Inf2A: The Pumping Lemma

Stuart Anderson

School of Informatics
University of Edinburgh

October 8, 2009
Outline

1. Deterministic Finite State Machines and Regular Languages
2. When is a Language not Regular?
3. Tools for Showing Languages are not Regular
   - The Pumping Lemma
   - Applications of the Pumping Lemma
4. Summary
The language of a DFA

\[ M = \left( Q, \Sigma, q_0, F, \delta \right) \]

- (Kozen): Define \( \hat{\delta}(q, \varepsilon) = q \) and \( \hat{\delta}(q, sa) = \delta(\hat{\delta}(q, s), a) \).
- We write \( q' \xrightarrow{x} q'' \) if and only if \( \hat{\delta}(q', x) = q'' \), i.e. if \( M \) is in state \( q' \) and reads the string \( x \in \Sigma^* \) then it will end up in state \( q'' \).
- \( M \) accepts a string \( x \in \Sigma^* \) if \( q_0 \xrightarrow{x} q \) where \( q \in F \).
- The language recognized by \( M \) is the language

\[ L(M) = \{ x \in \Sigma^* \mid M \text{ accepts } x \} \]

over the alphabet \( \Sigma \).
This is a DFA formally specified by:

\[ M = (\{0, 1, 2, 3\}, \{a, b\}, 0, \{3\}, \delta) \]

where the transition function \( \delta \) is defined by:

\[
\begin{array}{c|cc}
\delta & a & b \\
\hline
0 & 1 & 0 \\
1 & 2 & 0 \\
2 & 3 & 0 \\
3 & 3 & 3 \\
\end{array}
\]
Definition: A language $L \subseteq \Sigma^*$ is regular if there is a DFA $M$ such that $L = L(M)$.

Examples: The following languages are regular:

$$L_1 = \{ xaaay \mid x, y \in \{a, b\}^* \} \subseteq \{a, b\}^*,$$
$$L_2 = \{ x1102 \mid x \in \{0, 1, \ldots, 9\}^* \} \subseteq \{0, 1, \ldots, 9\}^*,$$
$$L_3 = \{ x \in \{0, 1\}^* \mid x \text{ contains an even number of } 0\text{s} \} \subseteq \{0, 1\}^*. $$
Which of the following languages are regular?

\[
L_4 = \{abc^*x \mid x \in \{a, b, c\}^* \} \subseteq \{a, b, c\}^*,
\]

\[
L_5 = \{\varepsilon\} \subseteq \{a, b\}^*,
\]

\[
L_6 = \\{(n)^n \mid n \in \mathbb{N}_0\},
\]

\[
L_7 = \{a^{f(n)} \mid n \in \mathbb{N}_0\}
\]

The language \(L_7\) is really a family of languages where we let \(f\) be a particular quadratic function that always has positive values.
Non-regular Languages

- To prove that a language is regular, we just have to produce a DFA accepting the language, but . . .
- . . . *how can we prove that a language is NOT regular?*
- For example, suppose we want to show that the language $\{a^{n^2} \mid n \geq 0\}$ is *not* regular, how do we do it?
The Pumping Lemma

Let

\[ M = (Q, \Sigma, q_0, F, \delta) \]

be a DFA with \( k \) states, and let \( x \in L(M) \) be a string that it accepts.

If \( |x| \geq k \) then there exist three strings \( u, v, w \in \Sigma^* \) such that the four properties below all hold:

1. \( uvw = x \),
2. \( |v| \geq 1 \),
3. \( |uv| \leq k \),
4. for every \( i \in \mathbb{N}_0 \): \( uv^i w \in L(M) \).
Proof of the Pumping Lemma

- Suppose $q_0 \xrightarrow{x} q_F$ where $q_F \in F$.
- $|x| \geq k, |Q| = k \implies$ There is a state $q \in Q$ that occurs twice in the first $k$ steps of the computation.
- Thus we can write

$$q_0 \xrightarrow{u} q \xrightarrow{v} q \xrightarrow{w} q_F$$

for suitable $u, v, w \in \Sigma^*$ satisfying (1)–(3).
Proof: concluded

But then we also have

\[ q_0 \xrightarrow{u} q \xrightarrow{w} q_F \]
\[ q_0 \xrightarrow{u} q \xrightarrow{v} q \xrightarrow{v} q \xrightarrow{w} q_F \]
\[ q_0 \xrightarrow{u} q \xrightarrow{v} q \xrightarrow{v} q \xrightarrow{v} q \xrightarrow{w} q_F \]
\[ \ldots \]

In other words, for every \( i \geq 0 \) we have

\[ q_0 \xrightarrow{u} q \xrightarrow{v^i} q \xrightarrow{w} q_F. \text{ Thus } M \text{ accepts } uv^i w. \]
**Theorem:** The language $L = \{0^n1^n \mid n \in \mathbb{N}_0\} \subseteq \{0, 1\}^*$ is not regular.

**Proof:** Suppose for contradiction that $L = L(M)$ for an automaton $M$ with $k$ states.

1. Pumping Lemma applied to $x = 0^k1^k \in L$ yields $u, v, w \in \{0, 1\}^*$ satisfying (i)–(iv).

2. (iii) $\implies$ $u, v$ only consist of 0s.

3. Thus (ii) $\implies$ $uv^2w \notin L$.

4. But (iv) $\implies$ $uv^2w \in L(M)$ — contradiction!
Applications of the Pumping Lemma II

**Theorem:** The language

\[ L = \{ x \in \{ (, ) \}^* \mid \text{the parenthesis in } x \text{ are well balanced} \} \]

is not regular.

**Proof:** Suppose for contradiction that \( L = L(M) \) for an automaton \( M \) with \( k \) states.

1. Pumping Lemma applied to \( x = (k)^k \in L \) yields \( u, v, w \in \{ (, ) \}^* \) satisfying (i)–(iv).
2. (iii) \( \implies u, v \) only consist of ‘(’s.
3. Thus (ii) \( \implies uv^2w \notin L. \)
4. But (iv) \( \implies uv^2w \in L(M) \) — contradiction!
Theorem: The language \( \text{JAVA} \subseteq \text{ASCII}^* \) consisting of all syntactically correct JAVA programs is not regular.

Hint for the proof:
Can you create a sequence of well-formed JAVA programs that could demonstrate the language is not regular?
Begin by *assuming that the language is regular*. We use the Pumping Lemma to reach a contradiction from this assumption.

Because $L$ is assumed to be regular, there must be some DFA $M$ that recognises it. Write $k$ for the number of states in $M$.

Choose some string $x$ in $L$ with $|x| \geq k$.

Apply the Pumping Lemma to $x$. The Pumping Lemma breaks $x$ up into suitable $u$, $v$ and $w$.

Choose $i \in \mathbb{N}_0$ so that $uv^iw \not\in L$.

This contradicts property (iv) of the Pumping Lemma, which guarantees that $uv^iw \in L(M)$, since $L = L(M)$.

Having reached the sought contradiction, conclude that the initial assumption (that $L$ is regular) is flawed.
Applications of the Pumping Lemma IV

Theorem: The language $L = \{a^p \mid p \text{ is a prime number}\} \subseteq \{a\}^*$ is not regular.

Proof: Suppose for contradiction that $L = L(M)$ for an automaton $M$ with $k$ states.

1. Pumping Lemma applied to $x = a^p$, where $p$ is a prime number bigger than $k$, yields $u, v, w \in \{a\}^*$ satisfying (i)–(iv).

2. Let $x' = uv^{p+1}w$. Then (iv) $\implies x' \in L(M)$.

3. Let $l = |u|$ and $m = |v|$, so that $|w| = p - m - l$ with $m \geq 1$ and $l + m \leq k$. Now $|x'| = l + m(p + 1) + (p - m - l) = (m + 1)p$, which is not a prime number.

4. Thus $x' \notin L$ — contradiction!
To show a language is regular we just need to exhibit a machine that recognises the language.

The pumping lemma shows that all sufficiently long strings in regular languages have some substring that can be iterated to generate more members of the language.

We can use proof by contradiction together with the Pumping Lemma to demonstrate a language is not regular.