

4.3 Identifying Nonregular Languages



Using the Pigeonhole Principle

The term “pigeonhole principle” is used by mathematicians to refer to the following simple observation. If we put n objects into m boxes (pigeonholes), and if $n > m$, then at least one box must have more than one item in it. This is such an obvious fact that it is surprising how many deep results can be obtained from it.

Example 4.6

Is the language $L = \{a^n b^n : n \geq 0\}$ regular? The answer is no, as we show using a proof by contradiction.

Suppose L is regular. Then some dfa $M = (Q, \{a, b\}, \delta, q_0, F)$ exists for it. Now look at $\delta^*(q_0, a^i)$ for $i = 1, 2, 3, \dots$. Since there are an unlimited number of i 's, but only a finite number of states in M , the pigeonhole principle tells us that there must be some state, say q , such that

$$\delta^*(q_0, a^n) = q$$

and

$$\delta^*(q_0, a^m) = q,$$

with $n \neq m$. But since M accepts $a^n b^n$ we must have

$$\delta^*(q, b^n) = q_f \in F.$$

From this we can conclude that

$$\begin{aligned}\delta^*(q_0, a^m b^n) &= \delta^*(\delta^*(q_0, a^m), b^n) \\ &= \delta^*(q, b^n) \\ &= q_f.\end{aligned}$$

This contradicts the original assumption that M accepts $a^m b^n$ only if $n = m$, and leads us to conclude that L cannot be regular. ■

[2] A Pumping Lemma

The following result, known as the **pumping lemma** for regular languages, uses the pigeonhole principle in another form. The proof is based on the observation that in a transition graph with n vertices, any walk of length n or longer must repeat some vertex, that is, contain a cycle.

Theorem 4.8

Let L be an infinite regular language. Then there exists some positive integer m such that any $w \in L$ with $|w| \geq m$ can be decomposed as

$$w = xyz$$

with

$$|xy| \leq m,$$

and

$$|y| \geq 1,$$

such that

$$w_i = xy^i z, \tag{4.2}$$

is also in L for all $i = 0, 1, 2, \dots$

n+1 states

Proof: If L is regular, there exists a dfa that recognizes it. Let such a dfa have states labeled $q_0, q_1, q_2, \dots, q_n$. Now take a string w in L such that $|w| \geq m = n + 1$. Since L is assumed to be infinite, this can always be done.

Consider the set of states the automaton goes through as it processes w , say

Let $w = w_1 \dots w_{|w|}$

w_1 w_2 $w_{|w|}$
 $q_0, q_i, q_j, \dots, q_f$

Since this sequence has exactly $|w| + 1$ entries, at least one state must be repeated, and such a repetition must start no later than the n th move.

Thus, the sequence must look like

$q_0, q_i, q_j, \dots, q_r, \dots, q_r, \dots, q_f,$

indicating there must be substrings x, y, z of w such that

$$\delta^*(q_0, x) = q_r,$$

$$\delta^*(q_r, y) = q_r,$$

$$\delta^*(q_r, z) = q_f,$$

with $|xy| \leq n + 1 = m$ and $|y| \geq 1$. From this it immediately follows that

$$\delta^*(q_0, xz) = q_f,$$

as well as

$$\delta^* (q_0, xy^2z) = q_f,$$

$$\delta^*(q_0, xy^3z) = q_f,$$

and so on, completing the proof of the theorem. \blacksquare

Example 4.7

Use the pumping lemma to show that $L = \{a^n b^n : n \geq 0\}$ is not regular. Assume that L is regular, so that the pumping lemma must hold. We do not know the value of m , but whatever it is, we can always choose $n = m$. Therefore, the substring y must consist entirely of a 's. Suppose $|y| = k$. Then the string obtained by using $i = 0$ in Equation (4.2) is

$$w_0 = a^{m-k} b^m$$

and is clearly not in L . This contradicts the pumping lemma and thereby indicates that the assumption that L is regular must be false. ■

Example 4.8

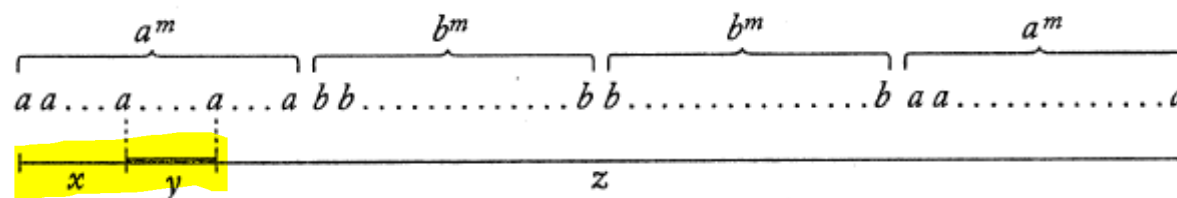
Show that

$$L = \{ww^R : w \in \Sigma^*\}$$

is not regular.

Whatever m the opponent picks on Step 1, we can always choose a w as shown in Figure 4.5. Because of this choice, and the requirement that $|xy| \leq m$, the opponent is restricted in Step 3 to choosing a y that consists entirely of a 's. In Step 4, we use $i = 0$. The string obtained in this fashion has fewer a 's on the left than on the right and so cannot be of the form ww^R . Therefore, L is not regular.

Figure 4.5



Example 4.9

Let $\Sigma = \{a, b\}$. The language

$$L = \{w \in \Sigma^* : n_a(w) < n_b(w)\}$$

is not regular.

Suppose we are given m . Since we have complete freedom in choosing w , we pick $w = a^m b^{m+1}$. Now, because $|xy|$ cannot be greater than m , the opponent cannot do anything but pick a y with all a 's, that is

$$y = a^k, \quad 1 \leq k \leq m.$$

We now pump up, using $i = 2$. The resulting string

$$w_2 = a^{m+k} b^{m+1}$$

is not in L . Therefore, the pumping lemma is violated, and L is not regular.

Example 4.10


The language

$$L = \{(ab)^n a^k : n > k, k \geq 0\}$$

is not regular.

Given m , we pick as our string

$$w = (ab)^{m+1} a^m,$$

which is in L . Because of the constraint $|xy| \leq m$, both x and y must be in the part of the string made up of ab 's. The choice of x does not affect the argument, so let us see what can be done with y . If our opponent picks $y = a$, we choose $i = 0$ and get a string not in L ($(ab)^* a^*$). If the opponent picks $y = ab$, we can choose $i = 0$ again. Now we get the string $(ab)^m a^m$, which is not in L . In the same way, we can deal with any possible choice by the opponent, thereby proving our claim. 

Example 4.11

Show that

$$L = \{a^n : n \text{ is a perfect square}\}$$

is not regular.

Given the opponent's choice of m , we pick

$$w = a^{m^2}.$$

If $w = xyz$ is the decomposition, then clearly

$$y = a^k$$

with $1 \leq k \leq m$. In that case,

$$w_0 = a^{m^2 - k}$$

But $m^2 - k > (m-1)^2$, so that w_0 cannot be in L . Therefore, the language is not regular.

Example 4.12

Show that the language

$$L = \{a^n b^k c^{n+k} : n \geq 0, k \geq 0\}$$

is not regular.

It is not difficult to apply the pumping lemma directly, but it is even easier to use closure under homomorphism. Take

$$h(a) = a, h(b) = a, h(c) = c,$$

then

$$\begin{aligned} h(L) &= \{a^{n+k} c^{n+k} : n + k \geq 0\} \\ &= \{a^i c^i : i \geq 0\}. \end{aligned}$$

But we know this language is not regular; therefore, L cannot be regular either.

Example 4.13

Show that the language

$$L = \{a^n b^l : n \neq l\}$$

is not regular.

Here we need a bit of ingenuity to apply the pumping lemma directly. Choosing a string with $n = l + 1$ or $n = l + 2$ will not do, since our opponent can always choose a decomposition that will make it impossible to pump the string out of the language (that is, pump it so that it has an equal number of a 's and b 's). We must be more inventive. Let us take $n = m!$ and $l = (m + 1)!$. If the opponent now chooses a y (by necessity consisting of all a 's) of length $k < m$, we pump i times to generate a string with $m! + (i - 1)k$ a 's. We can get a contradiction of the pumping lemma if we can pick i such that

$$m! + (i - 1)k = (m + 1)!$$

This is always possible since

$$i = 1 + \frac{m \cdot m!}{k}$$

and $k \leq m$. The right side is therefore an integer, and we have succeeded in violating the conditions of the pumping lemma.

However, there is a much more elegant way of solving this problem. Suppose L were regular. Then, by Theorem 4.1, \bar{L} and the language

$$L_1 = \bar{L} \cap L(a^*b^*)$$

would also be regular. But $L_1 = \{a^n b^n : n \geq 0\}$, which we have already classified as nonregular. Consequently, L cannot be regular.
