THEORY OF AUTOMATA & FORMAL LANGUAGES

DFA Minimization

From equivalent states to a smaller machine

Reachability → Partition → Refine → Merge → Verify

Why minimize a DFA?

Preserve the language while reducing the number of states.

Reachability

A state is reachable if some input leads to it.

Start at the initial state.

Follow every outgoing transition.

Repeat until no new state is found.

Remove all unreachable states.

Equivalence

Two states are equivalent if every suffix

is accepted from both or rejected from both.

An accepting and a rejecting state differ on ε.

Equivalent states can be merged.

A minimum DFA has no equivalent distinct states.

Remove unreachable states before refining partitions.

Worked example — original DFA

Σ = {a,b}; initial A; accepting states A, A′, B, B′.

Original transition table

State	a	b
→ * A	A′	B
* A′	A	B′
* B	D	B′
* B′	D	B

Remaining states and reachability

State	a	b
D	D	D
U	U	U

From A: a → A′; b → B; bb → B′.

ba → D; U is never reached.

Original states: 6. Remove U; 5 reachable states remain.

Step 1 — Initial partition

Separate accepting states from non-accepting states.

Partition P_0

$F = \{A, A', B, B'\}$

$Q - F = \{D\}$

$P_0 = \{\{A,A',B,B'\}, \{D\}\}$

Call the first block X, the second Y.

States in different blocks cannot merge.

Why this split is necessary

The empty suffix ε is already a test.

From A: ε is accepted.

From D: ε is rejected.

Thus A and D are distinguishable.

Check the remaining states using a and b.

The initial partition distinguishes states by acceptance of ε.

Step 2 — Refine the partition

Compare destination blocks for every symbol.

Transition signatures under P_0

State	a-block	b-block
A	X	X
A′	X	X
B	Y	X
B′	Y	X

Split block X

A and A′ have signature (X,X).

B and B′ have signature (Y,X).

D has signature (Y,Y).

$P_1 = \{\{A,A'\}, \{B,B'\}, \{D\}\}$

The suffix a distinguishes A from B.

Split any block whose members have different transition signatures.

Step 3 — Test stability

Rename blocks X={A,A′}, Y={B,B′}, Z={D}.

Signatures under P_1

State	a-block	b-block
A, A′	X	Y
B, B′	Z	Y
D	Z	Z

All members of each block agree.

Stable partition

$P_2 = P_1$; no further split is possible.

A ≡ A′ and B ≡ B′.

X and Y differ on suffix a.

X and Z differ on ε; Y and Z differ on ε.

Therefore all three blocks are distinct.

A stable partition of reachable states gives the minimum DFA.

Step 4 — Draw the minimum DFA

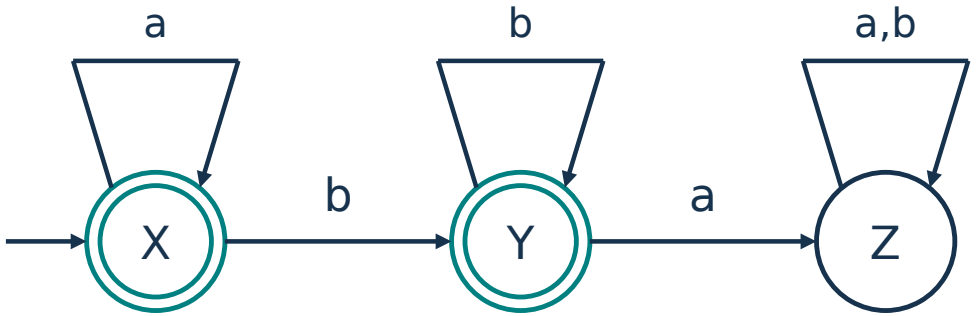
X and Y are accepting; Z is a rejecting sink.

Minimum transition table

State	a	b
→ * X	X	Y
* Y	Z	Y
Z	Z	Z

X={A,A′}; Y={B,B′}; Z={D}

Resulting DFA



The language is a*b*: zero or more a symbols followed by zero or more b symbols.

Step 5 — Verify equivalence

Compare complete executions of the original and minimized machines.

Accepted input: aabb

Original: A → A′ → A → B → B′

Inputs: a a b b

Minimum: X → X → X → Y → Y

Final B′ belongs to accepting block Y.

Both machines accept aabb.

Rejected input: aba

Original: A → A′ → B′ → D

Inputs: a b a

Minimum: X → X → Y → Z

Final D belongs to rejecting block Z.

Both machines reject aba.

The stable partition proves equivalence for all strings, not just these tests.

Practice and summary

Apply reachability and refinement systematically.

Practice questions

1. Can an unreachable state affect acceptance?

2. Can a final and nonfinal state merge?

3. When should refinement stop?

4. How many states remain in our example?

5. What suffix distinguishes X and Y?

Explained answers

1. No: no input reaches it.

2. No: ε distinguishes them.

3. When no block can be split.

4. Three, from six original states.

5. a: X accepts it; Y rejects it.

Minimum DFA is unique up to renaming of states (for a fixed language and alphabet).