DISCLAIMER: These notes are not necessarily an accurate representation of what I said during the class. They are mostly what I intend to say, and have not been carefully edited.

• Today’s topics:
  1. Pumping lemma for regular languages
  2. Non-regular languages
  3. Quiz 4
  4. Proof of pumping lemma
  5. More nonregular languages

Given a regular language, we now know a method to prove that it is regular - simply build a finite automaton (DFA or NFA) that accepts the language. But how do we prove that a language is non-regular?

Consider the following example language: \( B = \{0^n1^n \mid n \geq 0\} \). Can we build a DFA/NFA for it? It might require somehow keeping track of the number of 0s seen (which could be arbitrarily large). It seems that might not be possible using a finite number of states. But can we prove definitively that there cannot be a DFA/NFA accepting \( B \)?

## 1 The pumping lemma

**Theorem 1.1** (Pumping Lemma for Regular Languages). *If \( A \) is a regular language, then there is a number \( p \) (the pumping length) where, if \( s \) is a string in \( A \) and \( |s| \geq p \), 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. \( |xy| \leq p \).
2 Proof of nonregularity using the lemma

Let’s go back to our example language \( B = \{0^n1^n \mid n \geq 0 \} \). We will now use the pumping lemma to prove that \( B \) is not regular.

Assume for the sake of contradiction, that \( B \) is regular. Then, let \( p \) be the pumping length given by the lemma. Pick a string \( s = 0^p1^p \). Since \( s \in B \) and \( |s| \geq p \), the pumping lemma implies that \( s \) can be divided into three pieces \( s = xyz \), such that \( xy^i z \in B \) for all \( i \geq 0 \). Let us consider the ways in which \( s \) cannot be divided into \( x, y \) and \( z \):

1. Suppose \( y \) has only 0s. Then \( xy^2z \) has more 0s than 1s (note that it cannot be length 0 from condition 2 of the lemma), so it cannot be in \( B \).

2. The case where \( y \) has only 1s follows similarly.

3. Suppose \( y \) has both 0s and 1s. Then, in \( xy^2z \), some 1s appear before 0s and so it is not in \( B \).

Hence, there is no division of \( s \) which satisfies the properties of the lemma. This must mean that our initial assumption that \( B \) is regular was false.

Things to note

- You cannot choose \( p \) - you have to assume this is given by the lemma.
- You can choose the string \( s \).
- You cannot choose one particular division of \( s \). You have to prove that no possible division can satisfy the three conditions.

3 Proof of the pumping lemma

We applied the lemma as a black box, but let us see why it is true.

Proof of Theorem 1.1. Since \( A \) is regular, we can construct a DFA \( M \) recognizing it. Let this be \( M = (Q, \Sigma, \delta, q_1, F) \). Let us assign the pumping length to be \( p = |Q| + 1 \). Now, let \( s \) be any string accepted by \( M \) of length at least \( p \). (What if there are no such strings? Then the lemma is true by default!)

Consider the set of states visited by \( M \) on input \( s \). Since \( s \) has at least \( |Q| + 1 \) characters, at least one state must be visited twice. Let \( q_i \) be the first such repeated state. The sequences of states visited look like:

\[ q_1, \ldots, q_i, \ldots, q_i, \ldots, q_F \]

Let \( x \) denote the first part of \( s \) which the machine reads before reaching \( q_i \) the first time. Let \( y \) denote the part of \( s \) reads between \( q_i \) and returning to \( q_i \). Let \( z \) be the rest of
the string. Now, it is easy to see that replacing \( y \) with \( y_i \) for any \( i \geq 0 \) will also lead the machine to reach \( q_F \) (condition 1). \( y \) cannot be empty since is is between two separate occurrences of a state (condition 2). Lastly, since \( q_i \) is the first repetition of a state, the number of characters read until then is at most \( |Q| + 1 = p \) (condition 3).

\[ \square \]

**Note:** The converse of the pumping lemma is not true! That is, a language satisfying the lemma may still be non-regular.

### 4 A few more examples

**Example 1: pumping up**

Let \( C \) be the language \( \{1^{n^2} \mid n \geq 0\} \). \( C \) is a unary language, which only accepts strings whose lengths are perfect squares. Let’s use the pumping lemma to prove that \( C \) is not regular.

Assume for the sake of contradiction, that \( C \) is regular. Then, let \( p \) be the pumping length given by the lemma. Pick a string \( s = 1^{p^2} \). Since \( s \in C \) and \( |s| \geq p \), the pumping lemma implies that \( s \) can be divided into three pieces \( s = xyz \), such that \( xy^iz \in D \) for all \( i \geq 0 \). So, we have that \( |xy^iz| \) is always a perfect square.

From condition 3, we know that \( |xy| \leq p \implies |y| \leq p \). Then, \( |xy^2z| = |xyz| + |y| \leq p^2 + p < (p + 1)^2 \). Hence, the only way \( |xy^2z| \) can be a perfect square is if \( |y| = 0 \), which contradicts condition 2 of the lemma.

**Example 2: pumping down**

Let \( D \) be the language \( \{0^i1^j \mid i > j\} \). Let’s use the pumping lemma to prove that \( D \) is not regular.

Assume for the sake of contradiction, that \( D \) is regular. Then, let \( p \) be the pumping length given by the lemma. Pick a string \( s = 0^{p+1}1^p \). Since \( s \in C \) and \( |s| \geq p \), the pumping lemma implies that \( s \) can be divided into three pieces \( s = xyz \), such that \( xy^iz \in D \) for all \( i \geq 0 \).

From condition 3, we know that \( |xy| \leq p \implies y \) has only 0s. Then, \( xy^0z = xz \) has equal or lesser 0s than 1s, which contradicts condition 1 of the lemma.