THEORY OF AUTOMATA & FORMAL LANGUAGES

Systematic Construction of NFA

From language conditions to alternative paths

Describe → Design → Trace → Verify

A systematic design method

Every example uses $\Sigma = \{a,b\}$; no ε-transitions.

Step 1: describe the language

Identify the alphabet and condition.

List accepted and rejected strings.

Split the condition into milestones.

Assign a state to each milestone.

Choose initial and accepting states.

Step 2: construct and verify

Add edges for allowed symbols.

Use loops for arbitrary repetitions.

Use branches for alternative matches.

Missing edges mean an empty set.

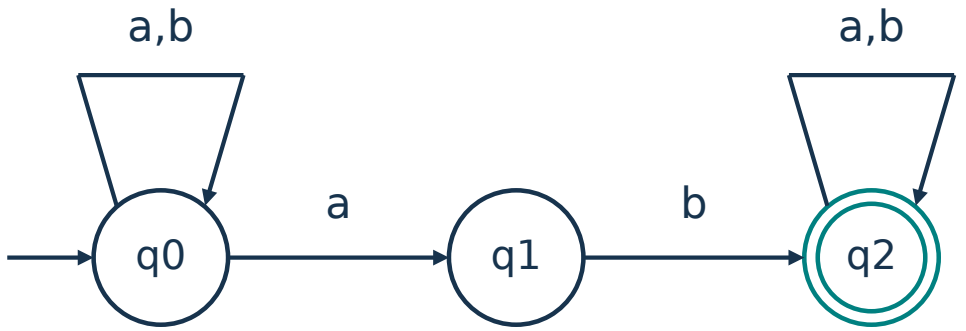
Trace the entire string before deciding.

An NFA accepts when at least one complete path reaches an accepting state.

Example 1: strings containing ab

Regular expression: (a|b)*ab(a|b)*.

Complete NFA



Double circle = accepting state

State meanings

q_0: scan an arbitrary prefix.

q_1: a possible starting a was read.

q_2: ab has been found.

q_0 loops on a,b; also a → q_1.

q_2 loops on a,b to retain the match.

Unlike a suffix machine, this machine accepts even if more symbols follow ab.

Verify the substring NFA

Track every reachable state after each symbol.

Transition table

State	a	b
q_0	$\{q_0,q_1\}$	$\{q_0\}$
q_1	$\varnothing$	$\{q_2\}$
q_2	$\{q_2\}$	$\{q_2\}$

Start: q_0; final: q_2.

Trace: baab

Start: $\{q_0\}$

b → $\{q_0\}$

a → $\{q_0,q_1\}$

a → $\{q_0,q_1\}$

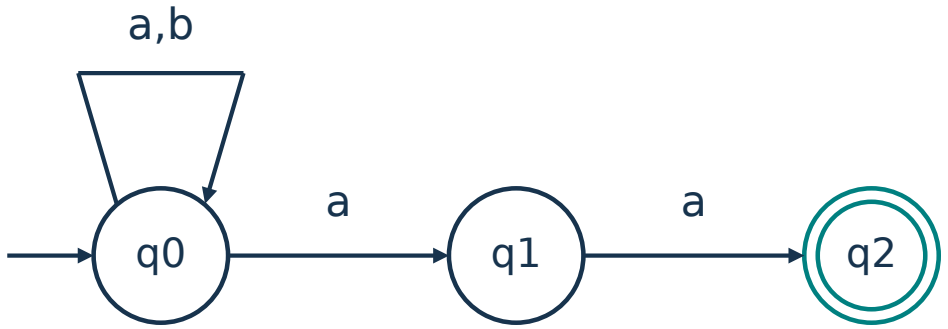
b → $\{q_0,q_2\}$: ACCEPT.

For bba the final set is $\{q_0,q_1\}$; no accepting state, so reject.

Example 2: strings ending in aa

Regular expression: (a|b)*aa.

Complete NFA



Double circle = accepting state

State meanings

q_0: scan any prefix.

q_1: guess the first of final aa.

q_2: both final a symbols matched.

q_0 loops on a,b; also a → q_1.

q_2 has no outgoing transitions.

The guessed match must finish exactly when the input ends.

Verify the suffix NFA

A failed guess does not invalidate other paths.

Transition table

State	a	b
q_0	$\{q_0,q_1\}$	$\{q_0\}$
q_1	$\{q_2\}$	$\emptyset$
q_2	$\emptyset$	$\emptyset$

Start: q_0; final: q_2.

Trace: aaa

Start: $\{q_0\}$

$a \rightarrow \{q_0,q_1\}$

$a \rightarrow \{q_0,q_1,q_2\}$

$a \rightarrow \{q_0,q_1,q_2\}$: ACCEPT.

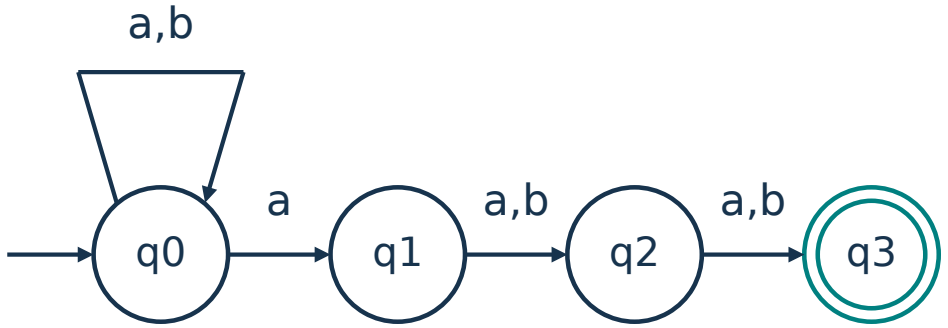
For aab: final set $\{q_0\}$: reject.

The accepting path can start its final match at the second a.

Example 3: third symbol from the end is a

Regular expression: (a|b)*a(a|b)(a|b).

Complete NFA



Double circle = accepting state

State meanings

q_0: scan arbitrary prefix.

q_1: guess the required a.

q_2: consume one following symbol.

q_3: consume second following symbol.

No outgoing edges from q_3.

Guess the position of a, then require exactly two more symbols.

Verify the position-based NFA

Input: baab. The third symbol from the end is a.

Transition table

State	a	b
q_0	$\{q_0,q_1\}$	$\{q_0\}$
q_1	$\{q_2\}$	$\{q_2\}$
q_2	$\{q_3\}$	$\{q_3\}$
q_3	$\emptyset$	$\emptyset$

Trace: baab

Start: $\{q_0\}$

b → $\{q_0\}$

a → $\{q_0,q_1\}$

a → $\{q_0,q_1,q_2\}$

b → $\{q_0,q_2,q_3\}$: ACCEPT.

For ab, only two symbols exist; q_3 cannot be reached, so reject.

Practice and comparison

Use the state meanings to predict each decision.

Practice: decide membership

Contains ab: aba? Yes.

Ends in aa: aaba? No.

Ends in aa: baa? Yes.

Third from end a: abb? Yes.

Third from end a: bab? No.

Design principles

Substring: loop after the match.

Suffix: no loop after the match.

Fixed position: count remaining input.

Multiple destinations are allowed.

Do not accept before input ends.

Next: ε-NFA — transitions that consume no input symbol.