Nondeterminism

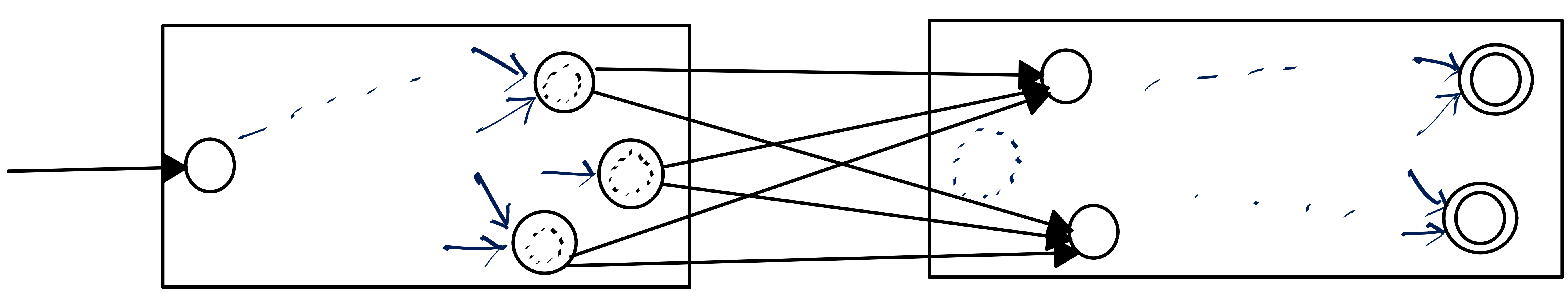
<u>Recap</u>: Reg is closed under union, intersection, and complementation

<u>Today</u>: Other operations, nondeterminism

<u>Concatenation</u>:

If A and B are regular (s.t. $A = \mathcal{L}(M_1)$ and $B = \mathcal{L}(M_2)$),
is $A \circ B = \{ xy \mid x \in A, \ y \in B \}$ regular?

Suppose $A \circ B$ is regular. How can we construct a DFA M for it?



Consider the following languages over $\Sigma = \{a, b\}$

A : all strings containing at least one a (M_1)

B : all strings containing at least one b (M_2)

What do M_1, M_2, and M look like?

The machine needs to "know" when a "relevant" substring ends, and check membership in the appropriate language accordingly.

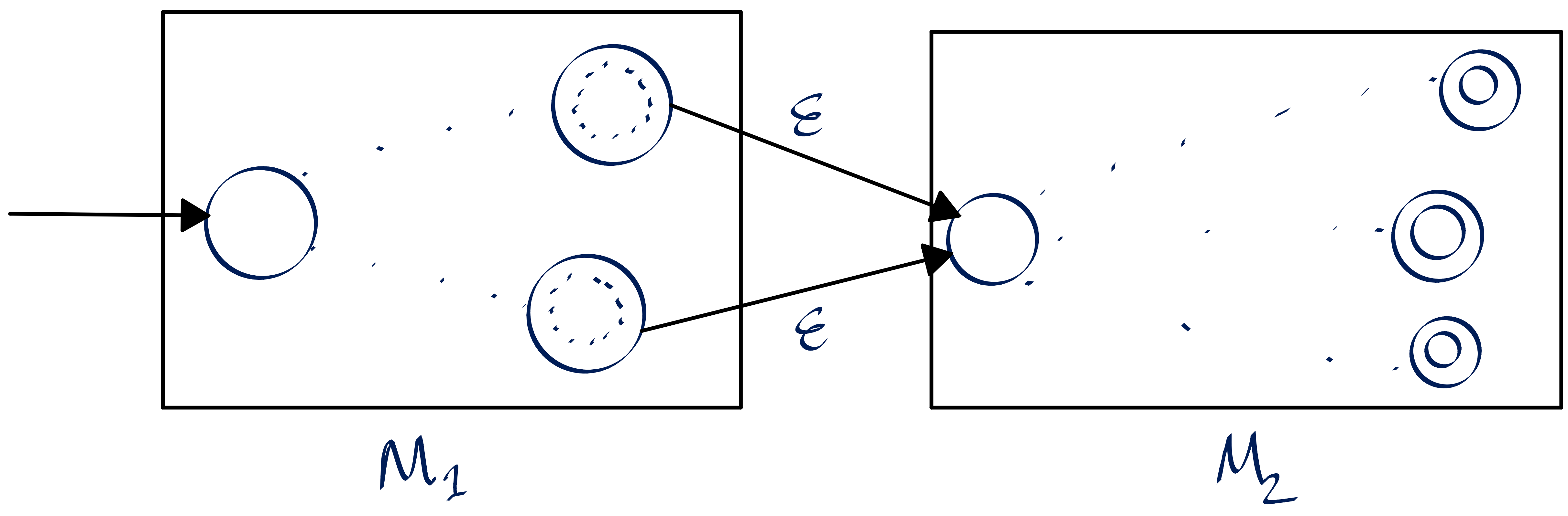
How can it know such a thing? Magic!

Suppose the machine could correctly guess when the substring x ends and y begins, s.t. $x \in A$ and $y \in B$.
Then we add the transitions between the "appropriate" states, and done!

The question is: what labels do these transitions take on?

Must not affect the behaviour of M_1 and M_2,
but still allow this "magically correct" guess!

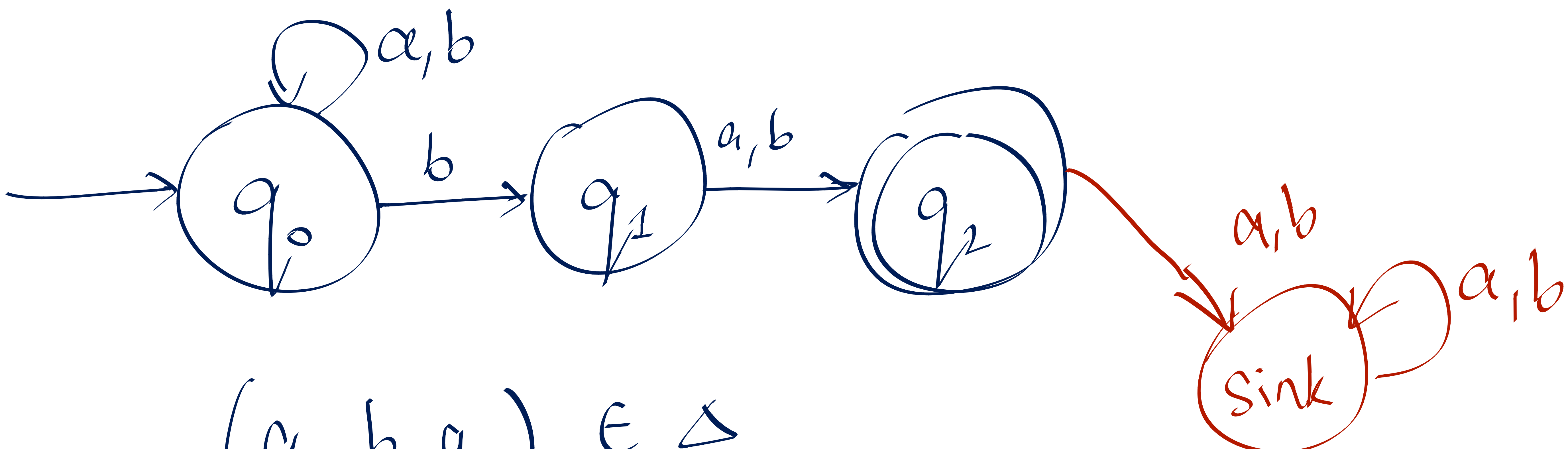
We move, therefore, to an extended model of computation,
a nondeterministic finite-state automaton (NFA).



ε
ε
M1
M2

$\mathcal{L} = \{ wbl \mid w \in \Sigma^*, l \in \Sigma \}$ over $\Sigma = \{a, b\}$

All strings with b as the penultimate letter.



$(q_0, b, q_0) \in \Delta$

$(q_0, b, q_1) \in \Delta$

DFA

$M = (Q, \Sigma, \delta, q_0, F)$

Q: finite set of states

Σ: alphabet

δ: transition function

$\delta : Q \times \Sigma \to Q$

q_0: initial state $\in Q$

F: set of final states $\subseteq Q$

M accept a word w iff the run of M on w terminates in a final state from F.

NFA

$M = (Q, \underbrace{\Sigma \cup \{\varepsilon\}}_{\Sigma_\varepsilon}, \Delta, Q_0, F)$

$+\ \varepsilon$

Δ: transition relation

$\Delta \subseteq Q \times \Sigma_\varepsilon \times Q$

Q_0: set of initial states $\subseteq Q$

M accepts a word w iff M has at least one run on w which terminates in a state $\in F$.