# 1.4 NONREGULAR LANGUAGES

To understand the power of finite automata, you must also understand their limitations. In this section, we show how to prove that certain languages cannot be recognized by any finite automaton.

Let's take the language $B = \{0^n1^n | \, n \geq 0\}$. If we attempt to find a DFA that recognizes $B$, we discover that the machine seems to need to remember how many 0s have been seen so far as it reads the input. Because the number of 0s isn't limited, the machine will have to keep track of an unlimited number of possibilities. But it cannot do so with any finite number of states.

Next, we present a method for proving that languages such as $B$ are not regular. Doesn't the argument already given prove nonregularity because the number of 0s is unlimited? It does not. Just because the language appears to require unbounded memory doesn't mean that it is necessarily so. It does happen to be true for the language $B$; but other languages seem to require an unlimited number of possibilities, yet actually they are regular. For example, consider two languages over the alphabet $\Sigma = \{0,1\}$:

$C = \{w | \, w$ has an equal number of 0s and 1s$\}$, and

$D = \{w | \, w$ has an equal number of occurrences of 01 and 10 as substrings$\}$.

At first glance, a recognizing machine appears to need to count in each case, and therefore neither language appears to be regular. As expected, $C$ is not regular, but surprisingly $D$ is regular![6] Thus our intuition can sometimes lead us astray, which is why we need mathematical proofs for certainty. In this section, we show how to prove that certain languages are not regular.

## THE PUMPING LEMMA FOR REGULAR LANGUAGES

Our technique for proving nonregularity stems from a theorem about regular languages, traditionally called the ***pumping lemma***. This theorem states that all regular languages have a special property. If we can show that a language does not have this property, we are guaranteed that it is not regular. The property states that all strings in the language can be "pumped" if they are at least as long as a certain special value, called the ***pumping length***. That means each such string contains a section that can be repeated any number of times with the resulting string remaining in the language.

[6]See Problem 1.48.

**THEOREM 1.70**

**Pumping lemma** If $A$ is a regular language, then there is a number $p$ (the pumping length) where if $s$ is any string in $A$ of length at least $p$, then $s$ may be divided into three pieces, $s = xyz$, satisfying the following conditions:

1. for each $i \geq 0$, $xy^iz \in A$,
2. $|y| > 0$, and
3. $|xy| \leq p$.

Recall the notation where $|s|$ represents the length of string $s$, $y^i$ means that $i$ copies of $y$ are concatenated together, and $y^0$ equals $\varepsilon$.

When $s$ is divided into $xyz$, either $x$ or $z$ may be $\varepsilon$, but condition 2 says that $y \neq \varepsilon$. Observe that without condition 2 the theorem would be trivially true. Condition 3 states that the pieces $x$ and $y$ together have length at most $p$. It is an extra technical condition that we occasionally find useful when proving certain languages to be nonregular. See Example 1.74 for an application of condition 3.

**PROOF IDEA** Let $M = (Q, \Sigma, \delta, q_1, F)$ be a DFA that recognizes $A$. We assign the pumping length $p$ to be the number of states of $M$. We show that any string $s$ in $A$ of length at least $p$ may be broken into the three pieces $xyz$, satisfying our three conditions. What if no strings in $A$ are of length at least $p$? Then our task is even easier because the theorem becomes *vacuously* true: Obviously the three conditions hold for all strings of length at least $p$ if there aren't any such strings.

If $s$ in $A$ has length at least $p$, consider the sequence of states that $M$ goes through when computing with input $s$. It starts with $q_1$ the start state, then goes to, say, $q_3$, then, say, $q_{20}$, then $q_9$, and so on, until it reaches the end of $s$ in state $q_{13}$. With $s$ in $A$, we know that $M$ accepts $s$, so $q_{13}$ is an accept state.

If we let $n$ be the length of $s$, the sequence of states $q_1, q_3, q_{20}, q_9, \ldots, q_{13}$ has length $n + 1$. Because $n$ is at least $p$, we know that $n + 1$ is greater than $p$, the number of states of $M$. Therefore, the sequence must contain a repeated state. This result is an example of the ***pigeonhole principle***, a fancy name for the rather obvious fact that if $p$ pigeons are placed into fewer than $p$ holes, some hole has to have more than one pigeon in it.

The following figure shows the string $s$ and the sequence of states that $M$ goes through when processing $s$. State $q_9$ is the one that repeats.

$$
\begin{array}{rccccccccccccccccc}
s = & & s_1 & & s_2 & & s_3 & & s_4 & & s_5 & & s_6 & & \cdots & & s_n & \\
& \uparrow & & \uparrow & & \uparrow & & \uparrow & & \uparrow & & \uparrow & & \uparrow & & \uparrow & & \uparrow \\
& q_1 & & q_3 & & q_{20} & & q_9 & & q_{17} & & q_9 & & q_6 & & q_{35} & & q_{13}
\end{array}
$$

**FIGURE 1.71**
Example showing state $q_9$ repeating when $M$ reads $s$

We now divide $s$ into the three pieces $x$, $y$, and $z$. Piece $x$ is the part of $s$ appearing before $q_9$, piece $y$ is the part between the two appearances of $q_9$, and piece $z$ is the remaining part of $s$, coming after the second occurrence of $q_9$. So $x$ takes $M$ from the state $q_1$ to $q_9$, $y$ takes $M$ from $q_9$ back to $q_9$, and $z$ takes $M$ from $q_9$ to the accept state $q_{13}$, as shown in the following figure.

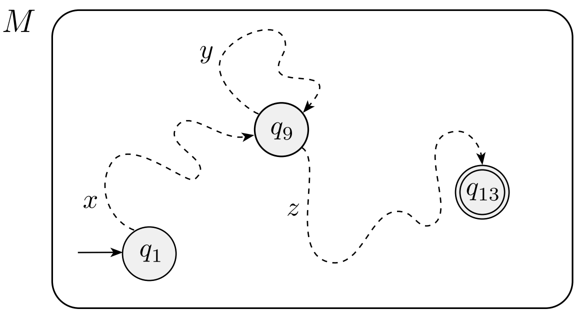


**FIGURE 1.72**
Example showing how the strings $x$, $y$, and $z$ affect $M$

Let's see why this division of $s$ satisfies the three conditions. Suppose that we run $M$ on input $xyyz$. We know that $x$ takes $M$ from $q_1$ to $q_9$, and then the first $y$ takes it from $q_9$ back to $q_9$, as does the second $y$, and then $z$ takes it to $q_{13}$. With $q_{13}$ being an accept state, $M$ accepts input $xyyz$. Similarly, it will accept $xy^iz$ for any $i > 0$. For the case $i = 0$, $xy^iz = xz$, which is accepted for similar reasons. That establishes condition 1.

Checking condition 2, we see that $|y| > 0$, as it was the part of $s$ that occurred between two different occurrences of state $q_9$.

In order to get condition 3, we make sure that $q_9$ is the first repetition in the sequence. By the pigeonhole principle, the first $p+1$ states in the sequence must contain a repetition. Therefore, $|xy| \leq p$.

**PROOF** Let $M = (Q, \Sigma, \delta, q_1, F)$ be a DFA recognizing $A$ and $p$ be the number of states of $M$.

Let $s = s_1s_2 \cdots s_n$ be a string in $A$ of length $n$, where $n \geq p$. Let $r_1, \ldots, r_{n+1}$ be the sequence of states that $M$ enters while processing $s$, so $r_{i+1} = \delta(r_i, s_i)$ for $1 \leq i \leq n$. This sequence has length $n + 1$, which is at least $p + 1$. Among the first $p + 1$ elements in the sequence, two must be the same state, by the pigeonhole principle. We call the first of these $r_j$ and the second $r_l$. Because $r_l$ occurs among the first $p+1$ places in a sequence starting at $r_1$, we have $l \leq p+1$. Now let $x = s_1 \cdots s_{j-1}$, $y = s_j \cdots s_{l-1}$, and $z = s_l \cdots s_n$.

As $x$ takes $M$ from $r_1$ to $r_j$, $y$ takes $M$ from $r_j$ to $r_j$, and $z$ takes $M$ from $r_j$ to $r_{n+1}$, which is an accept state, $M$ must accept $xy^iz$ for $i \geq 0$. We know that $j \neq l$, so $|y| > 0$; and $l \leq p+1$, so $|xy| \leq p$. Thus we have satisfied all conditions of the pumping lemma.

To use the pumping lemma to prove that a language $B$ is not regular, first assume that $B$ is regular in order to obtain a contradiction. Then use the pumping lemma to guarantee the existence of a pumping length $p$ such that all strings of length $p$ or greater in $B$ can be pumped. Next, find a string $s$ in $B$ that has length $p$ or greater but that cannot be pumped. Finally, demonstrate that $s$ cannot be pumped by considering all ways of dividing $s$ into $x$, $y$, and $z$ (taking condition 3 of the pumping lemma into account if convenient) and, for each such division, finding a value $i$ where $xy^iz \notin B$. This final step often involves grouping the various ways of dividing $s$ into several cases and analyzing them individually. The existence of $s$ contradicts the pumping lemma if $B$ were regular. Hence $B$ cannot be regular.

Finding $s$ sometimes takes a bit of creative thinking. You may need to hunt through several candidates for $s$ before you discover one that works. Try members of $B$ that seem to exhibit the "essence" of $B$'s nonregularity. We further discuss the task of finding $s$ in some of the following examples.

## EXAMPLE 1.73

Let $B$ be the language $\{\texttt{0}^n\texttt{1}^n | \, n \geq 0\}$. We use the pumping lemma to prove that $B$ is not regular. The proof is by contradiction.

Assume to the contrary that $B$ is regular. Let $p$ be the pumping length given by the pumping lemma. Choose $s$ to be the string $\texttt{0}^p\texttt{1}^p$. Because $s$ is a member of $B$ and $s$ has length more than $p$, the pumping lemma guarantees that $s$ can be split into three pieces, $s = xyz$, where for any $i \geq 0$ the string $xy^iz$ is in $B$. We consider three cases to show that this result is impossible.

1. The string $y$ consists only of 0s. In this case, the string $xyyz$ has more 0s than 1s and so is not a member of $B$, violating condition 1 of the pumping lemma. This case is a contradiction.
2. The string $y$ consists only of 1s. This case also gives a contradiction.
3. The string $y$ consists of both 0s and 1s. In this case, the string $xyyz$ may have the same number of 0s and 1s, but they will be out of order with some 1s before 0s. Hence it is not a member of $B$, which is a contradiction.

Thus a contradiction is unavoidable if we make the assumption that $B$ is regular, so $B$ is not regular. Note that we can simplify this argument by applying condition 3 of the pumping lemma to eliminate cases 2 and 3.

In this example, finding the string $s$ was easy because any string in $B$ of length $p$ or more would work. In the next two examples, some choices for $s$ do not work so additional care is required. ■

## EXAMPLE 1.74

Let $C = \{w | \, w \text{ has an equal number of 0s and 1s}\}$. We use the pumping lemma to prove that $C$ is not regular. The proof is by contradiction.

Assume to the contrary that $C$ is regular. Let $p$ be the pumping length given by the pumping lemma. As in Example 1.73, let $s$ be the string $\texttt{0}^p\texttt{1}^p$. With $s$ being a member of $C$ and having length more than $p$, the pumping lemma guarantees that $s$ can be split into three pieces, $s = xyz$, where for any $i \geq 0$ the string $xy^iz$ is in $C$. We would like to show that this outcome is impossible. But wait, it *is* possible! If we let $x$ and $z$ be the empty string and $y$ be the string $\texttt{0}^p\texttt{1}^p$, then $xy^iz$ always has an equal number of 0s and 1s and hence is in $C$. So it *seems* that $s$ can be pumped.

Here condition 3 in the pumping lemma is useful. It stipulates that when pumping $s$, it must be divided so that $|xy| \leq p$. That restriction on the way that $s$ may be divided makes it easier to show that the string $s = \texttt{0}^p\texttt{1}^p$ we selected cannot be pumped. If $|xy| \leq p$, then $y$ must consist only of 0s, so $xyyz \notin C$. Therefore, $s$ cannot be pumped. That gives us the desired contradiction.

Selecting the string $s$ in this example required more care than in Example 1.73. If we had chosen $s = (\texttt{01})^p$ instead, we would have run into trouble because we need a string that *cannot* be pumped and that string *can* be pumped, even taking condition 3 into account. Can you see how to pump it? One way to do so sets $x = \varepsilon$, $y = \texttt{01}$, and $z = (\texttt{01})^{p-1}$. Then $xy^iz \in C$ for every value of $i$. If you fail on your first attempt to find a string that cannot be pumped, don't despair. Try another one!

An alternative method of proving that $C$ is nonregular follows from our knowledge that $B$ is nonregular. If $C$ were regular, $C \cap \texttt{0}^*\texttt{1}^*$ also would be regular. The reasons are that the language $\texttt{0}^*\texttt{1}^*$ is regular and that the class of regular languages is closed under intersection, which we proved in footnote 3 (page 46). But $C \cap \texttt{0}^*\texttt{1}^*$ equals $B$, and we know that $B$ is nonregular from Example 1.73. ■

**EXAMPLE 1.75** ..................................................................

Let $F = \{ww \mid w \in \{\texttt{0,1}\}^*\}$. We show that $F$ is nonregular, using the pumping lemma.

Assume to the contrary that $F$ is regular. Let $p$ be the pumping length given by the pumping lemma. Let $s$ be the string $\texttt{0}^p\texttt{1}\texttt{0}^p\texttt{1}$. Because $s$ is a member of $F$ and $s$ has length more than $p$, the pumping lemma guarantees that $s$ can be split into three pieces, $s = xyz$, satisfying the three conditions of the lemma. We show that this outcome is impossible.

Condition 3 is once again crucial because without it we could pump $s$ if we let $x$ and $z$ be the empty string. With condition 3 the proof follows because $y$ must consist only of 0s, so $xyyz \notin F$.

Observe that we chose $s = \texttt{0}^p\texttt{1}\texttt{0}^p\texttt{1}$ to be a string that exhibits the "essence" of the nonregularity of $F$, as opposed to, say, the string $\texttt{0}^p\texttt{0}^p$. Even though $\texttt{0}^p\texttt{0}^p$ is a member of $F$, it fails to demonstrate a contradiction because it can be pumped. ■

## EXAMPLE 1.76

Here we demonstrate a nonregular unary language. Let $D = \{1^{n^2} \mid n \geq 0\}$. In other words, $D$ contains all strings of 1s whose length is a perfect square. We use the pumping lemma to prove that $D$ is not regular. The proof is by contradiction.

Assume to the contrary that $D$ is regular. Let $p$ be the pumping length given by the pumping lemma. Let $s$ be the string $1^{p^2}$. Because $s$ is a member of $D$ and $s$ has length at least $p$, the pumping lemma guarantees that $s$ can be split into three pieces, $s = xyz$, where for any $i \geq 0$ the string $xy^iz$ is in $D$. As in the preceding examples, we show that this outcome is impossible. Doing so in this case requires a little thought about the sequence of perfect squares:

$$0, 1, 4, 9, 16, 25, 36, 49, \ldots$$

Note the growing gap between successive members of this sequence. Large members of this sequence cannot be near each other.

Now consider the two strings $xyz$ and $xy^2z$. These strings differ from each other by a single repetition of $y$, and consequently their lengths differ by the length of $y$. By condition 3 of the pumping lemma, $|xy| \leq p$ and thus $|y| \leq p$. We have $|xyz| = p^2$ and so $|xy^2z| \leq p^2 + p$. But $p^2 + p < p^2 + 2p + 1 = (p+1)^2$. Moreover, condition 2 implies that $y$ is not the empty string and so $|xy^2z| > p^2$. Therefore, the length of $xy^2z$ lies strictly between the consecutive perfect squares $p^2$ and $(p+1)^2$. Hence this length cannot be a perfect square itself. So we arrive at the contradiction $xy^2z \notin D$ and conclude that $D$ is not regular. ■

## EXAMPLE 1.77

Sometimes "pumping down" is useful when we apply the pumping lemma. We use the pumping lemma to show that $E = \{0^i1^j \mid i > j\}$ is not regular. The proof is by contradiction.

Assume that $E$ is regular. Let $p$ be the pumping length for $E$ given by the pumping lemma. Let $s = 0^{p+1}1^p$. Then $s$ can be split into $xyz$, satisfying the conditions of the pumping lemma. By condition 3, $y$ consists only of 0s. Let's examine the string $xyyz$ to see whether it can be in $E$. Adding an extra copy of $y$ increases the number of 0s. But, $E$ contains all strings in $0^*1^*$ that have more 0s than 1s, so increasing the number of 0s will still give a string in $E$. No contradiction occurs. We need to try something else.

The pumping lemma states that $xy^iz \in E$ even when $i = 0$, so let's consider the string $xy^0z = xz$. Removing string $y$ decreases the number of 0s in $s$. Recall that $s$ has just one more 0 than 1. Therefore, $xz$ cannot have more 0s than 1s, so it cannot be a member of $E$. Thus we obtain a contradiction. ■