x\backslash L.$$

▶

The following theorem shows that what is defined in Definition 4.7.5 is actually isomorphic to the right minimal automaton $\mathcal{M}_{_r}(L) \stackrel{\text{def}}{=} \mathcal{U}^r_\Sigma/\simeq^r_L$.

Theorem 5.7.5 Minimal (right) abstract automaton (explicit version)
The abstract automaton defined in Definition 4.7.5 is isomorphic to the right abstract automaton $\mathcal{U}^r_\Sigma/\simeq^r_L$ describing L, with the minimum number of states, as its right output profile.

Proof. In order to show this isomorphism, consider the map $\sigma : \mathcal{U}^r_\Sigma/\simeq^r_L \longmapsto \mathcal{M}_{_r}(L)$ defined as $\sigma([x]_{\simeq^r_L}) \stackrel{\text{def}}{=} x\backslash L$ for any $x \in \Sigma^*$. For any choice of representatives x and y, where $x \simeq^r_L y$, we have

$$\begin{aligned} [x]_{\simeq^r_L} = [y]_{\simeq^r_L} \quad &\Leftrightarrow \quad x \simeq^r_L y \\ &\Leftrightarrow \quad x\backslash L = y\backslash L \\ &\Leftrightarrow \quad \sigma([x]_{\simeq_L}) = \sigma([y]_{\simeq_L}). \end{aligned}$$

Hence, the map is well-defined and one to one. It is also trivially onto, since for any $x\backslash L \in Q_{\mathcal{M}}$ there exists $[x]_{\simeq_L}$ in the state space of $\mathcal{U}^r_\Sigma/\simeq^r_L$ such that

$$\sigma([x]_{\simeq_L}) = x\backslash L.$$

Finally, let us prove that the map is a homomorphism. For any $i \in \Sigma$ and $x \in \Sigma^*$, we have

$$\begin{aligned} \tau_{_i}(\sigma([x]_{\simeq^r_L})) \quad &= \quad \tau_{_i}(x\backslash L) \\ &= \quad (xi)\backslash L \\ &= \quad \sigma([xi]_{\simeq^r_L}) \\ &= \quad \sigma(\tilde{r}_{_i}([x]_{\simeq^r_L})). \end{aligned}$$

□

As we mentioned earlier, the computational model of an abstract automaton is essentially trivial without any restrictions on the number of states, since any language can be considered as the language of a trivial infinite abstract automaton as stated in Proposition 2.7.5. This setup

is also quite uninteresting since it contradicts the very fundamental and basic assumption of the theory of computation stating that computational models must have finite descriptions. These facts motivate the following definition. Assumptions

Definition 6.7.5 Regular languages

A language $L \subseteq \Sigma^*$ is said to be a *regular* language, if the (right) congruence relation $\simeq_L^r$ has a finite number of equivalence classes. Note that Theorem 5.7.5 implies that any regular language can be finitely described by a finite abstract automaton, and consequently, admits finite description by regular computers. ▶ → Sec.7.1

Example 7.7.5

Consider the language $L \stackrel{\text{def}}{=} \{1\,0^n \ : \ n \in \mathbb{N}\}$. Let us verify, using Definition 6.7.5, that L is a regular language. To do this, we should somehow verify that the number of sets of the form $x\backslash L$ is finite. Hence, we make a list as follows.

- $\epsilon\backslash L = L$,
- $0\backslash L = \emptyset$,
- $1\,0^j\backslash L \stackrel{\text{def}}{=} Z \qquad \forall\, j \in \mathbb{N}$,
- $0u\backslash L = \emptyset \qquad u \in \Sigma^*$,
- $11u\backslash L = \emptyset \qquad u \in \Sigma^*$,
- $101u\backslash L = \emptyset \qquad u \in \Sigma^*$.

Now its easy to verify, using Definition 4.7.5, that the minimal (right) abstract automaton of L is exactly the preautomaton depicted in Figure 7.6a, with the output map γ that maps q_1 to one and the other states to zero. ◇

7.6 Left or right?

You are given a set $L \in \Sigma^*$ and you are supposed to find a finite description for it. Your strategy is to, somehow, find a pattern in strings that can characterize L in Σ^*. As far as you are looking for patterns, you may not be necessarily concerned about which side of the string is its beginning and which side is its end. However, if you decide to find such a characterization by solving the membership problem-type, using a computer, then you have to decide how you are supposed to feed each string into the machine. → Sec. 1.2

Assume that you are living on a planet $\mathcal{E}$ in which people always feed strings into computers starting from the leftmost bit of the string, meaning that the rightmost bit is the *end of input*. Also, assume that your friend, who is facing the same problem with L, lives on the planet $\mathcal{J}$ in which people feed strings into computers starting from the rightmost bit of the string, where the leftmost bit is the end of input. Needless to say, as far as the language L is concerned, you and your friend both are facing the same problem, however, since you have decided to find descriptions which are based on solving the membership problem-type using computers, it is possible that you find different answers on $\mathcal{E}$ or $\mathcal{J}$.

To see this, consider a simple language as

$$L \stackrel{\text{def}}{=} \{w1 \ : \ w \in \{0,1\}^*\}.$$

Given a string $u \in \{0,1\}^*$, in order to decide whether u is in L or not, one has to verify whether the rightmost bit of u is equal to 1 or not. However, clearly, there is a simple difference in these verification procedures if one starts from the left or from the right of the word u, at least in terms of the time one has to take to reach the rightmost bit.

This simple example shows that solving the membership problem-type from the left or from the right, using computers, may come to different conclusions. Also, if you are interested in solving your problem on planet $\mathcal{E}$ but using the machinery and facilities of your friend on planet $\mathcal{J}$, you may send the language L^R to your friend on $\mathcal{J}$ and ask her to solve the membership

problem under $\mathcal{J}$'s standards. Hence, one understands that, to complete the scenario of our *right* theory developed so far, one may either use a dual *left* theory or one may apply the existing *right* theory to the language L^R. → Exr. 20.3.10

Therefore, one may go through similar steps we took in Sections 7.2, 7.3, 7.4 and 7.5, *mutatis mutandis*, defining the minimal *left* abstract automaton and prove the following counterparts of our previous facts about right abstract automata.

Definition 1.7.6 The minimal left abstract automaton of $L \subseteq \Sigma^*$

Given a language $L \subseteq \Sigma^*$, we define the left abstract automaton $\mathcal{M}_l(L)$ as the abstract automaton with the set of states

$$Q^l_{\mathcal{M}} \stackrel{\text{def}}{=} \{x/L \ : \ x \in \Sigma^*\},$$

and the starting state (i.e. the nullary operation) $x/L = L \in Q^l_{\mathcal{M}}$. Also, for any $i \in \Sigma$, the transition maps τ_i is defined as *right quotient by* i, i.e.

$$\tau_i(x/L) \stackrel{\text{def}}{=} i/(x/L) = (xi)/L \in Q^l_{\mathcal{M}}.$$

Moreover, the output map $\gamma_{\mathcal{M}} : Q^l_{\mathcal{M}} \longmapsto \{0,1\}$ is defined as,

$$\gamma_{\mathcal{M}}(x/L) = 1 \quad \Leftrightarrow \quad \epsilon \in x/L.$$

▶

Theorem 2.7.6 Minimal left abstract automaton (algebraic left version)

Given a language $L \subseteq \Sigma^*$ the largest left congruence relation smaller than $\sim_L$, which is denoted by $\simeq^l_L$, exists and is defined as follows,

$$\forall\ \{x,y\} \subseteq \Sigma^*, \quad x \simeq^l_L y \ \Leftrightarrow \ x/L = y/L.$$

In other words, $x \simeq^l_L y$ if and only if for any $u \in \Sigma^*$, the words ux and uy are either both in L or both are not in L.

Theorem 3.7.6 Minimal left abstract automaton (explicit left version)

The abstract automaton defined in Definition 1.7.6 is isomorphic to the right abstract automaton $\mathcal{U}^l_\Sigma/\simeq^l_L$ describing L, with the minimum number of states, as its left output profile.

Now, a fundamental question that may be raised in relation to Definition 6.7.5 is,

"Does there exists a regular language L whose minimal left abstract automaton has an infinite number of states?"

Note that, by symmetry of theories, it is straight forward to see that analyzing the above mentioned question is essentially equivalent to analyzing the dual question concerning left-regular languages.

We leave it for the reader to answer this question in Exercise 7.12.7, while this will be discussed in more detail in Section 7.7 from another viewpoint. On the other hand, the following example shows that there are cases for which we have $\mathcal{M}_r(L) \cong \mathcal{M}_l(L)$.

Example 4.7.6

Again consider the language $L \stackrel{\text{def}}{=} \{1\,0^n \ : \ n \in \mathbb{N}\}$ of Example 7.7.5 and let us try to find its minimal left abstract automaton using Definition 1.7.6. Analyzing the right quotients yields the following list,

- $L/\epsilon = L/0 = L$,
- $L/1 = \{\epsilon\}$,
- $L/0^j = L \ \ \forall\ j \in \mathbb{N}$,
- $L/1\,0^j = \{\epsilon\} \ \ \forall\ j \in \mathbb{N}$,

- $L/u = \emptyset \quad u \neq \epsilon,\ u \neq 0^j, u \neq 1\,0^j \quad \forall\ j \in \mathbb{N},$

showing that the minimal left abstract automaton is isomorphic to that of right abstract automaton (see Figure 7.6a). ◇

7.7 Reversed automaton and nondeterminism

7.8 Some other descriptions

7.9 The four fundamental classes

7.10 Some extra properties