Properties of Regular Languages

Closure • decision problems • constructions

Why regular languages stay regular under common operations.

Learning outcomes

- Define closure
- Construct product DFAs
- Explain closure proofs
- Decide membership, emptiness, finiteness and equivalence

What is closure?

A class is closed under an operation if applying it to members always produces a member of the same class.

For regular languages L_1 and L_2, their union and intersection are regular.

Proof strategy: construct a finite automaton or regular expression for the result.

Closure properties

Boolean operations: union, intersection, complement, difference, symmetric difference.

String operations: concatenation, Kleene star and reversal.

Mappings: homomorphism and inverse homomorphism.

Product DFA construction

Given complete DFAs M_1 and M_2 over the same alphabet Σ.

States $Q = Q_1 \times Q_2$; start $= (s_1, s_2)$.

$\delta((p,q),a) = (\delta_1(p,a), \delta_2(q,a))$.

Final pairs: union uses OR; intersection uses AND; difference uses first accepts AND second rejects.

Example: component DFAs

L_1: even number of 1s. States E (start, final) and O. On 0 stay; on 1 switch.

L_2: strings ending in 0. States N (start) and Z (final). On 0 go to Z; on 1 go to N.

Goal: $L_1 \cap L_2$, even number of 1s and ending in 0.

Intersection: product transition table

Product state	Input 0	Input 1	Accept?
→ (E,N)	(E,Z)	(O,N)	No
(E,Z)	(E,Z)	(O,N)	Yes
(O,N)	(O,Z)	(E,N)	No
(O,Z)	(O,Z)	(E,N)	No

Start: (E,N) • **Sole final: (E,Z)**

Trace the product DFA

110: (E,N) → (O,N) → (E,N) → (E,Z). Accept.

10: (E,N) → (O,N) → (O,Z). Reject.

Union finals: (E,N), (E,Z), (O,Z). Difference $L_1 - L_2$ final: (E,N).

Complement and difference

For a complete DFA, keep transitions and replace final set F with Q − F.

Complement of strings ending in 0: ε or (0∪1)*1.

$L_1 - L_2 = L_1 \cap \text{complement}(L_2)$. Complete partial DFAs before complementing.

Concatenation and Kleene star

Concatenation: connect final states of first disjoint ε-NFA to start of second using ε. Only second finals remain final.

Star: add fresh start which is final, with ε to old start. Add ε from old finals back to old start and to fresh final.

Example: $\{01\}^* = \{\varepsilon, 01, 0101, 010101, \ldots\}$.

Reversal

Reverse all NFA edges and make old start the sole final.

Introduce a fresh start with ε-edges to each old final.

Example: (0∪1)*01 reverses to 10(0∪1)*.

Homomorphism and inverse homomorphism

A homomorphism h maps symbols to strings and extends by concatenation.

If h(0)=ab and h(1)=c, then h(01*)=ab(c)*. Substitute images in a regular expression.

For inverse image of a DFA language, use $\delta'(q,a)=\delta^*(q,h(a))$; keep start and finals.

Decision problems

Membership: simulate DFA on w and test final state. Time O(length(w)).

Emptiness: no accepting state reachable from start.

Finiteness: a DFA language is infinite iff a directed cycle occurs on a path from start to an accepting state.

Equivalence and inclusion

Equivalence: $L_1 = L_2$ iff symmetric difference is empty. Product finals accept when exactly one component accepts.

Inclusion: $L_1 \subseteq L_2$ iff $L_1 - L_2$ is empty.

Closure can prove nonregularity: if A is regular, then $A \cap R$ is regular for every regular R.

Classroom exercises

Q1. Find complement of $1(0 \cup 1)^*$ over binary alphabet.

Q2. Use slide 7 to test 0010 and 00.

Q3. Reverse 0^*11. How can we decide whether two DFAs are equivalent?

Solutions and recap

Q1. ε or 0(0∪1)*.

Q2. 0010 rejected (one 1); 00 accepted (zero 1s, ends in 0).

Q3. 110*. Test emptiness of symmetric-difference product DFA.

Next: Pumping Lemma for Regular Languages.