Languages that are Regular and Languages that are not

November 7, 2011
CS 81: Computability & Logic
Closure Properties

A family of languages is a set of languages.

✓ The family of all finite languages
✓ The family of all languages
✓ The family of all regular languages

A family $F$ is closed under an operation if applying the operation to languages in $F$ always produces a result in $F$. 
Finite Languages

Is the family of finite languages closed under:

- ✔ Union? \( (A \cup B) \)
- ✔ Intersection \( (A \cap B) \)
- ✔ Concatenation? \( (AB) \)
- ✔ Star \( (A^*) \)
- ✔ Complement \( (A^c) \)
Regular Languages

The regular languages are closed under

✓ Union? \((A \cup B)\)
✓ Intersection \((A \cap B)\)
✓ Concatenation? \((AB)\)
✓ Star \((A^*)\)
✓ Complement \((A^c)\)

Proofs?
COMPLEMENT?
**Complement!**

\[ \text{DFA } M = (\Sigma, K, \delta, q_0, A) \]

\[ \text{DFA } M^c = (\Sigma, K, b, c, q_0, \bar{A}) \]
INTERSECTION: DFA INPUTS

\[ A \xrightarrow{a} X \xrightarrow{a} B \]

\[ L \xrightarrow{a,b} K \xrightarrow{a} \]

\[ b \]

\[ X \]

\[ a \]

\[ b \]

\[ a,b \]

\[ b \]

\[ a \]

\[ b \]
**Intersection: Product Automaton**

\[ \text{DFA } M = (\Sigma, K, \rightarrow, q_0, A) \]
\[ \text{DFA } M' = (\Sigma, K', \rightarrow', q'_0, A') \]
\[ \text{DFA } M \cap M' = (\Sigma, K \times K', \rightarrow_{\text{both}}, \langle q_0, q'_0 \rangle, A \times A'). \]
State Machine Optimization

If two states have the same language, they can be merged without changing the language of the state machine.
Example

![Diagram of a finite automaton with states q0, q1, q2, q3, q4, q5, and transitions labeled with 'a'.]

- q0 transitions to q1, q3, and q5 on 'a'.
- q1 transitions to q2, q5, and q4 on 'a'.
- q2 transitions to q3, q4, and q0 on 'a'.
- q3 transitions to q4 and q0 on 'a'.
- q4 transitions to q5 and q1 on 'a'.
- q5 transitions to q2 and q3 on 'a'.

Closure Properties
Minimization
Myhill-Nerode
Maze Theory
Pumping Lemma
One DFA Minimization Algorithm

Assume all states are mergable unless there’s evidence otherwise:

✓ Accepting vs. nonaccepting.
✓ Same-symbol transitions to known-different states.
More Complex Example
**Bigger Example**
**Derivatives of a Language**

For any language $L$ and $x \in \Sigma^*$, define

$$\partial_x L := \{ y \in \Sigma^* | xy \in L \}$$

In terms of a state machine for $L$, the set $\partial_x L$ contains strings that will be accepted after you’ve already seen $x$.

So, if you run $x$ through a state machine for $L$, you end up in a state whose language is $\partial_x L$.

If our state machine is minimal, we should have exactly one state whose language is $\partial_x L$. 
Consequence

Theorem (Myhill-Nerode (essentially))

A language is regular iff \( \{ \partial_x L \mid x \in \Sigma^* \} \) is finite.

Proof idea: The size of this set is the size of the smallest deterministic state machine.

Consider

1. \( L := \{ a^{3n} \mid n \geq 0 \} \)
   \[ \partial_{\epsilon} L = \partial_{aaa} L = \cdots, \quad \partial_a L = \partial_{aaa} L = \cdots, \quad \partial_{aa} L = \partial_{aaaa} L = \cdots \]

2. \( L := \{ 0^n1^n \mid n \geq 0 \} \)
   \[ \partial_{\epsilon} L \neq \partial_0 L \neq \partial_{00} L \neq \partial_{000} L \neq \cdots \]
Maze Theory

✓ Suppose you are in a maze of twisty little passages, all alike. (but with doors that open from only one side)

✓ You happen to know that your maze has exactly 19 rooms. You enter the maze and wander through 27 rooms. What can you conclude?

✓ This wandering has brought you to an exit. What can you conclude about other solutions to the maze?

✓ Was there anything special about the numbers 19 and 27?
Finite Maze Theorem

For every finite maze there is a number $p$, such that

For every path through the maze $s$ with $|s| \geq p$:

- The path $s$ contains at least one loop, which starts and ends within the first $p$ steps.
- There are infinitely many paths through the maze (at least one shorter, and arbitrarily many longer) whose lengths differ by a multiple of some constant.
Finite Automata As Mazes
A Pumping Lemma

If \( L \) is a regular language, then there exists a number \( p \) such that for every \( s \in L \) with \( |s| \geq p \), we can decompose \( s \) into \( xyz \) where

1. \( y \neq \epsilon \)
2. \( |xy| \leq p \)
3. \( xy^iz \in L \) for every \( i \geq 0 \).
Deriving a Useful Corollary

The Pumping Lemma tells us that:

If \( L \) is regular,
then every long-enough string in \( L \) can be pumped.

What logically follows?

✓ If every long string in \( L \) can be pumped, then \( L \) is regular
✓ If there’s a long string in \( L \) that can be pumped, then \( L \) is regular
✓ If not every long string in \( L \) can be pumped, then \( L \) isn’t regular!
✓ If there’s a long string in \( L \) that can’t be pumped, then \( L \) isn’t regular!
Using the Pumping Lemma

To prove a language isn’t regular:
✓ Suppose \( L \) were regular, with pumping length \( p \)
  ▶ Carefully pick a long (\( \geq p \)) string \( s \in L \)
  ▶ Show that \( s \) cannot be pumped

✓ Contradiction. Therefore, \( L \) is not regular.

You cannot use it to prove a language is regular!
✓ E.g., non-regular languages with every string pumpable

\[
\{a^i b^j c^j | i \geq 1, j \geq 0\} \cup \{b^j c^k | j, k \geq 0\} \quad p = 1
\]
Let $L = \{ 0^n1^n \mid n \geq 0 \}$

Suppose $L$ were regular

✓ Let $p$ be the pumping length

✓ Consider, for example, $s := 0^p1^p$. (Note that $|s| \geq p$.)

✓ Consider all possible decompositions

$$s = xyz \text{ with } y \neq \epsilon \land |xy| \leq p.$$ 

✓ None of them work for pumping.

✓ Contradiction.

So $L$ is not regular. QED.
**Prove nonregular**

✓ $L = \{ w \in \{0, 1\}^* \mid w \text{ has as many 0's as 1's} \}$

✓ $L = \{ ww \mid w \in \{0, 1\} \}$

✓ $L = \{ 0^i 1^j \mid i < j \}$

✓ $L = \{ 0^i 1^j \mid i > j \}$