Lecture 5

Proving languages to be nonregular

We already know, for any alphabet Σ, that there exist languages $A \subseteq \Sigma^*$ that are nonregular. This is because there are *uncountably* many languages over Σ but only *countably* many regular languages over Σ. However, this observation does not allow us to conclude that specific nonregular languages are indeed nonregular. In this lecture we will discuss a method that can be used to prove that a fairly wide selection of languages are nonregular.

5.1 The pumping lemma for regular languages

We will begin by proving a simple fact—known as the *pumping lemma*—which establishes that a certain property must hold for all regular languages. Later in the lecture, in the section following this one, we will use this fact to conclude that certain languages are nonregular.

Lemma 5.1 (Pumping lemma for regular languages). *Let Σ be an alphabet and let $A \subseteq \Sigma^*$ be a regular language. There exists a positive integer n (called a* pumping length *of A) that possesses the following property. For every string $w \in A$ with $|w| \geq n$, it is possible to write $w = xyz$ for some choice of strings $x, y, z \in \Sigma^*$ such that*

1. *$y \neq \varepsilon$,*
2. *$|xy| \leq n$, and*
3. *$xy^iz \in A$ for all $i \in \mathbb{N}$.*

The pumping lemma is essentially a precise, technical way of expressing one simple consequence of the following fact:

> If a DFA with n or fewer states reads n or more symbols from an input string, at least one of its states must have been visited more than once.

This means that if a DFA with n states reads a particular string having length at least n, then there must be a substring of that input string that causes a loop, meaning that the DFA starts and ends on the same state. If the DFA accepts the original string, then by repeating that substring that caused a loop multiple times, or alternatively removing it altogether, we obtain a different string that is also accepted by the DFA. It may be helpful to try to match this intuition to the proof that follows.

Proof of Lemma 5.1. Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA that recognizes A and let $n = |Q|$ be the number of states of M. We will prove that the property stated in the pumping lemma is satisfied for this choice of n.

Let us note first that if there is no string contained in A that has length n or larger, then there is nothing more we need to do: the property stated in the lemma is trivially satisfied in this case. We may therefore move on to the case in which A does contain at least one string having length at least n. In particular, suppose that $w \in A$ is a string such that $|w| \geq n$. We may write

$$w = a_1 \cdots a_m \tag{5.1}$$

for $m = |w|$ and $a_1, \ldots, a_m \in \Sigma$. Because $w \in A$ it must be the case that M accepts w, and therefore there exist states

$$r_0, r_1, \ldots, r_m \in Q \tag{5.2}$$

such that $r_0 = q_0$, $r_m \in F$, and

$$r_{k+1} = \delta(r_k, a_{k+1}) \tag{5.3}$$

for every $k \in \{0, \ldots, m-1\}$.

Now, the sequence $r_0, r_1, \ldots, r_n$ has $n+1$ members, but there are only n different elements in Q, so at least one of the states of Q must appear more than once in this sequence.[1] Thus, there must exist indices $s, t \in \{0, \ldots, n\}$ satisfying $s < t$ such that $r_s = r_t$.

Next, define strings $x, y, z \in \Sigma^*$ as follows:

$$x = a_1 \cdots a_s, \qquad y = a_{s+1} \cdots a_t, \qquad z = a_{t+1} \cdots a_m. \tag{5.4}$$

It is the case that $w = xyz$ for this choice of strings, so to complete the proof, we just need to demonstrate that these strings fulfill the three conditions that are listed in the lemma. The first two conditions are immediate: we see that y has length $t - s$, which is at least 1 because $s < t$, and therefore $y \neq \varepsilon$; and we see that xy has

[1] This is an example of the so-called *pigeon hole principle*: if $n+1$ pigeons fly into n holes, then at least one of the holes must contain two or more pigeons.

length t, which is at most n because t was chosen from the set $\{0, \ldots, n\}$. It remains to verify that $xy^iz \in A$, which is equivalent to M accepting xy^iz, for every $i \in \mathbb{N}$. That fact that xy^iz is accepted by M follows from the verification that the sequence of states

$$r_0, \ldots, r_s, \underbrace{r_{s+1}, \ldots, r_t}_{\text{repeated } i \text{ times}}, r_{t+1}, \ldots, r_m \tag{5.5}$$

satisfies the definition of acceptance of the string xy^iz by the DFA M. □

The pumping lemma for an example DFA

If the proof of the pumping lemma, or the idea behind it, is not clear, it may be helpful to see it in action for an actual DFA and a long enough string accepted by that DFA. For instance, let us take M to be the DFA having the state diagram illustrated in Figure 5.1.

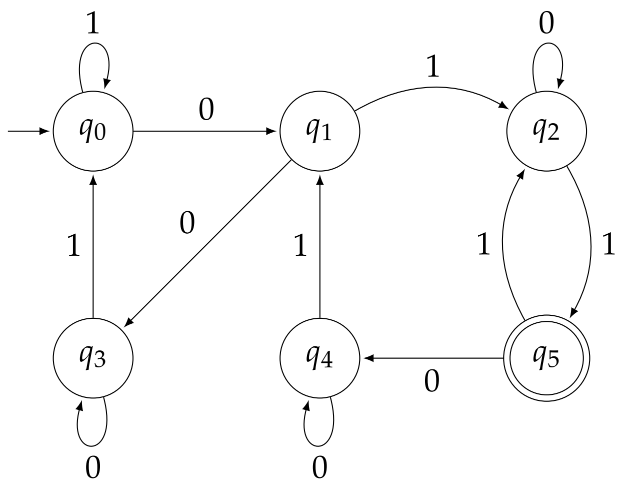

Figure 5.1: The state diagram of a DFA M, to be used to provide an example to explain the pumping lemma.

Now consider any string w having length at least 6 (which is the number of states of M) that is accepted by M. For instance, let us take $w = 0110111$. This causes M to move through this sequence of states:

$$q_0 \xrightarrow{0} q_1 \xrightarrow{1} q_2 \xrightarrow{1} q_5 \xrightarrow{0} q_4 \xrightarrow{1} q_1 \xrightarrow{1} q_2 \xrightarrow{1} q_5 \tag{5.6}$$

(The arrows represent the transitions and the symbols above the arrows indicate which input symbol has caused this transition.) Sure enough, there is at least one state that appears multiple times in the sequence, and in this particular case there are actually three such states: q_1, q_2, and q_5, each of which appear twice. Let us focus on the two appearances of the state q_1, just because this state happens to be the one that gets revisited first. It is the substring 1101 that causes M to move in a loop starting and ending on the state q_1. In the statement of the pumping lemma this corresponds to taking

$$x = 0, \qquad y = 1101, \qquad \text{and} \qquad z = 11. \tag{5.7}$$

Because the substring y causes M to move from the state q_1 back to q_1, it is as if reading y when M is in the state q_1 has no effect. So, given that x causes M to move from the initial state q_0 to the state q_1, and z causes M to move from q_1 to an accept state, we see that M must not only accept $w = xyz$, but it must also accept xz, $xyyz$, $xyyyz$, and so on.

There is nothing special about the example just described; something similar always happens. Pick any DFA whatsoever, and then pick any string accepted by that DFA that has length at least the number of states of the DFA, and you will be able to find a loop like we did above. By repeating input symbols in the most natural way so that the loop is followed multiple times (or no times) you will obtain different strings accepted by the DFA. This is essentially all that the pumping lemma is saying.

5.2 Using the pumping lemma to prove nonregularity

It is helpful to keep in mind that the pumping lemma is a statement about regular languages: it establishes a property that must always hold for every chosen regular language.

Although the pumping lemma is a statement about regular languages, we can use it to prove that certain languages are *not* regular using the technique of *proof by contradiction*. In particular, we take the following steps:

1. For A being the language we hope to prove is nonregular, we make the assumption that A *is* regular. Operating under the assumption that the language A is regular, we apply the pumping lemma to it.
2. Using the property that the pumping lemma establishes for A, we derive a contradiction. The contradiction will almost always be that we conclude that some particular string is contained in A that we know is actually not contained in A.
3. Having derived a contradiction, we conclude that it was our assumption that A is regular that led to the contradiction, and so we deduce that A is nonregular.

Examples of nonregularity proved through the pumping lemma

Let us illustrate this method for a few example languages. These examples will be stated as propositions, with the proofs showing you how the argument works.

Proposition 5.2. *Let $\Sigma = \{0, 1\}$ be the binary alphabet and define a language over Σ as follows:*

$$\text{SAME} = \{0^m 1^m : m \in \mathbb{N}\}. \tag{5.8}$$

The language SAME *is not regular.*

Remark 5.3. Whenever we give a language a special name like this, it is to be understood that this language is so defined for the remainder of the course (although reminders will often appear). It is a good idea to remember the languages that are given special names—they will serve well as examples. Two additional named languages will appear in this lecture.

Proof. Assume toward contradiction that SAME is regular. By the pumping lemma for regular languages, there must exist a pumping length $n \geq 1$ for SAME for which the property stated by that lemma holds. We will fix such a pumping length n for the remainder of the proof.

Define $w = 0^n 1^n$ (where n is the pumping length we just fixed). It is the case that $w \in \text{SAME}$ and $|w| = 2n \geq n$, so the pumping lemma tells us that there exist strings $x, y, z \in \Sigma^*$ so that $w = xyz$ and the following conditions hold:

1. $y \neq \varepsilon$,
2. $|xy| \leq n$, and
3. $xy^i z \in \text{SAME}$ for all $i \in \mathbb{N}$.

Now, because $xyz = 0^n 1^n$ and $|xy| \leq n$, the substring y cannot have any 1s in it, as the substring xy is not long enough to reach the 1s in xyz. This means that $y = 0^k$ for some choice of $k \in \mathbb{N}$, and because $y \neq \varepsilon$, we conclude moreover that $k \geq 1$. We may also conclude that

$$xy^2 z = xyyz = 0^{n+k} 1^n. \tag{5.9}$$

This is because $xyyz$ is the string obtained by inserting $y = 0^k$ somewhere in the initial portion of the string $xyz = 0^n 1^n$, before any 1s have appeared. More generally it holds that

$$xy^i z = 0^{n+(i-1)k} 1^n \tag{5.10}$$

for each $i \in \mathbb{N}$. (We do not actually need this more general formula for the sake of the current proof, but in other similar cases a formula like this can be helpful.)

However, because $k \geq 1$, we see that the string $xy^2z = 0^{n+k}1^n$ is *not* contained in SAME. This contradicts the third condition stated by the pumping lemma, which guarantees us that $xy^iz \in \text{SAME}$ for all $i \in \mathbb{N}$.

Having obtained a contradiction, we conclude that our assumption that SAME is regular was wrong. The language SAME is therefore nonregular, as required. □

Proposition 5.4. *Let $\Sigma = \{0,1\}$ be the binary alphabet and define a language over Σ as follows:*

$$A = \big\{0^m1^r \,:\, m, r \in \mathbb{N},\; m > r\big\}. \tag{5.11}$$

The language A is not regular.

Proof. Assume toward contradiction that A is regular. By the pumping lemma for regular languages, there must exist a pumping length $n \geq 1$ for A for which the property stated by that lemma holds. We will fix such a pumping length n for the remainder of the proof.

Define $w = 0^{n+1}1^n$. We see that $w \in A$ and $|w| = 2n + 1 \geq n$, so the pumping lemma tells us that there exist strings $x, y, z \in \Sigma^*$ so that $w = xyz$ and the following conditions are satisfied:

1. $y \neq \varepsilon$,
2. $|xy| \leq n$, and
3. $xy^iz \in A$ for all $i \in \mathbb{N}$.

Now, because $xyz = 0^{n+1}1^n$ and $|xy| \leq n$, it must be that $y = 0^k$ for some choice of $k \geq 1$. (The reasoning here is just like in the previous proposition.) This time we have

$$xy^iz = 0^{n+1+(i-1)k}1^n \tag{5.12}$$

for each $i \in \mathbb{N}$. In particular, if we choose $i = 0$, then we have

$$xy^0z = xz = 0^{n+1-k}1^n. \tag{5.13}$$

However, because $k \geq 1$, and therefore $n + 1 - k \leq n$, we see that the string xy^0z is *not* contained in A. This contradicts the third condition stated by the pumping lemma, which guarantees us that $xy^iz \in A$ for all $i \in \mathbb{N}$.

Having obtained a contradiction, we conclude that our assumption that A is regular was wrong. The language A is therefore nonregular, as required. □

Remark 5.5. In the previous proof, it was important that we could choose $i = 0$ to get a contradiction—no other choice of i would have worked.

Proposition 5.6. *Let $\Sigma = \{0\}$ and define a language over Σ as follows:*

$$\text{SQUARE} = \left\{ 0^{m^2} \,:\, m \in \mathbb{N} \right\}. \tag{5.14}$$

The language SQUARE *is not regular.*

Proof. Assume toward contradiction that SQUARE is regular. By the pumping lemma for regular languages, there exists a pumping length $n \geq 1$ for SQUARE for which the property stated by that lemma holds. We will fix such a pumping length n for the remainder of the proof.

Define $w = 0^{n^2}$. We observe that $w \in$ SQUARE and $|w| = n^2 \geq n$, so the pumping lemma tells us that there exist strings $x, y, z \in \Sigma^*$ so that $w = xyz$ and the following conditions are satisfied:

1. $y \neq \varepsilon$,
2. $|xy| \leq n$, and
3. $xy^iz \in$ SQUARE for all $i \in \mathbb{N}$.

There is only one symbol in the alphabet Σ, so this time it is immediate that $y = 0^k$ for some choice of $k \in \mathbb{N}$. Because $y \neq \varepsilon$ and $|y| \leq |xy| \leq n$, it must be the case that $1 \leq k \leq n$, and therefore

$$xy^iz = 0^{n^2+(i-1)k} \tag{5.15}$$

for each $i \in \mathbb{N}$. In particular, if we choose $i = 2$, then we have

$$xy^2z = xyyz = 0^{n^2+k}. \tag{5.16}$$

However, because $1 \leq k \leq n$, it cannot be that $n^2 + k$ is a perfect square; the number $n^2 + k$ is larger than n^2, but the next perfect square after n^2 is

$$(n+1)^2 = n^2 + 2n + 1, \tag{5.17}$$

which is strictly larger than $n^2 + k$ because $k \leq n$. The string xy^2z is therefore *not* contained in SQUARE, which contradicts the third condition stated by the pumping lemma, which guarantees us that $xy^iz \in$ SQUARE for all $i \in \mathbb{N}$.

Having obtained a contradiction, we conclude that the assumption of SQUARE being regular was wrong. The language SQUARE is therefore nonregular. □

In advance of the next example, let us introduce some notation that will be useful from time to time throughout the course. For a given string w, the string w^{R} denotes the *reverse* of the string w. Formally speaking, assuming w is a string over an alphabet Σ, we may define the string reversal operation inductively as follows:

1. $\varepsilon^{\mathrm{R}} = \varepsilon$, and
2. $(aw)^{\mathrm{R}} = w^{\mathrm{R}}a$ for every $w \in \Sigma^*$ and $a \in \Sigma$.

Let us also define the language

$$\mathrm{PAL} = \{w \in \Sigma^* : w = w^{\mathrm{R}}\} \tag{5.18}$$

over the binary alphabet $\Sigma = \{0,1\}$. This language is named PAL because it is short for *palindrome*, which (as you may know) is something that reads the same forward and backward.

Proposition 5.7. *The language* PAL *is not regular.*

Proof. Assume toward contradiction that PAL is regular. By the pumping lemma for regular languages, there must exist a pumping length $n \geq 1$ for PAL for which the property stated by that lemma holds. We will fix such a pumping length n for the remainder of the proof.

Define $w = 0^n 1 0^n$. We observe that $w \in \mathrm{PAL}$ and $|w| = 2n + 1 \geq n$, so the pumping lemma tells us that there exist strings $x, y, z \in \Sigma^*$ so that $w = xyz$ and the following conditions are satisfied:

1. $y \neq \varepsilon$,
2. $|xy| \leq n$, and
3. $xy^i z \in \mathrm{PAL}$ for all $i \in \mathbb{N}$.

Once again, we may conclude that $y = 0^k$ for $k \geq 1$. This time it is the case that

$$xy^i z = 0^{n+(i-1)k} 1 0^n \tag{5.19}$$

for each $i \in \mathbb{N}$. In particular, if we choose $i = 2$, then we have

$$xy^2 z = xyyz = 0^{n+k} 1 0^n. \tag{5.20}$$

Because $k \geq 1$, this string is not equal to its own reverse, and therefore xy^2z is therefore *not* contained in PAL. This contradicts the third condition stated by the pumping lemma, which guarantees us that $xy^i z \in \mathrm{PAL}$ for all $i \in \mathbb{N}$.

Having obtained a contradiction, we conclude that our assumption that PAL is regular was wrong. The language PAL is therefore nonregular, as required. □

The four propositions above should give you an idea of how the pumping lemma can be used to prove languages are nonregular. The set-up is always the same: we assume toward contradiction that a particular language is regular, and observe that the pumping lemma gives us a pumping length n. At that point it is time to choose the string w, try to use some reasoning, and derive a contradiction. It may not always be clear what string w to choose or how exactly to get a contradiction; these steps will depend on the language you are working with, there may be multiple good choices for w, and there may be some creativity and/or insight involved in getting it all to work.

If you do not know what string w to choose, take a guess and aim for a contradiction. If you do not succeed, you may find that you have gained some intuition on what a better choice might be. Of course, you should thoroughly be convinced by your own arguments and actively look for ways they might be going wrong; if you do not truly believe your own proof, it is not likely anyone else will believe it either.

5.3 Nonregularity from closure properties

Sometimes we can prove that a particular language is nonregular by combining together closure properties for regular languages our knowledge of other languages being nonregular. Here are two examples, again stated as propositions.

Proposition 5.8. *Let $\Sigma = \{0, 1\}$ and define a language over Σ as follows:*

$$B = \{w \in \Sigma^* : w \neq w^{\mathrm{R}}\}. \tag{5.21}$$

The language B is not regular.

Proof. Assume toward contradiction that B is regular. The regular languages are closed under complementation, and therefore $\overline{B}$ is regular. However, $\overline{B} = \mathrm{PAL}$, which we already proved is nonregular. This is a contradiction, and therefore our assumption that B is regular was wrong. We conclude that B is nonregular, as claimed. □

Proposition 5.9. *Let $\Sigma = \{0, 1\}$ and define a language over Σ as follows:*

$$C = \{w \in \Sigma^* : w \text{ has more 0s than 1s}\}. \tag{5.22}$$

The language C is not regular.

Proof. Assume toward contradiction that C is regular. We know that the language $\mathrm{L}(0^*1^*)$ is regular because it is the language matched by a regular expression. The regular languages are closed under intersection, so $C \cap \mathrm{L}(0^*1^*)$ is regular. However, we have that

$$C \cap \mathrm{L}(0^*1^*) = A, \tag{5.23}$$

the language defined in Proposition 5.4, which we already proved is nonregular. This is a contradiction, and therefore our assumption that C is regular was wrong. We conclude that C is nonregular. □

It is important to remember, when using this method, that it is the regular languages that are closed under operations such as intersection, union, and so on, not the nonregular languages. For instance, it is not always the case that the intersection of two nonregular languages is nonregular—so a proof would not be valid if it were to rely on such a claim.