COM2109

Automata

Robert Hierons

More on DFA: equivalence

Equivalence

Remember:

Finite Automata M and M' are equivalent if $L(M) = L(M')$

We can extend this to states s and s' of a finite automaton M:

s and s' are equivalent if $L(M(s))=L(M(s'))$

where, for a state t, $M(t)$ is the automaton formed by starting M in state t

Reducing a finite automaton

A DFA is minimal if there is no equivalent DFA that has fewer states.

We can reduce a DFA if it is not minimal.

We can simply:

Merge equivalent states

Reduction is checking for equivalence.

Remember: Intersection and DFA

Machine M_1

DFA for L_1

Machine M_2

DFA for L_2

Construct a new DFA M that accepts $L_1 \cap L_2$

M simulates in parallel M_1 and M_2

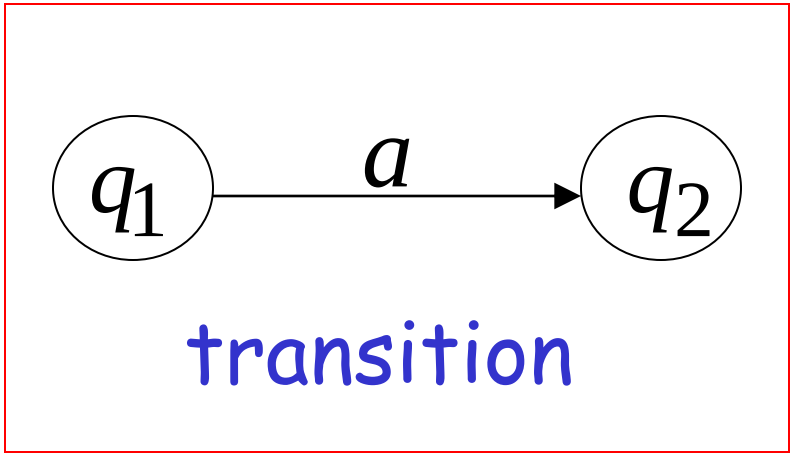
States in M

q_i, p_j

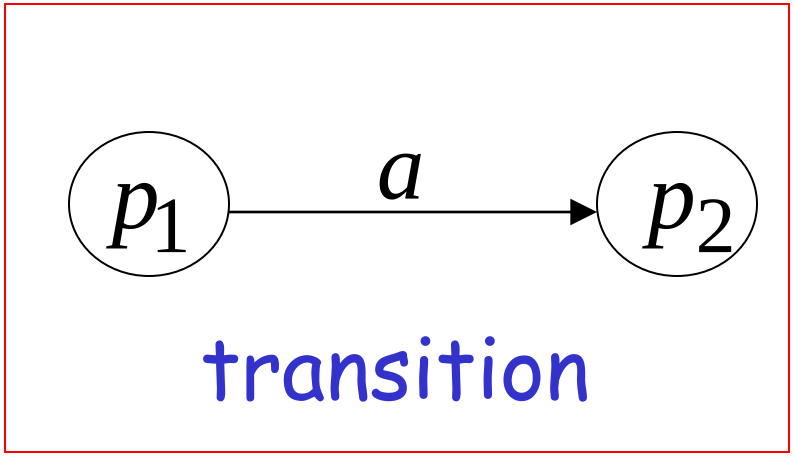
State in M_1

State in M_2

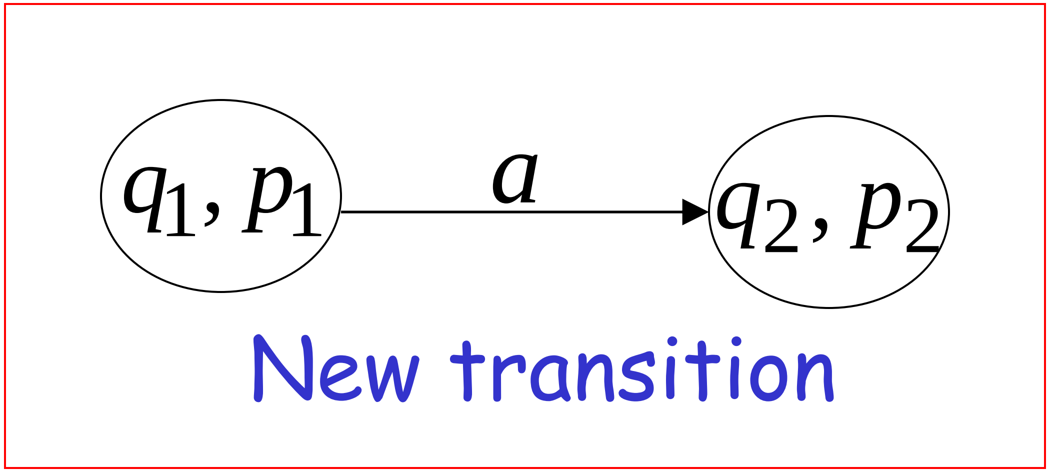
DFA M_1

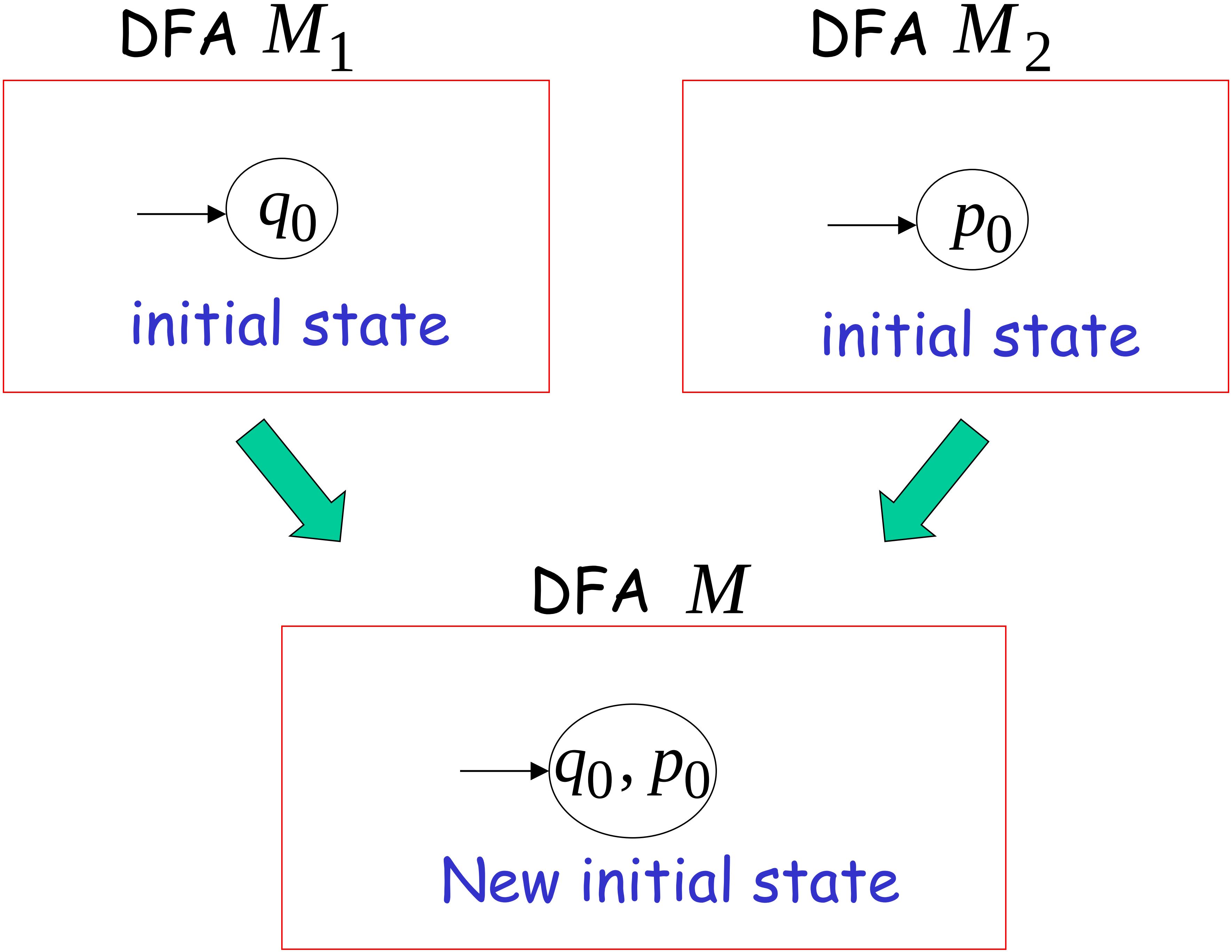


DFA M_2

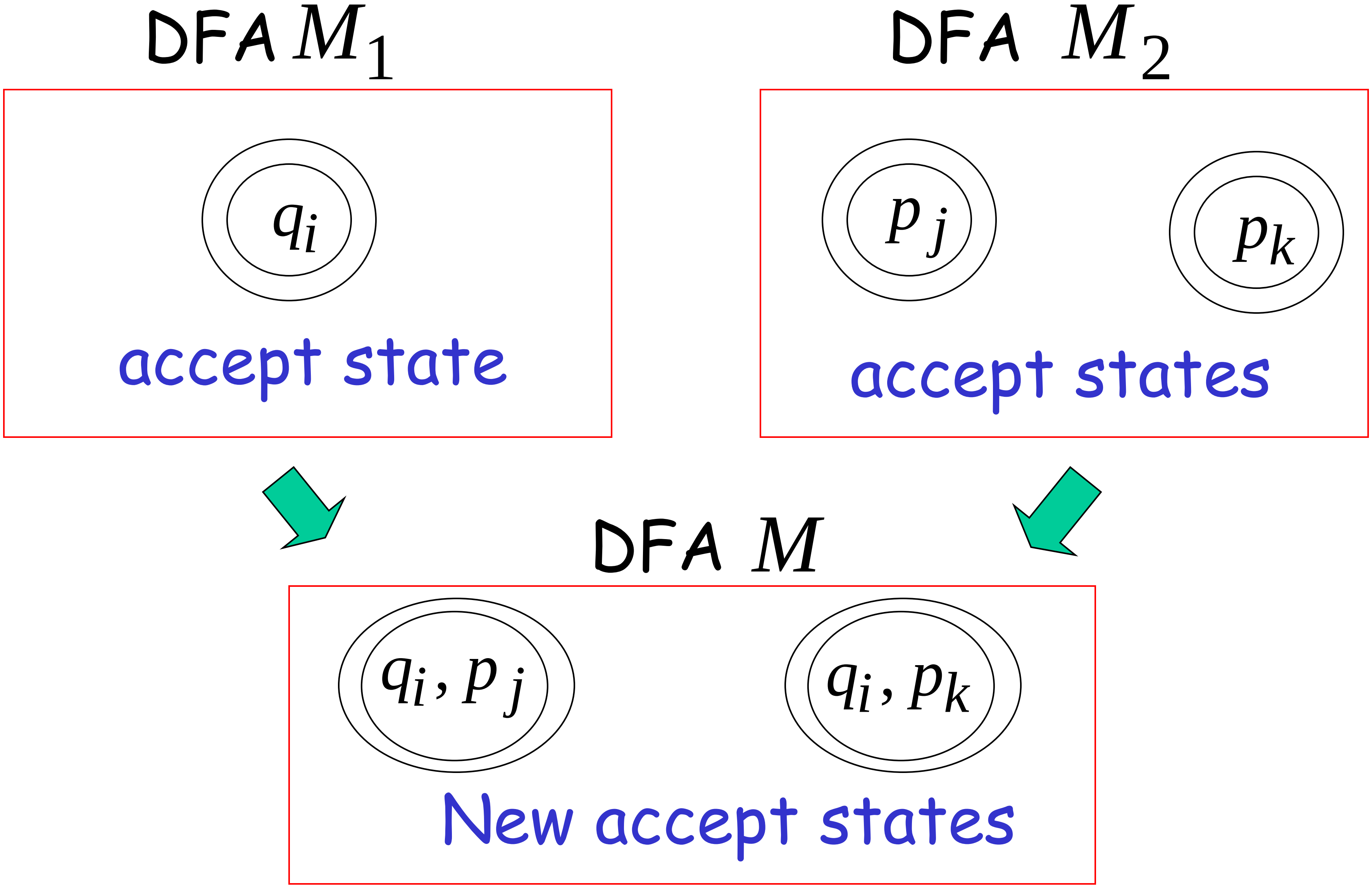


DFA M





DFA M_1
q_0
initial state
DFA M_2
p_0
initial state
DFA M
q_0, p_0
New initial state



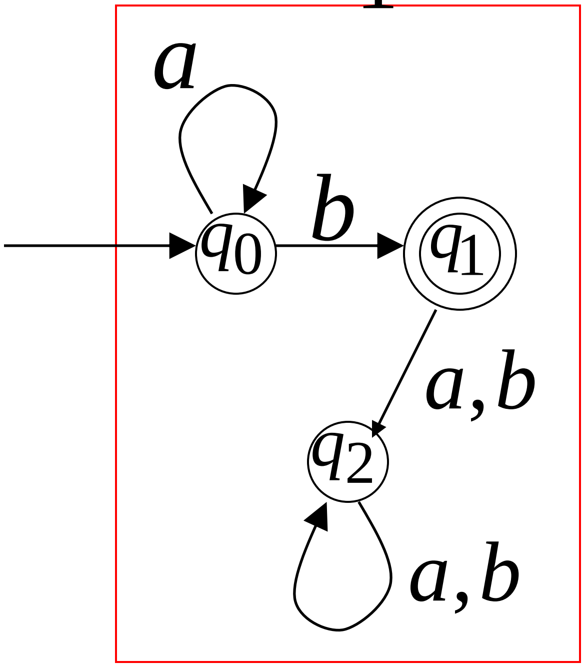
Both constituents must be accepting states

Example:

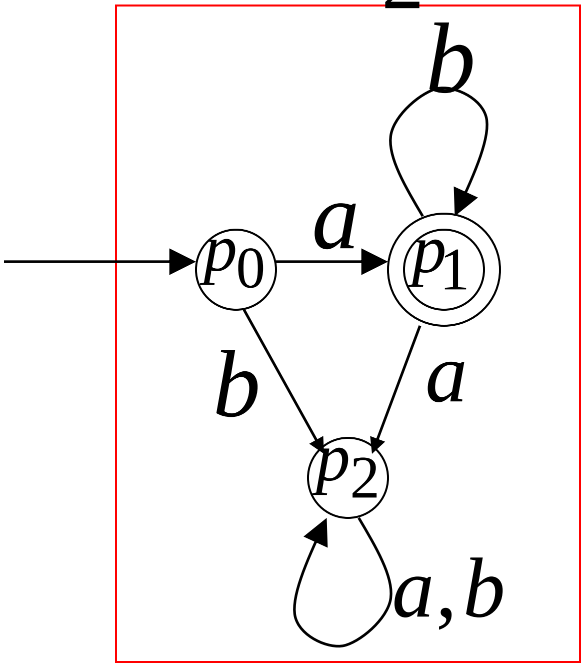
$$L_1 = \{a^n b\} \quad n \geq 0$$

$$L_2 = \{ab^m\} \quad m \geq 0$$

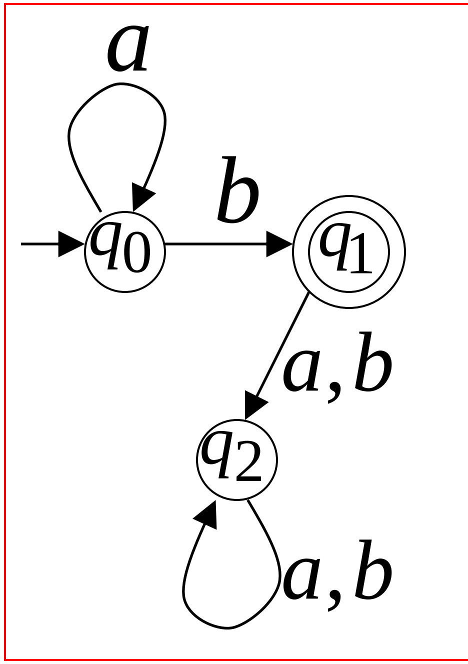
M_1



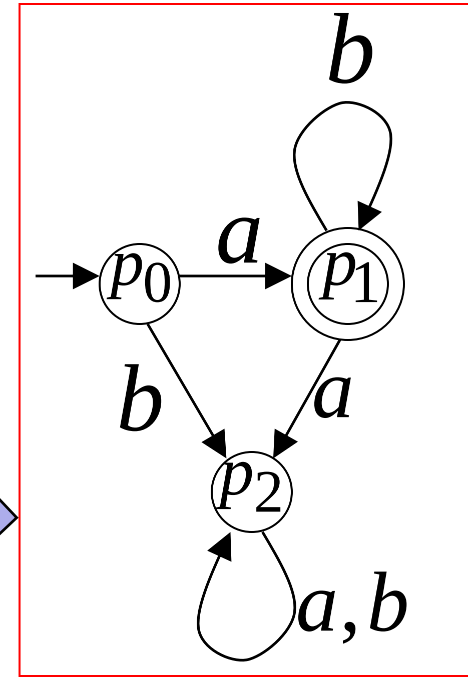
M_2

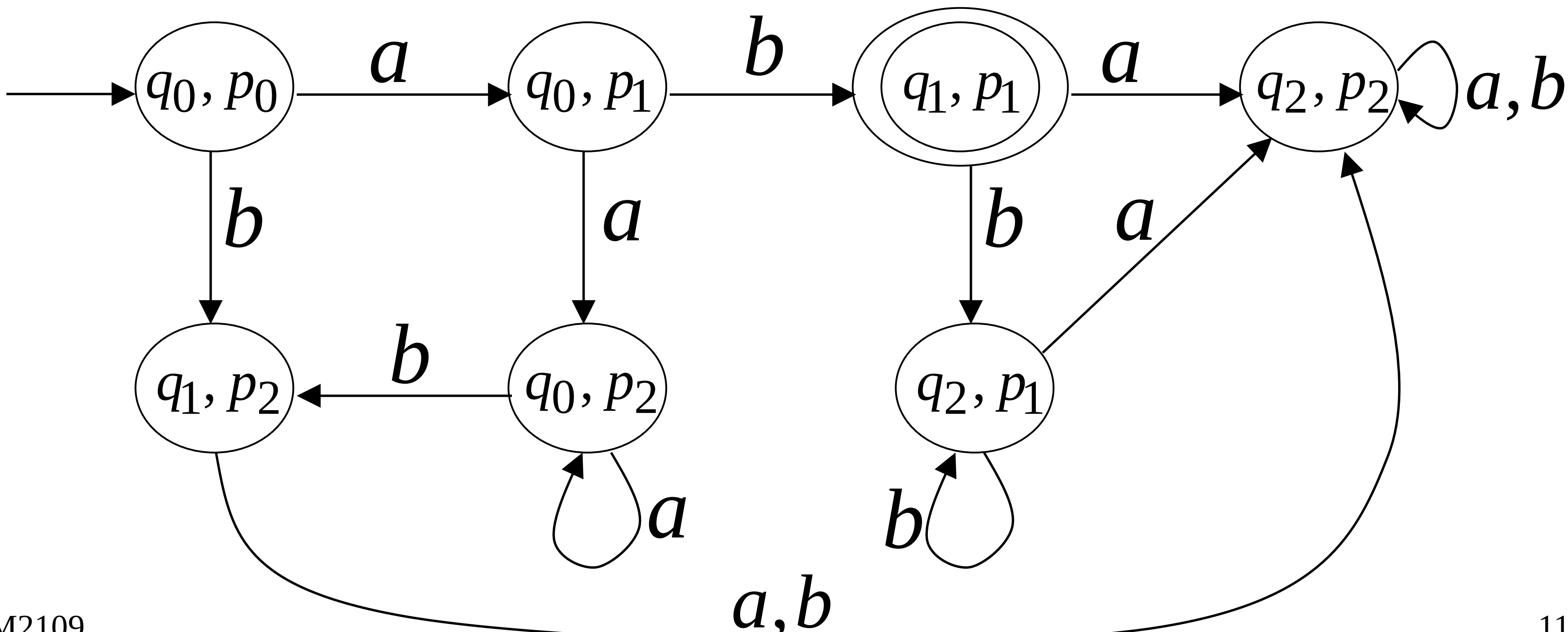


Automaton for intersection



$$L = \{a^n b\} \cap \{ab^n\} = \{ab\}$$





M simulates in parallel M_1 and M_2

M accepts string w if and only if:

M_1 accepts string w

and M_2 accepts string w

$$L(M) = L(M_1) \cap L(M_2)$$

How can we use this?

Simple:

A sequence separates M_1 and M_2 if either:

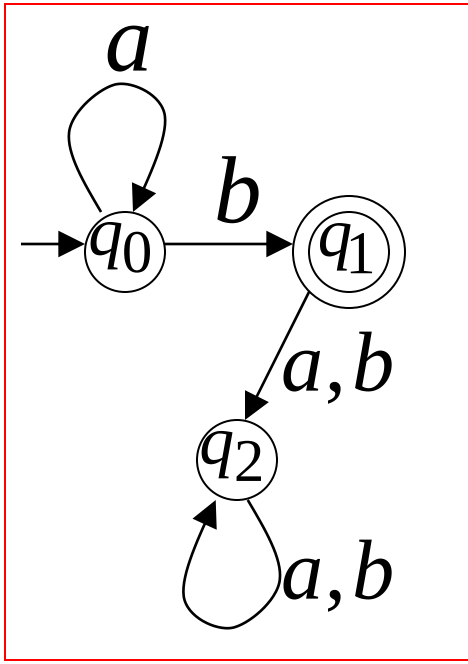
the sequence is in L_1 and not in L_2

or

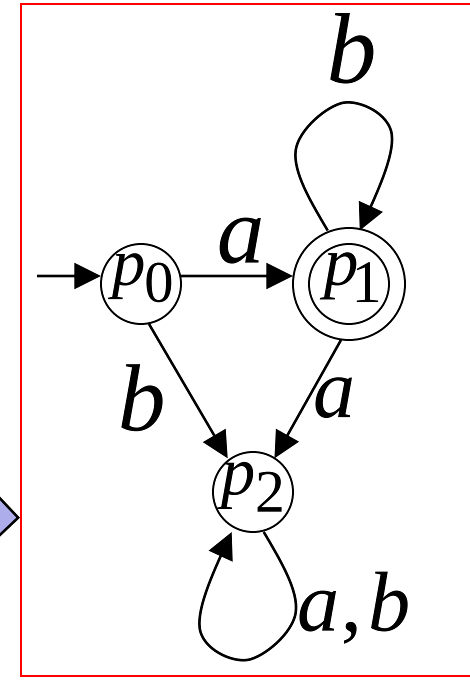
the sequence is in L_2 and not in L_1

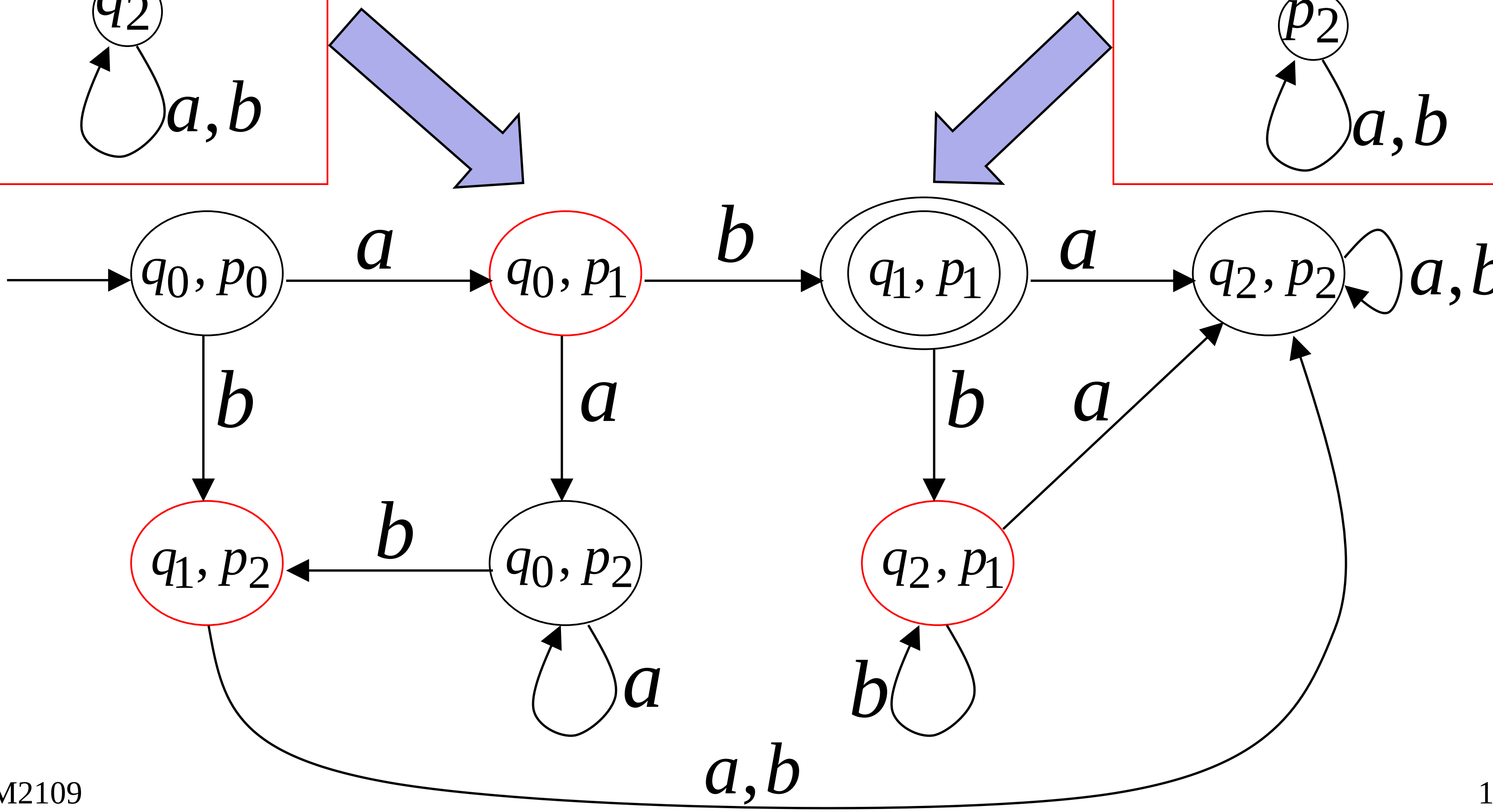
This is the case if we reach a pair of states (p,q) where one is accepting and the other is not.

Automaton for intersection



$$L = \{a^n b\} \cap \{ab^n\} = \{ab\}$$





Summary

There are efficient algorithms that:

Minimise a DFA

Determine whether DFA are equivalent

Determine whether the languages defined by two DFA intersect

Potential applications include verification of designs/models.