Finite Automata

Part Two

Recap from Last Time

Old MacDonald Had a Symbol, ♬ Σ-eye-ε-ey∈, Oh! ♫

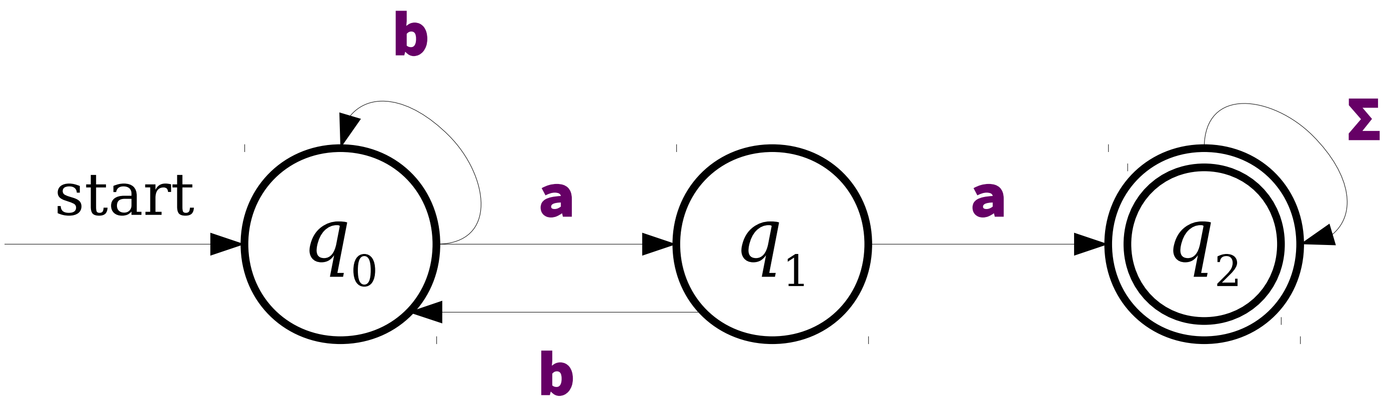
- You may have noticed that we have several letter-E-ish symbols in CS103, which can get confusing!
- Here's a quick guide to remembering which is which:
  - In automata theory, **Σ** refers to an ***alphabet***.
  - In automata theory, **ε** is the ***empty string***, which is length 0.
  - In set theory, use **∈** to say "is an ***element of***."
  - In set theory, use **⊆** to say "is a ***subset of***."

DFAs

- A ***DFA*** is a
  - ***D***eterministic
  - ***F***inite
  - ***A***utomaton
- DFAs are the simplest type of automaton that we will see in this course.

Recognizing Languages with DFAs

$L = \{ w \in \{\mathbf{a}, \mathbf{b}\}^* \mid w$ contains **aa** as a substring $\}$

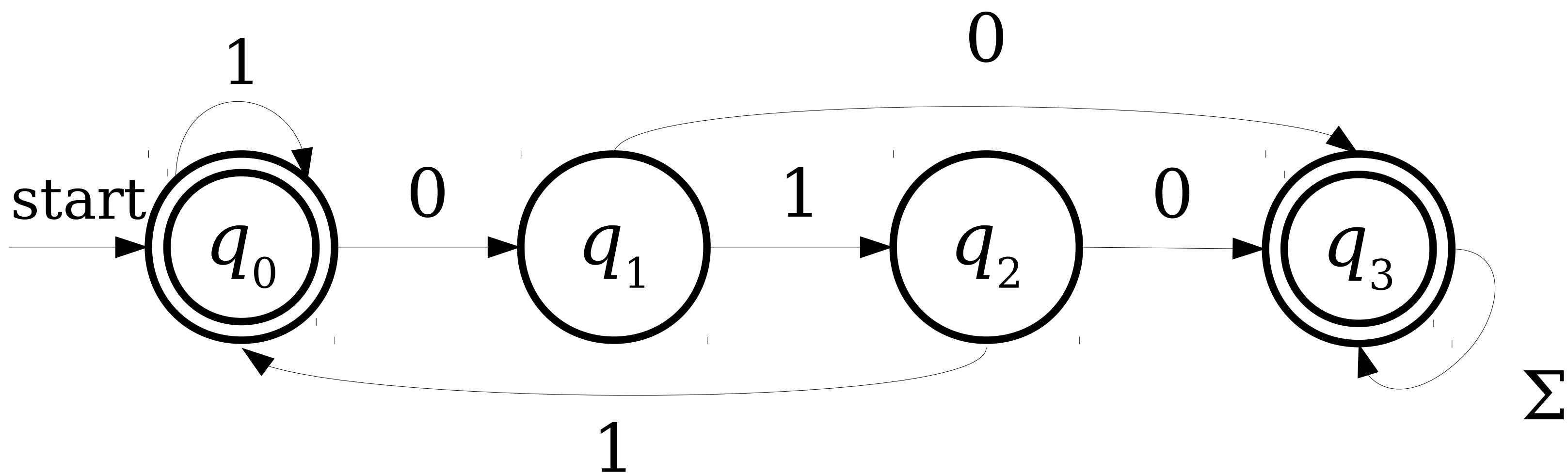


DFAs

- A DFA is defined relative to some alphabet Σ.
- For each state in the DFA, there must be ***exactly one*** transition defined for each symbol in Σ.
  - This is the "deterministic" part of DFA.
- There is a unique start state.
- There are zero or more accepting states.

New Stuff!

Which table best represents the transitions for the DFA shown below?



(A)

	0	1
q_0	q_1	q_0
q_1	q_3	q_2
q_2	q_3	q_0
q_3	q_3	q_3

(B)

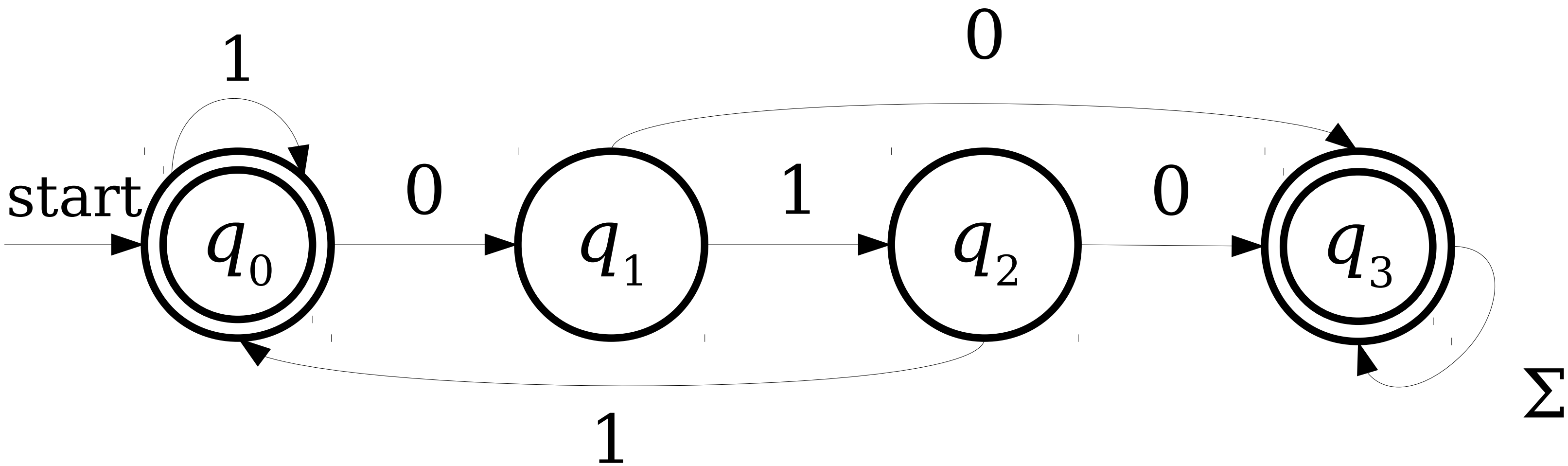
	0	1
q_0	q_0	q_1
q_1	q_2	q_3
q_2	q_0	q_3
q_3	q_3	q_3

(C)

	0	1	Σ
q_0	q_1	q_0	/
q_1	q_3	q_2	/
q_2	q_3	q_0	/
q_3	/	/	q_3

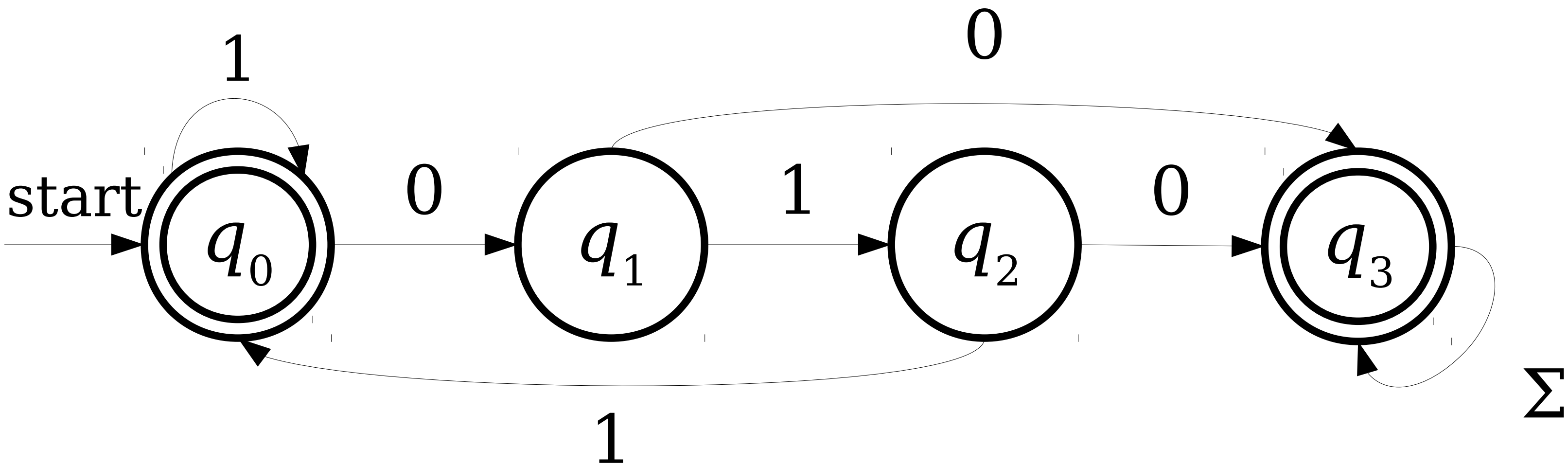
Answer at **PollEv.com/cs103** or text **CS103** to **22333** once to join, then **A**, **B**, **C**, or **D (none of the above)**.

Tabular DFAs



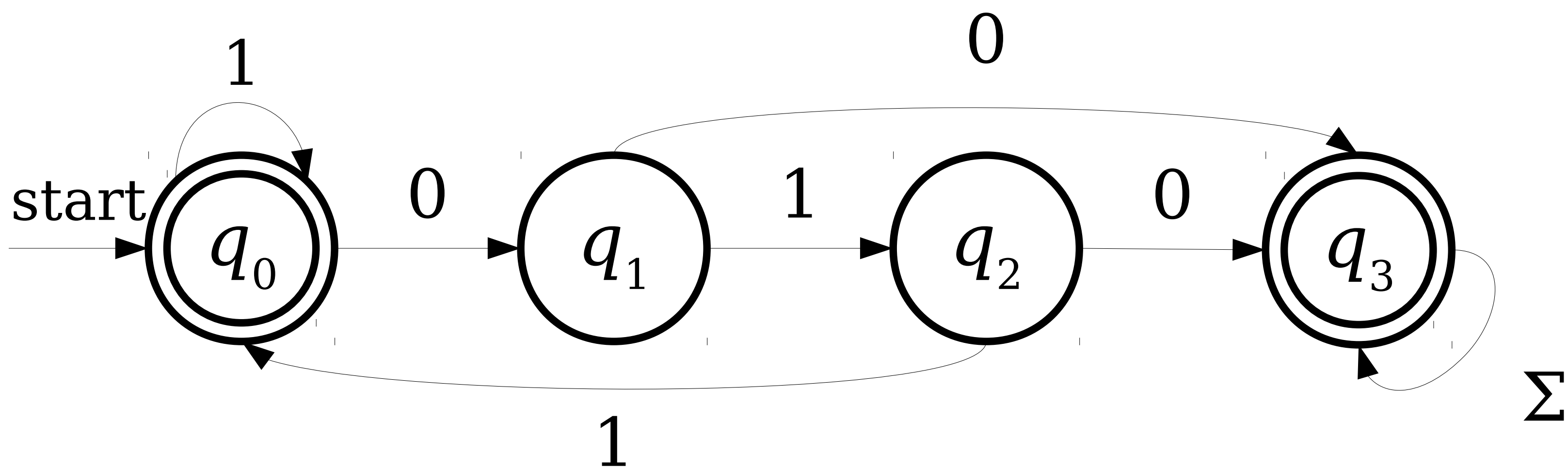
	0	1
q_0		
q_1		
q_2		
q_3		

Tabular DFAs



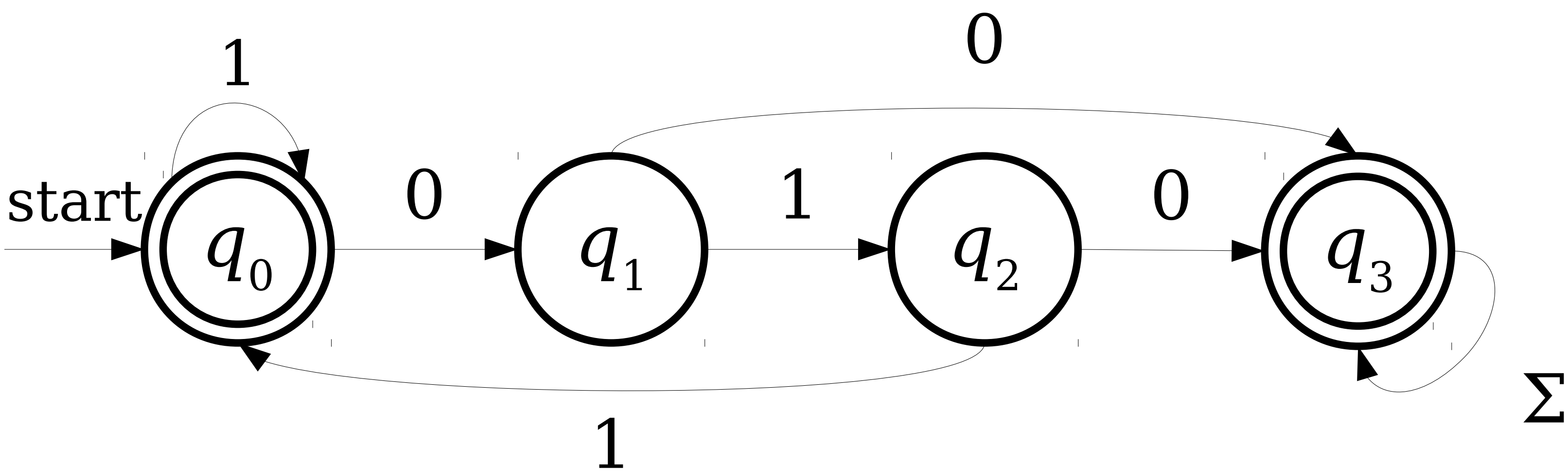
	0	1
q_0	q_1	q_0
q_1	q_3	q_2
q_2	q_3	q_0
q_3	q_3	q_3

Tabular DFAs



	0	1
$*q_0$	q_1	q_0
q_1	q_3	q_2
q_2	q_3	q_0
$*q_3$	q_3	q_3

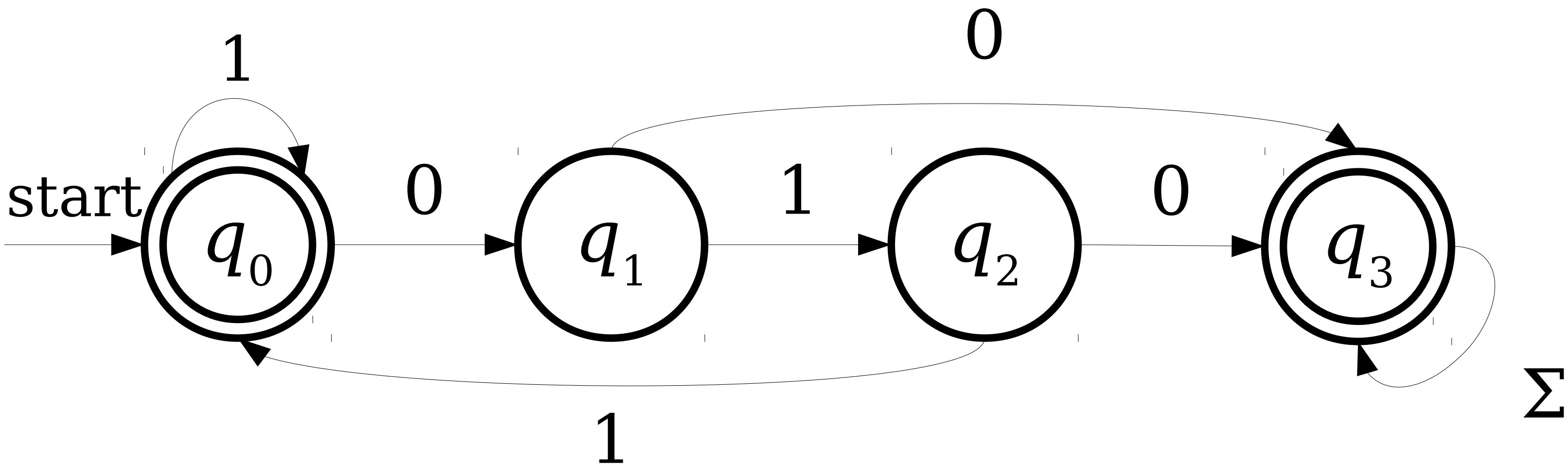
Tabular DFAs



	0	1
$*q_0$	q_1	q_0
q_1	q_3	q_2
q_2	q_3	q_0
$*q_3$	q_3	q_3

These stars indicate accepting states.

Tabular DFAs



	0	1
$*q_0$	q_1	q_0
q_1	q_3	q_2
q_2	q_3	q_0
$*q_3$	q_3	q_3

Since this is the first row, it's the start state.

My Turn to Code Things Up!

```
int kTransitionTable[kNumStates][kNumSymbols] = {
     {0, 0, 1, 3, 7, 1, …},
      …
};
bool kAcceptTable[kNumStates] = {
    false,
    true,
    true,
    …
};
bool SimulateDFA(string input) {
    int state = 0;
    for (char ch: input) {
        state = kTransitionTable[state][ch];
    }
    return kAcceptTable[state];
}
```

The Regular Languages

A language L is called a ***regular language*** if there exists a DFA D such that $\mathscr{L}(D) = L$.

If L is a language and $\mathscr{L}(D) = L$, we say that D ***recognizes*** the language L.

The Complement of a Language

- Given a language $L \subseteq \Sigma^*$, the ***complement*** of that language (denoted $\overline{L}$) is the language of all strings in Σ^* that aren't in L.
- Formally:

$$\overline{L} = \Sigma^* - L$$

The Complement of a Language

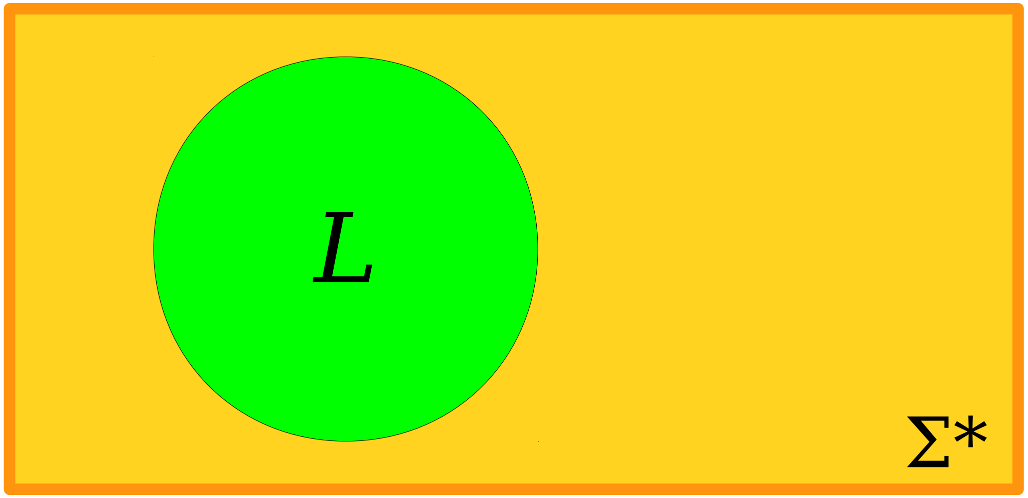
- Given a language $L \subseteq \Sigma^*$, the ***complement*** of that language (denoted $\overline{L}$) is the language of all strings in Σ* that aren't in L.
- Formally:

$$\overline{L} = \Sigma^* - L$$

The Complement of a Language

- Given a language $L \subseteq \Sigma^*$, the ***complement*** of that language (denoted $\overline{L}$) is the language of all strings in Σ^* that aren't in L.
- Formally:

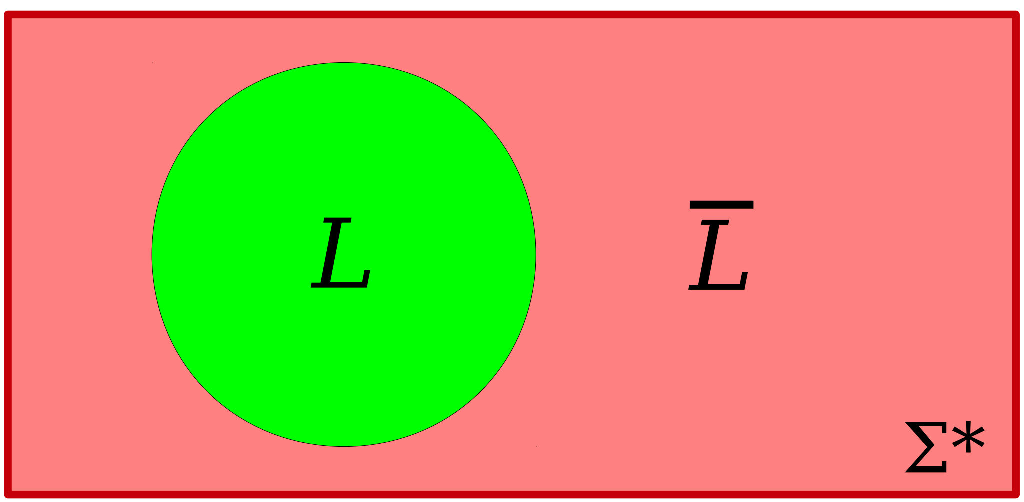
$$\overline{L} = \Sigma^* - L$$



The Complement of a Language

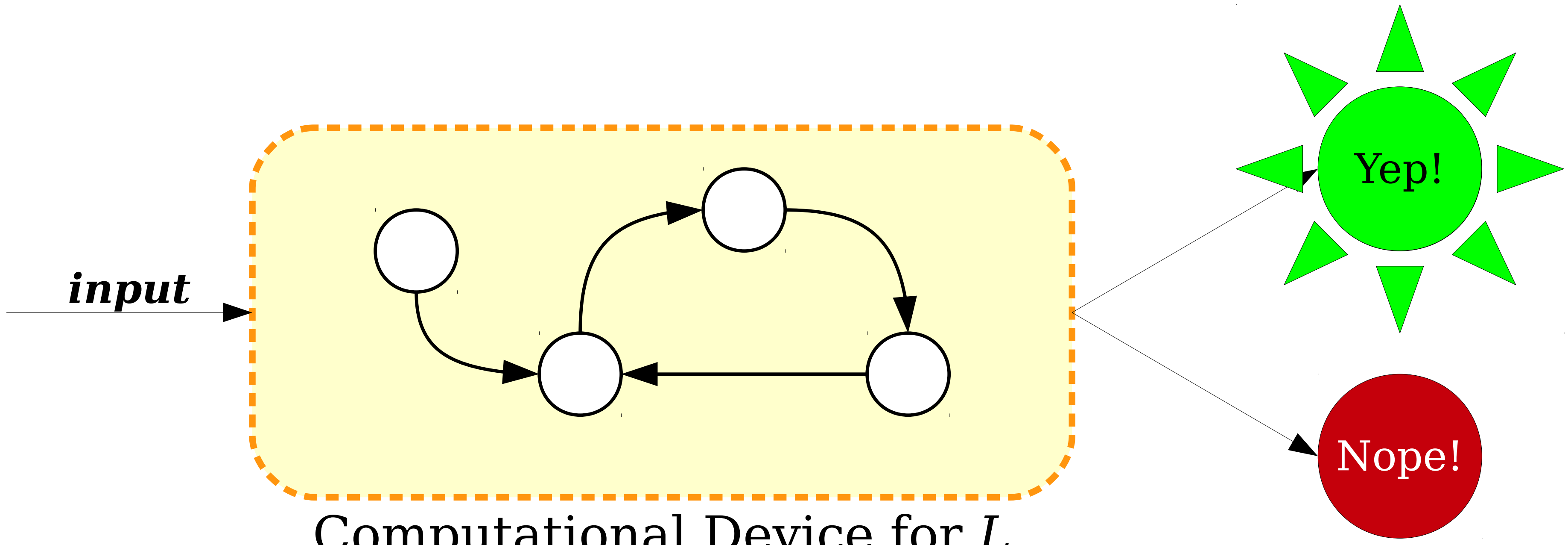
- Given a language $L \subseteq \Sigma^*$, the ***complement*** of that language (denoted $\overline{L}$) is the language of all strings in Σ^* that aren't in L.
- Formally:

$$\overline{L} = \Sigma^* - L$$

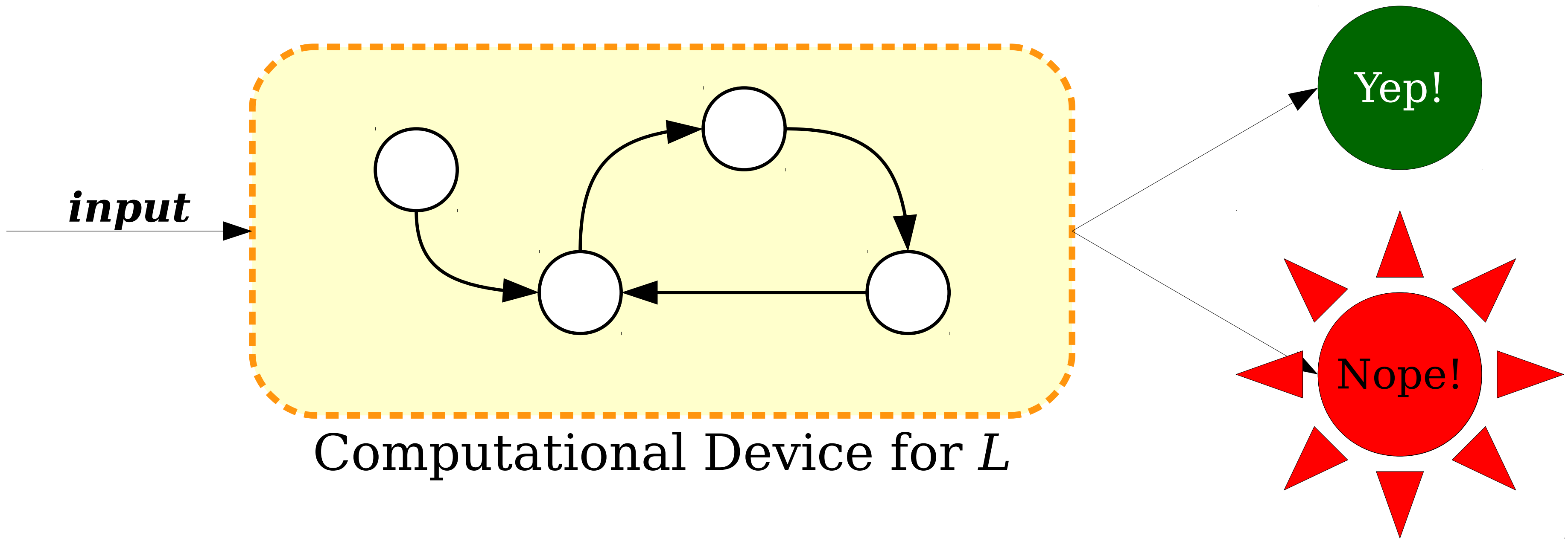


Complements of Regular Languages

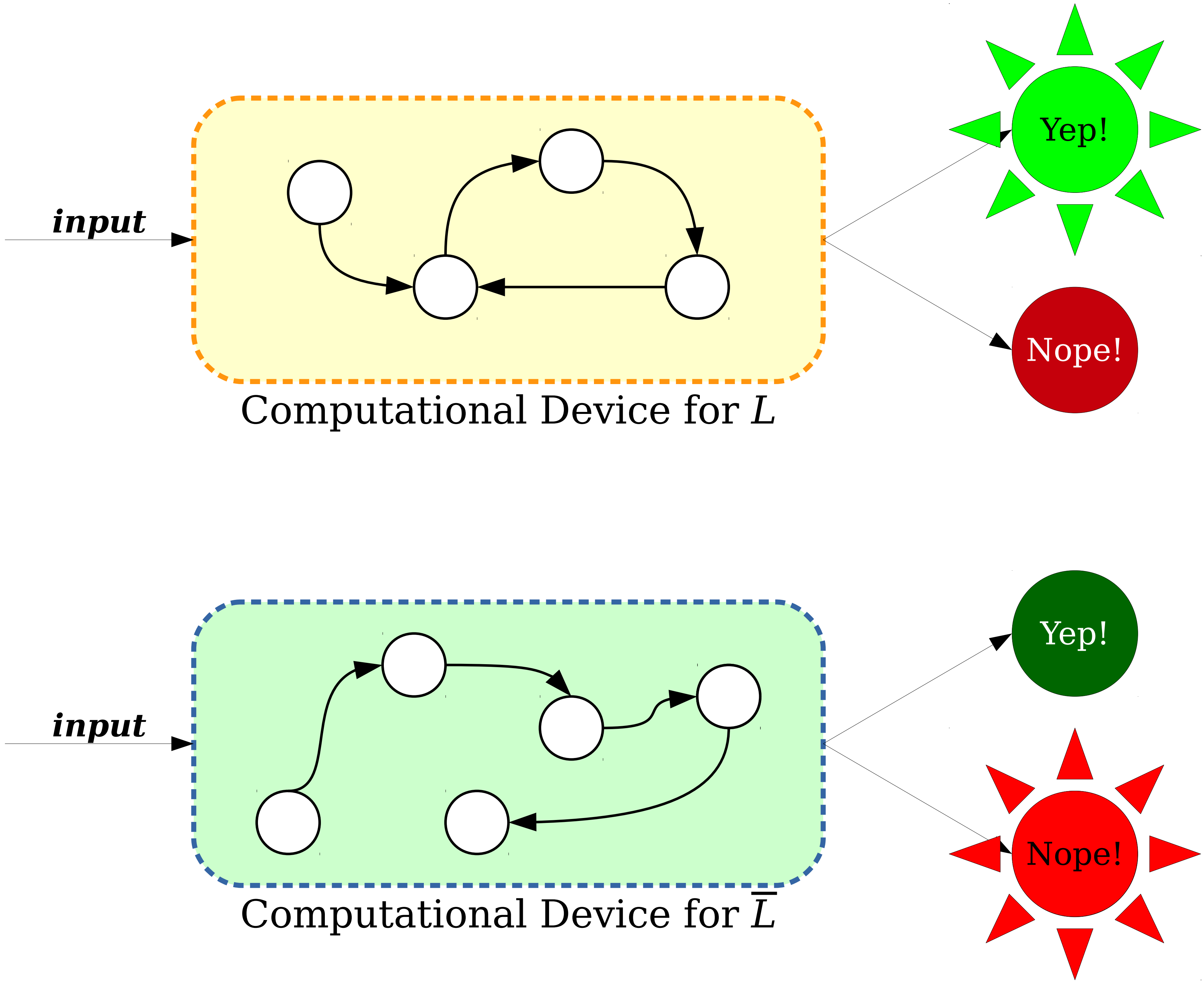
- As we saw a few minutes ago, a ***regular language*** is a language accepted by some DFA.
- ***Question:*** If L is a regular language, is $\overline{L}$ necessarily a regular language?
- If the answer is "yes," then if there is a way to construct a DFA for L, there must be some way to construct a DFA for $\overline{L}$.
- If the answer is "no," then some language L can be accepted by some DFA, but $\overline{L}$ cannot be accepted by any DFA.



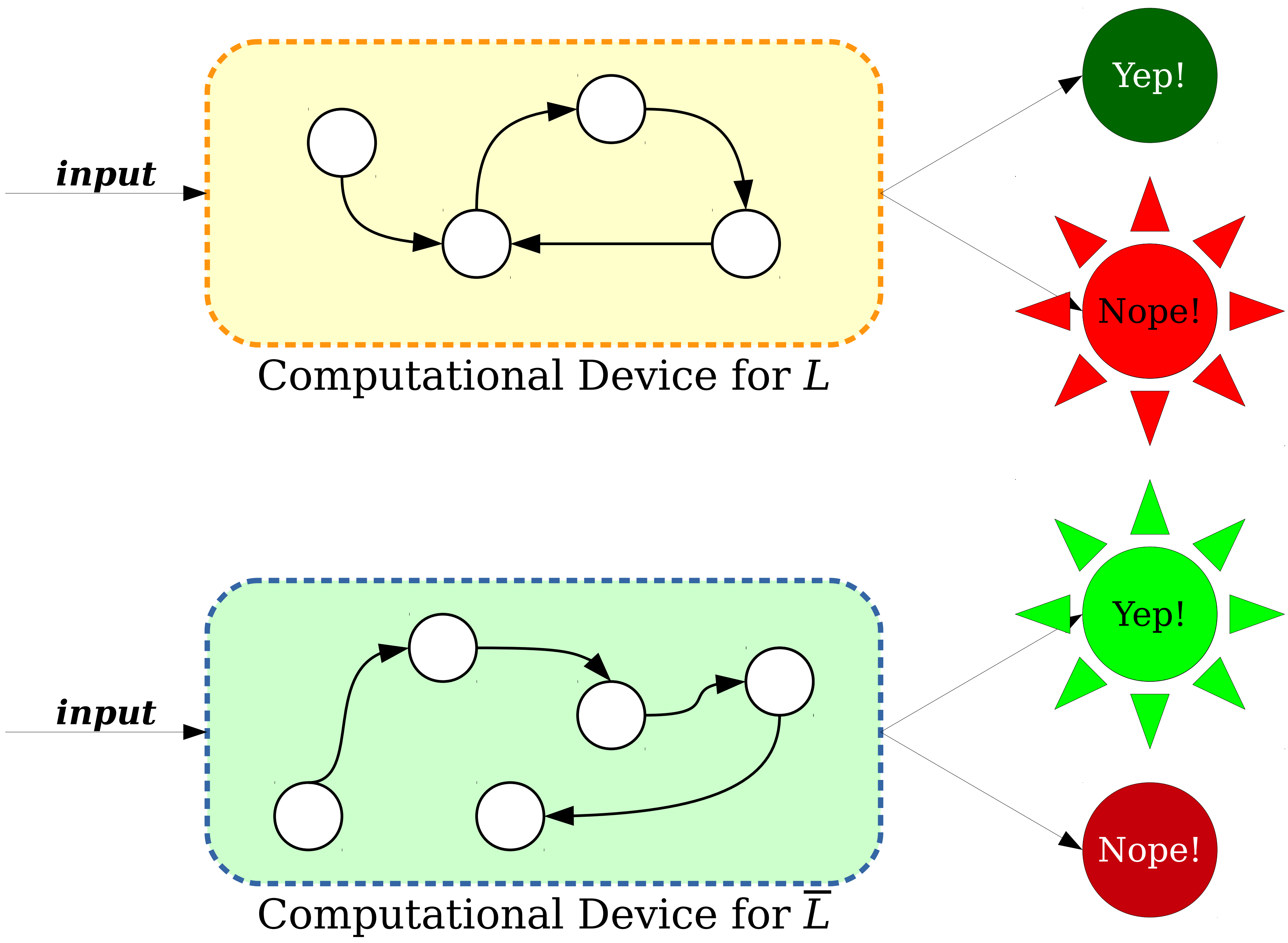
Computational Device for L



input
Yep!
Nope!
Computational Device for L



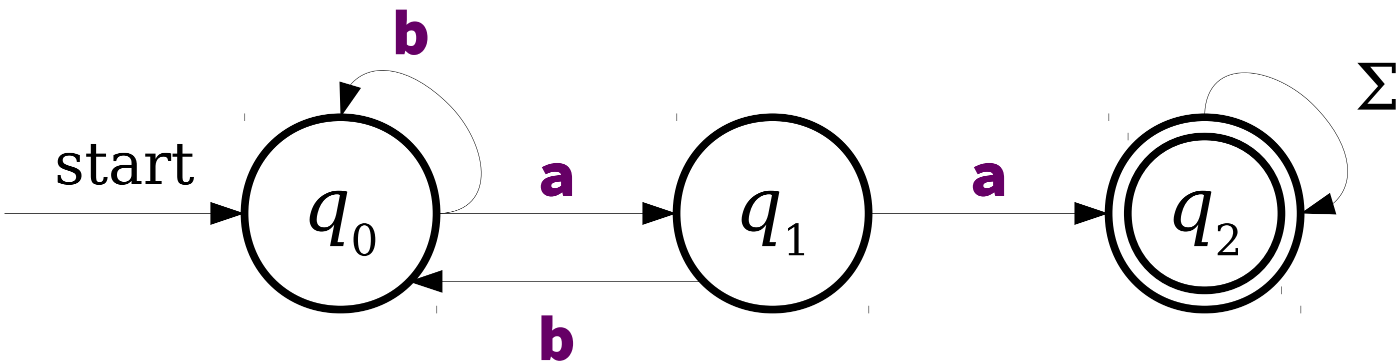
input
Computational Device for L
Yep!
Nope!
input
Computational Device for $\overline{L}$
Yep!
Nope!



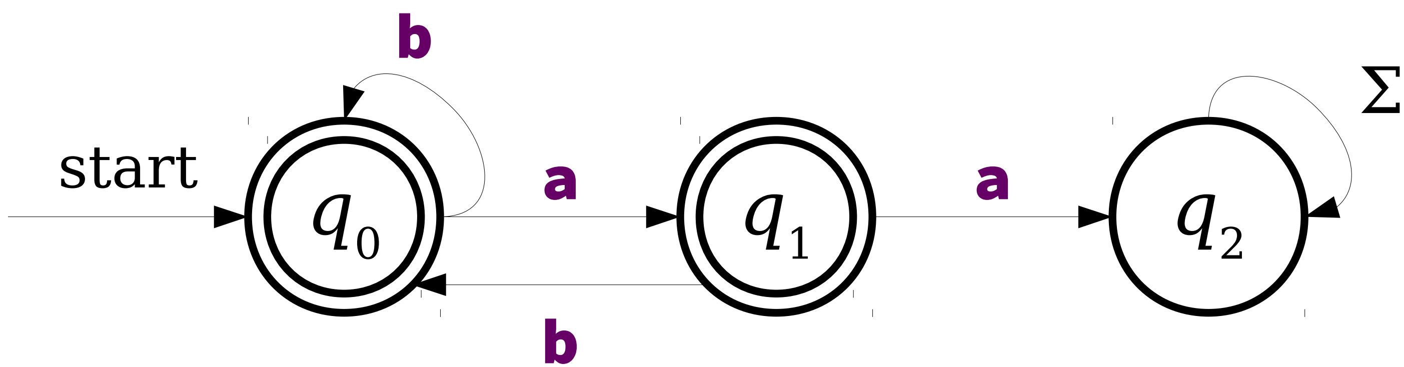
input
Computational Device for L
Yep!
Nope!
input
Computational Device for $\overline{L}$
Yep!
Nope!

Complementing Regular Languages

$L = \{ w \in \{$**a**, **b**$\}^* \mid w$ contains **aa** as a substring $\}$

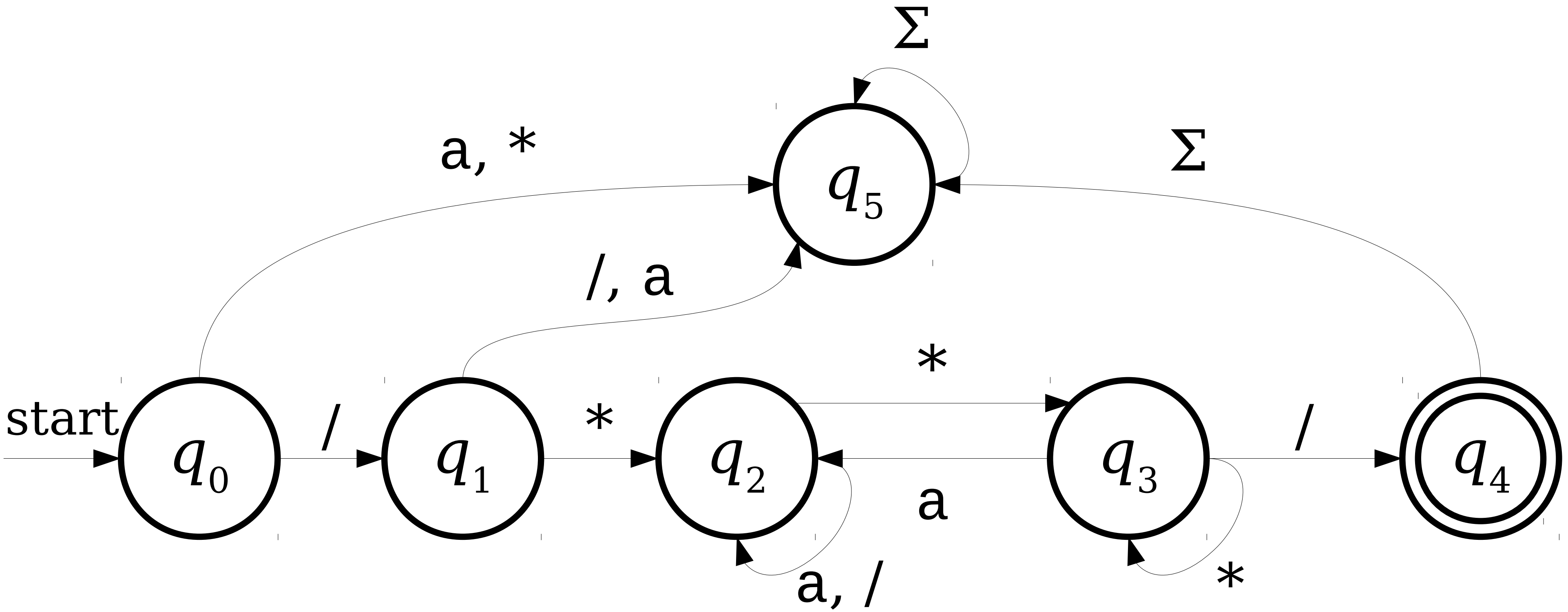


$\overline{L} = \{ w \in \{$**a**, **b**$\}^* \mid w$ ***does not*** contain **aa** as a substring $\}$



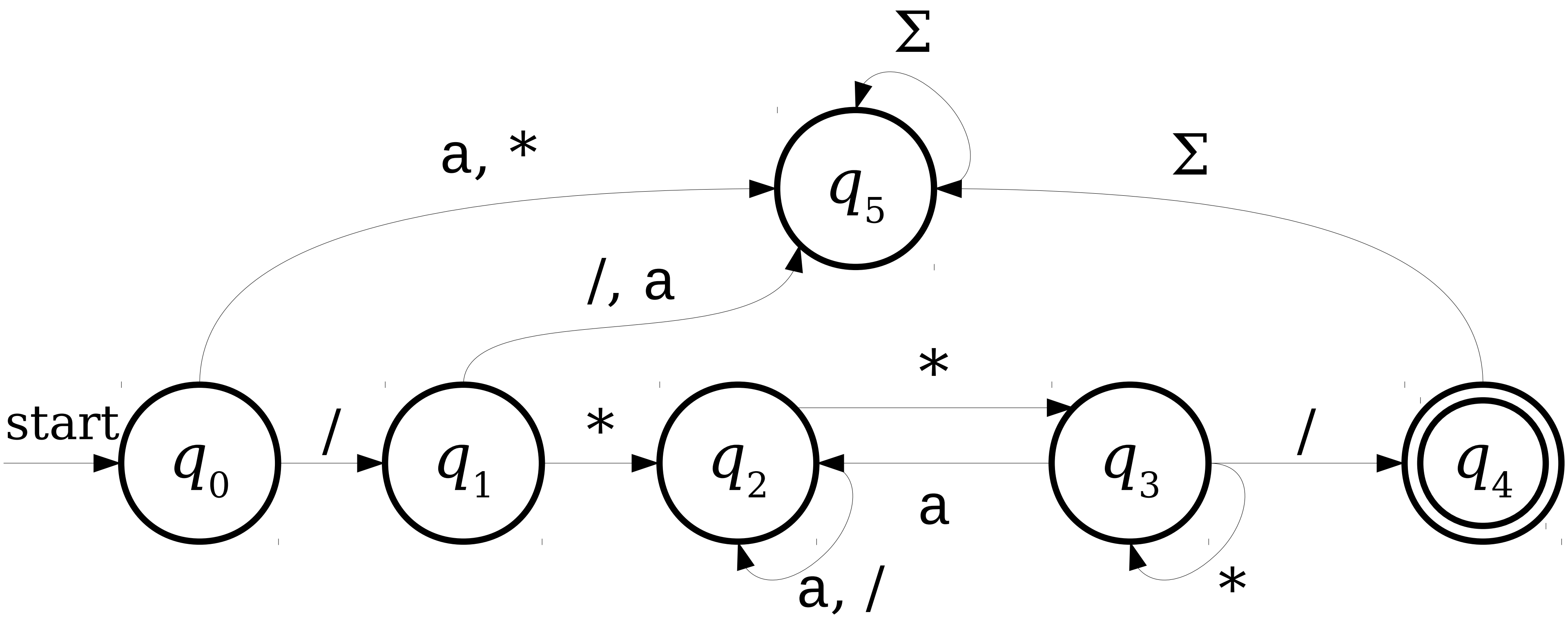
More Elaborate DFAs

$L = \{ \, w \in \{\texttt{a}, \texttt{*}, \texttt{/}\}^* \mid w$ represents a C-style comment $\}$



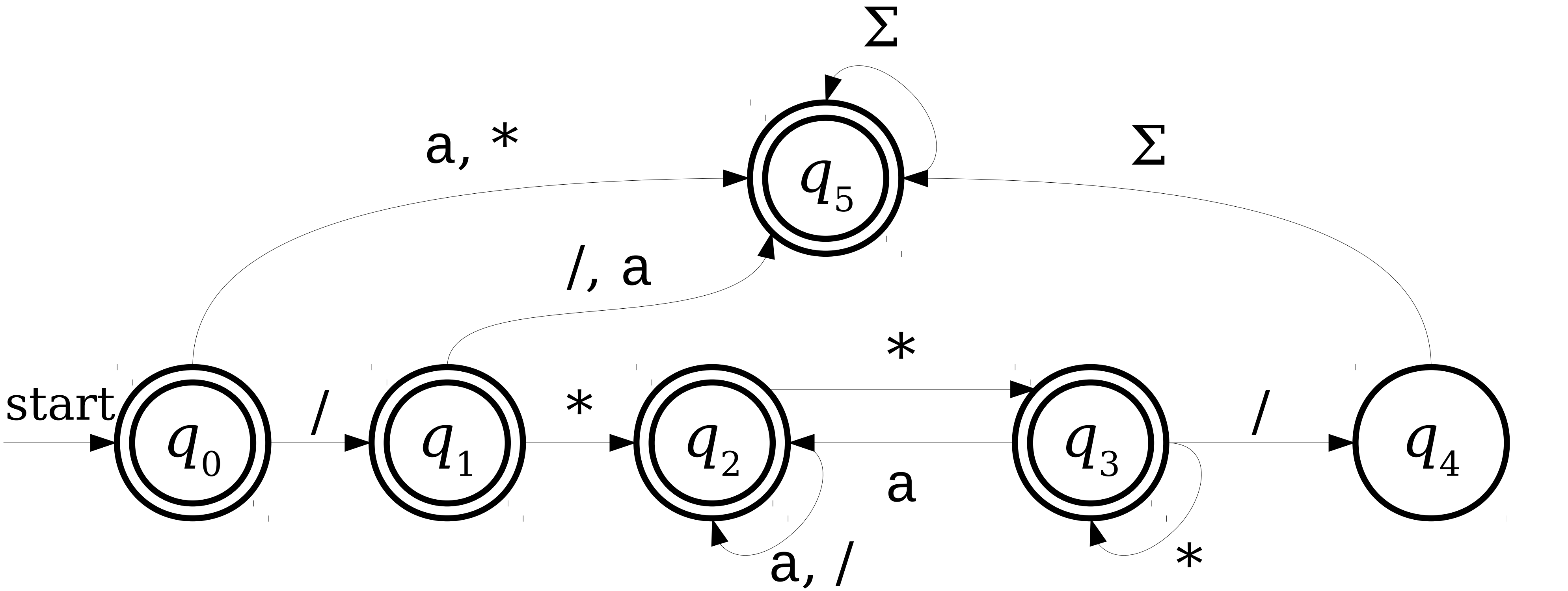
More Elaborate DFAs

$\overline{L}$ = { w ∈ {**a**, *****, **/**}* | *w doesn't* represent a C-style comment }



More Elaborate DFAs

$\overline{L}$ = { w ∈ {**a**, *****, **/**}* | *w doesn't* represent a C-style comment }



Closure Properties

- ***Theorem:*** If L is a regular language, then $\overline{L}$ is also a regular language.
- As a result, we say that the regular languages are ***closed under complementation***.