Non-regular Languages and Closure Properties

Previously:
Introduced the Pumping Lemma to show some languages are not regular.

Today:
Other approaches to show that languages are regular or non-regular.

Proving a language is non-regular without the pumping lemma

The pumping lemma isn't the only way we can prove a language is non-regular.

Other techniques:
- Show that the desired DFA would require infinite states to model the intended language
- Use closure properties to relate to other non-regular languages

Proof strategy: Infinite states
Consider the language \( \{ a^i b^i \mid i \geq 0 \} \) and a DFA to recognize it.

For any \( i \), let \( a_i \) be the state entered after processing \( a^i \), i.e.,
\[
\hat{\delta}(q_0, a^i) = a_i,
\]
Consider any \( i \) and \( j \) such that \( i \neq j \).
\[
\hat{\delta}(q_0, a^i b^i) \neq \hat{\delta}(q_0, a^j b^i) \text{ since the former is accepting and the latter is rejecting.}
\]
\[
\hat{\delta}(q_0, a^i b^i) = \delta(\delta(q_0, a^i), b^i) = \delta(a_i, b^i), \text{ by definition of } \hat{\delta} \text{ and definition of } a_i, \text{ respectively}
\]
\[
\hat{\delta}(q_0, a^j b^i) = \delta(\delta(q_0, a^j), b^i) = \delta(a_j, b^i), \text{ by the same reasoning}
\]

Since inputs \( a^i \) and \( a^j \) lead to different states on the same input, the states must be different: \( a_i \neq a_j \).

Since \( i \) and \( j \) were arbitrary, and since there are an infinite number of ways to pick them, there must be an infinite number of states.

Thus there is no DFA to recognize this language, and the language is non-regular.

Proof strategy: Closure properties

Certain operations on regular languages are guaranteed to produce regular languages.

Closure properties can be used to prove a language is regular – or that it’s non-regular.
Regular languages are closed under common set operations

Regular operations (used to construct REs):
- **Union**: \( L_1 \cup L_2 \)
- **Concatenation**: \( L_1L_2 \)
- **Star-closure**: \( L_1^* \)

The complement of a language

Given a language \( L \subseteq \Sigma^* \), the **complement** of that language (denoted \( \overline{L} \)) is the language of all strings in \( \Sigma^* \) not in \( L \).

Formally,
\[
\overline{L} = \{ w \mid w \in \Sigma^* \land w \notin L \}
\]

The complement of a language

A regular language is a language accepted by some DFA.

If \( L \) is a regular language, we can show \( \overline{L} \) is a regular language by constructing a DFA for it.
Complementing regular languages

A regular language is a language accepted by some DFA.

If $L$ is a regular language, we can show $\overline{L}$ is a regular language by constructing a DFA for it.

_Therefore, the regular languages are closed under complementation._

Regular languages are closed under common set operations

Regular operations (used to construct REs):

- **Union:** $L_1 \cup L_2$
- **Concatenation:** $L_1L_2$
- **Star-closure:** $L_1^*$

Additional set operations:

- **Complementation:** $\overline{L_1}$

Intersection of two languages

If $L_1$ and $L_2$ are languages over $\Sigma$, then $L_1 \cap L_2$ is the language of strings in both $L_1$ and $L_2$.

If $L_1$ and $L_2$ are both regular, is $L_1 \cap L_2$ regular?
Intersection of two languages

If $L_1$ and $L_2$ are languages over $\Sigma$, then $L_1 \cap L_2$ is the language of strings in both $L_1$ and $L_2$.

If $L_1$ and $L_2$ are both regular, is $L_1 \cap L_2$ regular?

De Morgan’s laws!

$L_1 \cup L_2$

Intersection of two languages

If $L_1$ and $L_2$ are languages over $\Sigma$, then $L_1 \cap L_2$ is the language of strings in both $L_1$ and $L_2$.

If $L_1$ and $L_2$ are both regular, is $L_1 \cap L_2$ regular?

Regular languages are closed under common set operations

Regular operations (used to construct REs):

- Union: $L_1 \cup L_2$
- Concatenation: $L_1L_2$
- Star-closure: $L_1^*$

Additional set operations:

- Complementation: $\overline{L_1}$
- Intersection: $L_1 \cap L_2$
Other closures

**Difference:** If \( L_1 \) and \( L_2 \) are regular, then \( L_1 - L_2 \) is also regular.

**Proof:**
Set difference is defined as 
\[ L_1 - L_2 = L_1 \cap \overline{L_2} \]
We know that if \( L_2 \) is regular, so is \( \overline{L_2} \). We also know regular languages are closed under intersection. Therefore, we know that \( L_1 \cap \overline{L_2} \) is regular.

Difference is sometimes notated as \( L_1 \setminus L_2 \)

Other closures

**Reversal:** If \( L \) is regular, then \( L^R \) is also regular.

**Proof sketch:**
Suppose \( L \) is a regular language. We can therefore construct an NFA with a single final state that accepts \( L \).
We can then make the start state of this NFA the final state, make the final state the start state, and reverse the direction of all arcs in the NFA.
The modified NFA accepts a string \( w^R \) if and only if the original NFA accepts \( w \).
Therefore the modified NFA accepts \( L^R \).

Using regular language closure properties

**To show a language is regular:**
Show that by using two or more known regular languages and one or more of the operations over which regular languages are closed, you can produce that language.

Basic template to show regularity

\[
L_{REG_1} \ <OP> \ L_{REG_2} = L_{REG_3}
\]

where \(<OP>\) is one of the operations over which regular languages are closed.

- \( L_{REG_3} \) is the language we need to prove is regular
- \( L_{REG_1} \) and \( L_{REG_2} \) are known regular languages

If the two languages on the left side of the operator are regular, then so too must be the one on the right side.
Example

Prove the language \( \{a^n b^m \mid n > 3 \text{ or } m > 3 \} \) is regular.

We can show that this language can be produced using regular language closure properties on known regular languages

\[
\begin{align*}
L_1 &= \{a^* b^* \} \\
L_2 &= \{\varepsilon, a, aa, aaa\} \\
L_3 &= \{\varepsilon, b, bb, bbb\}
\end{align*}
\]

as follows:

- **concatenation:** \( L_4 = L_2 \cdot L_3 \)
- **complementation:** \( L_5 = L_4 \)
- **intersection:** \( L_6 = L_5 \cap L_1 = \{a^n b^m \mid n > 3 \text{ or } m > 3\} \)

Why isn't this the language we need?

Example

Prove the language \( \{a^n b^m \mid n > 3 \text{ or } m > 3 \} \) is regular.

We can show that this language can be produced using regular language closure properties on known regular languages

\[
\begin{align*}
L_1 &= \{a^* b^* \} \\
L_2 &= \{\varepsilon, a, aa, aaa\} \\
L_3 &= \{\varepsilon, b, bb, bbb\}
\end{align*}
\]

as follows:

- **concatenation:** \( L_4 = L_2 \cdot L_3 \) (\( L_2 \) and \( L_3 \) contain the empty string, so when we concatenate them, the result includes \( L_2 \) and \( L_3 \))
- **complementation:** \( L_5 = L_4 \) (\( \Sigma^* - L_2 - L_3 - \{ab, abb, \ldots\} \))
- **intersection:** \( L_6 = L_5 \cap L_1 \) (\( \text{Force order to be } a^* b^* \)) \[ = \{a^n b^m \mid n > 3 \text{ or } m > 3\} \]

Example

Prove \( \{a^n b^m \mid n > 3 \text{ and } m > 3\} \) is regular.

Show that this language can be produced using regular language closure properties on known regular languages

\[
\begin{align*}
L_1 &= \{a^* b^* \} \\
L_2 &= \{\varepsilon, a, aa, aaa\} \\
L_3 &= \{\varepsilon, b, bb, bbb\}
\end{align*}
\]

Step 1: Things to remove:

- **concatenation:** \( L_4 = a^* L_3 \)
- **concatenation:** \( L_5 = L_2 b^* \)
- **union:** \( L_6 = L_4 \cup L_5 \)

Example

Prove \( \{a^n b^m \mid n > 3 \text{ and } m > 3\} \) is regular.

Show that this language can be produced using regular language closure properties on known regular languages

\[
\begin{align*}
L_1 &= \{a^* b^* \} \\
L_2 &= \{\varepsilon, a, aa, aaa\} \\
L_3 &= \{\varepsilon, b, bb, bbb\}
\end{align*}
\]

Step 1: Things to remove:

- **concatenation:** \( L_4 = a^* L_3 \)
- **concatenation:** \( L_5 = L_2 b^* \)
- **union:** \( L_6 = L_4 \cup L_5 \)
Example

Prove \( \{a^n b^m | n > 3 \text{ and } m > 3 \} \) is regular.

Show that this language can be produced using regular language closure properties on known regular languages:

\begin{align*}
L_1 &= \{a^n b^m | n > 3 \text{ and } m > 3 \} \\
L_2 &= \{a^n b^m | n > 3 \text{ and } m > 3 \} \\
L_3 &= \{a^n b^m | n > 3 \text{ and } m > 3 \} \\
concatenation: & \quad L_0 = a^n L_3 \\
concatenation: & \quad L_5 = L_2 b^m \\
union: & \quad L_4 = L_0 \cup L_2 \\
complementation: & \quad L_5 = \overline{L_4} \\
\end{align*}

Step 2: Remove them from the set of all strings.

Basic template to show regularity

\[ L_{\text{REG1}} \quad <\text{OP}> \quad L_{\text{REG2}} = L_{\text{REG3}} \]

We cannot assume that if the language on the right is regular, the languages on the left must be too!

Consider \( \{a, aa\} - \{a b\} = \{a, aa\} \)

\begin{align*}
\uparrow & \quad \uparrow & \quad \uparrow \\
\text{Regular} & \quad \text{Not regular!} & \quad \text{Still regular} \\
\text{because it's finite} & & \\
\end{align*}

Using regular language closure properties

To show a language is not regular:

Use the same template:

\[ L_{\text{REG1}} \quad <\text{OP}> \quad L_{\text{REG2}} = L_{\text{REG3}} \]

However the language in question is plugged into the template in the position of \( L_{\text{REG1}} \).

We want to use a known regular language for \( L_{\text{REG2}} \).

If we can use a known non-regular language as \( L_{\text{REG3}} \), then it must be the case that \( L_{\text{REG1}} \) is not regular.
Example

Show $L = \{w \mid w \text{ in } \{a,b\}^* \text{ and } w \text{ has equal number of } a \text{ and } b\}$ is non-regular.

- Instantiate the template: $L \cap \{a^*b^n \mid n \geq 0\}$
- If both languages on the left side are regular, then language on the right side is regular (due to closure of regular languages over intersection).
- $\{a^nb^n \mid n \geq 0\}$ is easily proved non-regular using the Pumping Lemma.
- We know $\{a^*b^n\}$ is regular.
- Therefore $L$ must be non-regular.

Using closure properties with Pumping Lemma

Show that $L = \{w \mid w \text{ has more instances of } a \text{ than } b\}$ is non-regular.

- Intersect with $b^*a^*$. This becomes language of strings $b^p a^p$ where $p > m$.
- Consider string $b^{np}a^{n+1}$ where $n$ is the Pumping Lemma number.
- If $L$ is regular, the conditions of the Pumping Lemma hold: $xy$ spans at most the first $n$ $b$s and $y$ consists of 1 to $n$ $b$s.
- However, regardless of how long $y$ is, when pumped it adds at least one $b$ and the resulting string contains at least as many $b$s as $a$s so is not in $L$.

Divide and conquer

$L = \{w \in \{a,b\}^* \mid w \text{ contains an even number of } a \text{ and an odd number of } b \text{ and all } a \text{ come in runs of three}\}$

Regular or non-regular?

Regular: $L = L_1 \cap L_2$, where
- $L_1 = \{w \in \{a,b\}^* \mid w \text{ contains an even number of } a \text{ and an odd number of } b\}$ and
- $L_2 = \{w \in \{a,b\}^* \mid \text{all } a \text{ come in runs of three}\}$
- To prove it, build an FA for each
  - Easier than FA for the original language

Because we can build a finite automaton for each of these languages, they are both regular.
- We get $L$ by intersecting these two languages.
- Regular languages are closed under intersection.
  
  Therefore, $L$ is regular!
What the closure theorem for union does not say

Closure theorem for union says: If $L_1$ and $L_2$ are regular, then $L = L_1 \cup L_2$ is regular.

What happens if (for example) $L$ is regular? Does that mean that $L_1$ and $L_2$ are also?

*Maybe.*

Example

We know $a^*$ is regular.

Consider two cases for $L_1$ and $L_2$:

- $a^* = \{a^n \mid n > 0 \text{ and } n \text{ is prime}\} \cup \{a^n \mid n > 0 \text{ and } n \text{ is not prime}\}$
  - $a^* = L_1 \cup L_2$
  - *Neither $L_1$ nor $L_2$ is regular!*
- $a^* = \{a^n \mid n > 0 \text{ and } n \text{ is even}\} \cup \{a^n \mid n > 0 \text{ and } n \text{ is odd}\}$
  - $a^* = L_1 \cup L_2$
  - *Both $L_1$ and $L_2$ are regular!*

What the closure theorem for concatenation does not say

Closure theorem for concatenation says: If $L_1$ and $L_2$ are regular, then $L = L_1 L_2$ is regular.

What happens (for example) if $L_2$ is not regular? Does that mean that $L$ isn’t regular?

*Maybe.*

Consider two examples:

- $\{ aba^n b^n \mid n \geq 0 \} = \{ab\} \cdot \{a^n b^n \mid n \geq 0\}$
  - $L = L_1 \cdot L_2$
  - *Neither $L$ nor $L_2$ is regular!*
- $\{aaa^*\} = \{a^*\} \cdot \{a^n \mid n \text{ is prime}\}$
  - $L = L_1 \cdot L_2$
  - $L_2$ is not regular, but $L$ is!
True or false?

If $L_1 \subseteq L_2$ and $L_1$ is not regular, then $L_2$ is not regular.

False! $\{a,b\}^*$ is regular, and it has a non-regular subset $\{a^n b^n \mid n \geq 0\}$

If $L_1 \subseteq L_2$ and $L_2$ is not regular, then $L_1$ is not regular.

False! Non-regular languages have finite subsets, and finite languages are regular

Hints

When you need a known regular language in a proof, remember that $\Sigma^*$, $\{\varepsilon\}$, $\{a^*\}$, $\{a^* b^*\}$, etc. are regular.

When you need a known non-regular language, use $a^n b^n$ or any language with a similar dependency.

Acknowledgments

This lecture incorporates material from:

Nancy Ide
Keith Schwarz