Crash Course on Regular Languages

The theory of finite automata has preserved from its origins a great diversity of aspects. From one point of view, it is a branch of mathematics connected with the algebraic theory of semigroups and associative algebras. From another point of view, it is a branch of algorithm design concerned with string manipulation and sequence processing. It is perhaps this diversity that has enriched the field to make it presently one with both interesting applications and significant mathematical problems.

Dominique Perrin [PER 90]

This chapter is intended to be short. Automata theory and formal language theory have been developed for more than 50 years. There are many textbooks devoted to these theories (and we can easily find series of exercises). To cite just a few (all these books cover in particular what is nowadays considered as classical material): [HOP 79] or [SUD 06] where the focus is oriented toward the computational models and the corresponding algorithms, the comprehensive [SAK 09] where a wider perspective, a more general algebraic framework and emphasis on the underlying structures are provided. I personally like the presentation and the material covered in [SHA 08]. I can also mention [LAW 04] (for a light
introduction to the syntactic monoid of a language) or the classic [EIL 74]. See also, the survey [YU 97] or [PER 90] for a condensed exposition.

We have already encountered the notion of automaton in several sections of this book: the formal presentation of deterministic automaton first appeared in definition 2.23. Then the extended model with output was given in definition 2.27, Volume 1, to deal with automatic sequences. We will work only with these types of finite machines\(^1\).

First, we will provide summary of some basic results about finite automata and regular languages. Then, we will select some particular topics related to the main themes of the book: recognizable sets of numbers and morphic or automatic words. The last section will present regular languages of polynomial growth. Thus, their characterization is applied to growing letters in a morphic word. Also, this chapter will serve as a preparation for the last chapter of the book where automata will play an important role when dealing with decision procedures. For an easy understanding of Chapter 3, the readers familiar with automata theory should consult sections 1.3 and 1.7.

1.1. Automata and regular languages

We begin with a definition that we have already described in definition 2.23, Volume 1. But, here the focus is put on the words, and thus on the language that is accepted by such a machine.

DEFINITION 1.1.– A *deterministic finite automaton*, or DFA for short, over an alphabet \(B\) is given by a 5-tuple \(\mathcal{A} = (Q, q_0, B, \delta, F)\), where \(Q\) is a finite set of states, \(q_0 \in Q\) is

---

\(^1\) The readers could also remember that cellular automata were introduced in section 1.3.1, Volume 1, but such a model of computation is out of the scope of the present chapter.
the initial state, $\delta : Q \times B \to Q$ is the transition function and $F \subseteq Q$ is the set of final states. The map $\delta$ can be extended to $Q \times B^*$ by setting $\delta(q, \varepsilon) = q$ and $\delta(q, wa) = \delta(\delta(q, w), a)$ for all $q \in Q$, $a \in B$ and $w \in B^*$. The language accepted or recognized by $A$ is

$$L(A) = \{w \in B^* \mid \delta(q_0, w) \in F\}.$$

A regular language is a language accepted by some DFA.

Given a DFA, we have a partition of $B^* = L(A) \cup (B^* \setminus L(A))$. A word $w$ is trivially either accepted whenever $\delta(q_0, w) \in F$ or, rejected whenever $\delta(q_0, w) \notin F$. Since the automaton is deterministic\(^2\), exactly one of the two situations occurs. So to speak, given a word $w$, we start reading this word from the initial state, one letter at a time from left to right. Following the transitions of $\delta$, the current state is updated until the whole word has been read. At that time, we test if the reached state is final or not.

**Example 1.2.–** Consider the DFA over $\{a, b\}$ having $\{0, 1, 2\}$ as set of states. The transition function is defined by $\delta(0, a) = 0$, $\delta(0, b) = 1$, $\delta(1, a) = 2$, $\delta(1, b) = 1$, $\delta(2, a) = 2$, $\delta(2, b) = 2$. The initial state (represented with an in-coming arrow) is 0, and the final states are 0, 1 (represented with an out-going arrow). This automaton is depicted in Figure 1.1. For a word $w = w_0 \cdots w_\ell$, we can consider the behavior (or state transition sequence) of the DFA, that is the sequence of $\ell + 2$ states

$$q_0, \delta(q_0, w_0), \ldots, \delta(q_0, w_0 \cdots w_i), \ldots, \delta(q_0, w_0 \cdots w_\ell).$$

\(^2\) Also, the domain of $\delta$ is the whole set $Q \times B^*$. If we allow $\delta$ to be a partial function, i.e. defined on a subset of $Q \times B^*$, then $\delta(q, w)$ could be undefined for some $w \in B^*$, in that case, the word $w$ is rejected. It is indeed common to consider only states that may lead eventually to some final state. Therefore, if we remove useless states (and the corresponding transitions), this means that the corresponding transition function is a partial function. In such a case, the automaton is still considered to be deterministic even though *stricto sensu* the domain of $\delta$ should be the whole set $Q \times B$ and by extension $Q \times B^*$. 
Figure 1.1. A DFA accepting \( \{a^i b^j \mid i, j \geq 0\} \)

Reading \( aabb \) or \( ababb \) gives, respectively,

\[
0 \xrightarrow{a} 0 \xrightarrow{a} 0 \xrightarrow{b} 1 \xrightarrow{b} 1 \\
\text{and } 0 \xrightarrow{a} 0 \xrightarrow{b} 1 \xrightarrow{a} 2 \xrightarrow{b} 2 \xrightarrow{b} 2.
\]

The word \( aabb \) is accepted because \( \delta(q_0, aabb) = 1 \in F \) but \( ababb \) is not because \( \delta(q_0, ababb) = 2 \notin F \). It is easy to see that the language accepted by this DFA is \( \{a^i b^j \mid i, j \geq 0\} \).

As we will see soon (when dealing with operations on regular languages), it is also interesting to introduce a more general model of machines: the non-deterministic automata.

**Definition 1.3.** – A non-deterministic finite automaton, or NFA for short, over an alphabet \( B \) is given by a 5-tuple \( N = (Q, I, B, \Delta, F) \), where \( Q \) is a finite set of states, \( I \subseteq Q \) is the set of initial states, \( \Delta \subseteq Q \times B^* \times Q \) is the (finite) transition relation and \( F \subseteq Q \) is the set of final states. A word \( w \) is accepted by \( N \) if there exists an integer \( i \), some (possibly empty) words \( v_1, \ldots, v_i \) and a sequence of states \( q_1, \ldots, q_{i+1} \) such that \( w = v_1 \cdots v_i \) and

\[
-(q_1, v_1, q_2), (q_2, v_2, q_3), \ldots, (q_i, v_i, q_{i+1}) \in \Delta \\
-q_1 \in I, q_{i+1} \in F.
\]

Otherwise stated, there is at least one accepting path from an initial state to some final state with label \( w \). The language accepted by \( N \) is the set of accepted words. We can assume that \( (q, \varepsilon, q) \) belongs to \( \Delta \) for all \( q \in Q \).
The acceptance condition for an NFA $N$ can be written in a different way. It is useful to introduce the following notation (that we will use in the determinization algorithm). Let $R \subseteq Q$ be a subset of states of $N$. Let $w$ be a finite word. We let

$$R.w$$

to denote the set of states defined as follows. The state $r \in Q$ belongs to $R.w$ if and only if there exists an integer $i$, some (possibly empty) words $v_1, \ldots, v_i$ and a sequence of states $q_1, \ldots, q_i$ such that $w = v_1 \cdots v_i$,

$$q_1 \in R \quad \text{and} \quad (q_1, v_1, q_2), (q_2, v_2, q_3), \ldots, (q_i, v_i, r) \in \Delta.$$

Otherwise stated, there is a path of label $w$ starting in a state belonging to $R$ and leading to $r$. Therefore, the language accepted by $N$ is

$$\{ w \in B^* | I.w \cap F \neq \emptyset \}.$$

**Example 1.4.**— We have depicted an NFA in Figure 1.2. The readers may notice that we have two initial states: 0 and 3. There are several $\varepsilon$-transitions, i.e. transitions labeled by the empty word, for instance, from 1 to 4. Reading the symbol $a$ from state 0 can lead to any of the four states 0, 1, 3, 4. With the notation introduced above

$$\{0\}.a = \{0, 1, 3, 4\} = \{0, 3\}.a$$

and $\{5\}.b = \{1, 2, 4, 5\}$. Indeed we have, for instance, that

$$(5, \varepsilon, 1), (1, b, 1), (1, \varepsilon, 4) \in \Delta.$$  

There are several ways to accept the word $aa$ (but also some non-accepting paths, although the existence of one accepting path is enough).
Figure 1.2. A non-deterministic automaton over \( \{a, b\} \)

**Remark 1.5.** Every DFA is a special case of an NFA. It has a unique initial state, and the relation \( \Delta \) is a subset of \( Q \times B \times Q \) that is the graph of a function.

We might suspect that the set of languages accepted by deterministic finite automata over \( B \) is a (strict) subset of the set of languages accepted by non-deterministic automata over \( B \). Nevertheless, any language accepted by an NFA is also accepted by a DFA (this is the Rabin–Scott theorem from 1959). We do not prove this result here, but below we present a determinization algorithm (without showing its correctness).

First, we can assume that the given NFA is *elementary*. This means that every transition \( (q, w, q') \in \Delta \) is such that \( |w| \leq 1 \). Indeed, if we have a transition \( (q, w, q') \), where \( w = w_1 \cdots w_\ell \) is a word of length \( \ell \geq 2 \), then we can replace this transition with \( \ell \) new transitions (where the states \( q_1, \ldots, q_{\ell-1} \) are newly created states):

\[
(q, w_1, q_1), (q_1, w_2, q_2), \ldots, (q_{\ell-2}, w_{\ell-1}, q_{\ell-1}), (q_{\ell-1}, w_\ell, q')
\]

For instance, the NFA in Figure 1.2 is elementary.

**Algorithm 1.6 (subsets construction).** The input is a non-deterministic automaton \( \mathcal{N} = (Q, I, B, \Delta, F) \) accepting some language \( L \). We can assume that \( \mathcal{N} \) is elementary. The output
is a DFA $\mathcal{A}$ accepting the same language $L$. The states of $\mathcal{A}$ are subsets of $Q$. The deterministic automaton $\mathcal{A}$ is obtained as follows.

- The initial state of $\mathcal{A}$ is $I.\varepsilon \subseteq Q$. It is the unique state of $\mathcal{A}$ produced when initializing the algorithm.

- At each step of the algorithm, for each state $S$ in $\mathcal{A}$ newly created during the previous step, compute $S.b$ for all $b \in B$. These are states of $\mathcal{A}$, i.e. subsets of states of $N$. If some new state appears, then it is a newly created state (and we iterate this step of the procedure).

The algorithm stops when no new state is created (the procedure always terminates because there are at most $2^{\text{Card}(Q)}$ states in $\mathcal{A}$).

- The transition function of $\mathcal{A}$ is $\delta(S, b) = S.b$ for all states $S$ and $b \in B$.

- A state $S$ of $\mathcal{A}$ is final whenever $S \cap F \neq \emptyset$.

Because the algorithm takes into account only states created from some previous step, the DFA $\mathcal{A}$ computed by the above algorithm is an accessible automaton, i.e. for every state $q$ in $\mathcal{A}$, there exists a word $w$ such that reading $w$ from the initial state of $\mathcal{A}$ leads to $q$. Otherwise stated, every state can be reached from the initial state.

We can also see that if an NFA has $n$ states, we could possibly obtain some DFA with up to $2^n$ states. In some cases, such an exponential blow-up is unavoidable\(^3\). Such an observation is crucial when dealing with practical

\(^3\) Let $p_1, \ldots, p_k$ be the first $k$ prime numbers. Consider the language $L_k = \{a^n \mid n \in p_1 \mathbb{N} \cup \cdots \cup p_k \mathbb{N}\}$. It is not difficult to prove that this language is accepted by an NFA with $p_1 + \cdots + p_k$ states, but any DFA accepting $L_k$ has at least $p_1 \cdots p_k$ states. (Left as an exercise.) Working over a unary alphabet reduces the arguments to number-theoretic results like the application of Bézout’s identity.
implementations like those that we will consider in Chapter 3.

**Example 1.7.**– If we apply the subsets construction to the NFA given in Figure 1.2, we get the following subsets as states of a DFA

\[
\{0, 3\}, \{0, 1, 3, 4\}, \{4\}, \{0, 1, 3, 4, 5\}, \{1, 4\}, \{1, 4, 5\}, \emptyset, \{1, 2, 4, 5\}.
\]

Starting with \(\{0, 3\}\), we compute sets of the form \(R.a\) and \(R.b\) as follows.

<table>
<thead>
<tr>
<th>(R)</th>
<th>(R.a)</th>
<th>(R.b)</th>
</tr>
</thead>
<tbody>
<tr>
<td>({0, 3})</td>
<td>({0, 1, 3, 4})</td>
<td>({4})</td>
</tr>
<tr>
<td>({0, 1, 3, 4})</td>
<td>({0, 1, 3, 4, 5})</td>
<td>({1, 4})</td>
</tr>
<tr>
<td>({4})</td>
<td>({1, 4, 5})</td>
<td>(\emptyset)</td>
</tr>
<tr>
<td>({0, 1, 3, 4, 5})</td>
<td>({0, 1, 3, 4, 5})</td>
<td>({1, 2, 4, 5})</td>
</tr>
<tr>
<td>({1, 4})</td>
<td>({1, 4, 5})</td>
<td>({1, 4})</td>
</tr>
<tr>
<td>({1, 4, 5})</td>
<td>({1, 4, 5})</td>
<td>({1, 2, 4, 5})</td>
</tr>
<tr>
<td>(\emptyset)</td>
<td>(\emptyset)</td>
<td>(\emptyset)</td>
</tr>
<tr>
<td>({1, 2, 4, 5})</td>
<td>({1, 4, 5})</td>
<td>({1, 2, 4, 5})</td>
</tr>
</tbody>
</table>

The final states are the subsets containing a final state of the initial NFA:

\[
\{0, 3\}, \{0, 1, 3, 4\}, \{0, 1, 3, 4, 5\}, \{1, 4, 5\}, \{1, 2, 4, 5\}.
\]

The corresponding DFA is depicted in Figure 1.3.

**Theorem 1.8.**– The set of regular languages over a fixed alphabet \(B\), i.e. languages accepted by some finite (deterministic or non-deterministic) automaton, is closed under the following operations: union, complement, concatenation, Kleene star, image by morphism and reversal. Moreover, all the corresponding automata can be effectively obtained.

**Proof.**– Let \(\mathcal{A} = (Q, q_0, B, \delta, F)\) and \(\mathcal{A}' = (Q', q'_0, B, \delta', F')\) be two DFAs. We build new automata based on \(\mathcal{A}\) and \(\mathcal{A}'\).
The language $L(A) \cup L(A')$ is accepted by the NFA obtained by “merging” the automata $A$ and $A'$ (there is no transition between the two automata). We have one “big” automaton that is the disjoint union of the first two ones. It simply has two initial states $q_0$ and $q'_0$. The set of states is $Q \cup Q'$. Indeed, a word is accepted if it is accepted by at least one of the two automata.

The language $B^* \setminus L(A)$ is accepted by the DFA obtained from $A$ where the updated set of final states is $Q \setminus F$.

The language $L(A) L(A')$ is accepted by the NFA obtained by considering sequentially the two automata: the unique initial state is $q_0$, the final states are those of $A'$ (the states in $F$ are no longer final), and we add $\varepsilon$-transitions from all states in $F$ to $q'_0$.

The language $L(A)^*$ is accepted by the NFA where we add a new state $r_0$ to $A$ and $\varepsilon$-transitions from $r_0$ to $q_0$ and from

---

**Figure 1.3. Determinization of a non-deterministic automaton over \{a, b\}**
every state in $F$ to $r_0$. The state $r_0$ is the unique initial and final state of this new automaton.

Let $f : B^* \rightarrow A^*$ be a morphism. We consider the NFA where each label $b$ is replaced accordingly with $f(b)$. It is indeed an NFA because the two letters can have the same image by $f$ or the morphism can be erasing. This NFA accepts the language $f(L(A))$.

For the reversal of $L(A)$, consider the NFA where the set of initial states is $F$, the unique final state is $q_0$ and all the transitions have been reversed: if $\delta(q,b) = r$ then $(r,b,q)$ belongs to the transition relation of the NFA.

In many constructions (like the conjunction of two logical formulae discussed in the last chapter), the intersection of two languages is an important operation to carry on. Slightly modifying the construction given below also allows us to derive – almost for free – a closure result about the shuffle of two regular languages (even though such an operation is not as useful as an intersection\(^4\)).

**Theorem 1.9.** The set of regular languages over a fixed alphabet $B$, i.e. languages accepted by some finite (deterministic or non-deterministic) automaton, is closed under intersection and shuffle.

To show that the set of regular languages is closed under intersection and shuffle, we introduce the *product* of two DFAs. The idea is that the product should mimic (i.e. keep track of) the behavior of both original automata.

**Definition 1.10.** Let $\mathcal{A} = (Q,q_0,B,\delta,F)$ and $\mathcal{A}' = (Q',q'_0,B,\delta',F')$ be two DFAs over $B$. Consider the *product*}

\(^4\) Nevertheless, an interesting application is to compare the growth function (definition 1.33, Volume 1) of the languages $L$, $M$ and $L \sqcup M$. See, for instance, [RIG 02a] where the shuffle product is used to obtain languages with a given polynomial growth function.
automaton $P$ where
- $Q \times Q'$ is the set of states;
- $(q_0, q'_0)$ is the initial state;
- The transition function $\lambda : (Q \times Q') \times B \rightarrow Q \times Q'$ is defined by
  \[ \lambda((q, q'), b) = (\delta(q, b), \delta'(q', b)), \quad \forall (q, q') \in Q \times Q', \ b \in B. \]

In the product automaton, if the set of final states is $F \times F'$, then this automaton accepts exactly $L(A) \cap L(A')$.

**Example 1.11.**– In Figure 1.4, at the top we have presented a DFA accepting the language \{a^i b^j \mid i, j \geq 0\} considered in example 1.2. On the lower left, we have presented a DFA accepting words over \{a, b\} with no factor bb. On the lower right of the figure, we have depicted the accessible part of the corresponding product automaton.

Recall that the (ordinary) shuffle of two finite words $u$ and $v$ is the set $u \shuffle v$ of words obtained when merging $u$ and $v$ from left to right, but choosing the next symbol arbitrarily from $u$ or $v$. As an example (having words over disjoint alphabets permits us to easily compute the shuffle),
\[ ab \shuffle cd = \{abcd, acbd, acdb, cabd, cdab\}. \]

We can define accordingly the shuffle of two languages $L$ and $M$ as
\[ L \shuffle M = \bigcup_{u \in L, v \in M} u \shuffle v. \]

We can easily modify the product automaton given in definition 1.10 to accept the language $L(A) \shuffle L(A')$ where $A$ and $A'$ are DFAs. First, we can assume that $A$ and $A'$ are defined over disjoint alphabets $B$ and $B'$ (if this is not the case, we can replace one of the two alphabets with new
symbols, and we apply an injective morphism $h : B^* \rightarrow B'^*$ that maps $b_i$ to $c_i$). Consider the same construction as in definition 1.10 but consider the transition function defined by

$$\lambda'((q, q'), c) = \begin{cases} (\delta(q, c), q'), & \text{if } c \in B; \\ (q, \delta'(q', c)), & \text{if } c \in B'. \end{cases}$$

**Figure 1.4. Two DFAs and the corresponding product**

Note that $\lambda'$ is well-defined because $B \cap B' = \emptyset$. The set of final states is, as for the intersection of languages, $F \times F'$. Having two disjoint alphabets permits us to act precisely on one of the two automata. If initially, the two automata were not given over disjoint alphabets, we can finish the proof by applying the morphism $h^{-1}$ to the labels belonging to $B'$.

**Remark 1.12.**– Every finite language is regular. The readers will be convinced by the following example. Take the language $\{abba, ab, bba\}$. Simply consider the subgraph of the full binary tree as depicted in Figure 1.5. In particular, this
means that adding or removing a finite number of words to a regular language will not alter its regularity. Regular languages are closed under finite modifications.

To conclude with this introductory section, we say a few words about regular expressions. DFAs and NFAs are acceptors of regular languages, while regular expressions are generators of these languages. Let \( B \) be an alphabet not containing the symbols \(+, \cdot, *, (, ), e, 0\). A regular expression over \( B \) is a word over \( B \cup \{+, \cdot, *, (, ), e, 0\} \) that is obtained by applying a finite number of times the following rules:

- \( 0, e \) and \( b \), for all \( b \in B \) are regular expressions;
- if \( R, S \) are regular expressions, so are \((R + S), (R \cdot S)\) and \( R^* \).

For instance, \(((a \cdot b)^* + e) + ((a \cdot b)^* + e)^*\) is a regular expression. We have just defined a way to produce syntactically “valid” expressions. Now we add some semantics: with each regular expression a language is associated. Let \( \text{Reg}(B) \) denote the set of regular expressions over \( B \). We define a map \( \varphi : \text{Reg}(B) \to B^* \) as follows. Let \( R, S \in \text{Reg}(B) \). We set

- \( \varphi(0) = \emptyset; \)
- \( \varphi(e) = \{\varepsilon\}; \)
- \( \varphi(b) = \{b\} \) for all \( b \in B; \)
- \( \varphi((R + S)) = \varphi(R) \cup \varphi(S); \)
- \( \varphi((R \cdot S)) = \varphi(R) \varphi(S); \)
- \( \varphi(R^*) = (\varphi(R))^*. \)
It is common to identify a regular expression $R$ with the language $\varphi(R)$. Also, several expressions may correspond to the same language. Kleene’s theorem states that a language is regular if and only if it is generated by some regular expression. There are several classical methods for building an automaton accepting the language described by a regular expression. Conversely, from a regular expression $R$, we can derive an automaton that accepts $\varphi(R)$. From this, the readers may be convinced that the set of regular languages over $B$ is the smallest collection of languages containing the finite languages and closed under union, concatenation and Kleene’s star.

### 1.2. Adjacency matrix

The adjacency matrix of a DFA $A = (Q, q_0, B, \delta, F)$ is a matrix $M(A)$ whose entries are indexed by $Q$ and where $(M(A))_{p,q} = \text{Card}\{b \in B \mid \delta(p, b) = q\}$.

Note that this matrix is similar to the one introduced in section 2.2.1, Volume 1, and associated with a morphism. Actually, take a morphism $f$ and the associated matrix $M_f$. We have associated with some automaton $A_f$; see, for instance, examples 2.24 and 2.31, Volume 1. If $f$ is a constant-length morphism, then the DFA $A_f$ has a total transition function. Otherwise, the transition function is partial. More about these links is discussed at the end of section 1.8.

It is easy to see that the adjacency matrix of the DFA $A_f$ is the transpose of the matrix $M_f$. We could decide to have a uniform presentation, but if we stick to the usual conventions, then let us make a distinction when dealing with morphisms or automata. In any event, spectral properties of matrices are invariant under transposition.

**Proposition 1.13.** Let $A = (Q, q_0, B, \delta, F)$ be a DFA. Let $M(A) \in \mathbb{N}^{Q \times Q}$ be the adjacency matrix of $A$. The number of
paths of length $n \geq 0$ from state $p$ to state $q$ is given by $(M(A)^n)_{p,q}$.

**Proof.**— proceed by induction on $n$. ■

The number of words of length $n$ belonging to $L = L(A)$ (that is, the growth function of the language; see definition 1.33, Volume 1) is given by

$$g_L(n) = \sum_{q \in F} (M(A)^n)_{q_0,q}.$$  

We could also count the number of accepted words of length $n$ starting from some state $r$; that is, the growth function $g_r$ of the language $\{ w \in B^* \mid \delta(r,w) \in F \}$. We have a similar formula:

$$g_r(n) = \sum_{q \in F} (M(A)^n)_{r,q}. \quad [1.1]$$

Let $P \in \mathbb{Z}[X]$ be the characteristic polynomial of $M(A)$. It is a monic polynomial of degree $d = \text{Card}(Q)$. From the Cayley–Hamilton theorem, we know that $P(M(A)) = 0$. Therefore, there exists integers $c_{d-1}, \ldots, c_0$ such that

$$M(A)^d = \sum_{i=0}^{d-1} c_i M(A)^i.$$  

Multiplying both sides by $M(A)^n$ shows that $(M(A)^j)_{j \geq 0}$ (and thus every component of this matrix) satisfies a linear recurrence equation. From [1.1], we deduce that the sequences $(g_r(n))_{n \geq 0}$ satisfy, for all $r \in Q$, this linear recurrence equation. But of course, the initial conditions depend on the state $r$. For the same reasons, $(g_L(n))_{n \geq 0}$ satisfies the same linear recurrence equation.

**Remark 1.14.**— If the DFA has $d$ states, we obtain a recurrence relation of order $d$. We will see that several
automata can recognize the same language. It is, therefore, meaningful to consider a DFA with a minimal number of states. We will develop a theory of the minimal automaton in section 1.5. There exists, up to isomorphism, a unique DFA with a minimal number of states that recognize a given regular language.

Without this theory about the minimal automaton, we can still remove useless states to obtain a recurrence relation of smaller order. States that cannot be reached from the initial state, or states from which no final state can be reached are useless. So removing those states, we have a DFA (with a partial transition function) having fewer states. The resulting automaton is usually said to be trim.

Let us recall formally the definition of a trim DFA.

**Definition 1.15.**——Let $A = (Q, q_0, B, \delta, F)$ be a DFA. A state $q$ is accessible if there exists a word $w$ such that $\delta(q_0, w) = q$. A state $q$ is coaccessible if there exists a word $w$ such that $\delta(q, w) \in F$. The subautomaton obtained by considering only states that are accessible and coaccessible is said to be trim. A sink state (also called dead state) is a non-accepting state $q$ such that, for all letters $a$, $\delta(q, a) = q$. So, whenever a DFA enters a sink state, there is no way to leave it and reach an accepting state.

It is probably worth recalling a standard result about linear recurrence sequences; see, for instance, [GRA 89].

**Theorem 1.16.**——Let $d \geq 1$ and $r_0, \ldots, r_{d-1} \in \mathbb{R}$. Let $(U_n)_{n \geq 0}$ be a sequence satisfying, for all $n \geq 0$,

$$U_{n+d} = r_{d-1}U_{n+d-1} + \cdots + r_0U_n.$$ 

If $\alpha_1, \ldots, \alpha_t \in \mathbb{C}$ are the roots of the polynomial $X^d - r_{d-1}X^{d-1} - \cdots - r_0$ with respective multiplicities $m_1, \ldots, m_t$, then there exists polynomials $P_1, \ldots, P_t$ of degree,
respectively, less than $m_1, \ldots, m_t$ and depending only on the initial conditions $U_0, \ldots, U_{d-1}$ such that

$$\forall n \geq 0, \quad U_n = P_1(n) \alpha_1^n + \cdots + P_t(n) \alpha_t^n.$$  \[1.2\]

### 1.3. Multidimensional alphabet

In the first section of this chapter, we have thus far considered automata over alphabets such as $\{0, 1, 2\}$ or $\{a, b\}$. But, there is no objection to considering other finite sets as alphabets such as

$$\{0, 1\}^2 = \left\{ \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right\}.$$

With this alphabet, the corresponding recognized language is a subset of $(\{0, 1\}^2)^*$. We make no distinction between pairs written horizontally or vertically, but it seems more natural to write them as a column vector because that is what the machine should read at once. Of course, there is no objection to taking $n$-tuples instead of pairs and also, we can have alphabets with more than two symbols. Note that the Cartesian product of alphabets was already considered to define the direct product of words (see definition 1.70 on page 71 of Volume 1).

**Example 1.17.**—Consider the DFA depicted in Figure 1.6 (the missing transitions lead to a non-final sink). This DFA accepts pairs of words $(u, v)$ over $\{0, 1\}$ of the same length where $v$ contains exactly one symbol 1 and this symbol occurs in the same position as the last 1 occurring in $u$ (and $u$ contains at least one symbol 1). Since we interchangeably use horizontal or vertical representations, let us write the two components one next to the other to fit it on a line of text. For instance, $(1, 1), (101010, 000010)$ or $(11100, 00100)$ belong to the recognized language. If words over $\{0, 1\}$ are interpreted as
base-2 expansions, then this DFA recognizes⁵ pairs of positive integers \((x, y)\) (written in base 2 and possibly allowing leading zeroes) such that \(y\) is the largest power of 2 dividing \(x\).

\[
\begin{align*}
(0,0), & \quad (1,0) \quad \rightarrow \\
(1,1) & \quad \rightarrow (0,0) \quad \rightarrow (0,0)
\end{align*}
\]

**Figure 1.6. A DFA accepting a language over \(\{0,1\}^2\)**

**Example 1.18.**— The DFA depicted in Figure 1.7 recognizes those pairs \((x, y)\) of positive integers (written in base 2 and possibly allowing leading zeroes) such that \(x < y\). Again, we have not represented the sink states and the corresponding transitions.

\[
\begin{align*}
(0,0), & \quad (1,1) \quad \rightarrow \\
(0,0), & \quad (0,1), \quad (1,0) \quad \rightarrow (1,1)
\end{align*}
\]

**Figure 1.7. A DFA accepting a language over \(\{0,1\}^2\)**

**Example 1.19.**— As a third example, we consider the alphabet \(\{0,1\}^3\). The DFA depicted in Figure 1.8 is intended to mimic base-2 addition. It recognizes 3-tuples of words \((u, v, w)\) such that \(u, v\) and \(w\) have the same length and \(\text{val}_2(u) + \text{val}_2(v) = \text{val}_2(w)\). The idea is that we read least significant digits first (this is not a problem because regular languages are closed under reversal). State 1 corresponds to the situation where there is a carrying to be taken into

---

⁵ We can say that the sequence \((V_2(n))_{n \geq 0}\) is 2-synchronized. This terminology was introduced in section 2.8, Volume 1.
account. For instance, \( \text{val}_2(01110) = 14 \), \( \text{val}_2(00111) = 7 \) and \( \text{val}_2(10101) = 21 \). Reading the corresponding 3-tuples starting with the least significant digit gives the sequence

\[
\begin{align*}
(0) & \quad (1) & \quad (1) & \quad (1) & \quad (0) \\
0 & \rightarrow 0 & \rightarrow 1 & \rightarrow 1 & \rightarrow 1 & \rightarrow 0.
\end{align*}
\]

\[
\begin{align*}
(0) & , (1) , (0) \\
0 & , 0 , 1
\end{align*}
\]

\[
\begin{align*}
0 & \rightarrow 0 & \rightarrow 1 & \rightarrow 0 \\
1 & \rightarrow 1 & \rightarrow 0 & \rightarrow 1
\end{align*}
\]

\[
\begin{align*}
0 & , (0) , (1) \\
0 & , 0 , 1
\end{align*}
\]

\[
\begin{align*}
1 & \rightarrow 1 & \rightarrow 0 \\
0 & \rightarrow 0 & \rightarrow 1
\end{align*}
\]

Figure 1.8. A DFA accepting a language over \( \{0, 1\}^3 \)

Let \( A, B \) be finite alphabets. With these examples, we see that given an automaton over an alphabet \( A \times B \), it recognizes a subset of \( A^* \times B^* \), i.e. a relation over \( A^* \times B^* \). In examples 1.17 and 1.19, we had moreover graphs of functions. These examples are in fact 2-synchronized functions (as briefly introduced in section 2.8, Volume 1). There is a vast amount of results about functions or relations that can be realized by finite automata or transducers, but this is not the place to take such a direction. See, for instance, [SAK 09].

1.4. Two pumping lemmas

The so-called pumping lemma is a classical result that is typically used to prove that some language is not regular. Its proof relies on the pigeonhole principle: any sufficiently long path goes through the same state twice.
LEMMA 1.20 (Pumping lemma).– Let \( L \) be an infinite regular language. There exists an integer \( k \) such that, for all words \( w \in L \) of length at least \( k \), there exists words \( x, y, z \) where \( y \) is non-empty and such that

\[
\begin{align*}
- w &= xyz; \\
- |xy| &\leq k; \\
- xy^nz &\in L \text{ for all } n \geq 0.
\end{align*}
\]

This last condition means that the language \( xy^*z \) is a sublanguage of \( L \).

PROOF. – Since \( L \) is regular, there exists a DFA \( A \) with \( k \) states accepting \( L \). Let \( w = w_1 \cdots w_\ell \) be a word of length \( \ell \geq k \). Such a word exists because \( L \) is infinite, and thus \( L \) contains arbitrarily long words. By the pigeonhole principle, the state transition sequence of length \( \ell + 1 \) contains at least twice the same state (and at least one such repetition must occur in the initial segment of length \( k + 1 \)): there exists \( i, j \) such that \( 0 \leq i < j \leq k \) such that

\[
\delta(q_0, w_1 \cdots w_i) = \delta(q_0, w_1 \cdots w_j).
\]

It is naturally understood that if \( i = 0 \), then \( w_1 \cdots w_0 = \varepsilon \).

To complete the proof, take \( x = w_1 \cdots w_i \), \( y = w_{i+1} \cdots w_j \) and \( z = w_{j+1} \cdots w_\ell \). We have detected a cycle labeled \( y \) starting from the state \( \delta(q_0, w_1 \cdots w_i) \).

EXAMPLE 1.21.– The archetypical example of a non-regular language is the language \( C = \{a^n b^n \mid n \geq 0\} \). Assume, to get a contradiction, that \( C \) is regular. We can, therefore, make use of the pumping lemma for some constant \( k \). Consider the word \( w = a^k b^k \) in \( C \). There must exist a factorization of \( w \) of the form \( xyz \) with \( |xy| \leq k \). Hence, the non-empty word \( y \) contains only \( a \)'s. The pumping lemma implies that \( xy^nz \) belongs to \( C \) for any \( n > 0 \) leading to a contradiction (such a word has more \( a \)'s than \( b \)'s).
Here is another result that can be useful to prove non-regularity results. If $L$ is a regular language, then we can prove that the set of lengths

$$|L| = \{ n \mid \exists w \in L : |w| = n \}$$

is a finite union of arithmetic progressions. The converse does not hold: the non-regular language $C = \{ a^n b^n \mid n \geq 0 \}$ is such that $|C| = 2N$. For an example, the language $\{ a^{n^2} \mid n \geq 0 \}$ is not a regular language.

We present and give a proof of a stronger version of the pumping lemma\(^6\). In contrast with lemma 1.20, we have here a necessary and sufficient condition for regularity. Note also that the following result is about any word in $B^*$ and not only words in $L$.

**Theorem 1.22 (J. Jaffe).**– An infinite language $L \subseteq B^*$ is regular if and only if there exists a constant $k > 0$ such that, for all words $w \in B^*$, if $|w| \geq k$, then there exists $x, y, z \in B^*$ such that $w = xyz$ with $y$ non-empty and, for all $i \geq 0$ and all $v \in B^*$,

$$wv \in L \iff wy^izv \in L. \quad [1.3]$$

**Proof.**– The fact that the condition is necessary follows the same lines as the proof of the pumping lemma. The interesting part is to prove the sufficient condition for regularity. Assume that $L$ is a language satisfying the assumptions of the statement. We consider the following DFA where

- the set of states is $Q = \{ q_w \mid w \in B^{\leq k-1} \}$;
- the initial state is $q_\epsilon$; and
- a state $q_w$ is final whenever $w \in L$.

\(^6\) There is another nice version of this result in [STA 82].
The transition function is defined as follows. If $|w| < k - 1$, then

$$\delta(q_w, b) = q_{wb}, \quad \forall b \in B.$$ 

So the readers may observe that this DFA is similar to a trie (as introduced in example 1.36, Volume 1) built for the (finite) prefix closed language $B^\leq k - 1$. Let $b \in B$. If $|w| = k - 1$, then $wb$ is a word of length $k$, and we can make use of the assumption. There is at least one factorization $wb = xyz$ verifying the properties given in the statement of the result. If such a factorization is not unique, we can choose the one for which $xy$ is the shortest and then the one for which $y$ is the shortest. In that case, since $y$ is non-empty, $|xz| = k - |y| < k$, and the following definition is, therefore, legitimate:

$$\delta(q_w, b) = q_{xz}.$$ 

To complete the proof, we show that this DFA accepts exactly the language $L$. Proceed by induction on the length of the input word. If the input word $w$ has length less than $k$, then by definition of the DFA, this word is trivially accepted: $\delta(q_\varepsilon, w) = q_w \in F$. Let $n \geq k$. Now assume that the DFA accepts exactly the words in $L$ of length less than $n$. Let us prove that it also accepts the words in $L$ of length $n$. Let $w$ be a word of length $n$ having $p$ as prefix of length $k$: $w = pv$ for some suffix $v$. By definition of the DFA, there exists some $x, y, z \in B^*$ such that

$$\delta(q_\varepsilon, p) = q_{xz} \text{ and } p = xyz \text{ with } y \neq \varepsilon.$$ 

In particular, applying [1.3] for $i = 0$ means that $w$ belongs to $L$ if and only if $xzv$ belongs to $L$. In the DFA, we have

$$\delta(q_\varepsilon, w) = \delta(q_\varepsilon, pv) = \delta(q_{xz}, v).$$ 

This means that $w$ is accepted by the DFA if and only if $xzv$ is accepted. But observe that $|xzv| < n$ (because $y$ is non-empty). By the induction hypothesis, $xzv$ is accepted by the DFA if and only if it belongs to $L$. We conclude that $w$ is in $L$ if and only if it is accepted by the DFA. $\blacksquare$
1.5. The minimal automaton

Infinitely many DFAs accept a given infinite regular language. Among all these automata, we seek an automaton having a minimal number of states. As we will see, up to isomorphism, this automaton is unique. Moreover, there exists important relations between a DFA and the minimal automaton accepting the same language. These relations are expressed by morphisms of automata.

The notion of minimal automaton can be defined for any language (even for a non-regular language but in that case, the automaton that we will define will be infinite).

Let \( L \subseteq B^* \) be a language. Let \( w \) be a word. We let \( w^{-1}L \) denote the set

\[
\{ u \in B^* \mid uw \in L \}.
\]

Such a set is called a derivative\(^7\) or a quotient. We can, therefore, introduce an equivalence relation over \( B^* \). Two words \( u, v \in B^* \) are equivalent with respect to \( \sim_L \) if and only if \( u^{-1}L = v^{-1}L \). It is easy to see that this relation is indeed a right congruence (with respect to concatenation of words): for all \( b \in B \),

\[
u \sim_L v \Rightarrow ub \sim_L vb.
\]

The relation \( \sim_L \) is often referred to as the Nerode congruence.

Observe that, for all words \( u, v \), we have

\[
(wv)^{-1}L = v^{-1}(u^{-1}L).
\]

\(^7\) There is no possible confusion with the derived sequence of a uniformly recurrent word introduced in section 3.2, Volume 1. Here, a derivative is a language of finite words.
EXAMPLE 1.23.– Let \( L \) be the language made up of words \( w \) over \( \{a, b\} \) such \( |w|_a \equiv 0 \ (\text{mod} \ 3) \). For instance, we have ababa \( \sim_L \) aaa, b \( \not\sim_L \) ab, aba \( \not\sim_L \) bab and a \( \sim_L \) ababa. Indeed, it is easy to see that the two words \( u, v \) are such that \( u \sim_L v \) if and only if \( |u|_a \equiv |v|_a \ (\text{mod} \ 3) \).

Let \( A = (Q, q_0, B, \delta, F) \) be a deterministic\(^8\) automaton. Let \( q \in Q \) and \( R \) be a subset of states. Similarly to the derivative of a language, we define \( q^{-1}R \) as the set

\[
\{w \in B^* \mid \delta(q, w) \in R\}.
\]

In particular, the language accepted by \( A \) is \( L(A) = q_0^{-1}F \).

Let \( q \) be a state and \( w \) be a word such that \( \delta(q_0, w) = q \). We have

\[
q^{-1}F = w^{-1}L.
\]  \[\text{[1.5]}\]

**Definition 1.24.–** A deterministic automaton \( A = (Q, q_0, B, \delta, F) \) is reduced if, for all states \( p, q \in Q \),

\[
p^{-1}F = q^{-1}F \Rightarrow p = q.
\]

This means that the languages accepted from any two distinct states differ.

**Definition 1.25 (Minimal automaton of \( L \)).–** Let \( L \subseteq B^* \) be a language. We define the deterministic automaton \( A_L = (Q_L, q_{0,L}, B, \delta_L, F_L) \) where

- the set of states is \( Q_L = \{w^{-1}L \mid w \in B^*\} \);
- the initial state is \( q_{0,L} = \epsilon^{-1}L = L \);
- the set of final states is \( F_L = \{w^{-1}L \mid w \in L\} \);

\(^8\) We do not say that this automaton is finite. The theory could be carried on for automata with infinitely many states.
the transition function $\delta_L : Q_L \times B \rightarrow Q_L$ is defined, for all $b \in B$ and $q \in Q_L$, by

$$\delta(q, b) = b^{-1}q.$$ 

The transition function is well-defined. Assume that two words $u$ and $v$ are such that $u \sim_L v$, i.e. $u^{-1}L = v^{-1}L$. Then these two words correspond to the same state of $A_L$. Notice that, from [1.4], we get

$$(ub)^{-1}L = b^{-1}(u^{-1}L) = b^{-1}(v^{-1}L) = (vb)^{-1}L.$$ 

Otherwise stated, the fact that $\sim_L$ is a right congruence implies that the value of $\delta(q, w)$ does not depend on the word $u$ such that $q = u^{-1}L$.

The function $\delta$ can be extended to $Q_L \times B^*$: for a word $w$, we get $\delta(q, w) = w^{-1}q$.

Also, $\delta_L(q_0, L, w) = w^{-1}L$ and $w^{-1}L$ is a final state if and only if $w$ belongs to $L$. This means that the minimal automaton of $L$ accepts the language $L$ (this is the property that we could have expected).

From this definition, $A_L$ can be seen to be accessible and reduced. (Left as an exercise. See, for instance, [SAK 09].)

The fundamental result about minimal automata is the following one.

**Theorem 1.26.**—Let $L$ be a language. Let $A_L = (Q_L, q_0, L, B, \delta_L, F_L)$ be the minimal automaton of $L$. Let $B = (Q, q_0, B, \delta, F)$ be a deterministic and accessible automaton accepting the same language $L$. There exists a map $\Phi : Q \rightarrow Q_L$ such that

- $\Phi$ is onto;
- $\Phi(q_0) = q_0, L$;
– for all $b \in B$ and $q \in Q$, $\Phi(\delta(q, b)) = \delta_L(\Phi(q), b)$;
– $\Phi(F) = F_L$.

We say that such a map $\Phi$ is a morphism of automata: transitions occurring in $A_L$ are compatible with those occurring in $B$. The fact that $\Phi$ is onto implies that if $B$ is a finite automaton (hence $L$ is a regular language), then the minimal automaton of $L$ has a number of states that is less or equal to the number of states of $B$. In particular, $L$ is a regular language if and only if its minimal automaton is finite. This is also equivalent (from the definition of $A_L$) to say that the Nerode congruence $\sim_L$ is of finite index: the quotient $A^*/\sim_L$ is finite.

**Example 1.27.** – In the upper part of Figure 1.9, we have depicted an accessible DFA accepting $a^*b^*$. In the lower part of the figure is represented the minimal automaton of the same language. The map $\Phi$ is such that $\Phi(i) = \Phi(i + 3) = i + 6$ for $i = 0, 1, 2$.

Note that if a map $\Phi : Q \to Q_L$ satisfies the properties given in the previous theorem, then $\Phi(q) = q^{-1}F$ for all $q \in Q$. Indeed, the following holds. Let $q \in Q$. Since $B$ is accessible, there exists $w \in B^*$ such that $\delta(q_0, w) = q$. Then

\[
\Phi(q) = \Phi(\delta(q_0, w)) = \delta_L(\Phi(q_0), w) = \delta_L(q_{0,L}, w)
\]

\[
= w^{-1}q_{0,L} = w^{-1}L = q^{-1}F.
\]

For the last equality, we have used \[1.5\].

**Corollary 1.28.** – Let $L$ be a language. Let $A_L = (Q_L, q_{0,L}, B, \delta_L, F_L)$ be the minimal automaton of $L$. Let $B = (Q, q_0, B, \delta, F)$ be a deterministic and accessible automaton accepting the same language $L$. The DFA $B$ is reduced if and only if the map $\Phi : Q \to Q_L$ from theorem 1.26 is one-to-one.
PROOF.— The map $\Phi$ given in theorem 1.26 is one-to-one if and only if, for all $p, q \in Q$, $\Phi(p) = \Phi(q)$ implies $p = q$. But, we have observed that $\Phi(p) = p^{-1}F$ and $\Phi(q) = q^{-1}F$. So, we are exactly back to the definition of a reduced automaton. ■

The previous corollary means that, up to isomorphism, i.e. up to a bijective morphism of automata, the minimal automata of a language is unique.

**Corollary 1.29.**— Let $L$ be a regular language. Let $A_L = (Q_L, q_{0,L}, B, \delta_L, F_L)$ be the minimal automaton of $L$. Let $B = (Q, q_0, B, \delta, F)$ be a accessible DFA accepting the same language $L$. The minimal polynomial of the adjacency matrix of $A_L$ divides the one for $B$.

**Proof.**— Let $A$ (respectively, $B$) be the adjacency matrix of $A_L$ (respectively, $B$). Consider the map $\Phi : Q \to Q_L$ given by theorem 1.26. Let $p, q$ be states in $A_L$ and $w$ be a word such
that $\delta_L(p, w) = q$. The key observation\(^9\) is the following. There exists $p' \in Q$ such that $\Phi(p') = p$ and

$$(A^n)_{p,q} = \sum_{q' \in \Phi^{-1}(q)} (B^n)_{p',q'}.$$ 

Therefore, if $P(B) = 0$ for some polynomial $P$, then $P(A) = 0$. In particular, the minimal polynomial $M_B$ of $B$ is such that $M_B(B) = 0$. Therefore, $M_B(A) = 0$. This implies that $M_B$ is a multiple of the minimal polynomial of $A$. (This is a consequence of [1.1] on page 7 of Volume 1.) \(\blacksquare\)

To conclude with this section, we show that the inverse image of a regular language by a morphism is again a regular language. Note that this closure property of the set of regular languages was not listed by theorems 1.8 and 1.9.

**Proposition 1.30.** \(\text{Let } f : A^* \to B^* \text{ be a morphism. Let } L \subseteq B^* \text{ be a regular language. The language } f^{-1}(L) \subseteq A^* \text{ is regular.}\)

**Proof.** To prove that $f^{-1}(L) = \{u \in A^* \mid f(u) \in L\}$ is regular, we will show that its minimal automaton has a finite number of states.

Let $w \in A^*$. Consider languages\(^10\) of the form $w^{-1}f^{-1}(L)$. We will show that the set

$$\{w^{-1}f^{-1}(L) \mid w \in A^*\}$$

is a finite set (proving that the corresponding minimal automaton is finite). We have

$$w^{-1}f^{-1}(L) = \{u \in A^* \mid wu \in f^{-1}(L)\} = \{u \in A^* \mid f(wu) \in L\}$$

\(^9\) Think twice about it, we have to count each path of length $n$ exactly once.

\(^{10}\) I want to stress on the fact that the language $w^{-1}f^{-1}(L)$ may be an infinite set of words. But this is not a problem, what we are looking for is the number of distinct sets of the type $w^{-1}f^{-1}(L)$ that we can get.
\[ \{ u \in A^* \mid f(w)f(u) \in L \} = \{ u \in A^* \mid f(u) \in f(w)^{-1}L \} = f^{-1}(f(w)^{-1}L). \]

But notice that \( L \) is a regular language, hence \( f(w)^{-1}L \) takes a finite number of values. So, \( f^{-1}(f(w)^{-1}L) \) takes only a finite number of values. ■

What is usually understood as state complexity is to relate operations on languages to the size of the corresponding minimal automata. For instance, what can be said about the number of states of \( A_{L\cup M} \) in terms of the number of states of \( A_L \) and \( A_M \)? What kinds of lower bounds or upper bounds can be achieved? Most of the time, worst-case analysis is carried out. (But of course, an average-case analysis would be meaningful for practical reasons). We can also be interested in a family of languages depending on some parameters, let us say, some sequence of regular languages \( (L_n)_{n \geq 0} \), and estimate the size of \( A_{L_n} \) in terms of \( n \). For a short survey, see [YU 05].

1.6. Some operations preserving regularity

In this section, we consider a few operations that given any regular language, will extract a sublanguage that is again regular. For these operations, we will assume that the languages are genealogically ordered (see definition 1.11 on page 14 of Volume 1). About regularity preserving operations, the paper [BER 06a] is worth reading.

Let \( L \) be a language over a totally ordered alphabet \((B, <)\). Since the language is genealogically ordered (induced by the ordering of \( B \)), we define

\[
\max_{\text{lg}}(L) = \{ u \in L \mid \forall v \in L, |u| = |v| \Rightarrow u \geq v \},
\]

\[
\min_{\text{lg}}(L) = \{ u \in L \mid \forall v \in L, |u| = |v| \Rightarrow u \leq v \}.
\]
Otherwise stated, $\text{maxlg}(L)$ (respectively, $\text{minlg}(L)$) contains the largest (respectively, smallest) word of each length with respect to the genealogical ordering of $L$.

**Theorem 1.31.**– Let $L$ be a regular language over a totally ordered alphabet $(B, <)$. The languages $\text{maxlg}(L)$ and $\text{minlg}(L)$ are regular.

**Proof.**– It is convenient to make a proof exploiting non-determinism. Let $A$ be a DFA accepting $L$. The idea to accept a word $w$ in $\text{maxlg}(L)$ is first to test if $w$ belongs to $L$ making use of $A$. Next, we have to guess (we will make this statement precise) other words of the same length. The guessed word $u$ (or if you prefer, a word picked non-deterministically) is tested to be in $L$ due to $A$. If we cannot guess any such word $u \in L$ larger than $w$, then $w$ belongs to $\text{maxlg}(L)$. Testing if $u < w$ can be realized with an automaton such as the one depicted in example 1.18.

It is easier to first build an NFA accepting $L \setminus \text{maxlg}(L)$, and then take the complement in $B^{*}$ and intersection with $L$. First from $A$, we construct a new DFA $A_1$ accepting the language

$$L_1 = \{(u, v) \mid |u| = |v| \text{ and } u \in L\}.$$  

We simply replace each label $a$ of $A$ with $\text{Card}(B)$ labels $\left(\begin{array}{c}a \\ b \end{array}\right)$, one for each $b \in B$. If we apply this procedure to the automaton recognizing $a^{*}b^{*}$ given in Figure 1.1, we get the DFA depicted in Figure 1.10.

We do the same with

$$L_2 = \{(u, v) \mid |u| = |v| \text{ and } v \in L\}.$$  

As in example 1.18, there exists a DFA accepting the language

$$L_{<} = \{(u, v) \in (B \times B)^{*} \mid u < v\}.$$  

11 A proof of this result appeared in [SHA 94].
Since regular languages are closed under intersection, the language $L_1 \cap L_2 \cap L_<$ is regular and accepted by some DFA $B$ over the alphabet $B \times B$. Note that this language is exactly
$$\{(u, v) \in L \times L \mid |u| = |v| \text{ and } u < v\}.$$ 

If we replace every label $\left(\begin{array}{c} a \\ b \end{array}\right)$ in $B$ with $a$, we get an NFA accepting $L \setminus \max\lg(L)$. Indeed, if there exists an accepting path for a word $u$ in this NFA, this means that in the original DFA $B$ there is at least an accepting path whose first component has label $u$. Hence, there exists a word $v$ such that $(u, v) \in L_1 \cap L_2 \cap L_<$, i.e. $u$ is in $L$ but not a maximal word in $L$.

**Remark 1.32.**– The last argument developed in the previous proof: replacing $\left(\begin{array}{c} a \\ b \end{array}\right)$ with $a$ is particularly important for Chapter 3, where similar constructions will be applied several times. Indeed, for the first time, we are somehow explaining how to deal with an existential quantifier.

Let $0 \leq r < m$. The **periodic decimation** of $L$ is the sublanguage obtained as follows. First, enumerate the words of $L$ in terms of increasing genealogical ordering: $L = \{w_0 < w_1 < w_2 < \cdots\}$. Then, we define
$$\text{per}_{r,m}(L) = \{w_{nm+r} \mid n \geq 0\}.$$
Indeed, this notion is similar to the periodic decimation of an infinite word as considered in proposition 2.36, Volume 1.

**THEOREM 1.33.**— Let $L$ be a regular language over a totally ordered alphabet $(B, <)$. Let $0 \leq r < m$. The periodic decimation $\text{per}_{r,m}(L)$ is regular.

**PROOF.**— The first proof was originally given in [LEC 01]. Two different proofs of this result can be found with details and examples in [BER 10, pp.123–126]. See also [KRI 09] where another proof was given later. ■

**REMARK 1.34.**— Interestingly, for a context-free language $L$ (this classical notion in formal language theory is not discussed in this book), it is known that $\max \text{lg}(L)$ (respectively, $\min \text{lg}(L)$) is also context-free [BER 97]. Nevertheless, there exists a context-free language $C$ such that $\text{per}_{0,2}(C)$ is not context-free [KRI 09].

### 1.7. Links with automatic sequences and recognizable sets

Let $b \geq 2$ be an integer. Recall that the base-$b$ expansion of $n \geq 0$ (with no leading zeroes) is denoted by $\text{rep}_b(n)$. Given a set of integers $X \subseteq \mathbb{N}$, we are interested in the language $\text{rep}_b(X) = \{\text{rep}_b(n) | n \in X\}$.

**DEFINITION 1.35.**— Let $b \geq 2$ be an integer. A set $X \subseteq \mathbb{N}$ is $b$-recognizable if $\text{rep}_b(X)$ is a regular language over $[0, b-1]$.

**EXERCISE 1.7.1.**— Let $b \geq 2$. Show that $X \subseteq \mathbb{N}$ is $b$-recognizable if and only if $0^* \text{rep}_b(X)$ is a regular language.

Since DFAs are a very simple model of computation, it is meaningful to study these $b$-recognizable sets. In a sense, they are “simple” from an algorithmic point of view: when written accordingly in base $b$, they are recognized by the simplest model of computation. Moreover, a DFA uses linear
time in terms of the size of the input to decide whether this input belongs to the set.

REMARK 1.36.– The definition of a $b$-recognizable set can be extended to subsets of $\mathbb{N}^n$ by considering DFA over the alphabet $[0, b - 1]^n$ reading $n$-tuples of words (of the same length, the shortest ones being padded with leading zeroes) as in section 1.3. Details will be provided in definition 2.45. Such an extension will be extensively used in the last chapter of this book.

At this point of the book, the following results should be seen as easy exercises. (Hint: see proposition 2.25, Volume 1).

PROPOSITION 1.37.– Let $b \geq 2$. A set $X \subseteq \mathbb{N}$ is $b$-recognizable if and only if its characteristic sequence is $b$-automatic.

DEFINITION 1.38.– Let $x \in A^\mathbb{N}$. The fibers of $x = x_0x_1x_2\cdots$ are sets of integers of the form

$$\text{fiber}_x(a) = \{ n \in \mathbb{N} \mid x_n = a \}, \ a \in A.$$

PROPOSITION 1.39.– Let $a \in A$. If $x \in A^\mathbb{N}$ is $b$-automatic, then the fiber $\text{fiber}_x(a)$ is $b$-recognizable. Conversely if, for all $a \in A$, $\text{fiber}_x(a)$ is $b$-recognizable, then $x \in A^\mathbb{N}$ is $b$-automatic.

It is interesting to see that every divisibility criterion can be expressed in every base.

PROPOSITION 1.40.– Let $b \geq 2$. Let $m, r \in \mathbb{N}$. The arithmetic progression $m \mathbb{N} + r$ is $b$-recognizable. Hence, every ultimately periodic set of integers is also $b$-recognizable.

From proposition 1.37, we deduce that any ultimately periodic word is $k$-automatic for all $k \geq 2$.

PROOF.– Since regular languages are closed under finite modifications, we may assume that $0 \leq r < m$ and $m \geq 2$. We consider a DFA reading base-$b$ expansions most significant
digit first and having \([0, m - 1]\) as set of states. Now, we define the transition function as

\[
\delta(r, c) = b \cdot r + c \mod m, \quad \forall c \in [0, b - 1], r \in [0, m - 1].
\]

This corresponds simply to the fact that \(\text{val}_b(wc) = b \cdot \text{val}_b(w) + c\). The initial state is 0, and the final state is \(r\).

As an example of the construction given in the previous proof, a DFA accepting exactly the binary expansions of the integers congruent to 3 (mod 4) is given in Figure 1.11.

![Figure 1.11. A finite automaton accepting \(0^* \text{rep}_2(4N + 3)\)](image)

**Remark 1.41.**—The above construction does not guarantee that the obtained automaton is minimal. Indeed, Alexeev studied the number of states of the minimal automaton of the language \(0^* \text{rep}_b(mN)\), that is, the set of \(b\)-ary representations of the multiples of \(m \geq 1\) [ALE 04]. The greatest common divisor of two integers \(a\) and \(b\) is denoted by \(\gcd(a, b)\). Let \(N, M\) be such that \(b^N < m \leq b^{N+1}\) and \(\gcd(m, 1) < \gcd(m, b) < \cdots < \gcd(m, b^M) = \gcd(m, b^{M+1}) = \gcd(m, b^{M+2}) = \cdots\). The number of states of the minimal automaton of \(0^* \text{rep}_b(mN)\) is exactly

\[
\frac{m}{\gcd(m, b^{N+1})} + \sum_{t=0}^{\inf\{N, M-1\}} \frac{b^t}{\gcd(m, b^t)}.
\]

We have just seen that arithmetic progressions are \(b\)-recognizable for all bases \(b\). To see that a given infinite ordered set \(X = \{x_0 < x_1 < x_2 < \cdots\}\) is \(b\)-recognizable for \(n\)
base $b \geq 2$ at all, we can use results such as the contrapositive of the following theorem 1.42 where the behavior of the ratio (respectively, difference) of any two consecutive elements in $X$ is studied through the quantities

$$R_X := \limsup_{i \to \infty} \frac{x_{i+1}}{x_i} \text{ and } D_X := \limsup_{i \to \infty} (x_{i+1} - x_i).$$

Observe that if $D_X$ is bounded, then this means exactly that $X$ is syndetic (we have seen this notion in definition 3.14, Volume 1).

The following “gap theorem” can be found in [COB 72] or in the classical book [EIL 74].

**Theorem 1.42.**– Let $b \geq 2$. If $X$ is a $b$-recognizable infinite subset of $\mathbb{N}$, then either $R_X > 1$ or $D_X < +\infty$.

**Corollary 1.43.**– Let $a \in \mathbb{N}_{\geq 2}$. The set of primes and the set \{n^a | n \geq 0\} are $b$-recognizable for no integer base $b \geq 2$.

**Proof.**– Application of the above theorem for \{n^a | n \geq 0\} is obvious. Let $\mathcal{P}$ be the set of primes. Since $n! + 2, \ldots, n! + n$ are $n - 1$ consecutive composite numbers, then $D_{\mathcal{P}}$ is infinite. It is known that the $n$th prime number belongs to the interval $(n \ln n, n \ln n + n \ln \ln n)$, and therefore $R_{\mathcal{P}} = 1$. See, for instance, [BAC 96].

From this result, the special case of the characteristic sequence of the squares given in example 2.5, Volume 1, is $k$-automatic for no $k \geq 2$.

**Corollary 1.44.**– The characteristic sequence of the set of squares is $k$-automatic for no integer $k \geq 2$.

We conclude this section with another important theorem given to us by A. Cobham. Recall from definition 2.74, Volume 1, that two integers $p, q \geq 2$ are multiplicatively independent if the only integers $k, \ell$ such that $p^k = q^\ell$ are $k = \ell = 0$. 

Otherwise, \( p \) and \( q \) are \emph{multiplicatively dependent}. A key argument in the proof of the next statement is given in exercise 1.3.2, Volume 1.

**Theorem 1.45 (Cobham [COB 69]).** Let \( p, q \geq 2 \) be two multiplicatively independent integers. If a set \( X \subseteq \mathbb{N} \) is both \( p \)-recognizable and \( q \)-recognizable, then \( X \) is ultimately periodic.

Obviously the set \( P_2 = \{2^n \mid n \geq 0\} \) of powers of two is 2-recognizable because \( \text{rep}_2(P_2) = 10^* \). But since \( P_2 \) is not ultimately periodic, the above theorem of Cobham implies that \( P_2 \) cannot be 3-recognizable.

**Exercise 1.7.2.** Let \( b, n \geq 2 \). Show that a set \( X \subseteq \mathbb{N} \) is \( b \)-recognizable if and only if it is \( b^n \)-recognizable.

As another example, we know that the Thue–Morse word \( t \) is not ultimately periodic (indeed, it is cube-free, see section 3.5, Volume 1). The set \( \text{fiber}_t(1) \) is the set of integers whose base-2 expansion contains an odd number of ones. This set is syndetic (because \( t \) is a concatenation of factors 01 and 10). As a result of Cobham’s theorem, \( \text{fiber}_t(1) \) is neither 3-recognizable nor 5-recognizable.

The base dependence of recognizability shown by Cobham’s result strongly motivates the general study of recognizable sets and the introduction of non-standard or exotic numeration systems discussed in Chapter 2. Defining new systems leads to new sets of integers recognizable by automata. In the last chapter of this book, we will show in particular that \( b \)-recognizable sets can be defined using the formalism of first-order logic.

From proposition 1.37, Cobham’s theorem can be restated (in a totally equivalent way) as follows. Let \( p, q \geq 2 \) be two multiplicatively independent integers. If an infinite word is both \( p \)-automatic and \( q \)-automatic, then it is ultimately periodic. Therefore, instead of considering only
constant-length morphisms, we could try to extend this result to morphic words. First, we have to decide what will replace the integers $p$ and $q$. This is exactly the notion of (pure) $\alpha$-substitutive word given in definition 2.71, Volume 1. Therefore, the nice extension is the Cobham–Durand theorem stated in theorem 2.75, Volume 1. To express this latter result in terms of numeration systems instead of morphic words, we will introduce abstract numeration systems.

To conclude this section, we mention an interesting extension of the notion of recognizability to sets of rational numbers [ROW 12b].

**Definition 1.46.**—Let $b \geq 2$ be an integer. A subset $X$ of $\mathbb{Q}$ is $b$-recognizable if there exists a DFA $A$ over $[0, b - 1]^2$ such that for each $x \in X$, there exists $p, q \in \mathbb{N}$ such that $x = p/q$, and for $n = \max\{|\text{rep}_b(p)|, |\text{rep}_b(q)|\}$, the word

$$(0^n - |\text{rep}_b(p)| \text{rep}_b(p), 0^n - |\text{rep}_b(q)| \text{rep}_b(q))$$

is accepted by $A$. Moreover, any pair $(u, v)$ accepted by $A$ is such that the rational number $\text{val}_b(u)/\text{val}_b(v)$ belongs to $X$.

The main difficulty with this definition is that a rational number is represented by infinitely many pairs of integers. But, it is only assumed that the DFA recognizes at least one of these representations.

**1.8. Polynomial regular languages**

We already know from section 1.2, that the growth function $g_L(n)$ of a regular language satisfies a linear recurrence equation (that can be derived from the characteristic polynomial of the adjacency matrix).

From the general form of the exponential polynomial given in [1.2] theorem 1.16, $g_L(n)$ exhibits either exponential or polynomial growth. This sentence should be understood as
follows. If $L$ is a regular language over $B$, then the language $(B^\ell)^*$ of words having a length that is a multiple of $\ell$ is regular, and thus $L \cap (B^\ell)^*$ is again regular. As an example, assume that $g_L(n) = 2^n$ for all $n \geq 0$. Then, the growth function of $L \cap (B^\ell)^*$ is

$$g_{L \cap (B^\ell)^*}(n) = \begin{cases} 2^n, & \text{if } n \equiv 0 \pmod{\ell}; \\ 0, & \text{otherwise}. \end{cases}$$

We still say that this function has an exponential behavior (even if it takes the value 0 infinitely often). As a trivial example, consider the language $L = \{aa, ab, ba, bb\}^*$. Since such a phenomenon cannot be avoided, we will use the asymptotic notation $g \in \Omega_\infty(f)$ if there exists a constant $C > 0$ such that, for an infinite increasing sequence $(n_i)_{i \geq 0}$, we have

$$g(n_i) \geq Cf(n_i), \quad \forall i \geq 0.$$ 

So, we stress the fact that this definition is a bit different from the usual one: $g \in \Omega(f)$ if $g(n) \geq Cf(n)$ for all large enough $n$. Accordingly, we use $g \in \Theta'(f)$ to say that $g \in O(f)$ and $g \in \Omega_\infty(f)$.

**REMARK 1.47.**—With regular languages, we can easily merge several behaviors. Let $\ell \geq 2$. Let $L_0, \ldots, L_{\ell-1}$ be regular languages over $B$. Consider the regular language

$$L = \bigcup_{i=0}^{\ell-1} \left( L_i \cap ((B^\ell)^*B_i) \right).$$

Then, we have $g_L(n\ell + i) = g_{L_i}(n\ell + i)$ for all $i \in [0, \ell - 1]$ and $n \geq 0$.

---

12 Interestingly, for formal series in $S\langle\langle X \rangle\rangle$ we can also define the relevant operation of merge of series as follows. Let $S_0, \ldots, S_{\ell-1} \in S\langle\langle X \rangle\rangle$. The merge of these series is the series $S$ defined by $(S, X^{m\ell+i}) = (S_i, X^m)$ for $i \in [0, \ell - 1]$, $m \in \mathbb{N}$. See [BER 11].
The class of regular languages splits into two subclasses according to whether the growth function is bounded by a polynomial or is an exponential function of order $2^{\Omega_\infty(n)}$. Of course, the Jordan normal form of the adjacency matrix can provide the information we are looking for. Nevertheless, we have chosen a combinatorial approach about paths in automata to present a characterization of the regular languages with polynomial growth. Namely, the organization of the cycles in the DFA permits us to determine the asymptotic growth.

The gap between polynomial and exponential languages is rendered by the following theorem. For a proof, see [SZI 92, Theorem 6].

**Theorem 1.48.**– Every regular language has a growth function that is either $O(n^k)$, for some integer $k$, or $2^{\Omega_\infty(n)}$.

We will focus on the languages of polynomial growth, the internal structure of the automata recognizing such languages and the corresponding regular expressions (see theorem 1.56). Their characterization follows from a series of five lemmas stated and proved below. We conclude this section with an application to morphic words.

Indeed, as we will see, the growth function depends on the structure of the DFA accepting the language. The following results are derived from and follow the presentation in [SZI 92]. Note that other approaches can be followed; see, for instance, [JUN 08]. The paper [GAW 10] is worth reading: the authors provided a polynomial time algorithm to determine whether an NFA accepts a language of polynomial or exponential growth. Given an NFA accepting a language of polynomial growth, the order of polynomial growth can also be efficiently determined. In the same spirit, a complete discussion and an algorithm for computing the growth rate of a regular language are given in [SHU 08].
1.8.1. **Tiered words**

Let $A = (Q, q_0, B, \delta, F)$ be a DFA. For any word $w = w_0 \cdots w_{n-1}$ of length $n$ over $B$, the state transition sequence of $A$ on $w$, or what is sometimes called the behavior, is a word over $Q$ (we add commas for readability) denoted by

$$\text{STS}_A(w) = q_0, \delta(q_0, w_0), \ldots, \delta(q_0, w_0 \cdots w_i), \ldots, \delta(q_0, w_0 \cdots w_{n-1}).$$

The philosophy of the next definition is to detect parts of the state transition sequence where the same state is repeated, and thus cycles that are taken several times consecutively. Moreover, there is an extra condition to express that it is not allowed to come back to a previously visited cycle. To have a good intuition about the following definition, see example 1.51.

**Definition 1.49.**— Let $A = (Q, q_0, B, \delta, F)$ be a DFA$^{13}$. A word $w \in B^*$ is said to be $t$-tiered$^{14}$, $t \geq 0$, with respect to $A$ if the state transition sequence of $w$ is given by

$$\text{STS}_A(w) = \alpha \beta_1^d \gamma_1 \cdots \beta_t^d \gamma_t$$

[1.6]

where

1) $0 \leq |\alpha| \leq \text{Card}(Q)$;

and, for all $i \in [1, t]$,

2) $\beta_i = q_{i,0} \cdots q_{i,k_i} \in Q^*$ and $\gamma_i = q_{i,0} r_{i,1} \cdots r_{i,\ell_i} \in Q^*, 0 \leq k_i, \ell_i < \text{Card}(Q)$;

$^{13}$ If the automaton is non-deterministic, then the state transition sequence is not uniquely defined.

$^{14}$ We stick to the terminology found in [SZI 92]. From the dictionary, *tiered* (adjective): having or arranged in tiers, rows or layers; *tier* (noun): a particular level in a group, organization, etc.
3) $q_{i,0}$ appears only once, as the first state, in $\beta_i$ and $\gamma_i$, i.e.

$$q_{i,1}, \ldots, q_{i,k_i}, r_{i,1}, \ldots, r_{i,\ell_i} \neq q_{0,i}$$

and if $i \neq j$, then $q_{i,0}$ does not appear in $\beta_j$, $\gamma_j$ nor $\alpha$;

4) $d_i > 0$.

Note that $|\beta_i| \geq 1$ because $\beta_i$ starts at least with $q_{i,0}$ and if $\ell_i = 0$, then $r_{i,1} \cdots r_{i,\ell_i} = \varepsilon$ but still $|\gamma_i| = 1$.

**REMARK 1.50.–** Let us make a few comments. First, let us emphasize the third condition given in the previous definition. The factor $\beta_i^{d_i} \gamma_i$ in the state transition sequence reflects that a cycle of length $|\beta_i|$ is taken $d_i$ times in the automaton. But the fact that $q_{i,0}$ does not appear in $\beta_j$, $\gamma_j$ nor $\alpha$ for $j \neq i$ implies that the cycle defined by $\beta_i$ does not appear anywhere else in the state transition sequence. The state $q_{i,0}$ has never been seen before and will never be seen after in the state transition sequence. Note that if $q_{i,0}$ appeared in some $\beta_j$ or $\gamma_j$ occurring later on, $j > i$, then we would have two cycles starting in $q_{i,0}$.

Obviously, two different words can have the same state transition sequence. Indeed, it could be the case whenever there exists two distinct letters $a, b$ such that $\delta(q, a) = \delta(q, b)$ for some state $q$.

Nevertheless, observe that if $\text{STS}_A(u) \neq \text{STS}_A(v)$, since the automaton $A$ is deterministic, then $u \neq v$. We will often make use of this observation.

Observe also that $|\text{STS}_A(u)| = |u| + 1$.

**EXAMPLE 1.51.–** Consider the trim automaton (see definition 1.15) depicted in Figure 1.12. We have

$$\text{STS}_A(\text{bbbbbabba}) = 0131312222 = \alpha \beta_1^2 \gamma_1 \beta_2^3 \gamma_2$$
with $\alpha = 0$, $\beta_1 = 13$, $\gamma_1 = 1$, $\beta_2 = 2$, $\gamma_2 = 2$. This means that after reading the first symbol, we enter a cycle of length $|\beta_1| = 2$ starting in state 1. We follow this cycle twice (the exponent of $\beta_1$). Then, we follow a path of length $|\gamma_1| = 1$ to enter the second cycle of length $|\beta_2| = 1$. We follow this cycle three times. Then, we have the last state given by $\gamma_2$. Otherwise stated, the word bbbbbabba is 2-tiered. Note that 1 (respectively, 2) occurs as the first state in $\beta_1$ and $\gamma_1$ (respectively, $\beta_2$ and $\gamma_2$) and does not appear anywhere else. Note that the word bbgbbabba gives exactly the same state transition sequence.

Figure 1.12. A (trim) DFA

As another example (where $\alpha = \varepsilon$), take the word aaabbbacbac. We get the state transition sequence

$$00001321321 = \beta_1^3\gamma_1\beta_2^2\gamma_2$$

with $\beta_1 = 0$, $\gamma_1 = 0$, $\beta_2 = 132$, $\gamma_2 = 1$. The word bcbbb is also 2-tiered: $\text{STS}_A(bcbbb) = (01)0(13)12$. In particular, this factorization (and the detection of cycles) shows that the language $(bc)^*b(bb)^*a$ is a sublanguage accepted by $A$. Note that the growth function of this sublanguage satisfies $g(2n) = n$ and $g(2n + 1) = 0$, thus $g \in \Omega_{\infty}(n)$. The word bbbbbbacaca is also 2-tiered. But we have to be careful in the choice of the factorization:

$$\text{STS}_A(bbbbbbacaca) = 01\underbrace{31}_{\beta_1}\underbrace{31}_{\gamma_1}\underbrace{2121}_{\beta_2^2}\underbrace{2}_{\gamma_2}.$$
This factorization shows that \( bb(bb)^*ba(ca)^* \) is a sublanguage accepted by the automaton. Nevertheless, we could also consider the factorization \( \alpha'\beta_1^2\gamma_1\beta_2^2\gamma_2 = 0(13)^21(21)^22 \), but here the state 1 is the first state in \( \beta_1' \) and \( \gamma_1' \), but it also appears in \( \beta_2' \) that is not allowed (third condition in definition 1.49). Actually, if we face such a factorization where the state 1 reappears after \( \beta_1 \) and \( \gamma_1 \), then this means that we have two cycles attached on state 1. If this is the case, this implies an exponential growth. In this example, looking at the state transition sequence, we see that there are two cycles with respective labels \( bb \) and \( ac \) starting from state 1. Therefore, the automaton accepts the words \( bb^{m}(ac)^{n}a \) for all \( m,n \). There are \( 2^{m+n} \) words of this form, and their length is \( 2(m + n + 1) \), i.e. \( g_L(\ell) \geq 2^{\ell/2-1} \) for all even \( \ell \geq 2 \). This observation turns out to be general.

We would like to characterize regular languages with a polynomial growth, we do hope that such an example makes clear the conditions defining a tiered word. Let us summarize roughly what we have observed. As soon as we detect two cycles with distinct labels attached to state 1, then we obtain a sublanguage of exponential growth. However, if we detect a path with several consecutive cycles and we never return to a state visited at an earlier stage, then we obtain a sublanguage of polynomial growth.

1.8.2. Characterization of regular languages of polynomial growth

**Lemma 1.52.**– [SZI 92, Lemma 1] Let \( A = (Q, q_0, B, \delta, F) \) be a DFA. If there exists a word \( w \in L(A) \), which is \( (t + 1) \)-tiered, for some \( t \geq 0 \), with respect to \( A \) then the growth function of \( L(A) \) is \( \Omega_\infty(n^t) \).

**Proof.**– Assume that \( w \in L \) is \( (t + 1) \)-tiered with respect to \( A \). Therefore, \( w \) can be factored with respect to [1.6]:

\[
w = x y_1^{d_1} z_1 \cdots y_{t+1}^{d_{t+1}} z_{t+1}, \text{ with } y_1, \ldots, y_{t+1} \neq \varepsilon.
\]
In particular, this means that we have detected cycles in $A$: for all $j \in [0, t]$, we have
$$\delta(q_0, x y_1^{d_1} z_1 \cdots y_{j+1}) = \delta(q_0, x y_1^{d_1} z_1 \cdots y_{j+1}^n), \forall n > 0.$$ Let $C = |y_1| \cdots |y_{t+1}|$ and $C_i = C/|y_i|$ for $i \in [1, t + 1]$. For all $i > 0$, we set
$$n_i = |xz_1 \cdots z_{t+1}| + i C.$$ For any $t + 1$ arbitrary non-negative integers $i_1, \ldots, i_{t+1}$ such that $i_1 + \cdots + i_{t+1} = i$, the word
$$x y_1^{i_1} z_1 \cdots y_{t+1}^{i_{t+1} C_{t+1}} z_{t+1}$$ belongs to $L$ and it is easy to check that its length is exactly $n_i$.

Let $(i_1, \ldots, i_{t+1}) \neq (j_1, \ldots, j_{t+1})$ be two $(t + 1)$-tuples of non-negative integers such that $i_1 + \cdots + i_{t+1} = i = j_1 + \cdots + j_{t+1}$. We show that the words
$$w_i = x y_1^{i_1} z_1 \cdots y_{t+1}^{i_{t+1} C_{t+1}} z_{t+1} \text{ and } w_j = x y_1^{j_1} z_1 \cdots y_{t+1}^{j_{t+1} C_{t+1}}$$ are distinct. There exists a smallest index $r$ such that $i_r \neq j_r$. Assume that $i_r < j_r$. Then, due to the third condition in the definition of a $(t + 1)$-tiered word, the state $q_{i_r, 0} = \delta(q_0, x y_1^{i_1} z_1 \cdots y_{r-1}^{i_{r-1} C_{r-1}} z_{r-1})$ appears more often in $\text{STS}_A(w_j)$ than in $\text{STS}_A(w_i)$. If the state transition sequences are different, the words $w_i$ and $w_j$ are distinct, because $A$ is deterministic.

Hence, the number of words of length $n_i$ belonging to $L$ is at least equal to the cardinality of the set
$$\left\{ (i_1, \ldots, i_{t+1}) \in \mathbb{N}^{t+1} : \sum_{j=1}^{t+1} i_j = i \right\}.$$
and it is a classical\(^\text{15}\) combinatorial exercise to show that the cardinality of this set is exactly

\[
\binom{i + t}{t} = \frac{(i + t) \cdots (i + 1)}{t!}.
\]

This latter quantity is sometimes called the number of *weak compositions* of \(i\). This quantity being larger than \((i + 1)^t/t!\), the conclusion follows, since \(n_i\) is a linear function of \(i\). We have, for all large enough \(i\),

\[g_L(n_i) > \frac{(n_i - |x_{z_1} \cdots z_{t+1}| + 1)^t}{t!}.
\]

\[
L E M M A 1.53.– \text{[SZI 92, Lemma 2]} \text{ Let } L \text{ be a regular language accepted by some DFA } A. \text{ If the growth function of } L \text{ is } O(n^k) \text{ for some integer } k \geq 0, \text{ then each word of } L \text{ is } t\text{-tiered with respect to } A \text{ for some non-negative } t \leq k + 1.
\]

**Proof.**– The proof is by induction on the length of the word \(w \in L\). If \(|w| < \text{Card}(Q)\), then the result holds trivially. We can write \(\text{STS}_A(w)\) as \(\alpha\), and \(w\) is 0-tiered.

Now assume that the statement holds for all words in \(L\) of length less than \(n\) for some \(n \geq \text{Card}(Q)\). Consider a word \(w\) in \(L\) of length exactly \(n\). Take \(\text{STS}_A(w)\). By the pigeonhole principle, there exists a state \(q\) such that \(\text{STS}_A(w) = \mu q \nu q \tau\) where in \(\nu q \tau\) all states are pairwise distinct (we look for the last repetition of a state). Consider all the occurrences of \(q\) in \(\text{STS}_A(w)\) and denote them by \(q^{(1)}, \ldots, q^{(h+1)}\). We have \(h \geq 1\). We can factor the word \(w\) accordingly:

\[
w = w_0 v_1 \cdots v_hw_1
\]

\[\text{[1.7]}
\]

\(^{15}\) See for instance, theorem 29 on page 80 and exercise 87 on page 102 in [COH 78].
and \( \delta(q_0, w_0) = q = \delta(q, v_i) \) for all \( i \in [1, h] \). In particular, \( |w_1| = |\tau| \) and \( |\nu q \tau| \leq \text{Card}(Q) \). Therefore, for all non-negative integers \( n_1, \ldots, n_h \), we have

\[ w_0 v_1^{n_1} \cdots v_h^{n_h} w_1 \in L. \]

Note that for any permutation \( \pi \) of \([1, h]\), the word \( w_0 v_1^{\pi(1)} \cdots v_h^{\pi(h)} w_1 \) also belongs to \( L \) because of the repetition of the state \( q \).

We prove that \( v_1 = \cdots = v_h \). Proceed by contradiction and assume that there exists \( i, j \in [1, h], i \neq j \), such that \( v_i \neq v_j \). Now \( v_i \) (respectively, \( v_j \)) is not a prefix of \( v_j \) (respectively, \( v_i \)) because otherwise \( \text{STS}_A(w) \) would contain more than \( h + 1 \) occurrences of \( q \). Now consider the words in \( \{v_i v_j, v_j v_i\}^* \). We claim that this set is isomorphic to \( \{a, b\}^* \). This means that there exists a bijective morphism between the two monoids. The morphism \( h : \{a, b\}^* \rightarrow \{v_i v_j, v_j v_i\}^* \) that maps \( a \) to \( v_i v_j \) and \( b \) to \( v_j v_i \) is a one-to-one correspondence. The fact that \( h \) is injective derives from the fact that \( v_i \) (respectively, \( v_j \)) is not a prefix\(^{16}\) of \( v_j \) (respectively, \( v_i \)). Now the number of words of length \( n|v_i v_j| \) in \( \{v_i v_j, v_j v_i\}^* \) is thus \( 2^n \). Therefore, there are at least \( 2^n \) words of length \( n|v_i v_j| + |w_0 w_1| \) in \( L \) contradicting the assumption that \( g_L \in \mathcal{O}(n^k) \). Hence, \( v_1 = \cdots = v_h \).

Now comes the conclusion. The word \( w_0 w_1 \) belongs to \( L \) and \( |w_0 w_1| < n \). Using the induction hypothesis, \( w_0 w_1 \) is \( t \)-tiered for some \( t \leq k + 1 \): \( \text{STS}_A(w_0 w_1) = \alpha \beta_1^{d_1} \gamma_1 \cdots \beta_t^{d_t} \gamma_t \), and this factorization satisfies the requirements of definition 1.49. From the choice of \( q \) and \([1.7] \), \( q \) appears only once in \( \gamma_t \) (if

---

16 Imagine, for instance, that \( v_i = u \) and \( v_j = uu \) for some word \( u \). Then, clearly \( h \) is not injective. For a rigorous proof, proceed by contradiction and assume that \( h \) is not injective. There exists two words \( x, y \in \{a, b\}^* \) such that \( h(x) = h(y) \). We have \( |x| = |y| \). There exists a smallest \( k \) such that \( x_k \neq y_k \). Hence, we must have \( v_j v_i = v_i v_j \). We deduce that \( v_i \) and \( v_j \) are powers of the same word that contradicts the fact that \( v_i \) (respectively, \( v_j \)) is not a prefix of \( v_j \) (respectively, \( v_i \)).
\( t \geq 1 \), otherwise \( t = 0 \) and \( \text{STS}_A(w_0w_1) = \alpha \) and \( q \) appears once in \( \alpha \). So, we write \( \gamma_t = \gamma'_t q \gamma''_t \). Hence, we get

\[
\text{STS}_A(w) = \alpha \beta_1^{d_1} \gamma_1 \ldots \beta_t^{d_t} \gamma_t \beta_{t+1}^h \gamma''_t
\]

where \( \beta_{t+1} \) is the state transition sequence obtained from \( q \) when reading \( v_1 \). To see that \( w \) is \( t \)-tiered for some \( t \leq k + 1 \), there are still two conditions to check. Let \( i \in \llbracket 1, t \rrbracket \). Let \( q_{i,0} \) be the first state in \( \beta_i \) and \( \gamma_i \). First, observe that \( q_{i,0} \) does not appear in \( \beta_{t+1} \) because if it were the case, we would have two cycles starting from \( q_{i,0} \). As shown at the end of example 1.51, this would imply an exponential growth not compatible with the assumption. Secondly, we must have \( t < k + 1 \). Indeed if \( t = k + 1 \), then we have found a \( (k + 2) \)-tiered word, and the previous lemma implies that the growth function is in \( \Omega_{\infty}(n^{k+1}) \) contradicting the assumption about the asymptotic behavior of the growth function.

\[ \Box \]

**Lemma 1.54.** [SZI 92, Lemma 3] Let \( A = (Q, q_0, B, \delta, F) \) be a DFA. If there exists \( k \geq 0 \) such that each word of \( L(A) \) is \( t \)-tiered with respect to \( A \), for some \( t \leq k \), then \( L(A) \) can be represented as a finite union of languages of the form

\[
x y_1^* z_1 \ldots y_t^* z_t
\]

with \( 0 \leq t \leq k \), \( |x|, |z_1|, \ldots, |z_t| < \text{Card}(Q) \) and \( 0 < |y_1|, \ldots, |y_t| \leq \text{Card}(Q) \).

**Proof.** As observed in example 1.51, if a word \( w \) in \( L(A) \) is \( t \)-tiered, then it can be factored as \( x y_1^{d_1} z_1 \ldots y_t^{d_t} z_t \) for some words satisfying \( |x|, |z_1|, \ldots, |z_t| < \text{Card}(Q) \) and \( 0 < |y_1|, \ldots, |y_t| \leq \text{Card}(Q) \). Let \( T(w) = x y_1^* z_1 \ldots y_t^* z_t \) be the corresponding language. We have \( T(w) \subseteq L(A) \).

Therefore, we get

\[
L(A) = \bigcup_{w \in L} T(w).
\]

At first glance, we might think that this union is infinite. But notice that the number of possible distinct languages \( T(w) \) is
finite. Indeed, $k$ is a constant, and $T(w)$ is formed from at most $2k + 1$ words with length bounded by $\text{Card}(Q)$. 

**Lemma 1.55.**—[SZI 92, Lemma 4] Let $t > 0$. Let $x, y_1, z_1, \ldots, y_t, z_t$ be words over $B$ where $y_1, \ldots, y_t$ are non-empty. The growth function of $L = x y_1^* z_1 \cdots y_t^* z_t$ is $O(n^{t-1})$.

**Proof.**— The argument is similar to that given in the proof of lemma 1.52. Let $n \geq 0$. A word $w \in L$ has length $n$ if and only if there exists $n_1, \ldots, n_t \in \mathbb{N}$ such that

$$n_1|y_1| + \cdots + n_t|y_t| + \underbrace{|xz_1 \cdots z_t|}_{C} = n.$$

The number of words of length $n$ in $L$ is at most the number of $t$-uples $(n_1, \ldots, n_t)$ such that $n_1|y_1| + \cdots + n_t|y_t| = n - C$. Consider the map

$$f : \mathbb{N}^t \to \mathbb{N}^t, \ (n_1, \ldots, n_t) \mapsto (n_1|y_1|, \ldots, n_t|y_t|).$$

This map is clearly one-to-one. Therefore, due to the injectivity of $f$, the number of words of length $n$ in $L$ is at most

$$\text{Card}\{(m_1, \ldots, m_t) \in \mathbb{N}^t \mid m_1 + \cdots + m_t = n - C\} = \binom{n - C + t - 1}{t - 1}.$$

To finish the proof, observe that this quantity is in $O(n^{t-1})$. 

In view of these lemmas, we can state the following result.

**Theorem 1.56.**— A regular language $L$ is such that its growth function $g_L$ is $O(n^k)$ for some $k \geq 0$ if and only if $L$ can be represented as a finite union of expressions of the form

$$x y_1^* z_1 \cdots y_t^* z_t$$

with $0 \leq t \leq k + 1$. 

We have also observed that if a trim automaton accepting \( L \) contains two distinct cycles (or even one cycle with different labels) attached to the same state, then the growth function is exponential. This can be reformulated as follows.

**Lemma 1.57.**– [SZI 92, Lemma 6] Let \( A = (Q, q_0, B, \delta, F) \) be a trim DFA. Each word in \( L(A) \) is \( t \)-tiered with respect to \( A \) for some \( t \leq \text{Card}(Q) \) if and only if there does not exist any state \( q \in Q \) such that there exists two distinct words \( x, y \) such that \( \delta(q, x) = q = \delta(q, y) \) and, for any non-trivial prefix \( x' \) (respectively, \( y' \)) of \( x \) (respectively, \( y \)), \( \delta(q, x') \neq q \) (respectively, \( \delta(q, y') \neq q \)).

**Proof.**– Assume that the states of \( A \) satisfy the above condition. Then each state in \( A \) appears in at most one cycle (with a unique possible label). Hence, each word in \( L(A) \) is \( t \)-tiered with respect to \( A \) for some \( t \leq \text{Card}(Q) \).

Conversely, assume that each word in \( L(A) \) is \( t \)-tiered with respect to \( A \) for some \( t \leq \text{Card}(Q) \). From lemmas 1.54 and 1.55, we know that the growth function of \( L(A) \) is in \( O(n^{\text{Card}(Q)-1}) \). Proceed by contradiction and assume that there exists a state in \( A \) from which two cycles start. Then, we already have observed that the growth function should be exponential, which is a contradiction. \( \blacksquare \)

**1.8.3. Growing letters in morphic words**

We conclude this section with an application to morphic words. Let \( f : A^* \rightarrow A^* \) be a morphism. With \( f \) associate a matrix \( M_f \) and an automaton \( A_f \) where all states are final. See, for instance, examples 2.24 and 2.31, Volume 1. If \( f \) is a constant-length morphism, then the DFA \( A_f \) has a total transition function. Otherwise, the transition function is partial.
What is necessary for making the connection between morphisms and regular languages is the following observation.

**Proposition 1.58.**– For all \(a \in A\), \(|f^n(a)|\) is exactly the number of words of length \(n\) starting from state \(a\) accepted by the DFA \(A_f\) associated with \(f\).

This can easily be shown by induction on \(n\). We have to recall that the transitions of the automaton are directly obtained from the images \(f(b), b \in A\). Note that since every state is final, we simply have to count the number of paths of length \(n\) starting from \(a\).

Recall (as defined in section 3.1, Volume 1) that a letter \(b\) is *growing* if \(\lim_{n \to +\infty} |f^n(b)| = +\infty\). With the above discussion, we can determine which letters have polynomial growth.

**Example 1.59.**– Take the morphism \(f : a \mapsto ab, b \mapsto bc, c \mapsto c\). For \(n \geq 1\), we can see that

\[
|f^n(a)| = \frac{n(n+1)}{2} + 1, \quad |f^n(b)| = n + 1, \quad |f^n(c)| = 1.
\]

Even though we know the exact behavior of \((|f^n(a)|)_{n \geq 0}\), \(a \in \{a, b, c\}\), we will illustrate how the characterization of languages of polynomial growth can help us to get some information. We have\(^{17}\) depicted the corresponding matrix \(M_f\) and the associated automaton in Figure 1.13.

\[
M_f = \begin{pmatrix}
1 & 0 & 0 \\
1 & 1 & 0 \\
0 & 1 & 1
\end{pmatrix}
\]

**Figure 1.13.** A DFA associated with a polynomial morphism

\(^{17}\) We can also use the fact that the dominating eigenvalue of \(M_f\) is 1.
The language accepted from $a$ is $0^*10^*10^* \cup 0^*10^* \cup 0^*$. In this language, there exists a word that is 3-tiered, e.g. $01010$: $STS(01010) = aabbcc$. From lemma 1.52, we know that $|f^n(a)|$ is in $\Omega(n^2)$. Moreover, from lemma 1.55, we also have that $|f^n(a)|$ is in $O(n^2)$. The language accepted from $b$ (respectively, $c$) is $0^*10^* \cup 0^*$ (respectively, $0^*$). With the same reasoning, we conclude that $|f^n(b)|$ (respectively, $|f^n(c)|$) is in $\Theta'(n)$ (respectively, $\Theta'(1)$).

**Example 1.60.**— We modify the previous example a little. We just add a letter $a$ in the image of $b$ by the morphism. Take the morphism $f' : a \mapsto ab$, $b \mapsto bca$, $c \mapsto ca$. We have depicted the corresponding matrix $M_{f'}$ and associated automaton in Figure 1.14.

![Figure 1.14. A DFA associated with an exponential morphism](image)

We observe directly that we have two cycles originating from $a$ (and the same holds for $b$). Hence, $|f'^n(a)|$ and $|f'^n(b)|$ have exponential growth. Nevertheless, words starting from $c$ are all 1-tiered, and thus we still have $|f'^n(c)|$ in $\Theta'(1)$.

**1.9. Bibliographic notes and comments**

About the **history** of automata theory, see [PER 95] (in French).

We have seen that a language $L$ is regular if and only if the Nerode congruence $\sim_L$ has finite index. There is another equivalence relation that shares the same kind of property. Let $L \subseteq B^*$ be a language. The **context** of a word $u$ is the set
$C_L(u)$ of pairs $(x, y)$ of words such that $xuy$ belongs to $u$. We define the equivalence relation $\equiv_L$ by $u \equiv_L v$ if and only if $C_L(u) = C_L(v)$. This equivalence is trivially a congruence and is called **syntactic congruence**. The quotient set $B^*/\equiv_L$ can be equipped with a product operation to get a monoid: the **syntactic monoid** of $L$. We can show that $L$ is regular if and only if the syntactic congruence $\equiv_L$ has finite index, that is, if and only if the syntactic monoid of $L$ is finite. Just like we can study state complexity (with respect to the number of states of the minimal automaton), we can also study the **syntactic complexity**: the number of equivalence classes for $\equiv_L$. For application of this concept to $b$-recognizable sets, see [LAC 12], where we compute the syntactic complexity of ultimately periodic sets of integers written in base $b$.

For **relations** and functions computed by automata; see, for instance, [FRO 93] and some chapters in [SAK 09].

There is an excellent survey about **$b$-recognizable sets** that served me since the beginning of my research work. You can consult [BRU 94]. I myself have written a survey about numeration systems giving other directions in [RIG 10].

Regarding remark 1.41 and Alexeev’s result: under some mild assumptions, the state complexity of the trim minimal automaton accepting the greedy representations of the multiples of $m \geq 2$ for a wide class of linear numeration systems is studied in [CHA 11]. As an example, the number of states of the trim minimal automaton accepting the greedy representations of $m \mathbb{N}$ in the Fibonacci system is exactly $2m^2$. This will be illustrated in example 2.49.

As regards deciding whether or not a $b$-recognizable set is ultimately periodic, a fast decision procedure in $O(n \log n)$ was recently presented in [MAR 13]. For subsets of $\mathbb{N}^d$, a result of Leroux was obtained earlier with quadratic complexity [LER 05].
Regarding **Cobham’s theorem** for integer bases, there is a subtle mistake that is repeated in several papers (thankfully not in the original paper of Cobham). See [RIG 06] for a patch and a complete description.

A language $L$ such that $g_L(n) \in O(1)$ is said to be *slender*. Regular slender languages are special cases of languages with **polynomial growth**. From the results given in section 1.8, it can be shown that a regular language is slender if and only if it is a finite union of the form $xy^*z$. See, for instance, [PÁU 95] or [SHA 94]. For periodic decimations within a slender language, see [CHA 08, Theorem 3].

We have not considered finite automata accepting **infinite words**. This subject is extensively studied. For instance, a Büchi automaton is a finite automaton devised to accept (or reject) infinite words [BÜC 60]. The acceptance condition is that a run, i.e. an infinite sequence of states corresponding to the considered infinite word, has to go through some final state infinitely often. Other acceptance conditions are also considered in the literature. To give a few references, see [PER 04]. There are several surveys [THO 90]. Some chapters of [KHO 01] are devoted to automata for infinite words.

To conclude this section, let us mention **Černý’s conjecture**. Let $\mathcal{A} = (Q, q_0, B, \delta, F)$ be a DFA. Here, the initial state and the set of final states do not matter. A word $w$ is *synchronizing* if there exists a state $r$ such that, for all states $q$, $\delta(q, w) = r$. If such a word $w$ exists, then the DFA is said to be *synchronizing*. Černý’s conjecture states that if a DFA with $n$ states is synchronizing, then it admits a synchronizing word of length at most $(n - 1)^2$. The DFA depicted in Figure 1.15 admits abbbabba as a synchronizing word.
There are many papers on this conjecture. A cubic bound is known. For a survey, see [VOL 08].