1 Nondeterministic Finite Automata, Part 3

We turn now to a method for demonstrating that some languages are not regular. For example, as we will see \( L = \{ a^n b^n | n \geq 1 \} \) is not a regular language. Intuition suggests that to recognize \( L \) we would need to count the number of \( a \)'s in the input string; in turn, this suggests that any automata recognizing \( L \) would need an unbounded number of vertices. But the question is how do we turn this into a convincing argument?

We use a proof by contradiction. So suppose for a contradiction that \( L \) were regular. Then there must be a DFA \( M \) that accepts \( L \). \( M \) has some number \( k \) of vertices. Let’s consider feeding \( M \) the input \( a^k b^k \). Look at the sequence of \( k + 1 \) vertices \( M \) goes through on reading \( a^k \): \( r_0 = \text{start}, r_1, r_2, \ldots, r_k \), where some of the \( r_i \)'s may be a repeated instance of the same vertex. This is illustrated in Figure 1.

![Figure 1: The vertices traversed on input \( a^k \).](image)

As there are only \( k \) distinct vertices, there is at least one vertex that is visited twice in this sequence, \( r_i = r_j \), say, for some pair \( i, j \), \( 0 \leq i < j \leq k \). This is shown in Figure 2. In fact we see that \( r_{j+1} = r_{h+1} \) if \( j + 1 \leq k \), \( r_{j+2} = r_{h+2} \) if \( j + 2 \leq k \), and so forth. But all we will need for our result is the presence of one loop in the path, so we will stick with the representation in the figure above.

It is helpful to partition the input into four pieces: \( a^i, a^{j-i}, a^{k-j}, b^k \). The first \( a^i \) takes \( M \) from vertex \( r_0 \) to \( r_i \), the next \( a^{j-i} \) takes \( M \) from \( r_i \) to \( r_j \), the final \( a^{k-j} \) takes \( M \) from \( r_j \) to \( r_k \), and \( b^k \) takes \( M \) from \( r_k \) to a final vertex, as shown in figure 3. What happens on input \( a^i a^{j-i} a^{k-j} b^k = a^{k+j-i} b^k \)? The initial \( a^i \) takes \( M \) from \( r_0 \) to \( r_i \), the first \( a^{j-i} \) takes \( M \) from \( r_i \) to \( r_j = r_i \), the second \( a^{j-i} \) takes \( M \) from \( r_i \) to \( r_j \), the \( a^{k-j} \) takes \( M \) from \( r_j \) to \( r_k \), and the \( b^k \) takes \( M \) from \( r_k \) to a final vertex. So
Figure 2: The path traversed on input $a^k$.

Figure 3: The path traversed on input $a^k$.

$M$ accepts $a^{k+j-i}b^k$, which is not in $L$ as $j - i > 0$. This is a contradiction, and thus the initial assumption, that $L$ was regular, must be incorrect.

We now formalize the above approach in the following lemma.

**Lemma 1** (Pumping Lemma) Let $L$ be a regular language. Then there is a number $p = p_L$, the pumping length for $L$, with the property that for each string $s$ in $L$ of length at least $p$, $s$ can be written as the concatenation of 3 substrings $x$, $y$, and $z$, that is $s = xyz$, and these substrings satisfy the following conditions.

1. $|y| > 0$,
2. $|xy| \leq p$,
3. For each integer $i \geq 0$, $xy^iz \in L$.

**Definition 2** We say $s$ can be pumped if $s \in L$ can be written as the concatenation of substrings $x$, $y$ and $z$, and conditions 1–3 of the Pumping Lemma hold.

**Proof:** Again, we use a proof by contradiction. So suppose that $L$ were regular and let $M$ be a DFA accepting $L$. Now suppose that $M$ has $p$ states. $p$, the number of states in $M$, will be the pumping length for $L$. 
Let $s$ be a string in $L$ of length $n \geq p$. Write $s$ as $s = s_1 s_2 \cdots s_n$, where each $s_i$ is a character in $\Sigma$, the alphabet for $L$. Consider the substring $s' = s_1 s_2 \cdots s_p$. We look at the path $M$ follows on input $s'$. It must go through $p + 1$ vertices, and as $M$ has only $p$ vertices, at least one vertex must be repeated. Let $\text{start} = r_0, r_1, \cdots, r_p$ be this sequence of vertices and suppose that $r_i = r_j$, where $0 \leq i < j \leq p$, is a repeated vertex, as shown in Figure 4. Let $x$ denote $s_1 \cdots s_i$, $y$ denote $s_i+1 \cdots s_j$, and $z$ denote $s_{j+1} \cdots s_{p}s_{p+1} \cdots s_n$. The path traversed on input $s = xyz$ is illustrated in Figure 5. As $j > i$, $|y| = j - i > 0$. Also $|xy| = |s_1 \cdots s_j| = j \leq p$. So (1) and (2) are true.

Clearly, $xz$ is recognized by $M$, for $x$ takes $M$ from $r_0$ to $r_i = r_j$ and $z$ takes $M$ from $r_j$ to a final vertex.

Similarly $xy'z$ is recognized by $M$ for any $i \geq 1$, for $x$ takes $M$ from $r_0$ to $r_i$, each repetition of $y$ takes $M$ from $r_i$ (back) to $r_j = r_i$, and then the $z$ takes $M$ from $r_j$ to a final vertex. Thus (3) is also true, proving the result. 

Now, to show that a language $L$ is non-regular we use the pumping Lemma in the following way. We begin by assuming $L$ is regular so as to obtain a contradiction. Next, we assert that there is a pumping length $p$ such that for each string $s$ in $L$ of length at least $p$ the three conditions of the Pumping Lemma hold. The next (and more substantial) task is to choose a particular string $s$ to which we will apply the
conditions of the Pumping Lemma, and condition (3) in particular, so as to obtain a contradiction.

**Example 3** Let us look at the language \( L = \{a^n b^n | n \geq 1\} \) again. The argument showing that \( L \) is not regular proceeds as follows.

**Step 1.** Suppose, so as to obtain a contradiction, that \( L \) were regular. Then \( L \) must have a pumping length \( p \geq 1 \) such that for any string \( s \in L \) with \(|s| \geq p\), \( s \) can be pumped.

**Step 2.** Choose \( s \). Recall that we chose the string \( s = a^p b^p \).

**Step 3.** By pumping \( s \), obtain a contradiction.

As \( s \) can be pumped (for \( s \in L \) and \(|s| = 2p \geq p\)), we can write \( s = xyz \), with \(|y| > 0\), \(|xy| \leq p\), and \( xy^iz \in L \) for all integer \( i \geq 0\).

As the first \( p \) characters of \( s \) are all \( a \)'s, the substring \( xy \) must also be a string of all \( a \)'s. By condition (3), with \( i = 0 \), we have that \( xz \in L \); but the string \( xz \) has removed \(|y| \) \( a \)'s from \( s \), that is \( xz = a^{p-|y|} b^p \), and this string is not in \( L \) since \( p - |y| \neq p \). This is a contradiction for we have shown that \( xz \in L \) and \( xz \notin L \).

**Step 4.** Consequently the initial assumption is incorrect; that is, \( L \) cannot be recognized by a DFA, so \( L \) is not regular.

Let me stress the sequence in which the argument goes. First the existence of pumping length \( p \) for \( L \) is asserted (different regular languages \( L \) may have different pumping lengths, but each regular language with arbitrarily long strings will have a pumping length). Then a suitable string \( s \) is chosen. The length of \( s \) will be a function of \( p \). \( s \) is chosen so that when it is pumped a string outside of \( L \) is obtained. An important point about the pumped substring \( y \) is that while we know \( y \) occurs among the first \( p \) characters of \( s \), we do not know exactly which ones form the substring \( y \). Consequently, a contradiction must arise for every possible substring \( y \) of the first \( p \) characters in order to show that it is not regular.

There is a distinction here: for a given value of \( p \), \( s \) is fully determined; e.g. for \( p = 3 \), \( s = aaabbb \). By contrast, all that we know about \( x \) and \( y \) is that together they contain between 1 and 3 characters, and that \( y \) has at least one character. There are 6 possibilities in all for the pair \((x, y)\), namely: \((\lambda, a)\), \((\lambda, aa)\), \((\lambda, aaa)\), \((a, a)\), \((a, aa)\), \((aa, a)\). In general, for \(|xy| \leq p\), there are \( \frac{1}{2}p(p + 1) \) choices of \( x \) and \( y \). The argument leading to a contradiction must work for every possible choice of \( p \) and every possible partition of \( s \) into \( x \), \( y \), and \( z \). Of course, you are not going to give a separate argument for each case given that there are infinitely many cases. Rather, the argument must work regardless of the value of \( p \) and regardless of which partition of \( s \) is being considered.

Next, we show an alternative Step 3 for Example 3.

**Alternative Step 3.** In applying Condition 3, use \( i = 2 \) (instead of \( i = 0 \)), giving \( xyyz \in L \). But \( xyyz \) adds \(|y| \) \( a \)'s to \( xyz \), namely \( xyyz = a^{p+|y|} b^p \) and this string is
not in \( L \) since \( p - |y| \neq p \). This is a contradiction for we have shown that \( xyyz \in L \) and \( xyyz \notin L \).

In Example 3 both pumping down (\( i = 0 \)) and pumping up (\( i = 2 \)) will yield a contradiction. This is not the case in every example. Sometimes only one direction works, and then only for the right choice of \( s \).

A common mistake. Not infrequently, an attempted solution may try to specify how \( s \) is partitioned into \( x, y, \) and \( z \). In Example 3, this might take the form of stating that \( x = \lambda, y = a^p, \) and \( z = b^p \), and then obtaining a contradiction for this partition. This is an incomplete argument, however. All that the Pumping Lemma states is that there is a partition; it does not tell you what the partition is. The argument showing a contradiction must work for every possible partition.

**Example 4** Let \( K = \{ww^R \mid w \in \{a, b\}^*\} \). For example, \( \lambda, abba, abaaba \in L, \) \( ab, a, aba \notin L \). We show that \( K \) is not regular.

**Step 1.** Suppose, so as to obtain a contradiction, that \( L \) were regular. Then \( L \) must have a pumping length \( p \geq 1 \) such that for any string \( s \in L \) with \( |s| \geq p \), \( s \) can be pumped.

**Step 2.** Choose \( s \). We choose the string \( s = a^pbba^p \).

**Step 3.** By pumping \( s \), obtain a contradiction.

As \( s \) can be pumped (for \( |s| = 2p + 2 \geq p \)), we can write \( s = xyz \), with \( |y| > 0, |xy| \leq p, \) and \( xy^iz \in L \) for all integer \( i \geq 0 \).

As the first \( p \) characters of \( s \) are all \( a \)'s, the substring \( xy \) must also be a string of all \( a \)'s. By condition (3), with \( i = 0 \), we have that \( xz \in L \); but the string \( xz \) has removed \( |y| \) \( a \)'s from \( s \), that is \( xz = a^{p-|y|}bba^p \), and this string is not in \( L \) since \( p - |y| \neq p \). This is a contradiction for we have shown that \( xz \in L \) and \( xz \notin L \).

**Step 4.** Consequently the initial assumption is incorrect; that is, \( L \) cannot be recognized by a DFA, so \( L \) is not regular.

Another common mistake. I have seen attempted solutions for the above example that set \( s = w^p(w^R)^p \). This is not a legitimate definition of \( s \). For \( w \) is an arbitrary string, so such a specification does not determine what is the string \( s \). To effectively apply the Pumping Lemma you are going to need to choose an \( s \), in which only \( p \) is left unspecified. So if you are told, for example, that \( p = 3 \), then you must be able to write down \( s \) as a specific string of characters (in Example 4, with \( p = 3, s = aaabbbaaa \)).

For the assumption in the application of the Pumping Lemma is that the language \( L \) under consideration is recognized by a DFA. What is not known is the assumed size of the DFA. What the argument leading to a contradiction shows is that regardless of its size the supposed DFA cannot recognize \( L \), but this requires the argument to work for any value of \( p \).
Example 5  \( J = \{ w \mid w \text{ has equal numbers of } a \text{'s and } b \text{'s} \} \). We show that \( J \) is not regular. To do this we introduce a new technique. Note that if \( A \) and \( B \) are regular then so is \( A \cap B = \overline{(A \cup B)} \) (for the union and complement of regular languages are themselves regular; alternatively, see Problem 2 in Problem Collection 2).

Now note that \( R = \{ a^i b^j \mid i, j \geq 0 \} \) is regular. Thus if \( J \) is regular, then so is \( J \cap R = \{ a^i b^i \mid i \geq 0 \} \). But we already showed \( J \cap R \) was not regular in Example 3. Thus \( J \) cannot be regular either. (Strictly, this is a proof by contradiction.)

We could also proceed as in Example 3, applying the Pumping Lemma to string \( s = a^p b^p \). The exact same argument will work.

Question  Does the proof of the Pumping Lemma work if we consider a \( p \)-state NFA that accepts \( L \), rather than a \( p \)-state DFA? Justify your answer.