CSE 105
THEORY OF COMPUTATION

"Winter" 2018

http://cseweb.ucsd.edu/classes/wi18/cse105-ab/
Today's learning goals

• Determine if a language is regular
• Apply closure properties to conclude that a language is or isn't regular
• Prove closure properties of the class of regular languages
Regular languages

- DFA $M$ over the alphabet $\Sigma$
  - For each string $w$ over $\Sigma$, $M$ either accepts $w$ or rejects $w$
  - The **language recognized by $M$** is the set of strings $M$ accepts
    a.k.a. the **language of $M$** is the set of strings $M$ accepts
    a.k.a. $L(M) = \{ w \mid w$ is a string over $\Sigma$ and $M$ accepts $w \}$

A language is **regular** iff there is some finite automaton that recognizes **exactly** it.
Justification?

To prove that the DFA we build, $M$, actually recognizes the language $L$

\[ \text{WTS} \quad L(M) = L \]

(1) Is every string accepted by $M$ in $L$?
(2) Is every string from $L$ accepted by $M$?

or contrapositive version: Is every string rejected by $M$ not in $L$?
A useful (optional) bit of terminology

When is a string accepted by a DFA?

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

**Computation of M on w**: where do we land when start at \( q_0 \) and read each symbol of \( w \) one-at-a time?

\[ \delta^*(q, w) = \]

\[ \delta^*(q, w) = \begin{cases} q & \text{if } w = \varepsilon \\ \delta((q, a)) & \text{if } w = a \in \Sigma \\ \delta((\delta^*(q, u), a)) & \text{if } w = ua \\ \delta((\delta^*(q, u), a)) & \text{for } u \in \Sigma^* \\ \delta((\delta^*(q, u), a)) & \text{for } a \in \Sigma \\ \end{cases} \]

Recursively defined function
Regular languages: bounds?

Is every finite language regular?

A. No: some finite languages are regular, and some are not.
B. No: there are no finite regular languages.
C. Yes: every finite language is regular.
D. I don't know.
\[ \Sigma = \{a, b, c\} \]
\[ L = \{w_1, \ldots, w_n\} \]

Make accepting state accept this path from it. Each path from it to the start state corresponds to a word in \( L \).
Building DFA

Remember

States are our only (computer) memory.

Design and pick states with specific roles / tasks in mind.

"Have not seen any of desired pattern yet"
"Trap state"
Complementation

$\left( \{q_1,q_2,q_3\}, \{a,b\}, \delta, q_1, ? \right)$

$L(M) = \emptyset$ if $F=\emptyset$

Modify:

$F = \emptyset q_3 \}$

new $L(M) = \{w \}$

$ab$ is a substring of $w$

Modify again of $w$

$F' = \{q_1, q_2 \}$

how $L(M)$?

$\{b's \text{ followed by } a\} = ab$ is not a substring of $w$
Building DFA

New strategy

Express \( L \) in terms of \textit{simpler languages} – use them as building blocks.

Example

\[ L = \{ w \mid w \text{ does not contain the substring } \text{baba} \} \]
\[ = \text{the complement of the set}\]
\[ \{ w \mid w \text{ contains the substring } \text{baba} \} \]
Complementation

**Claim**: If $A$ is a regular language over $\{0,1\}^*$, then so is $\overline{A}$
aka "the class of regular languages is closed under complementation"
Complementation

Claim: If $A$ is a regular language over $\{0,1\}^*$, then so is $\overline{A}$ aka "the class of regular languages is closed under complementation"

Proof: Let $A$ be a regular language. Then there is a DFA $M = (Q, \Sigma, \delta, q_0, F)$ such that $L(M) = A$. We want to build a DFA whose language is $\overline{A}$. Define

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

Claim of Correctness $L(M') = \overline{A}$

Proof of claim…
Consider arbitrary \( w \in L(M') \)

We wish we \( A \)

Computation of \( M' \) on \( w \) is accepting, i.e. \( \delta^*((q_0, w)) \in \overline{F} \)

i.e. \( \delta^*((q_0, w)) \notin \overline{F} \) so \( w \notin L(M) \)

so \( w \in \overline{L(M)} \) but since \( L(M) = A \), \( w \notin \overline{A} = \overline{B} \)

Exercise
Why closure proofs?

- General technique of proving a new language is regular
- Stretch the power of the model
- Puzzle!
Set operations

Input set(s) $\rightarrow$ OPERATION $\rightarrow$ Output set

Complementation $\checkmark$
Kleene star
Concatenation
Union
Intersection
Set difference
The regular operations

For $A$, $B$ languages over same alphabet, define:

$$A \cup B = \{x \mid x \in A \text{ or } x \in B\}$$

$$A \circ B = \{xy \mid x \in A \text{ and } y \in B\}$$

$$A^* = \{x_1x_2\ldots x_k \mid k \geq 0 \text{ and each } x_i \in A\}$$

These are operations on sets!
Theorem: The class of regular languages over fixed alphabet $\Sigma$ is closed under the union operation.

Proof:

What are we proving here?

A. For any set $A$, if $A$ is regular then so is $A \cup A$.
B. For any sets $A$ and $B$, if $A \cup B$ is regular, then so is $A$.
C. For two DFAs $M_1$ and $M_2$, $M_1 \cup M_2$ is regular.
D. None of the above.
E. I don't know.
Theorem: The class of regular languages over fixed alphabet $\Sigma$ is closed under the union operation.

Proof: Let $A_1$, $A_2$ be any two regular languages over $\Sigma$. WTS that $A_1 \cup A_2$ is regular.

Goal: build a machine that recognizes $A_1 \cup A_2$. 
Union

Goal: build a machine that recognizes $A_1 \cup A_2$.

Strategy: use machines that recognize each of $A_1, A_2$.

Accept if either (or both) accepts

**HOW?**
"Run in parallel"

\[ M = (Q_1 \times Q_2, \Sigma, \delta, q_0, F_1 \times F_2) \]

Start state: \( (q_0, q'_0) \)

Accept state(s):

Transition function:

\[ M_1 = (Q_1, \Sigma, F_1, q_0, F_1) \]

\[ M_2 = (Q_2, \Sigma, F_2, q'_0, F_2) \]

The set of accepting states for \( M \) is

A. \( F_1 \times F_2 \)
B. \( \{ (r, s) | r \text{ is in } F_1 \text{ and } s \text{ is in } F_2 \} \)
C. \( \{ (r, s) | r \text{ is in } F_1 \text{ or } s \text{ is in } F_2 \} \)
D. \( F_1 \cup F_2 \)
E. I don't know.
Proof: Let $A_1$, $A_2$ be any two regular languages over $\Sigma$. Given $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ such that $L(M_1) = A_1$ and $M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ such that $L(M_2) = A_2$.

WTS that $A_1 \cup A_2$ is regular.

Define
$M = (Q_1 \times Q_2, \Sigma, \delta, (q_1, q_2), \{(r, s) \in Q_1 \times Q_2 \mid r \in F_1 \text{ or } s \in F_2\})$
with
$\delta((r, s), x) = (\delta_1(r, x), \delta_2(s, x))$
for $(r, s)$ in $Q_1 \times Q_2$ and $x$ in $\Sigma$.

Why does $L(M) = A_1 \cup A_2$?
Aside: Intersection

• How would you prove that the class of regular languages is closed under intersection?
• Can you think of more than one proof strategy?

\[ A \cap B = \{ x \mid x \text{ in } A \text{ and } x \text{ in } B \} \]
Payoff

\{ w \mid w \text{ contains neither the substrings} \text{ aba nor baab} \}

Is this a regular set?
Payoff

\{ w \mid \text{w contains neither the substrings aba nor baab}\}

Is this a regular set?

A = \{ w \mid \text{w contains aba as a substring}\}
B = \{ w \mid \text{w contains baab as a substring}\}

\overline{A} \cap \overline{B} = \overline{A \cup B}
General proof structure/strategy

**Theorem:** For any $L$ over $\Sigma$, if $L$ is regular then [ the result of some operation on $L$ ] is also regular.

**Proof:**

*Given* name variables for sets, machines assumed to exist.  
*WTS* state goal and outline plan.  
*Construction* using objects previously defined + new tools working towards goal. Give formal definition and explain.  
*Correctness* prove that construction works.  
*Conclusion* recap what you've proved.
The regular operations

For A, B languages over same alphabet, define:

\[ A \cup B = \{ x \mid x \in A \text{ or } x \in B \} \]

\[ A \circ B = \{ xy \mid x \in A \text{ and } y \in B \} \]

\[ A^* = \{ x_1 x_2 \ldots x_k \mid k \geq 0 \text{ and each } x_i \in A \} \]

How can we prove that the concatenation of two regular languages is a regular language?
For next time

• Work on Group Homework 1 due Saturday

Pre class-reading for Friday:
- Page 48 (Figure 1.27 and description below it)
- Example 1.35 on page 52