Oberlin College
CSCI 383
Fall 2026
Lecture 4. Closure Properties
Kozen - Section 4

Def: A language A is *regular* if there exists a DFA M such that $L(M) = A$.

A language is an algorithmic problem. Saying it's regular: It can be solved with a DFA.

Thing to ask yourself: Are all languages regular?

Warmups: Give 2-state DFAs for

- strings ending in 0
- strings with an even number of 1s

How about strings not ending in 0?

Theorem: For any regular language L, $\sim L$ is regular.

Proof sketch: Just construct a new DFA $(Q, \Sigma, \delta, s, Q \setminus F)$.

We say the set of regular languages is *closed under complement*.

Closure Properties of Regular Languages

Given an operator on f languages, does f preserve regularity?

Example operators:

- $\sim$. Given regular L, is $\sim L$ regular? (Yes, we just proved it.)
- $\cup$. Given two regular languages L_1 and L_2, is $L_1 \cup L_2$ regular?
- $\cap$. How about $L_1 \cap L_2$?
- Concatenation. How about $L_1 L_2$?
- Asterate. How about L^*?

Consider $\cap$.

If we can solve L_1 and L_2 with a DFA, can we recongize strings that are yes instances to both at once?

I.e. strings where both machines would return true.

Instead of DFAs, let's say we had Java methods `isinL1()` and `isinL2()` that recognize membership in L_1 and L_2, respectively.

How would we construct a Java method that recognizes strings in $L_1 \cap L_2$?

Write the code.

With DFAs: Run both DFAs.

Can we come up with a DFA that runs both original DFAs?

The Product Construction

Let L_1 and L_2 be regular.

That is, there are DFAs:

- $M_1 = (Q_1, \Sigma, \delta_1, s_1, F_1)$
- $M_2 = (Q_2, \Sigma, \delta_2, s_2, F_2)$

Such that $L(M_1) = L_1$ and $L(M_2) = L_2$.

Note: The alphabets are the same.

Goal: Run both DFAs simultaneously.

Read a character, take a step of each. Read a character, take a step of each. etc.

New machine $M_3 = (Q_3, \Sigma, \delta_3, s_3, F_3)$:

- $Q_3 = Q_1 \times Q_2$, where $\times$ is the *Cartesian product*.

  Ex: $Q_1 = \{1, 2, 3\}$, $Q_2 = \{\text{square}, \text{circle}, \text{triangle}\}$
  $Q_3 = \{(1, \text{square}), (1, \text{circle}), (1, \text{triangle}), (2, \text{square}), \ldots\}$
- Same alphabet.
- How about δ_3? Should simulate δ_1 and δ_2.
  $\delta_3((q_1, q_2), a) = (\delta_1(q_1, a), \delta_2(q_2, a))$.
- $s_3 = (s_1, s_2)$
- $F_3 = F_1 \times F_2$

Work out for an example.

Want to show: $L(M_3) = L_1(M_1) \cap L_2(M_2) = L_1 \cap L_2$.

How would you prove this?

Define $\hat{\delta}_1$ and $\hat{\delta}_2$ the usual way:

- $\hat{\delta}_1(q_1, \varepsilon) = q_1$
- $\hat{\delta}_1(q_1, ya) = \delta(\hat{\delta}(q_1, y), a)$

Define $\hat{\delta}_3$ similarly:

- $\hat{\delta}_3((q_1, q_2), \varepsilon) = (q_1, q_2)$
- $\hat{\delta}_3((q_1, q_2), ya) = \delta_3(\hat{\delta}_3((q_1, q_2), y), a)$

Idea: $\hat{\delta}_3$ simulates $\hat{\delta}_1$ and $\hat{\delta}_2$ simultaneously.

Lem: For any $x \in \Sigma^*$, $\hat{\delta}_3((q_1, q_2), x) = (\hat{\delta}_1(q_1, x), \hat{\delta}_2(q_2, x))$.

Proof. (Structural induction)

Game: Why is this true?

Base case: $x = \varepsilon$.

Want to show: $\hat{\delta}_3((q_1, q_2), \varepsilon) = (\hat{\delta}_1(q_1, \varepsilon), \hat{\delta}_2(q_2, \varepsilon))$

- $\hat{\delta}_3((q_1, q_2), \varepsilon) = (q_1, q_2)$ (def of $\hat{\delta}_3$)
- $= (\hat{d}_1(q_1, \varepsilon), \hat{d}_2(q_2, \varepsilon))$ (def of $\hat{\delta}_1$, $\hat{\delta}_2$)

Inductive case: $x = ya$ for $y \in \Sigma^*$, $a \in \Sigma$.

Want to show: $\hat{\delta}_3((q_1, q_2), ya) = (\hat{\delta}_1(q_1, ya), \hat{\delta}_2(q_2, ya))$

- $\hat{\delta}_3((q_1, q_2), ya) = \delta_3(\hat{\delta}_3((q_1, q_2), y), a)$ (def of $\hat{\delta}_3$)
- $= \delta_3((\hat{\delta}_1(q_1, y), \hat{\delta}_2(q_2, y)), a)$ (induction)
- $= (\delta_1(\hat{\delta}_1(q_1, y), a), \delta_2(\hat{\delta}_2(q_2, y), a))$ (definition of δ_3)
- $= (\hat{\delta}_1(q_1, ya), \hat{\delta}_2(q_2, ya))$ (definition of $\hat{\delta}_1$, $\hat{\delta}_2$).

Thm: $L(M_3) = L_1 \cap L_2$.

Proof: $x \in L(M_3)$

- $\Leftrightarrow \hat{\delta}_3((s_1, s_2), x) \in F_1 \times F_2$ (def. of acceptance)
- $\Leftrightarrow (\hat{\delta}_1(s_1, x), \hat{\delta}_2(s_2, x)) \in F_1 \times F_2$ (lemma)
- $\Leftrightarrow x \in L_1$ and $X \in L_2$ (definition of L_1, L_2)
- $\Leftrightarrow x \in L_1 \cap L_2$ (def of intersect)

Hence, regular languages are closed under intersection.

What if we wanted a DFA for $L_1 \cup L_2$? (answer on next page)

$F_3 = (Q_1 \times F_2) \cup (F_1 \times Q_2)$