Chapter 5

Properties of Regular Languages

In the previous chapters we have introduced various tools, viz. grammars, automata, to understand regular languages. Also, we have noted that the class of regular languages is closed with respect to certain operations like union, concatenation, Kleene closure. Now, with this information, can we determine whether a given language is regular or not? If a given language is regular, then to prove the same we need to use regular expression, regular grammar, finite automata or Myhill-Nerode theorem. Is there any other way to prove that a language is regular? The answer is "Yes". If a given language can be obtained from some known regular languages by applying those operations which preserve regularity, then one can ascertain that the given language is regular. If a language is not regular, although we have Myhill-Nerode theorem, a better and more practical tool viz. pumping lemma will be introduced to ascertain that the language is not regular. If we were somehow know that some languages are not regular, then again closure properties might be helpful to establish some more languages that are not regular. Thus, closure properties play important role not only in proving certain languages are regular, but also in establishing non-regularity of languages. Hence, we are indented to explore further closure properties of regular languages.

5.1 Closure Properties

5.1.1 Set Theoretic Properties

Theorem 5.1.1. *The class of regular languages is closed with respect to complement.*

Proof. Let L be a regular language accepted by a DFA $\mathscr{A} = (Q, \Sigma, \delta, q_0, F)$. Construct the DFA $\mathscr{A}' = (Q, \Sigma, \delta, q_0, Q - F)$, that is, by interchanging the roles of final and nonfinal states of $\mathscr{A}$. We claim that $L(\mathscr{A}') = L^c$ so that L^c is regular. For $x \in \Sigma^*$,

$$\begin{aligned} x \in L^c &\iff x \notin L \\ &\iff \hat{\delta}(q_0, x) \notin F \\ &\iff \hat{\delta}(q_0, x) \in Q - F \\ &\iff x \in L(\mathscr{A}'). \end{aligned}$$

□

Corollary 5.1.2. *The class of regular languages is closed with respect to intersection.*

Proof. If L_1 and L_2 are regular, then so are L_1^c and L_2^c. Then their union $L_1^c \cup L_2^c$ is also regular. Hence, $(L_1^c \cup L_2^c)^c$ is regular. But, by De Morgan's law

$$L_1 \cap L_2 = (L_1^c \cup L_2^c)^c$$

so that $L_1 \cap L_2$ is regular. □

Alternative Proof by Construction. For $i = 1, 2$, let $\mathscr{A}_i = (Q_i, \Sigma, \delta_i, q_i, F_i)$ be two DFA accepting L_i. That is, $L(\mathscr{A}_1) = L_1$ and $L(\mathscr{A}_2) = L_2$. Set the DFA

$$\mathscr{A} = (Q_1 \times Q_2, \Sigma, \delta, (q_1, q_2), F_1 \times F_2),$$

where δ is defined point-wise by

$$\delta((p, q), a) = (\delta_1(p, a), \delta_2(q, a)),$$

for all $(p, q) \in Q_1 \times Q_2$ and $a \in \Sigma$. We claim that $L(\mathscr{A}) = L_1 \cap L_2$. Using induction on $|x|$, first observe that $\hat{\delta}\Big((p, q), x\Big) = \Big(\hat{\delta}_1(p, x), \hat{\delta}_2(q, x)\Big)$, for all $x \in \Sigma^*$.

Now it clearly follows that

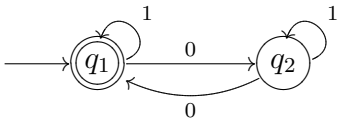
$$\begin{aligned} x \in L(\mathscr{A}) &\iff \hat{\delta}\Big((q_1, q_2), x\Big) \in F_1 \times F_2 \\ &\iff \Big(\hat{\delta}_1(q_1, x), \hat{\delta}_2(q_2, x)\Big) \in F_1 \times F_2 \\ &\iff \hat{\delta}_1(q_1, x) \in F_1 \text{ and } \hat{\delta}_2(q_2, x) \in F_2 \\ &\iff x \in L_1 \text{ and } x \in L_2 \\ &\iff x \in L_1 \cap L_2. \end{aligned}$$

□

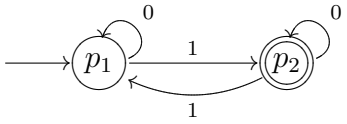
Example 5.1.3. Using the construction given in the above proof, we design a DFA that accepts the language

$$L = \{x \in (0+1)^* \mid |x|_0 \text{ is even and } |x|_1 \text{ is odd}\}$$

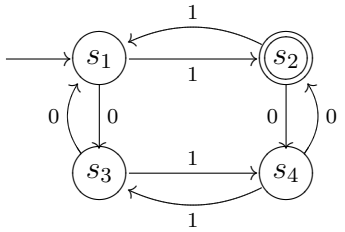
so that L is regular. Note that the following DFA accepts the language $L_1 = \{x \in (0+1)^* \mid |x|_0 \text{ is even}\}$.



Also, the following DFA accepts the language $L_2 = \{x \in (0+1)^* \mid |x|_1 \text{ is odd}\}$.



Now, let $s_1 = (q_1, p_1)$, $s_2 = (q_1, p_2)$, $s_3 = (q_2, p_1)$ and $s_4 = (q_2, p_2)$ and construct the automaton that accepts the intersection of L_1 and L_2 as shown below.



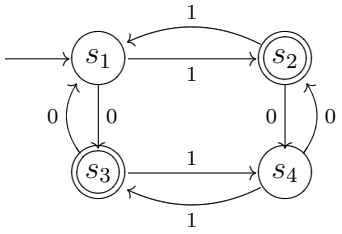
Further, we note the following regarding the automaton. If an input x takes the automaton (from the initial state) to the state

1. s_1, that means, x has even number of $0'$s and even number of $1'$s.
2. s_2, that means, x has even number of $0'$s and odd number of $1'$s (as desired in the current example).
3. s_3, that means, x has odd number of $0'$s and even number of $1'$s.
4. s_4, that means, x has odd number of $0'$s and odd number of $1'$s.

By choosing any combination of states among s_1, s_2, s_3 and s_4, appropriately, as final states we would get DFA which accept input with appropriate combination of $0's$ and $1'$s. For example, to show that the language

$$L' = \Big\{x \in (0+1)^* \;\Big|\; |x|_0 \text{ is even } \Leftrightarrow |x|_1 \text{ is odd}\Big\}.$$

is regular, we choose s_2 and s_3 as final states and obtain the following DFA which accepts L'.



Similarly, any other combination can be considered.

Corollary 5.1.4. *The class of regular languages is closed under set difference.*

Proof. Since $L_1 - L_2 = L_1 \cap L_2^c$, the result follows. □

Example 5.1.5. The language $L = \{a^n \mid n \geq 5\}$ is regular. We apply Corollary 5.1.4 with $L_1 = L(a^*)$ and $L_2 = \{\varepsilon, a, a^2, a^3, a^4\}$. Since L_1 and L_2 are regular, $L = L_1 - L_2$ is regular.

Remark 5.1.6. In general, one may conclude that the removal of finitely many strings from a regular language leaves a regular language.

5.1.2 Other Properties

Theorem 5.1.7. *If L is regular, then so is $L^R = \{x^R \mid x \in L\}$.*

To prove this we use the following lemma.

Lemma 5.1.8. *For every regular language L, there exists a finite automaton $\mathscr{A}$ with a single final state such that $L(\mathscr{A}) = L$.*

Proof. Let $\mathscr{A} = (Q, \Sigma, \delta, q_0, F)$ be a DFA accepting L. Construct $\mathscr{B} = (Q \cup \{p\}, \Sigma, \delta', q_0, \{p\})$, where $p \notin Q$ is a new state and δ' is given by

$$\delta'(q,a) = \begin{cases} \delta(q,a), & \text{if } q \in Q, a \in \Sigma \\ p, & \text{if } q \in F, a = \varepsilon. \end{cases}$$

Note that $\mathscr{B}$ is an NFA, which is obtained by adding a new state p to $\mathscr{A}$ that is connected from all the final states of $\mathscr{A}$ via ε-transitions. Here p is the only final state in $\mathscr{B}$ and all the final states of $\mathscr{A}$ are made nonfinal states. It is easy to prove that $L(\mathscr{A}) = L(\mathscr{B})$. □

Proof of the Theorem 5.1.7. Let $\mathscr{A}$ be a finite automaton with the initial state q_0 and single final state q_f that accepts L. Construct a finite automaton $\mathscr{A}^R$ by reversing the arcs in $\mathscr{A}$ with the same labels and by interchanging the roles of initial and final states. If $x \in \Sigma^*$ is accepted by $\mathscr{A}$, then there is a path q_0 to q_f labeled x in $\mathscr{A}$. Therefore, there will be a path from q_f to q_0 in $\mathscr{A}^R$ labeled x^R so that $x^R \in L(\mathscr{A}^R)$. Conversely, if x is accepted by $\mathscr{A}^R$, then using the similar argument one may notice that its reversal $x^R \in L(\mathscr{A})$. Thus, $L(\mathscr{A}^R) = L^R$ so that L^R is regular. □

Example 5.1.9. Consider the alphabet $\Sigma = \{a_0, a_1, \ldots, a_7\}$, where $a_i = \begin{pmatrix} b_i' \\ b_i'' \\ b_i''' \end{pmatrix}$ and $b_i' b_i'' b_i'''$ is the binary representation of decimal number i, for $0 \leq i \leq 7$. That is, $a_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$, $a_1 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$, $a_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$, …, $a_7 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$.

Now a string $x = a_{i_1} a_{i_2} \cdots a_{i_n}$ over Σ is said to represent correct binary addition if

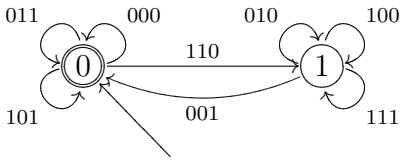
$$b_{i_1}' b_{i_2}' \cdots b_{i_n}' + b_{i_1}'' b_{i_2}'' \cdots b_{i_n}'' = b_{i_1}''' b_{i_2}''' \cdots b_{i_n}'''.$$

For example, the string $a_5 a_1 a_6 a_5$ represents correct addition, because $1011 + 0010 = 1101$. Whereas, $a_5 a_0 a_6 a_5$ does not represent a correct addition, because $1011 + 0010 \neq 1001$.

We observe that the language L over Σ which contain all strings that represent correct addition, i.e.

$$L = \{a_{i_1} a_{i_2} \cdots a_{i_n} \in \Sigma^* \mid b_{i_1}' b_{i_2}' \cdots b_{i_n}' + b_{i_1}'' b_{i_2}'' \cdots b_{i_n}'' = b_{i_1}''' b_{i_2}''' \cdots b_{i_n}'''\},$$

is regular. Consider the NFA shown in the following.



Note that the NFA accepts $L^R \cup \{\varepsilon\}$. Hence, by Remark 5.1.6, L^R is regular. Now, by Theorem 5.1.7, L is regular, as desired.

Definition 5.1.10. Let L_1 and L_2 be two languages over Σ. Right quotient of L_1 by L_2, denoted by L_1/L_2, is the language

$$\{x \in \Sigma^* \mid \exists y \in L_2 \text{ such that } \ xy \in L_1\}.$$

Example 5.1.11. 1. Let $L_1 = \{a, ab, bab, baba\}$ and $L_2 = \{a, ab\}$; then $L_1/L_2 = \{\varepsilon, bab, b\}$.

2. For $L_3 = 10^*1$ and $L_4 = 1$, we have $L_3/L_4 = 10^*$.

3. Let $L_5 = 0^*10^*$.

   (a) $L_5/0^* = L_5$.

   (b) $L_5/10^* = 0^*$.

   (c) $L_5/1 = 0^*$.

4. If $L_6 = a^*b^*$ and $L_7 = \{(a^nb^n)a^* \mid n \geq 0\}$, then $L_6/L_7 = a^*$.

Theorem 5.1.12. *If L is a regular language, then so is L/L', for any language L'.*

Proof. Let L be a regular language and L' be an arbitrary language. Suppose $\mathscr{A} = (Q, \Sigma, \delta, q_0, F)$ is a DFA which accepts L. Set $\mathscr{A}' = (Q, \Sigma, \delta, q_0, F')$, where

$$F' = \{q \in Q \mid \delta(q, x) \in F, \text{ for some } x \in L'\},$$

so that $\mathscr{A}'$ is a DFA. We claim that $L(\mathscr{A}') = L/L'$. For $w \in \Sigma^*$,

$$\begin{aligned} w \in L(\mathscr{A}') &\iff \hat{\delta}(q_0, w) \in F' \\ &\iff \hat{\delta}(q_0, wx) \in F, \text{ for some } x \in L' \\ &\iff w \in L/L'. \end{aligned}$$

Hence L/L' is regular. □

Note that Σ^* is a monoid with respect to the binary operation concatenation. Thus, for two alphabets Σ_1 and Σ_2, a mapping

$$h : \Sigma_1^* \longrightarrow \Sigma_2^*$$

is a homomorphism if, for all $x, y \in \Sigma_1^*$,

$$h(xy) = h(x)h(y).$$

One may notice that to give a homomorphism from Σ_1^* to Σ_2^*, it is enough to give images for the elements of Σ_1. This is because as we are looking for a homomorphism one can give the image of $h(x)$ for any $x = a_1a_2 \cdots a_n \in \Sigma_1^*$ by

$$h(a_1)h(a_2) \cdots h(a_n).$$

Therefore, a homomorphism from Σ_1^* to Σ_2^* is a mapping from Σ_1 to Σ_2^*.

Example 5.1.13. Let $\Sigma_1 = \{a, b\}$ and $\Sigma_2 = \{0, 1\}$. Define $h : \Sigma_1 \longrightarrow \Sigma_2^*$ by

$$h(a) = 10 \quad \text{and} \quad h(b) = 010.$$

Then, h is a homomorphism from Σ_1^* to Σ_2^*, which for example assigns the image 10010010 for the string *abb*.

We can generalize the concept of homomorphism by substituting a language instead of a string for symbols of the domain. Formally, a *substitution* is a mapping from Σ_1 to $\mathscr{P}(\Sigma_2^*)$.

Example 5.1.14. Let $\Sigma_1 = \{a, b\}$ and $\Sigma_2 = \{0, 1\}$. Define $h : \Sigma_1 \longrightarrow \mathscr{P}(\Sigma_2^*)$ by

$$\begin{aligned} h(a) &= \{0^n \mid n \geq 0\}, \text{ say } L_1; \\ h(b) &= \{1^n \mid n \geq 0\}, \text{ say } L_2. \end{aligned}$$

Then, h is a substitution. Now, for any string $a_1 a_2 \cdots a_n \in \Sigma_1^*$, its image under the above substitution h is

$$h(a_1 a_2 \cdots a_n) = h(a_1)h(a_2) \cdots h(a_n),$$

the concatenation of languages. For example, $h(ab)$ is the language

$$L_1 L_2 = \{0^m 1^n \mid m, n \geq 0\} = L(0^*1^*).$$

Given a substitution h from Σ_1 to Σ_2 one may naturally define $h(L)$ for a language L over Σ_1 by

$$h(L) = \bigcup_{x \in L} h(x).$$

Example 5.1.15. Consider the substitution h given in Example 5.1.14 and let $L = \{a^n b^n \mid n \geq 0\}$. For which,

$$\begin{aligned} h(L) &= \bigcup_{x \in L} h(x) \\ &= \bigcup_{n \geq 0} h(a^n b^n) \\ &= \bigcup_{n \geq 0} \overbrace{h(a) \cdots h(a)}^{n \text{ times}} \overbrace{h(b) \cdots h(b)}^{n \text{ times}} \\ &= \bigcup_{n \geq 0} \overbrace{0^* \cdots 0^*}^{n \text{ times}} \overbrace{1^* \cdots 1^*}^{n \text{ times}} \\ &= \{0^m 1^n \mid m, n \geq 0\} = 0^*1^*. \end{aligned}$$

Example 5.1.16. Define the substitution $h : \{a, b\} \longrightarrow \mathscr{P}(\{0, 1\}^*)$ by

$$\begin{aligned} h(a) &= \text{the set of strings over } \{0,\ 1\} \text{ ending with 1;} \\ h(b) &= \text{the set of strings over } \{0,\ 1\} \text{ starting with 0.} \end{aligned}$$

For the language $L = \{a^n b^m \mid n, m \geq 1\}$, we compute $h(L)$ through regular expressions described below.

Note that the regular expression for L is a^+b^+ and that are for $h(a)$ and $h(b)$ are $(0 + 1)^*1$ and $0(0 + 1)^*$. Now, write the regular expression that is obtained from the expression of L by replacing each occurrence of a by the expression of $h(a)$ and by replacing each occurrence of b by the expression of $h(b)$. That is, from a^+b^+, we obtain the regular expression

$$((0 + 1)^*1)^+(0(0 + 1)^*)^+.$$

This can be simplified as follows.

$$\begin{aligned} ((0 + 1)^*1)^+(0(0 + 1)^*)^+ &= ((0 + 1)^*1)^*((0 + 1)^*1)(0(0 + 1)^*)(0(0 + 1)^*)^* \\ &= ((0 + 1)^*1)^*(0 + 1)^*10(0 + 1)^*(0(0 + 1)^*)^* \\ &= (0 + 1)^*10(0 + 1)^*. \end{aligned}$$

The following Theorem 5.1.17 confirms that the expression obtained in this process represents $h(L)$. Thus, from the expression of $h(L)$, we can conclude that the language $h(L)$ is the set of all strings over $\{0, 1\}$ that have 10 as substring.

Theorem 5.1.17. *The class of regular languages is closed under substitutions by regular languages. That is, if h is a substitution on Σ such that $h(a)$ is regular for each $a \in \Sigma$, then $h(L)$ is regular for each regular language L over Σ.*

Proof. Let r be a regular expression for language L over Σ, i.e. $L(r) = L$, and, for each $a \in \Sigma$, let r_a be a regular expression for $h(a)$. Suppose r' is the expression obtained by replacing r_a for each occurrence of a (of Σ) in r so that r' is a regular expression. We claim that $L(r') = h(L(r))$ so that $h(L)$ is regular. We prove our claim by induction on the number of operations involved in r.

Assume that the number of operations involved in r is zero. Then there are three possibilities for r, viz. 1. $\varnothing$, 2. ε and 3. a for some $a \in \Sigma$.

1. If $r = \varnothing$, then $r' = \varnothing$ and $h(L(\varnothing)) = \varnothing$ so that the result is straightforward.

2. In case $r = \varepsilon$, $r' = \varepsilon$ and hence we have

$$h(L(r)) = h(\{\varepsilon\}) = \{\varepsilon\} = L(r').$$

3. For other case, let $r = a$, for some $a \in \Sigma$. Then, $r' = r_a$ so that

$$h(L(r)) = h(\{a\}) = L(r_a) = L(r').$$

Hence basis of the induction is satisfied.

For inductive hypothesis, assume $L(r') = h(L(r))$ for all those regular expressions r which have k or fewer operations. Now consider a regular expression r with $k + 1$ operations. Then

$$r = r_1 + r_2 \quad \text{or} \quad r = r_1 r_2 \quad \text{or} \quad r = r_1^*$$

for some regular expressions r_1 and r_2. Note that both r_1 and r_2 have k or fewer operations. Hence, by inductive hypothesis, we have

$$L(r_1') = h(L(r_1)) \quad \text{and} \quad L(r_2') = h(L(r_2)),$$

where r_1' and r_2' are the regular expressions which are obtained from r_1 and r_2 by replacing r_a for each a in r_1 and r_2, respectively.

Consider the case where $r = r_1 + r_2$. The expression r' (that is obtained from r) is nothing else but replacing each r_a in the individual r_1 and r_2, we have

$$r' = r_1' + r_2'.$$

Hence,

$$\begin{aligned} L(r') &= L(r_1' + r_2') \\ &= L(r_1') \cup L(r_2') \\ &= h(L(r_1)) \cup h(L(r_2)) \\ &= h(L(r_1) \cup L(r_2)) \\ &= h(L(r_1 + r_2)) \\ &= h(L(r)) \end{aligned}$$

as desired, in this case. Similarly, other two cases, viz. $r = r_1 r_2$ and $r = r_1^*$, can be handled.

Hence, the class of regular languages is closed under substitutions by regular languages. □

Corollary 5.1.18. *The class of regular languages is closed under homomorphisms.*

Theorem 5.1.19. *Let $h : \Sigma_1 \longrightarrow \Sigma_2^*$ be a homomorphism and $L \subseteq \Sigma_2^*$ be regular. Then, inverse homomorphic image of L,*

$$h^{-1}(L) = \{x \in \Sigma_1^* \mid h(x) \in L\}$$

is regular.

Proof. Let $\mathscr{A} = (Q, \Sigma_2, \delta, q_0, F)$ be a DFA accepting L. We construct a DFA $\mathscr{A}'$ which accepts $h^{-1}(L)$. Note that, in $\mathscr{A}'$, we have to define transitions for the symbols of Σ_1 such that $\mathscr{A}'$ accepts $h^{-1}(L)$. We compose the homomorphism h with the transition map δ of $\mathscr{A}$ to define the transition map of $\mathscr{A}'$. Precisely, we set $\mathscr{A}' = (Q, \Sigma_1, \delta', q_0, F)$, where the transition map

$$\delta' : Q \times \Sigma_1 \longrightarrow Q$$

is defined by

$$\delta'(q, a) = \hat{\delta}(q, h(a))$$

for all $q \in Q$ and $a \in \Sigma_1$. Note that $\mathscr{A}'$ is a DFA. Now, for all $x \in \Sigma_1^*$, we prove that

$$\hat{\delta}'(q_0, x) = \hat{\delta}(q_0, h(x)).$$

This gives us $L(\mathscr{A}') = h^{-1}(L)$, because, for $x \in \Sigma_1^*$,

$$\begin{aligned} x \in h^{-1}(L) &\iff h(x) \in L \\ &\iff \hat{\delta}(q_0, h(x)) \in F \\ &\iff \hat{\delta}'(q_0, x) \in F \\ &\iff x \in L(\mathscr{A}'). \end{aligned}$$

We prove our assertion by induction on $|x|$. For basis, suppose $|x| = 0$. That is $x = \varepsilon$. Then clearly,

$$\hat{\delta}'(q_0, x) = q_0 = \hat{\delta}(q_0, \varepsilon) = \hat{\delta}(q_0, h(x)).$$

Here $h(\varepsilon) = \varepsilon$, because h is a homomorphism. Further, by definition of δ', we have

$$\hat{\delta}'(q_0, a) = \delta'(q_0, a) = \hat{\delta}(q_0, h(a)),$$

for all $a \in \Sigma_1^*$, so that the assertion is true for $|x| = 1$ also. For inductive hypothesis, assume

$$\hat{\delta}'(q_0, x) = \hat{\delta}(q_0, h(x))$$

for all $x \in \Sigma_1^*$ with $|x| = k$. Let $x \in \Sigma_1^*$ with $|x| = k$ and $a \in \Sigma_1$. Now,

$$\begin{aligned}
\hat{\delta}'(q_0, xa) &= \hat{\delta}'(\hat{\delta}'(q_0, x), a) \\
&= \hat{\delta}'(\hat{\delta}(q_0, h(x)), a) && \text{(by inductive hypothesis)} \\
&= \delta'(\hat{\delta}(q_0, h(x)), a) \\
&= \hat{\delta}(\hat{\delta}(q_0, h(x)), h(a)) && \text{(by definition of } \delta') \\
&= \hat{\delta}(q_0, h(x)h(a)) \\
&= \hat{\delta}(q_0, h(xa)). && \text{(by the property of homomorphism)}
\end{aligned}$$

Hence the result. □

Example 5.1.20. Suppose $L \subseteq (0+1)^*$ is a regular language. Now we observe that the language

$$L' = \{a_1 b_1 \cdots a_n b_n \in (0+1)^* \mid a_1 \cdots a_n \in L \text{ and } a_i = 0 \text{ iff } b_i = 1\}$$

is regular. For instance, define

$$f : \{0,1\} \longrightarrow \{0,1\}^*$$

by $f(0) = 01$ and $f(1) = 10$ so that f is a homomorphism. Now note that

$$\begin{aligned}
f(L) &= \{f(x) \mid x \in L\} \\
&= \{f(a_1 \cdots a_n) \mid a_1 \cdots a_n \in L\} \\
&= \{f(a_1) \cdots f(a_n) \mid a_1 \cdots a_n \in L\} \\
&= \{a_1 b_1 \cdots a_n b_n \mid a_1 \cdots a_n \in L \text{ and } a_i = 0 \text{ iff } b_i = 1\} \\
&= L'
\end{aligned}$$

Being homomorphic image of a regular language, L' is regular.

5.2 Pumping Lemma

Theorem 5.2.1 (Pumping Lemma)**.** *If L is an infinite regular language, then there exists a number κ (associated to L) such that for all $x \in L$ with $|x| \geq \kappa$, x can be written as uvw satisfying the following:*

1. *$v \neq \varepsilon$, and*
2. *$uv^i w \in L$, for all $i \geq 0$.*

Proof. Let $\mathscr{A} = (Q, \Sigma, \delta, q_0, F)$ be a DFA accepting L. Let $|Q| = \kappa$. Since L is infinite, there exists a string $x = a_1 a_2 \cdots a_n$ in L with $n \geq \kappa$, where $a_i \in \Sigma$. Consider the accepting sequence of x, say

$$q_0, q_1, \ldots, q_n$$

where, for $0 \leq i \leq n-1$, $\delta(q_i, a_{i+1}) = q_{i+1}$ and $q_n \in F$. As there are only κ states in Q, by pigeon-hole principle, at least one state must be repeated in the accepting sequence of x. Let $q_r = q_s$, for $0 \leq r < s \leq n$. Write $u = a_1 \cdots a_r$, $v = a_{r+1} \cdots a_s$ and $w = a_{s+1} \cdots a_n$; so that $x = uvw$. Note that, as $r < s$, we have $v \neq \varepsilon$. Now we will prove that $uv^i w \in L$, for all $i \geq 0$.

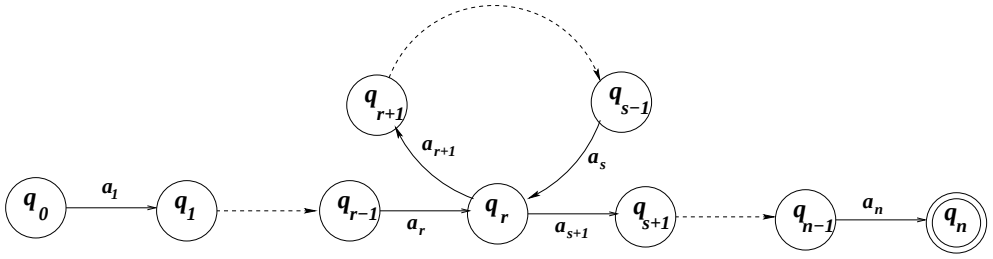


Figure 5.1: Pumping Sequence

For a given $i \geq 0$, $uv^i w = a_1 \cdots a_r (a_{r+1} \cdots a_s)^i a_{s+1} \cdots a_n$. Since $q_r = q_s$, we see that there is a computation for $uv^i w$ in $\mathscr{A}$, as given below:

$$\begin{aligned}
(q_0, uv^i w) &\vdash^* (q_r, v^i w) \\
&\vdash^* (q_s, v^{i-1w}) \\
&= (q_r, v^{i-1} w) \\
&\vdash^* (q_s, v^{i-2} w) \\
&\vdots \\
&\vdash^* (q_s, w) \\
&\vdash^* q_n
\end{aligned}$$

Thus, for $i \geq 0$, $uv^i w \in L$. □

Remark 5.2.2. If L is finite, then by choosing $\kappa = 1 + \max\{|x| \mid x \in L\}$ one may notice that L vacuously satisfies the pumping lemma, as there is no string of length greater than or equal to κ in L. Thus, the pumping lemma holds good for all regular languages.

A Logical Formula. If we write

$P(L)$: L satisfies pumping lemma and

$R(L)$: L is regular,

then the pumping lemma for regular languages is $R(L) \Longrightarrow P(L)$. The statement $P(L)$ can be elaborated by the following logical formula:

$$(\forall L)(\exists \kappa)(\forall x)\Big[x \in L \text{ and } |x| \geq \kappa \Longrightarrow (\exists u, v, w)\Big(x = uvw, v \neq \varepsilon \Longrightarrow (\forall i)(uv^i w \in L)\Big)\Big].$$

A Tool to Ascertain Non-regularity. If a language fails to satisfy the pumping lemma, then it cannot be regular. That is, $\neg P(L) \Longrightarrow \neg R(L)$ – the contrapositive form of the pumping lemma. The $\neg P(L)$ can be formulated as below:

$$(\exists L)(\forall \kappa)(\exists x)\Big[x \in L \text{ and } |x| \geq \kappa \text{ and } (\forall u, v, w)\Big(x = uvw, v \neq \varepsilon \text{ and } (\exists i)(uv^i w \notin L)\Big)\Big].$$

This statement can be better explained via the following adversarial game. Given a language L, if we want to show that L is not regular, then we play as given in the following steps.

1. An opponent will give us an arbitrary number κ.
2. Given κ, we pickup a string $x \in L$ with $|x| \geq \kappa$.
3. Opponent will divide x into u, v and w, arbitrarily, with $v \neq \varepsilon$. (Here $x = uvw$.)
4. We pickup an i such that $uv^i w \notin L$.

Example 5.2.3. We wish to show that $L = \{a^n b^n \mid n \geq 0\}$ is not regular. An opponent will give us a number κ. Then, we choose a string $x = a^m b^m \in L$ with $|x| \geq \kappa$. The opponent will give us u, v and w, where $x = uvw$ and $v \neq \varepsilon$. Now, there are three possibilities for v, viz. (1) a^p, for $p \geq 1$ (2) b^q, for $q \geq 1$ (3) $a^p b^q$, for $p, q \geq 1$. In each case, we give an i such that $uv^i w \notin L$, as shown below.

Case-1 ($v = a^p$, $p \geq 1$). Choose $i = 2$. Then $uv^i w = a^{m+k} b^m$ which is clearly not in L.

(In fact, except for $i = 1$, for all i, $uv^i w \notin L$)

Case-2 ($v = b^q$, $q \geq 1$). Similar to the Case-1, one may easily observe that except for $i = 1$, for all i, $uv^iw \notin L$. As above, say $i = 2$, then $uv^2w \notin L$.

Case-3 ($v = a^pb^q$, $p, q \geq 1$). Now, again choose $i = 2$. Note that

$$uv^iw = a^{m-p}(a^pb^q)^2b^{m-q} = a^mb^qa^pb^m.$$

Since $p, q \geq 1$, we observe that there is an a after a b in the resultant string. Hence $uv^2w \notin L$.

Hence, we conclude that L is not regular.

Example 5.2.4. We prove that $L = \{ww \mid w \in (0+1)^*\}$ is not regular. On contrary, assume that L is regular. Let κ be the pumping lemma constant associated to L. Now, choose $x = 0^n1^n0^n1^n$ from L with $|x| \geq \kappa$. If x is written as uvw with $v \neq \varepsilon$, then there are ten possibilities for v. In each case, we observe that pumping v will result a string that is not in L. This contradicts the pumping lemma so that L is not regular.

Case-1 (For $p \geq 1$, $v = 0^p$ with $x = 0^{k_1}v0^{k_2}1^n0^n1^n$).
Through the following points, in this case, we demonstrate a contradiction to pumping lemma.

1. In this case, v is in the first block of $0'$s and $p < n$.
2. Suppose v is pumped for $i = 2$.
3. If the resultant string uv^iw is of odd length, then clearly it is not in L.
4. Otherwise, suppose $uv^iw = yz$ with $|y| = |z|$.
5. Then, clearly, $|y| = |z| = \frac{4n+p}{2} = 2n + \frac{p}{2}$.
6. Since z is the suffix of the resultant string and $|z| > 2n$, we have $z = 1^{\frac{p}{2}}0^n1^n$.
7. Hence, clearly, $y = 0^{n+p}1^{n-\frac{p}{2}} \neq z$ so that $yz \notin L$.

Using a similar argument as given in Case-1 and the arguments shown in Example 5.2.3, one can demonstrate contradictions to pumping lemma in each of the following remaining cases.

Case-2 (For $q \geq 1$, $v = 1^q$ with $x = 0^n1^{k_1}v1^{k_2}0^n1^n$). That is, v is in the first block of $1'$s.

Case-3 (For $p \geq 1$, $v = 0^p$ with $x = 0^n1^n0^{k_1}v0^{k_2}1^n$). That is, v is in the second block of $0'$s.

Case-4 (For $q \geq 1$, $v = 1^q$ with $x = 0^n1^n0^n1^{k_1}v1^{k_2}$). That is, v is in the second block of $1'$s.

Case-5 (For $p, q \geq 1$, $v = 0^p1^q$ with $x = 0^{k_1}v1^{k_2}0^n1^n$). That is, v is in the first block of 0^n1^n.

Case-6 (For $p, q \geq 1$, $v = 1^p0^q$ with $x = 0^n1^{k_1}v0^{k_2}1^n$). That is, v is in the block of 1^n0^n.

Case-7 (For $p, q \geq 1$, $v = 0^p1^q$ with $x = 0^n1^n0^{k_1}v1^{k_2}$). That is, v is in the second block of 0^n1^n.

Case-8 (For $p, q \geq 1$, $v = 0^p1^n0^q$ with $x = 0^{k_1}v0^{k_2}1^n$). That is, v is in the block of $0^n1^n0^n$.

Case-9 (For $p, q \geq 1$, $v = 1^p0^n1^q$ with $x = 0^n1^{k_1}v1^{k_2}$). That is, v is in the block of $1^n0^n1^n$.

Case-10 (For $p, q \geq 1$, $v = 0^p1^n0^n1^q$ with $x = 0^{k_1}v1^{k_2}$). That is, v extended over all the blocks of 0's and $1'$s.

Hence, L is not regular.

Remark 5.2.5. Although it is sufficient to choose a particular string to counter the pumping lemma, it is often observed that depending on the string chosen there can be several possibilities of partitions as uvw that are to be considered as we have to check for all possibilities. For instance, in Example 5.2.3 we have discussed three cases. On the other hand, in Example 5.2.4 instead of choosing a typical string we have chosen a string which reduces the number of possibilities to discuss. Even then, there are ten possibilities to discuss.

In the following, we show how the number of possibilities, to be considered, can be reduced further. In fact, we observe that it is sufficient to consider the occurrence of v within the first κ symbols of the string under consideration. More precisely, we state the assertion through the following theorem, a restricted version of pumping lemma.

Theorem 5.2.6 (Pumping Lemma – A Restricted Version)**.** *If L is an infinite regular language, then there exists a number κ (associated to L) such that for all $x \in L$ with $|x| \geq \kappa$, x can be written as uvw satisfying the following:*

1. *$v \neq \varepsilon$,*
2. *$|uv| \leq \kappa$, and*
3. *$uv^iw \in L$, for all $i \geq 0$.*

Proof. The proof of the theorem is exactly same as that of Theorem 5.2.1, except that, when the pigeon-hole principle is applied to find the repetition of states in the accepting sequence, we find the repetition of states within the first $\kappa + 1$ states of the sequence. As $|x| \geq \kappa$, there will be at least $\kappa + 1$ states in the sequence and since $|Q| = \kappa$, there will be repetition in the first $\kappa + 1$ states. Hence, we have the desired extra condition $|uv| \leq \kappa$. □

Example 5.2.7. Consider that language $L = \{ww \mid w \in (0 + 1)^*\}$ given in Example 5.2.4. Using Theorem 5.2.6, we supply an elegant proof for L is not regular. If L is regular, then let κ be the pumping lemma constant associated to L. Now choose the string $x = 0^\kappa 1^\kappa 0^\kappa 1^\kappa$ from L. Using the Theorem 5.2.6, if x is written as uvw with $|uv| \leq \kappa$ and $v \neq \varepsilon$ then there is only one possibility for v in x, that is a substring of first block of $0'$s. Hence, clearly, by pumping v we get a string that is not in the language so that we can conclude L is not regular.

Example 5.2.8. We show that the language $L = \{x \in (a + b)^* \mid x = x^R\}$ is not regular. Consider the string $x = 0^\kappa 1^\kappa 1^\kappa 0^\kappa$ from L, where κ is pumping lemma constant associated to L. If x is written as uvw with $|uv| \leq \kappa$ and $v \neq \varepsilon$ then there is only one possibility for v in x, as in the previous example. Now, pumping v will result a string that is not a palindrome so that L is not regular.

Example 5.2.9. We show that the language

$$L = \{xcy \mid x, y \in \{a, b\}^* \text{ and } |x| = |y|\}$$

is not regular. On contrary, assume that L is regular. Then, since regular languages are closed with respect to intersection, $L' = L \cap a^*cb^*$ is regular. But, note that

$$L' = \{a^n cb^n \mid n \geq 0\},$$

because $xcy \in a^*cb^*$ and $|x| = |y|$ implies $xcy = a^n cb^n$. Now, using the homomorphism h that is defined by

$$h(a) = 0,\ h(b) = 1,\ \text{and } h(c) = \varepsilon$$

we have

$$h(L') = \{0^n 1^n \mid n \geq 0\}.$$

Since regular languages are closed under homomorphisms, $h(L')$ is also regular. But we know that $\{0^n 1^n \mid n \geq 0\}$ is not regular. Thus, we arrived at a contradiction. Hence, L is not regular.