Finite Automata

Part Two

Recap from Last Time

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

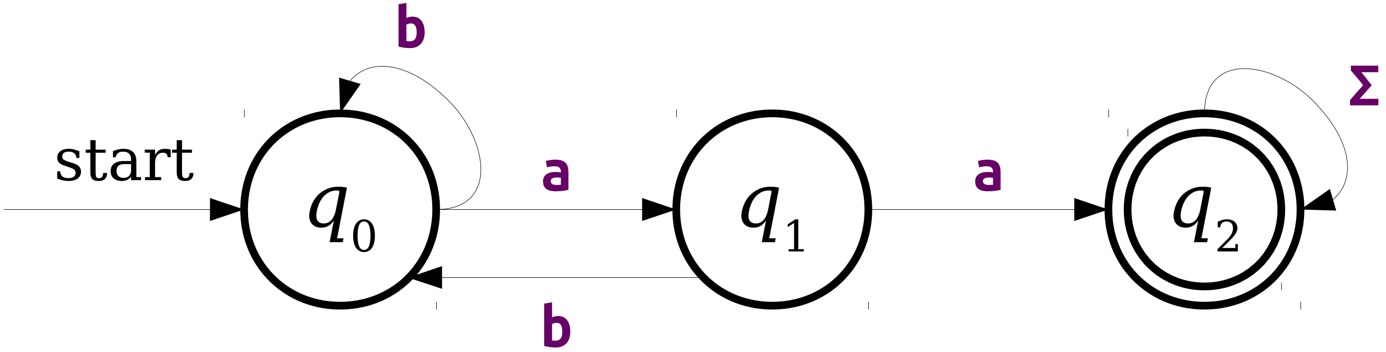
- You may have noticed that we have several letter-E-ish symbols in CS103, which can get confusing!
- Here's a quick guide to remembering which is which:
  - Typically, we use the symbol **Σ** to refer to an ***alphabet***.
  - The ***empty string*** is length 0 and is denoted **ε**.
  - In set theory, 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

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

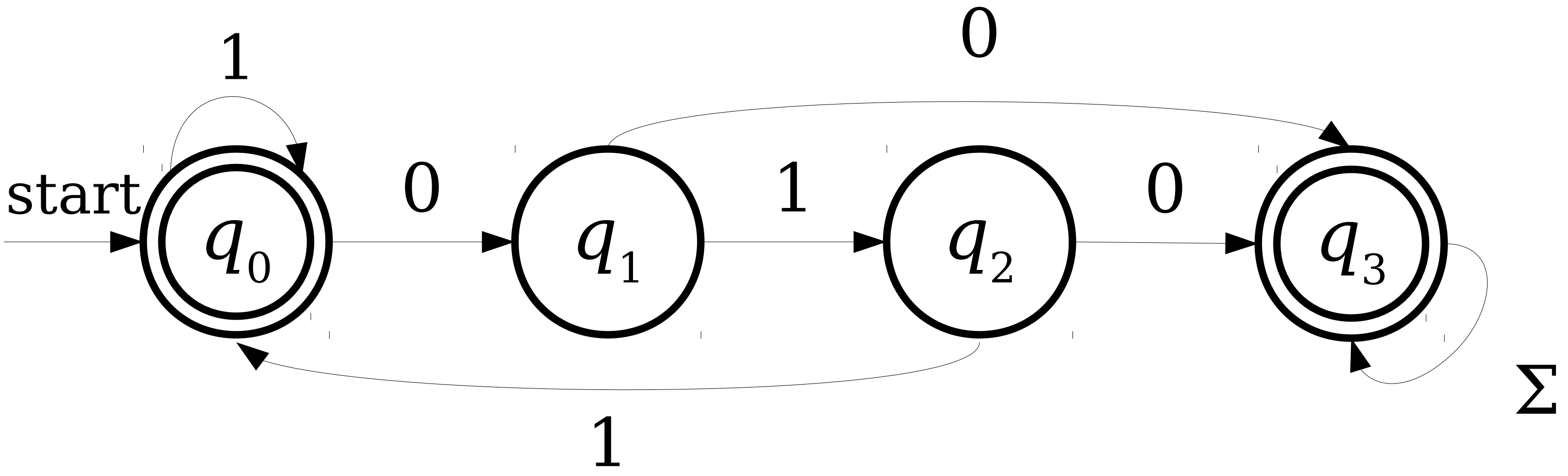


DFAs

- 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)

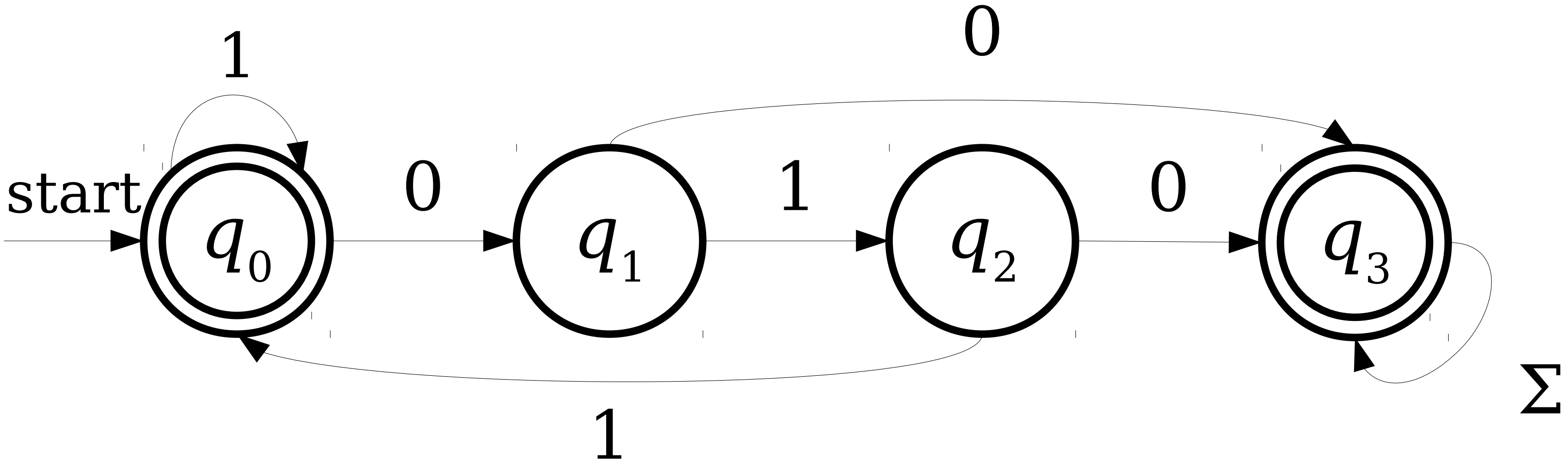
	0	1
q_0	q_0	q_1
q_1	q_2	q_3
q_2	q_0	q_3
q_3	q_3	q_3

(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

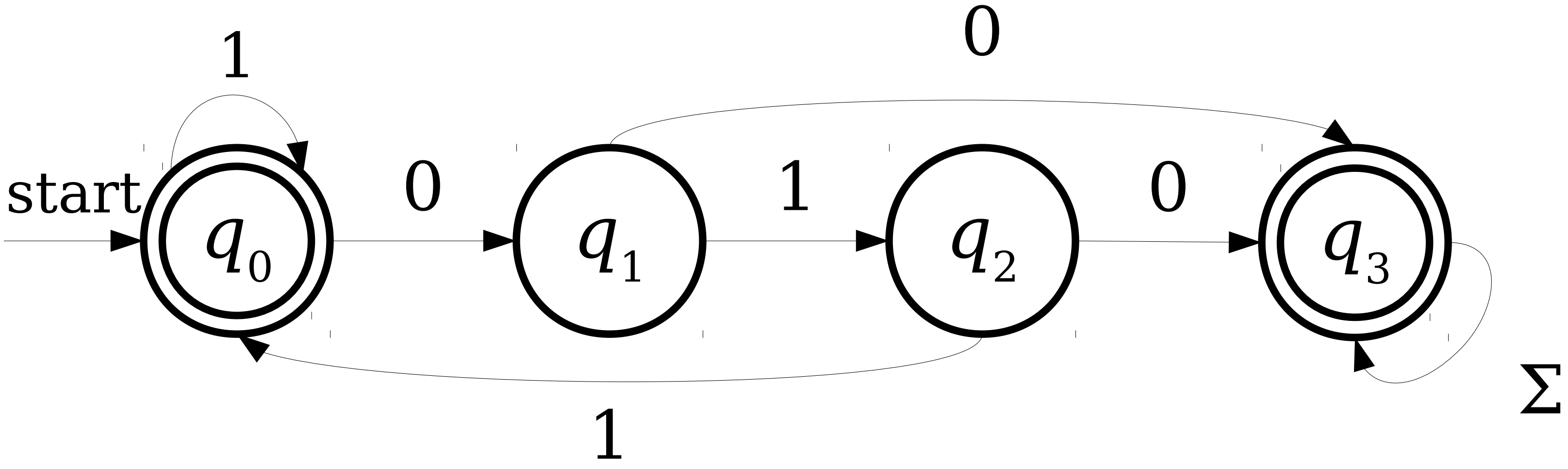
Answer at **PollEv.com/cs103** or text **CS103** to **22333** once to join, then **A**, **B**, **C**, or **D (none of the above)**.

Tabular DFAs



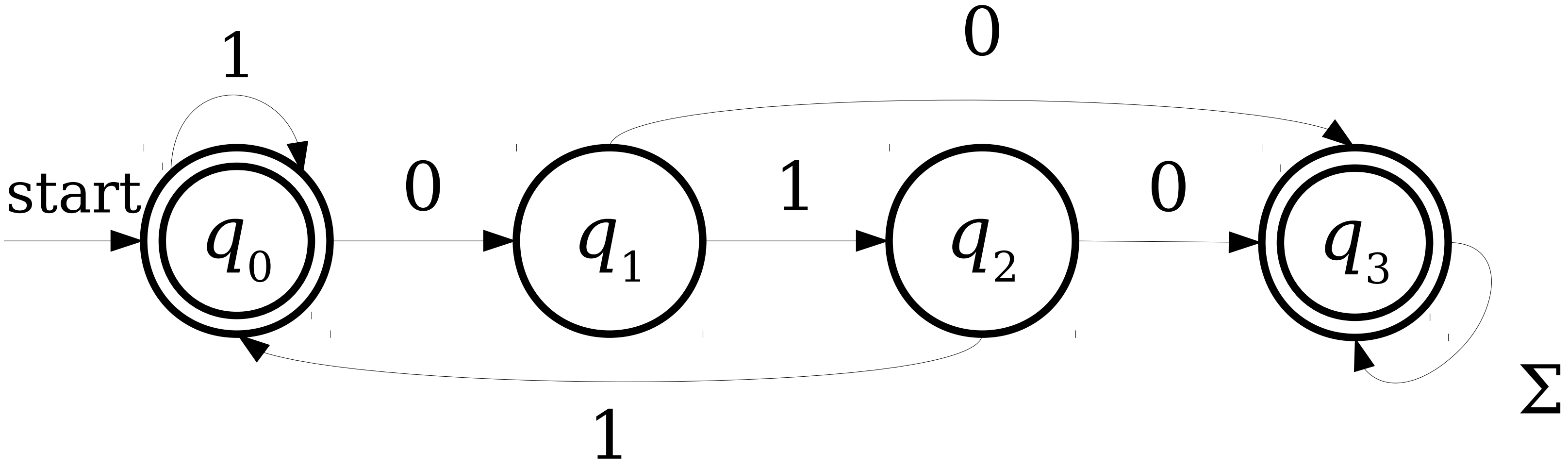
	0	1
q_0		
q_1		
q_2		
q_3		

Tabular DFAs



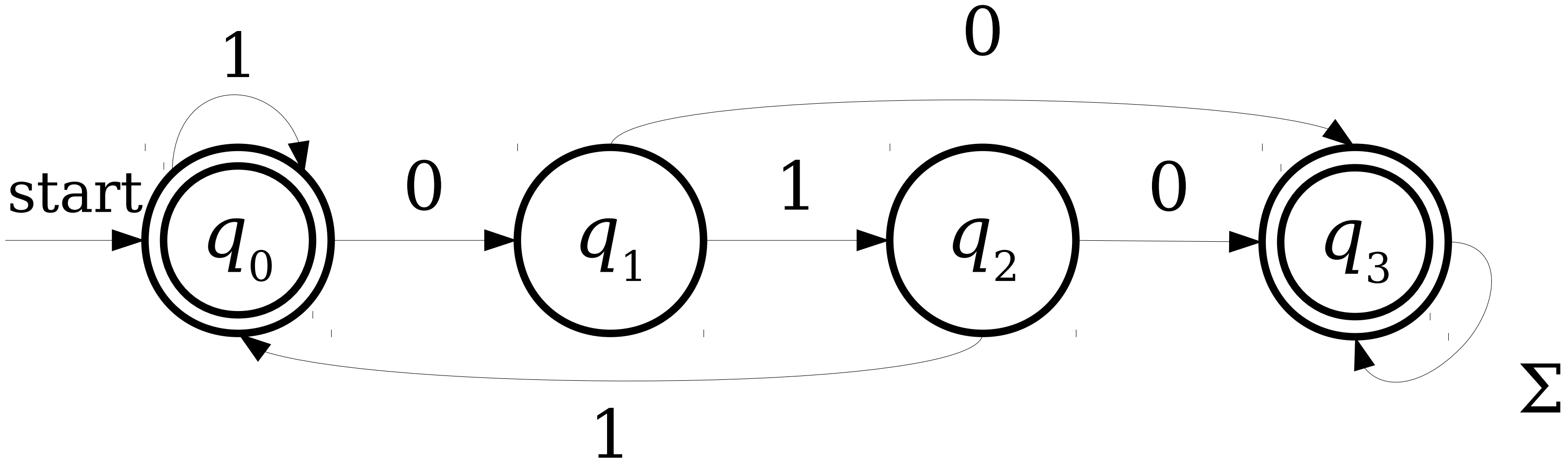
	0	1
q_0	q_1	q_0
q_1	q_3	q_2
q_2	q_3	q_0
q_3	q_3	q_3

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

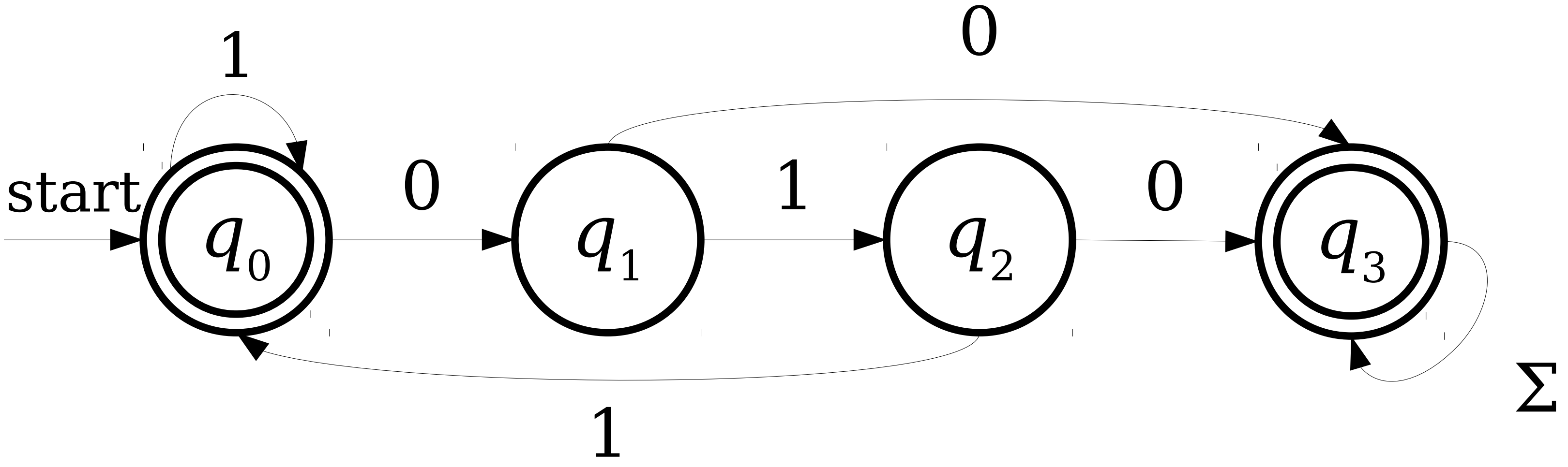
Tabular DFAs



	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

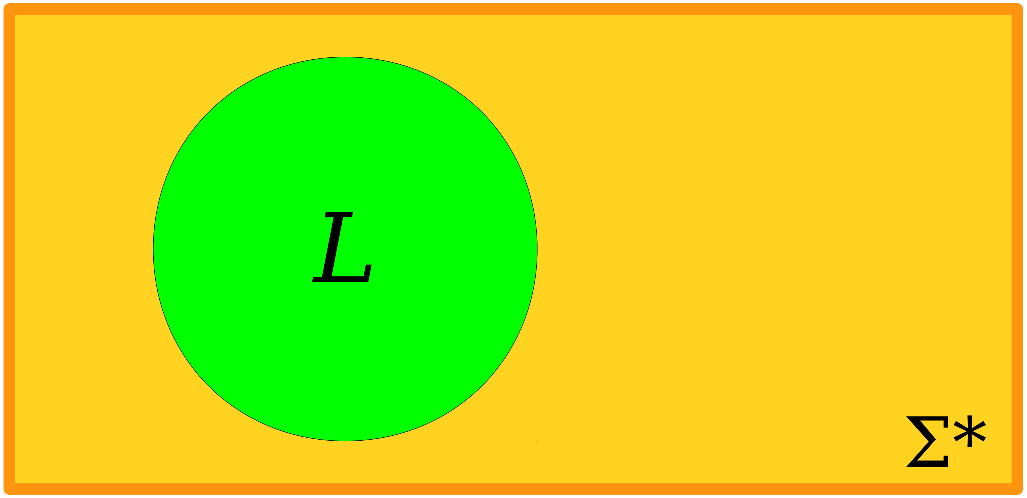
- Given a language $L \subseteq \Sigma^*$, the ***complement*** of that language (denoted $\overline{L}$) is the language of all strings in Σ^* that aren't in L.
- Formally:

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

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:

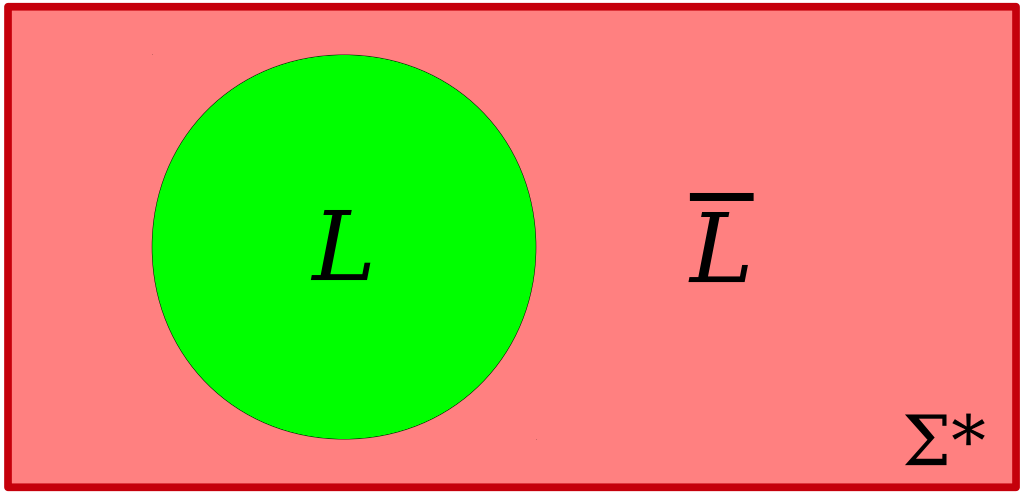
$$\overline{L} = \Sigma^* - L$$



The Complement of a Language

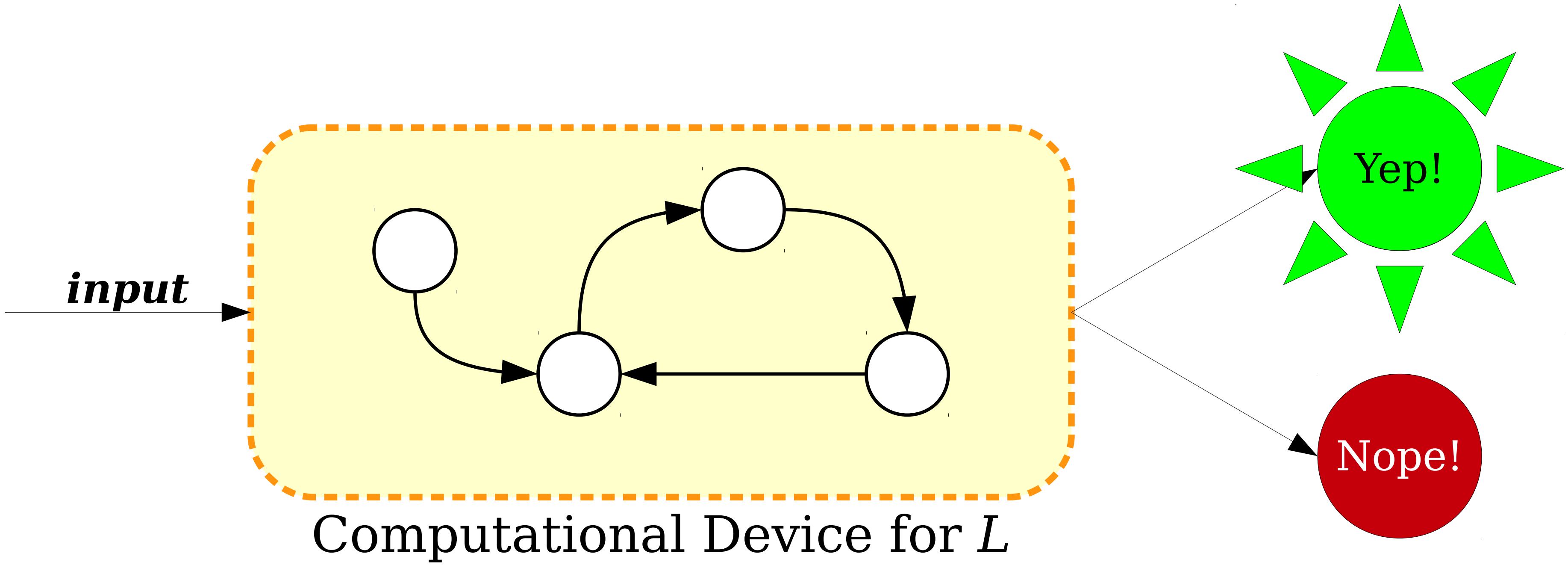
- Given a language $L \subseteq \Sigma^*$, the ***complement*** of that language (denoted $\overline{L}$) is the language of all strings in Σ* that aren't in L.
- Formally:

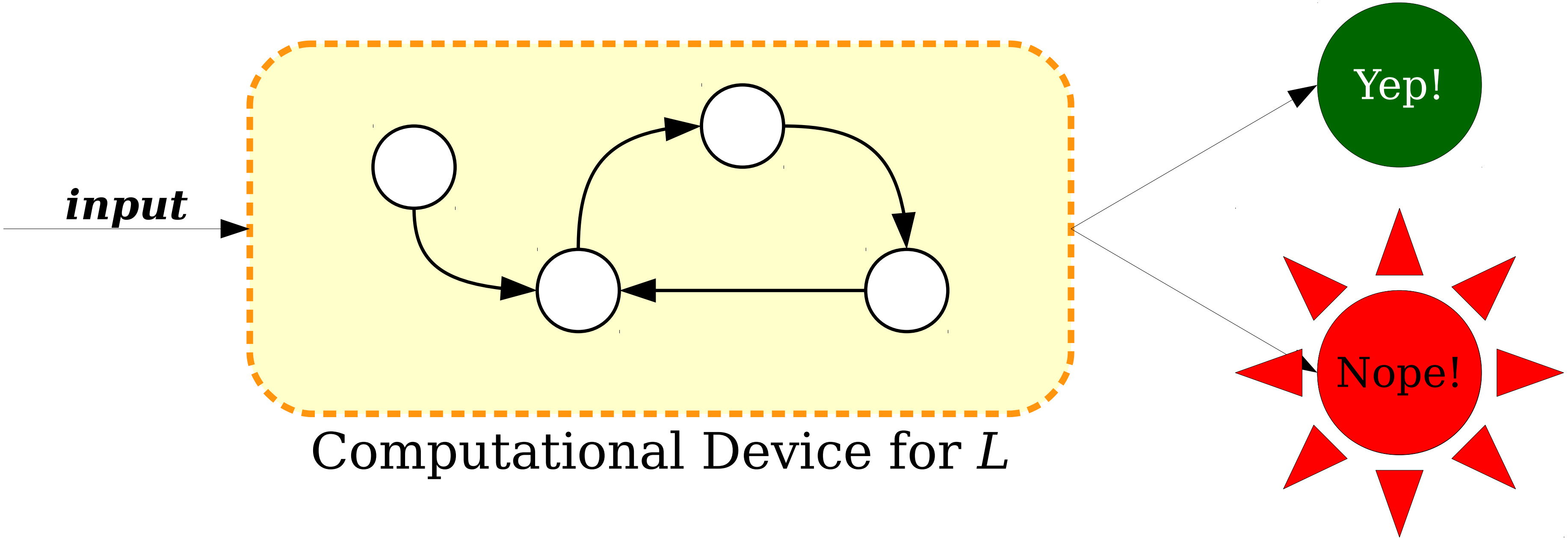
$$\overline{L} = \Sigma^* - L$$

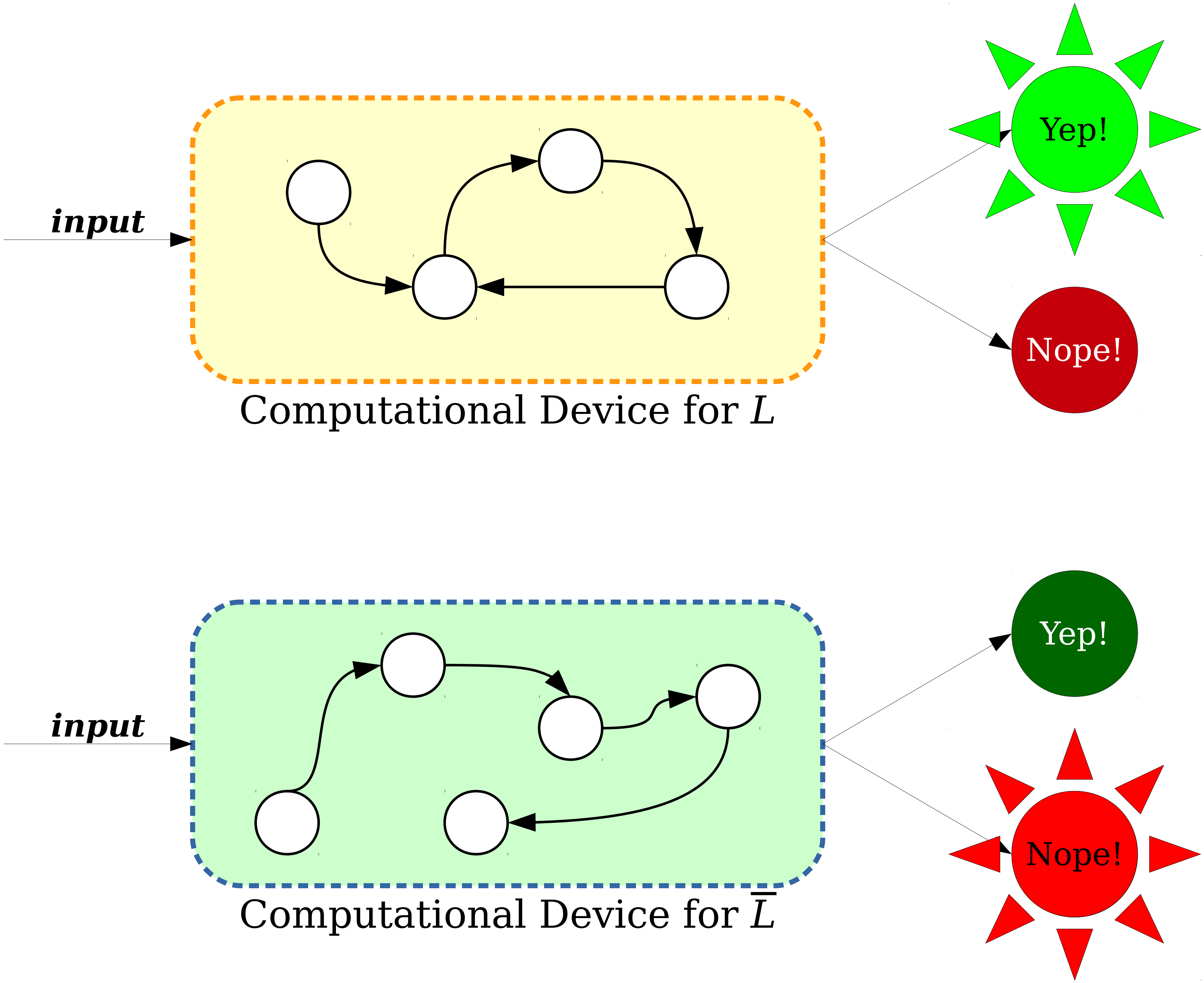


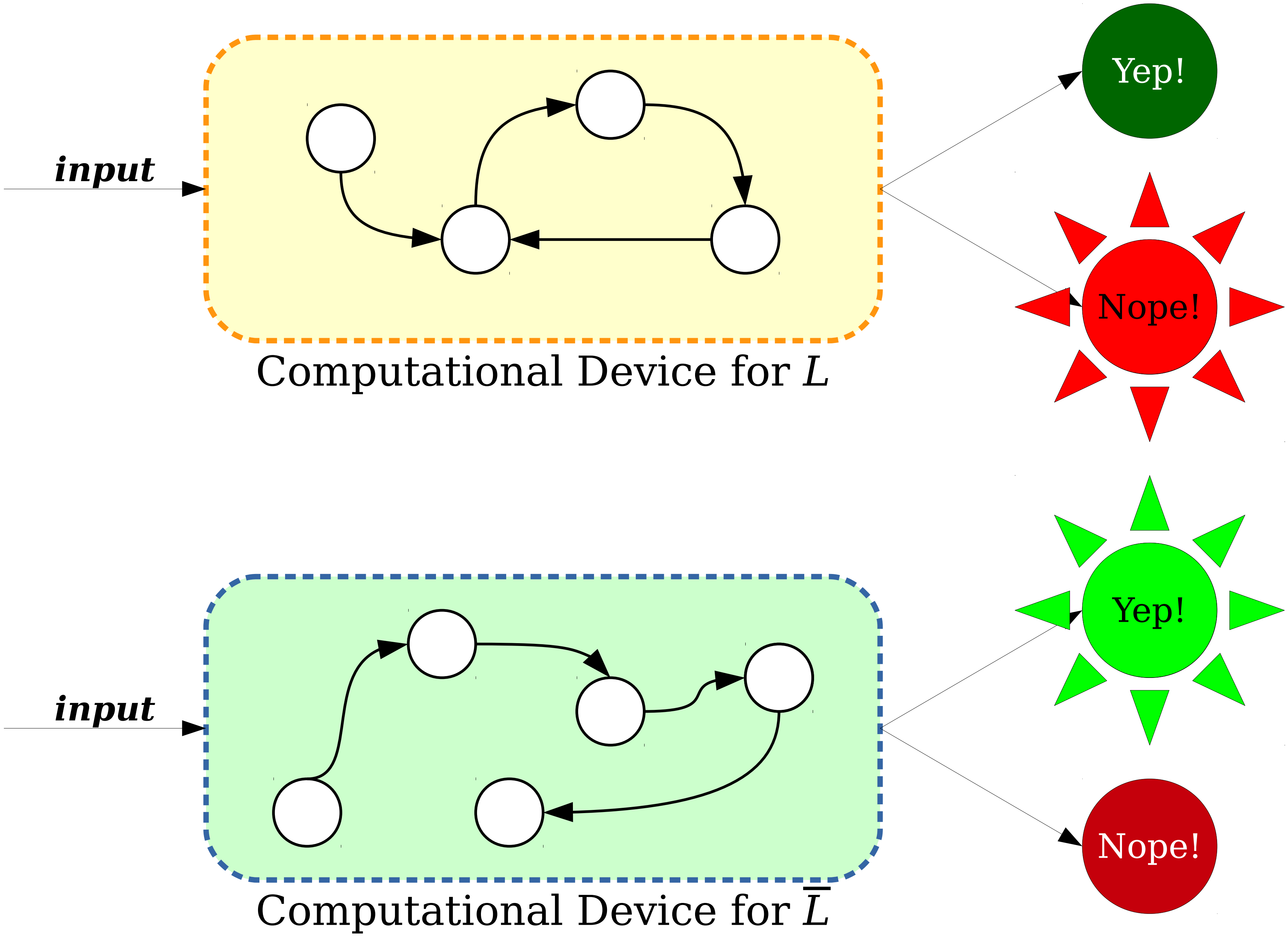
Complements of Regular Languages

- 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.


input
Yep!
Nope!
Computational Device for L

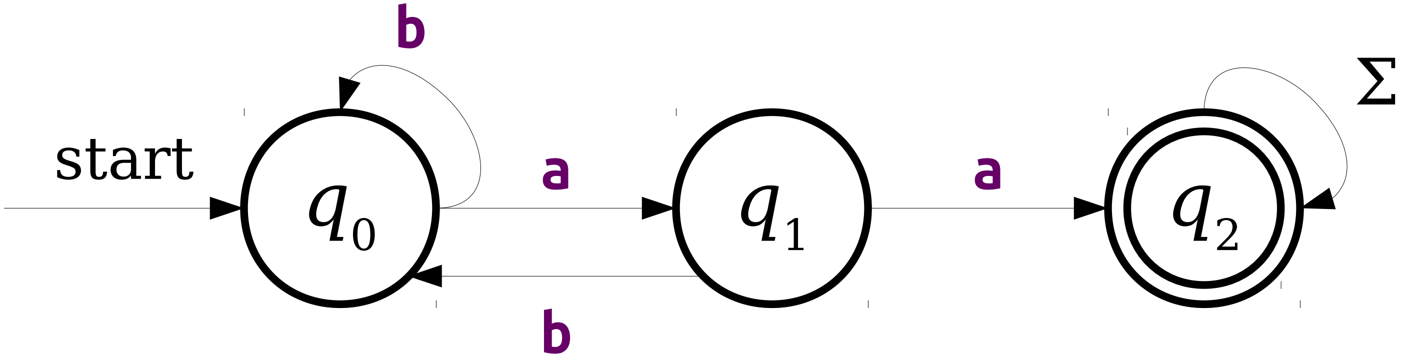

input
Computational Device for L
Yep!
Nope!


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

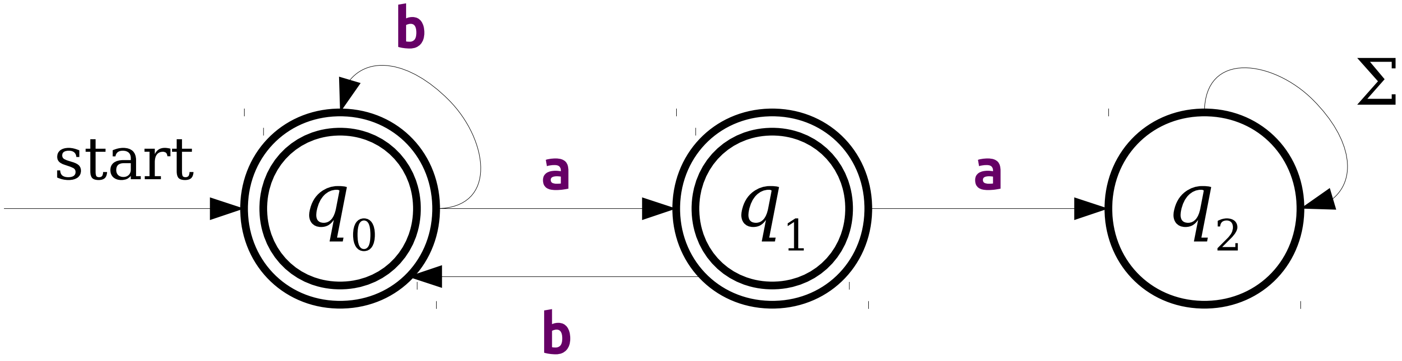

input
Computational Device for L
Yep!
Nope!
input
Computational Device for $\overline{L}$
Yep!
Nope!

Complementing Regular Languages

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

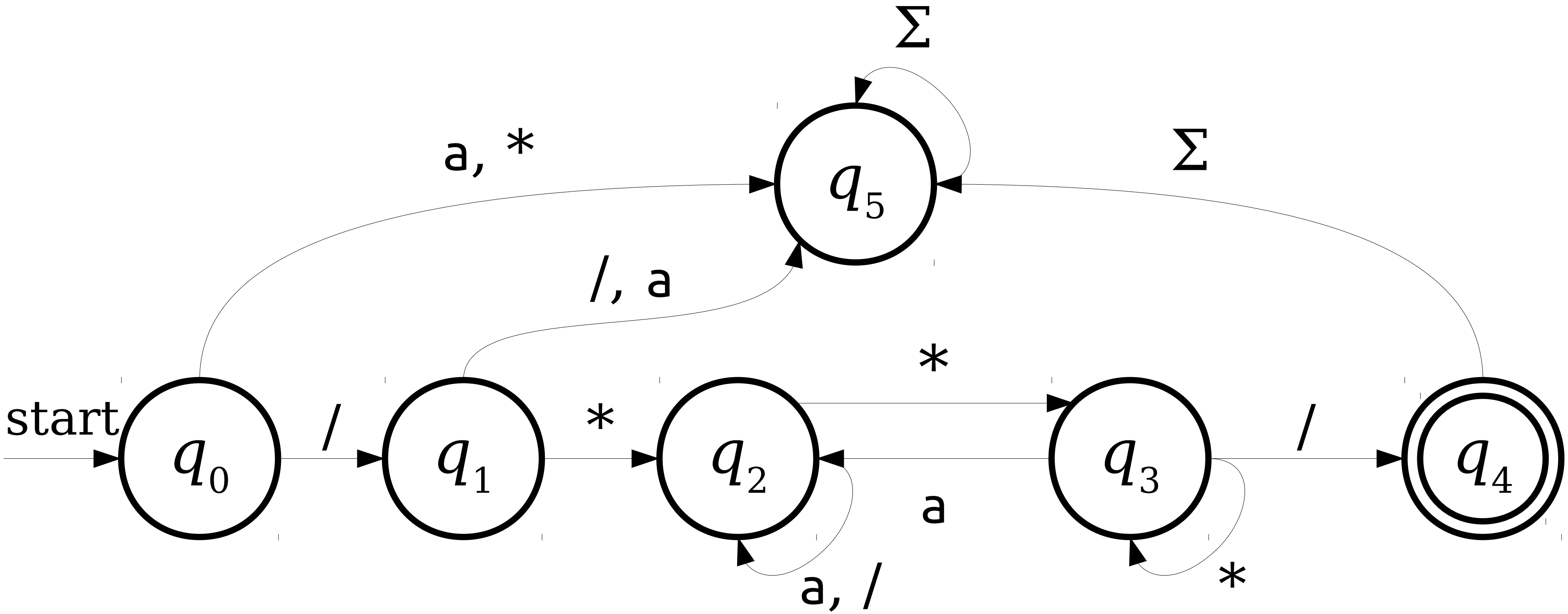


$\overline{L} = \{\ w \in \{\texttt{a}, \texttt{b}\}^* \mid w$ ***does not*** contain **aa** as a substring $\}$



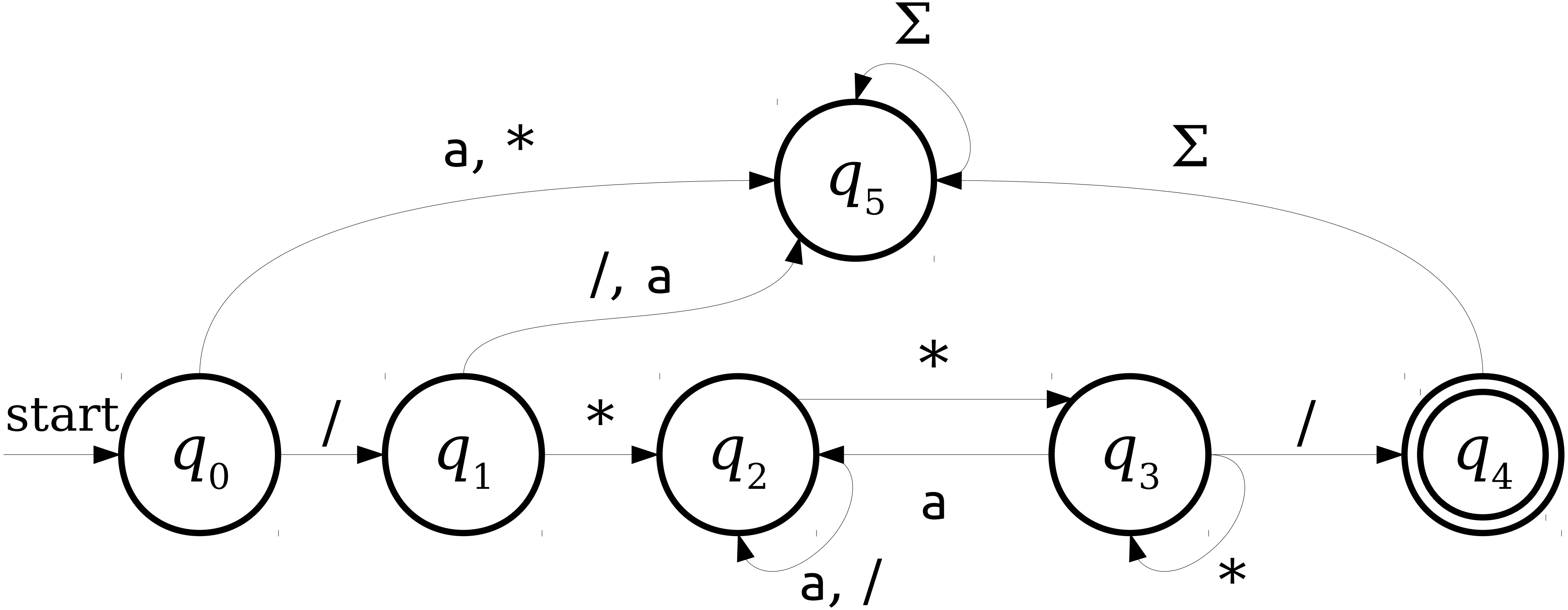
More Elaborate DFAs

L = { w ∈ {**a**, *****, **/**}* | w represents a C-style comment }



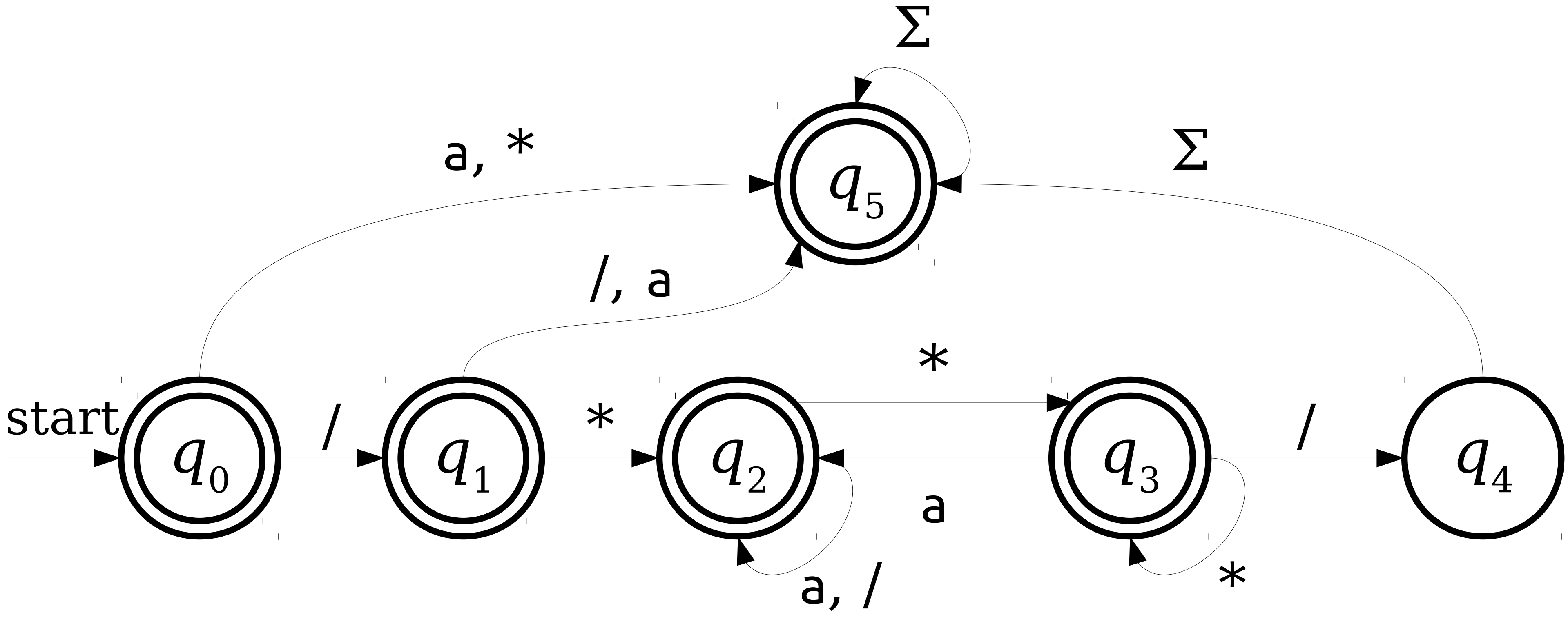
More Elaborate DFAs

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



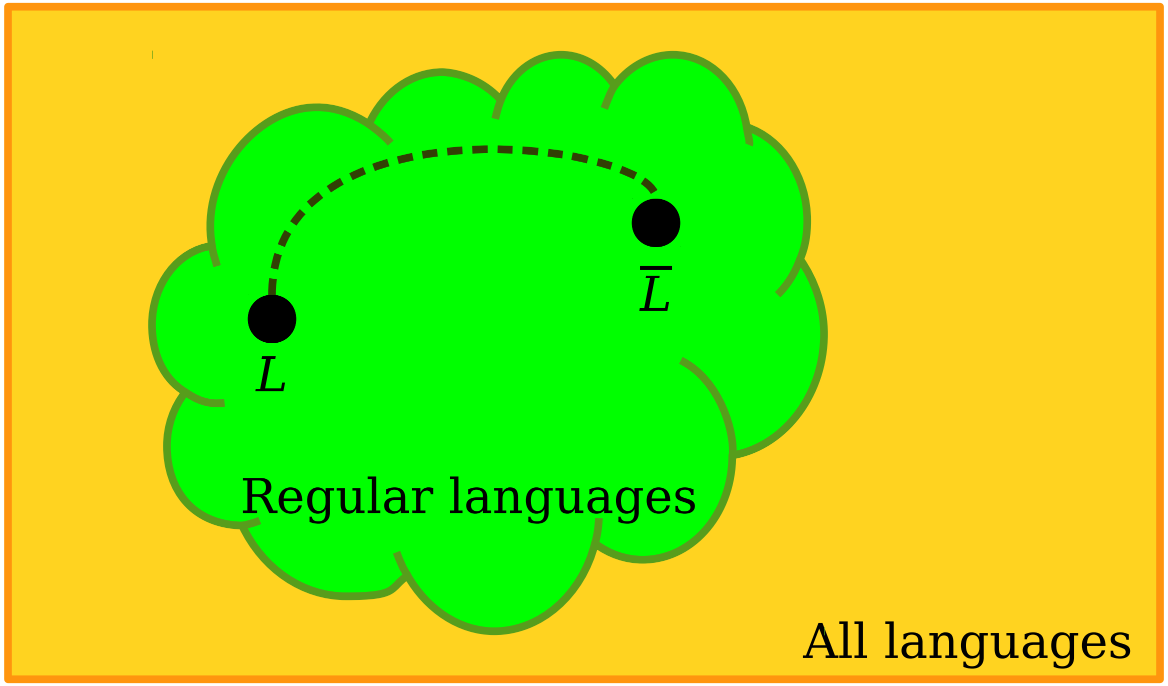
More Elaborate DFAs

$\overline{L}$ = { 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***.



Time-Out For Announcements!

Additional Practice

- Looking to improve your performance in CS103? We’ve released two handouts:
    - Handout 31: How to Improve in CS103
    - Handout 32: Extra Practice Problems 2
- Set aside a few minutes each day to get some additional practice. That will add up extremely quickly!

treehacks

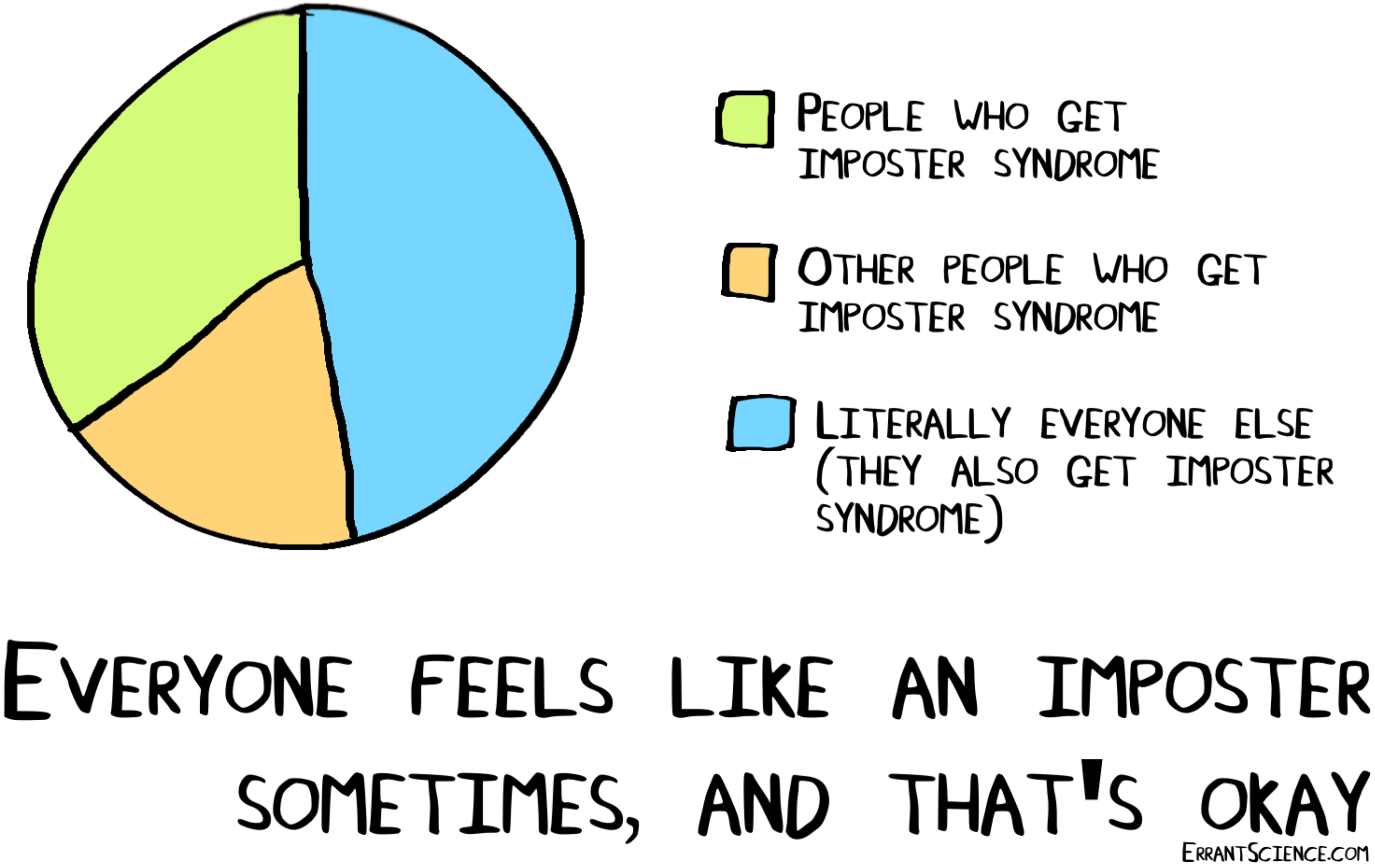
STANFORD’S NATIONAL HACKATHON

THIS WEEKEND FEB 16-18

HUANG ENGINEERING CENTER

BEGINNERS WELCOME!

Ever felt you weren't good enough to be in STEM? Afraid of being "found out" because you don't think you belong?


🟩 People who get imposter syndrome
🟧 Other people who get imposter syndrome
🟦 Literally everyone else (they also get imposter syndrome)
Everyone feels like an imposter sometimes, and that's okay
ErrantScience.com

Learn how to combat Imposter Phenomenon at a very special workshop led by Dr. Nicole Cabrera Salazar! Lunch will be served.

~Wed. 2/14, 11:30 am - 1:30 pm, PAB 232~

Back to CS103!

NFAs

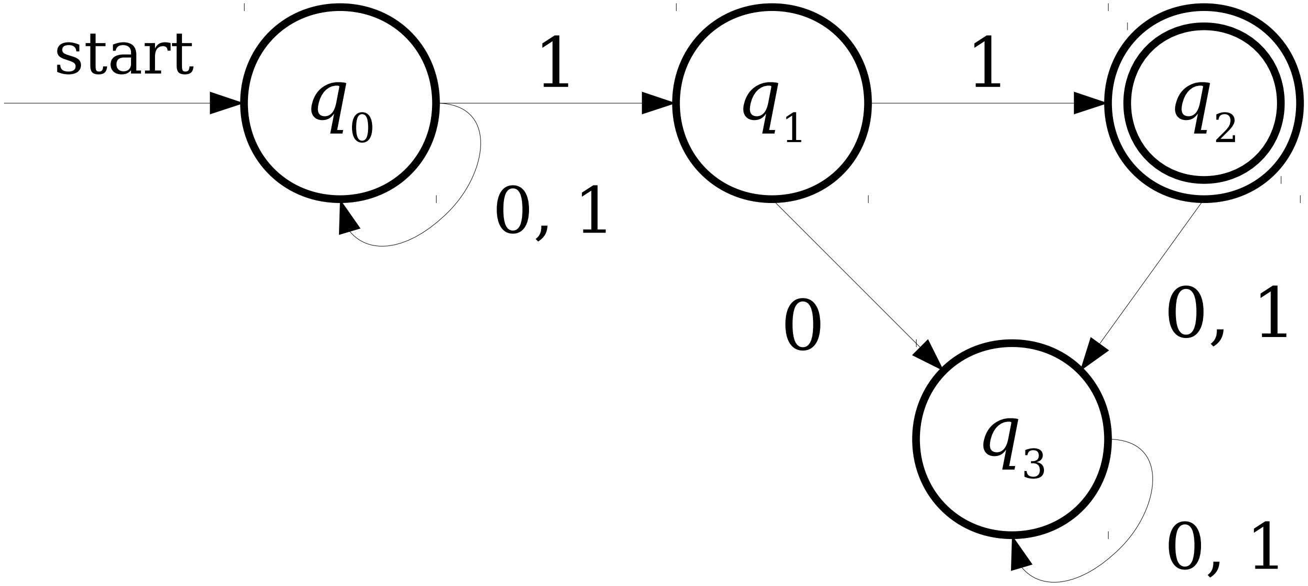
NFAs

- An ***NFA*** is a
  - ***N***ondeterministic
  - ***F***inite
  - ***A***utomaton
- Structurally similar to a DFA, but represents a fundamental shift in how we'll think about computation.

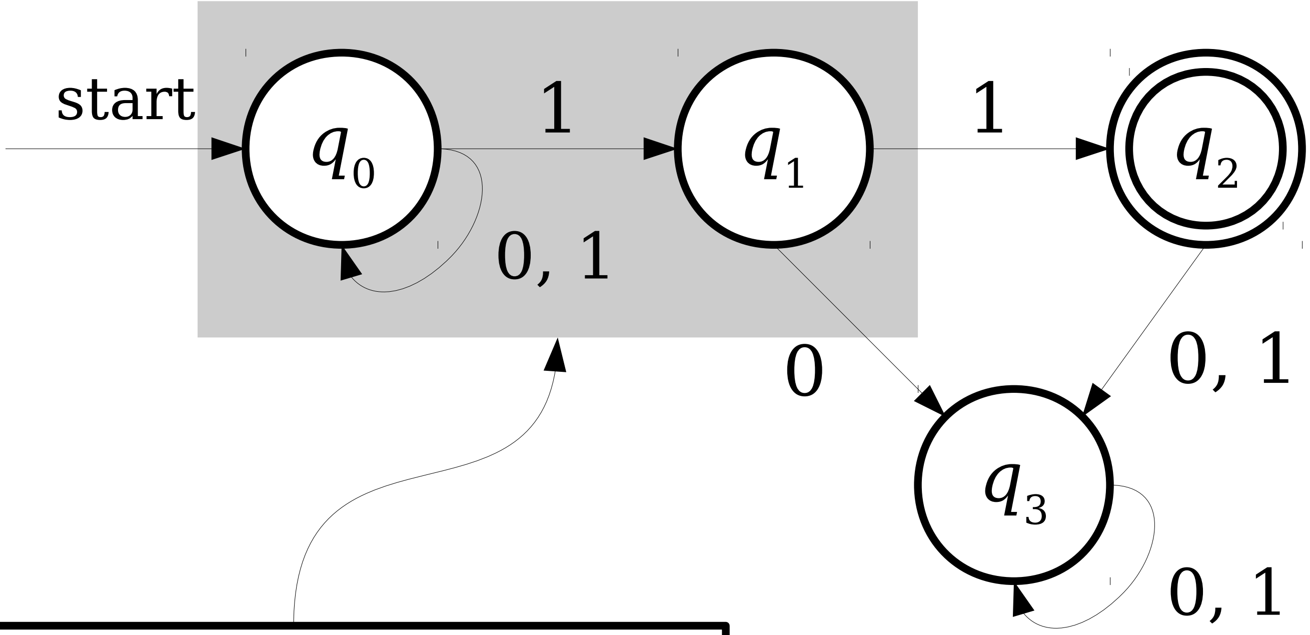
(Non)determinism

- A model of computation is ***deterministic*** if at every point in the computation, there is exactly one choice that can make.
- The machine accepts if that series of choices leads to an accepting state.
- A model of computation is ***nondeterministic*** if the computing machine may have multiple decisions that it can make at one point.
- The machine accepts if ***any*** series of choices leads to an accepting state.
  - (This sort of nondeterminism is technically called ***existential nondeterminism***, the most philosophical-sounding term we'll introduce all quarter.)

A Simple NFA

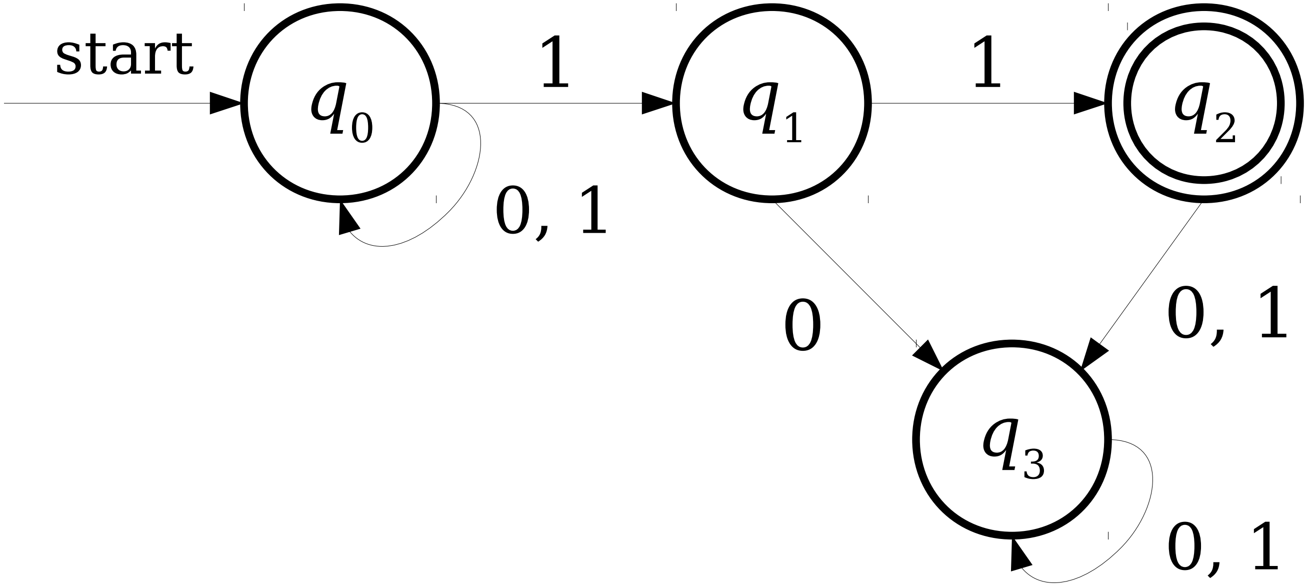


A Simple NFA



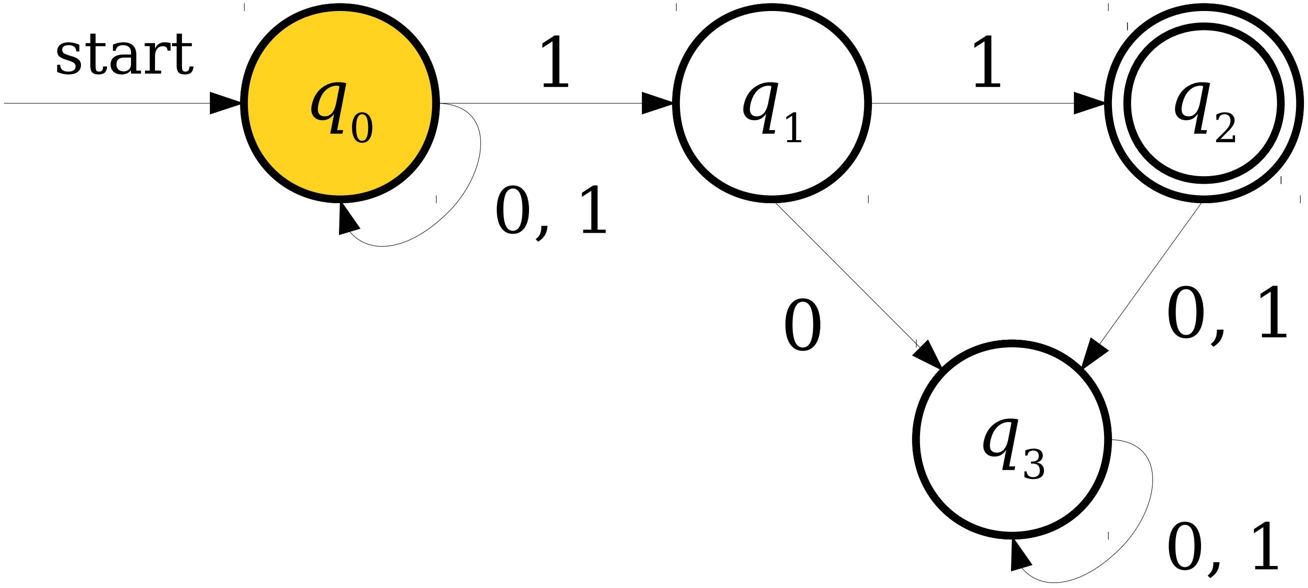
q_0 has two transitions defined on 1!

A Simple NFA



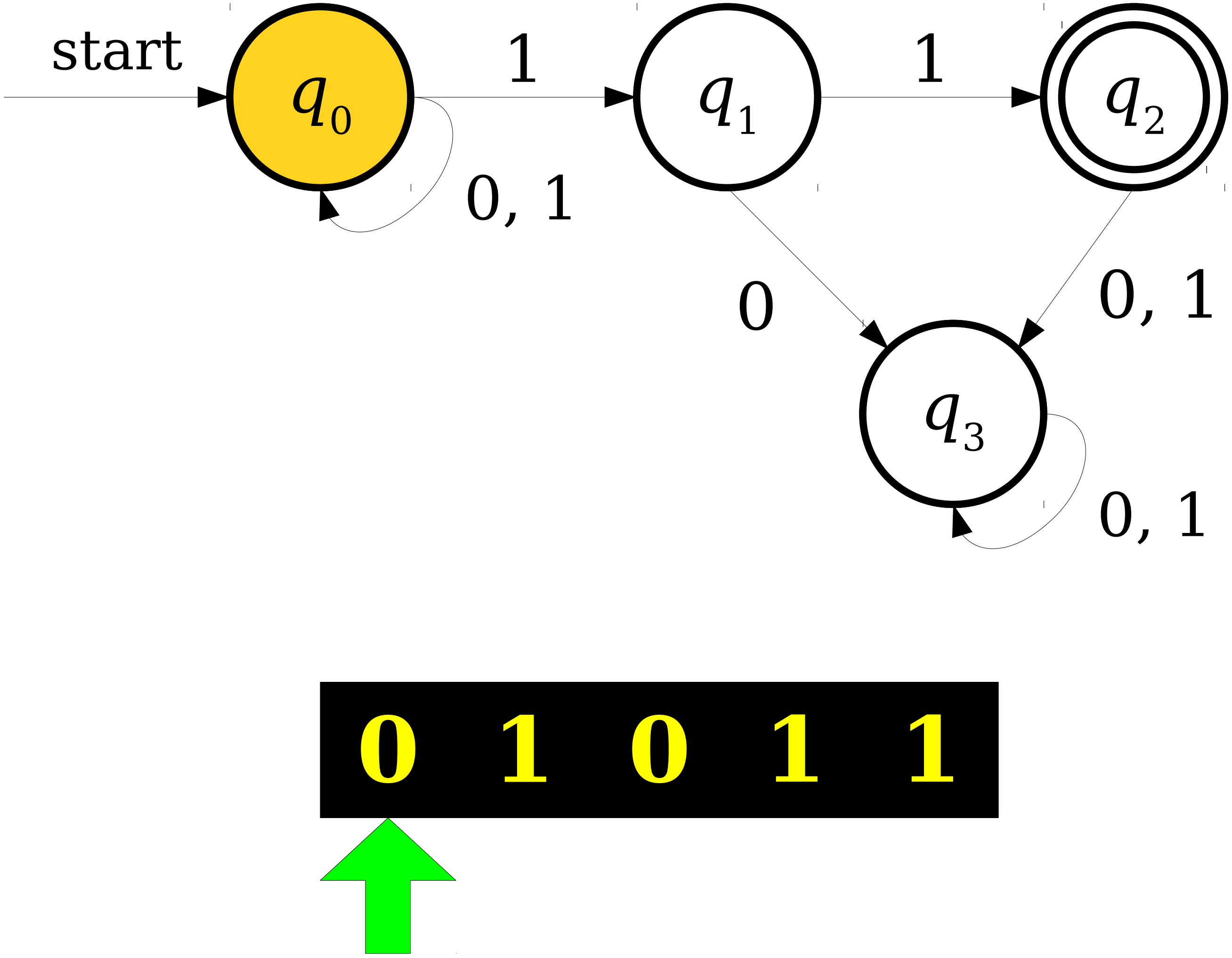
0 1 0 1 1

A Simple NFA

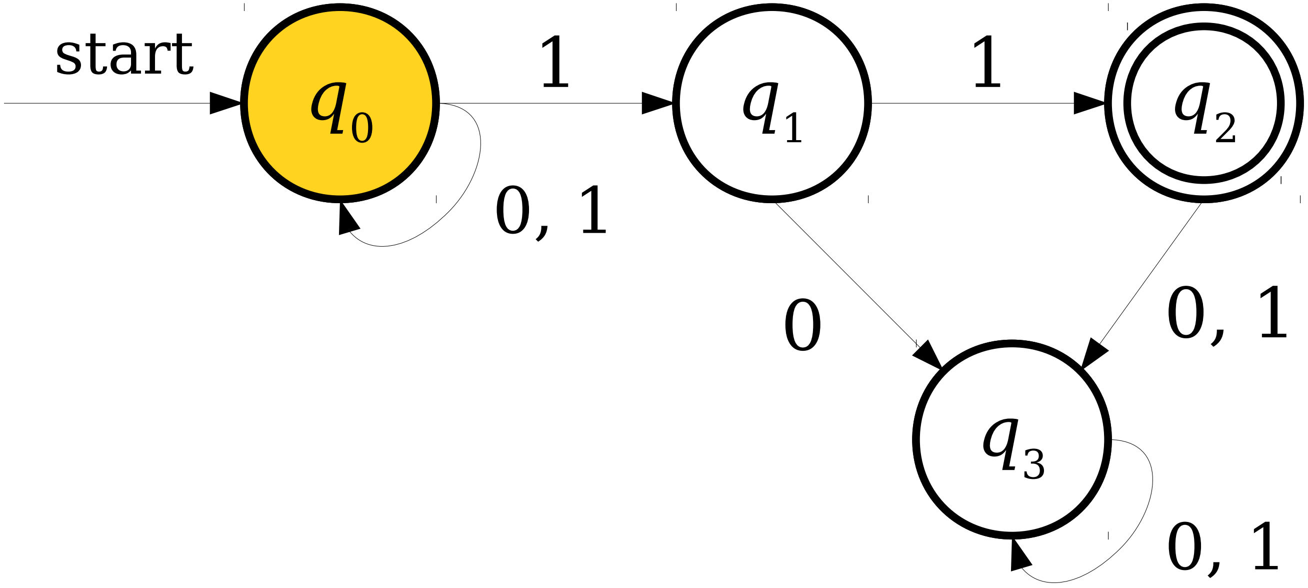


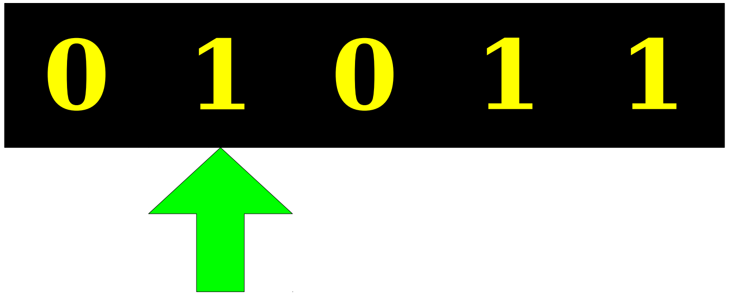
0 1 0 1 1

A Simple NFA

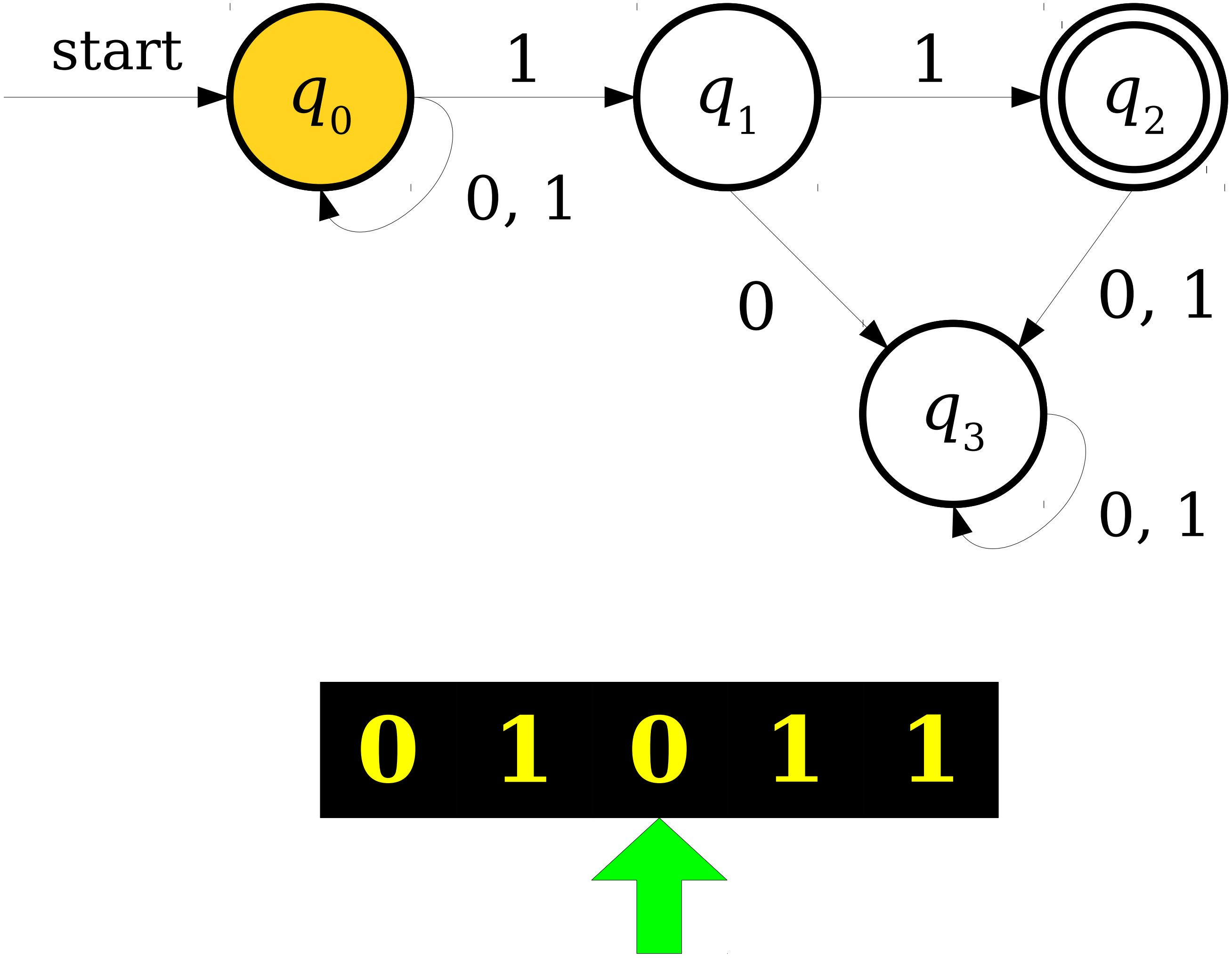


A Simple NFA

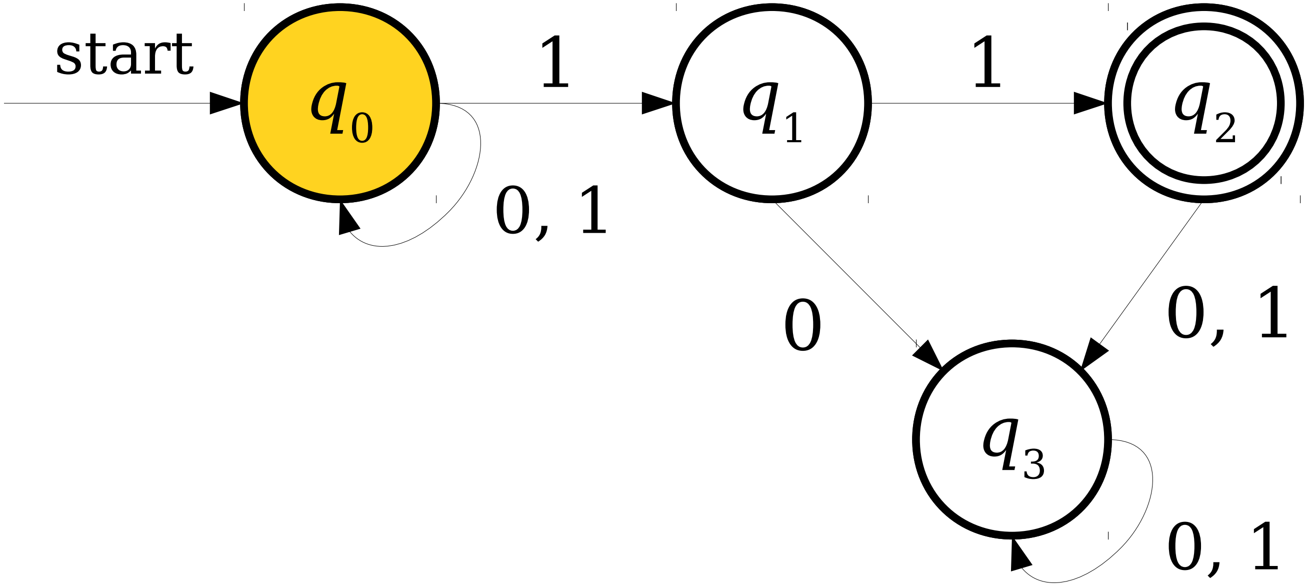


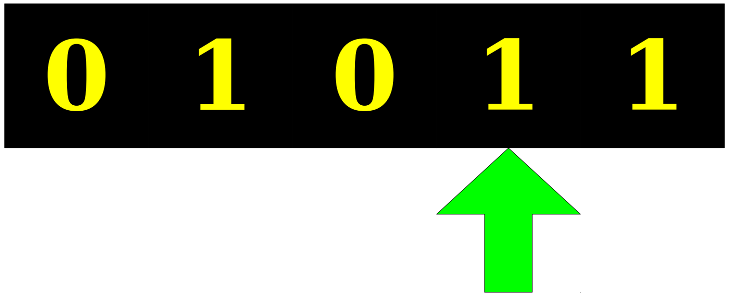


A Simple NFA

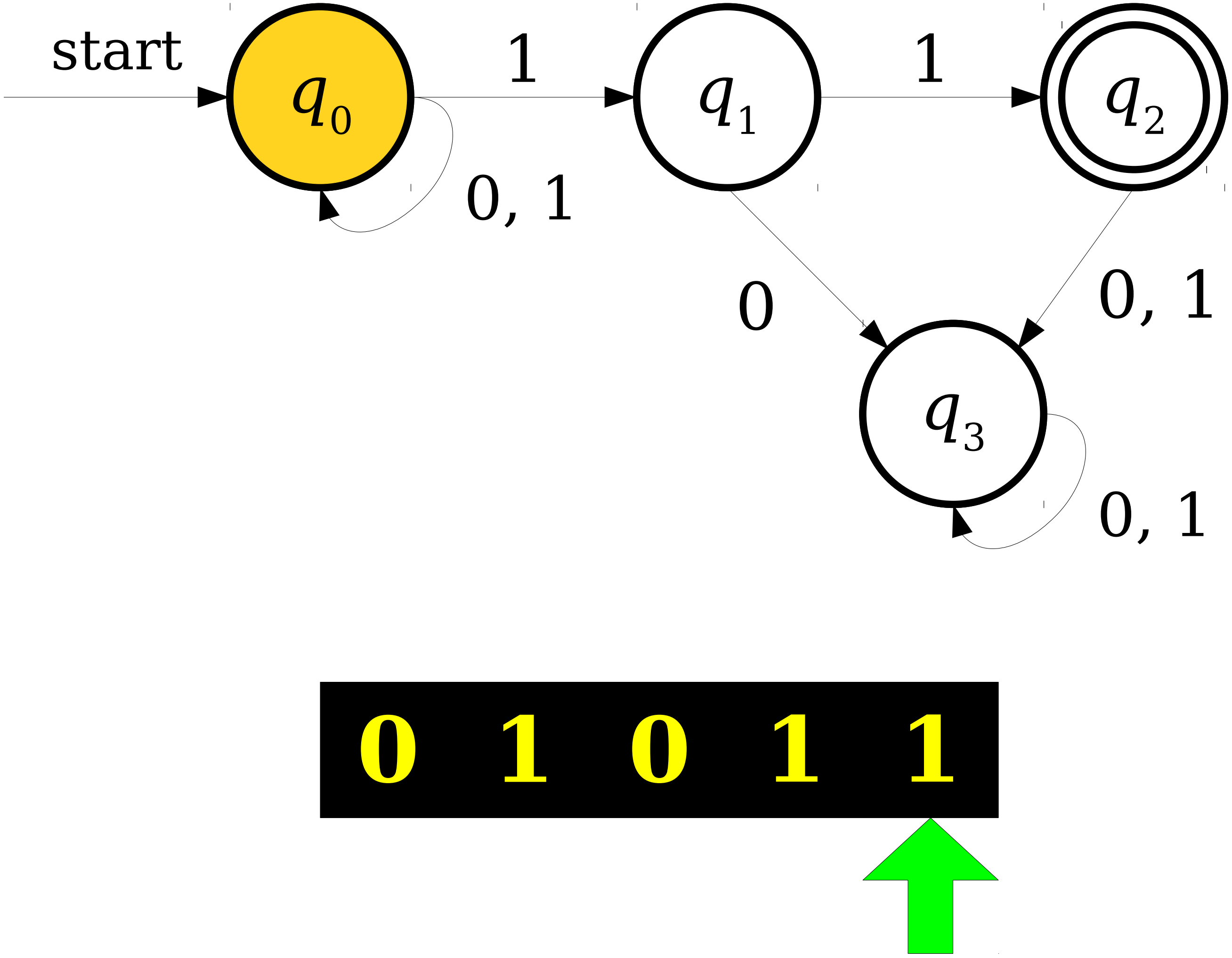


A Simple NFA

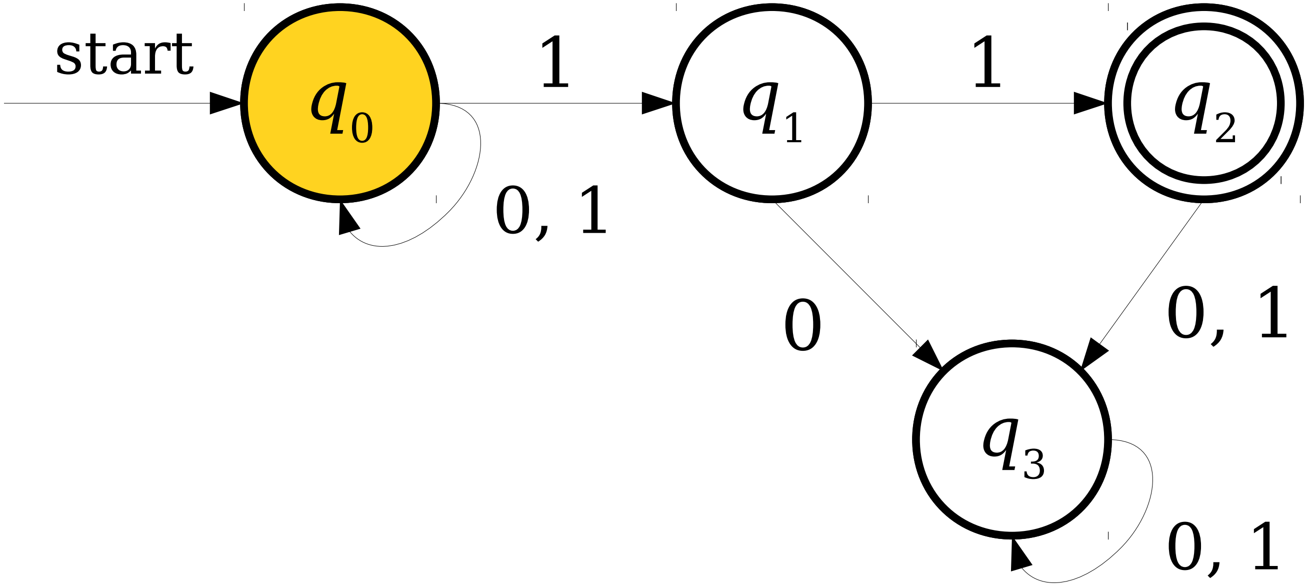




A Simple NFA

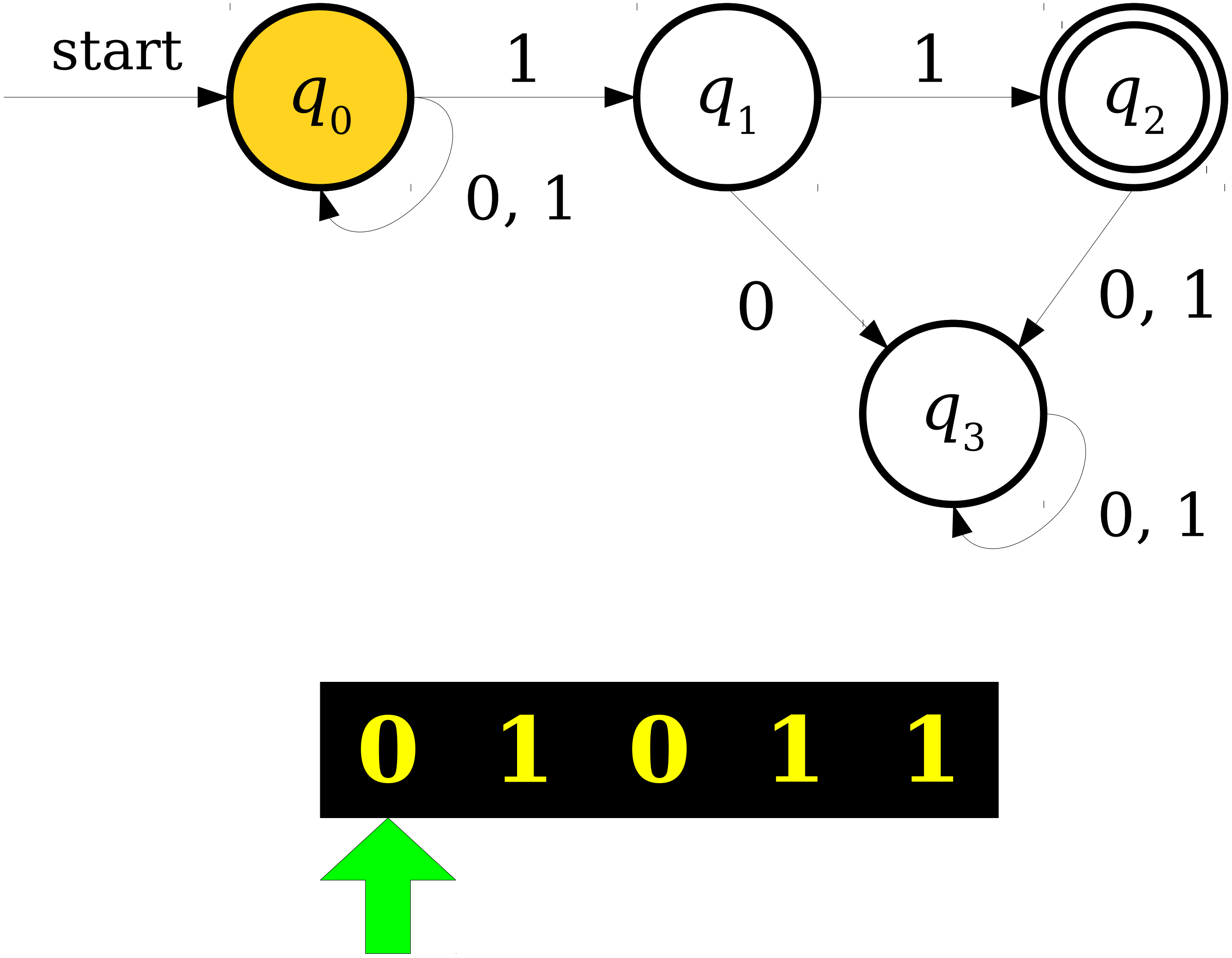


A Simple NFA

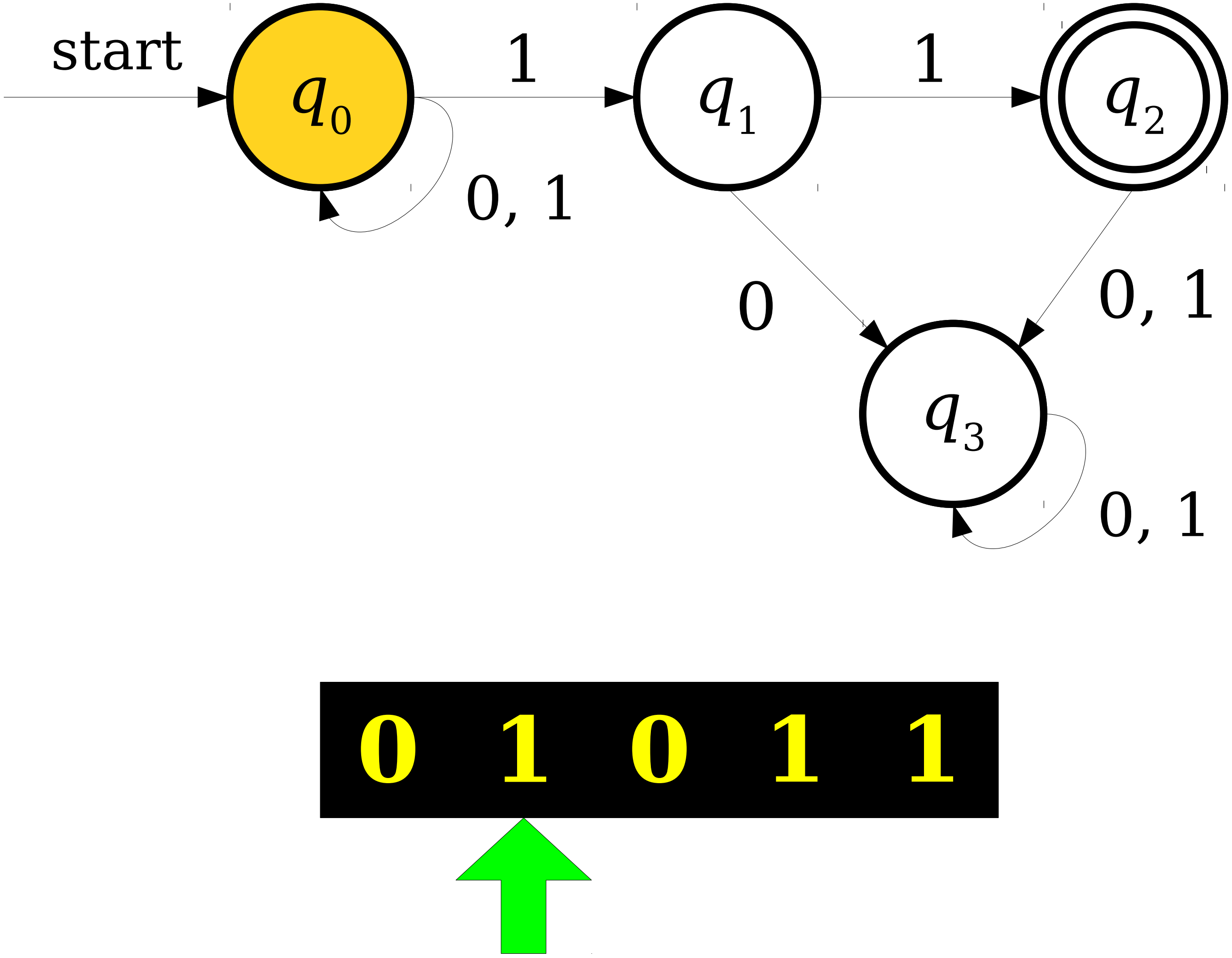


0	1	0	1	1

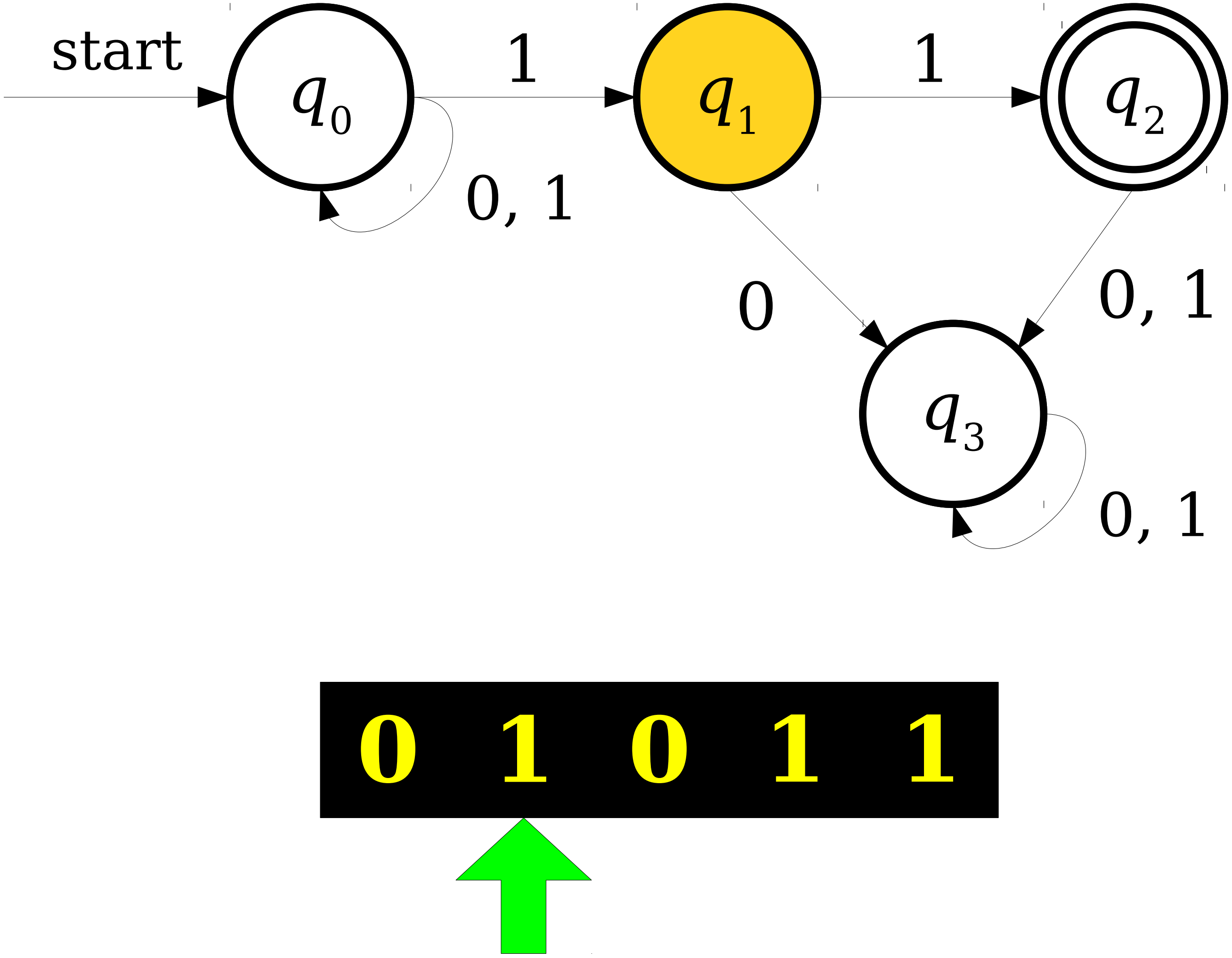
A Simple NFA



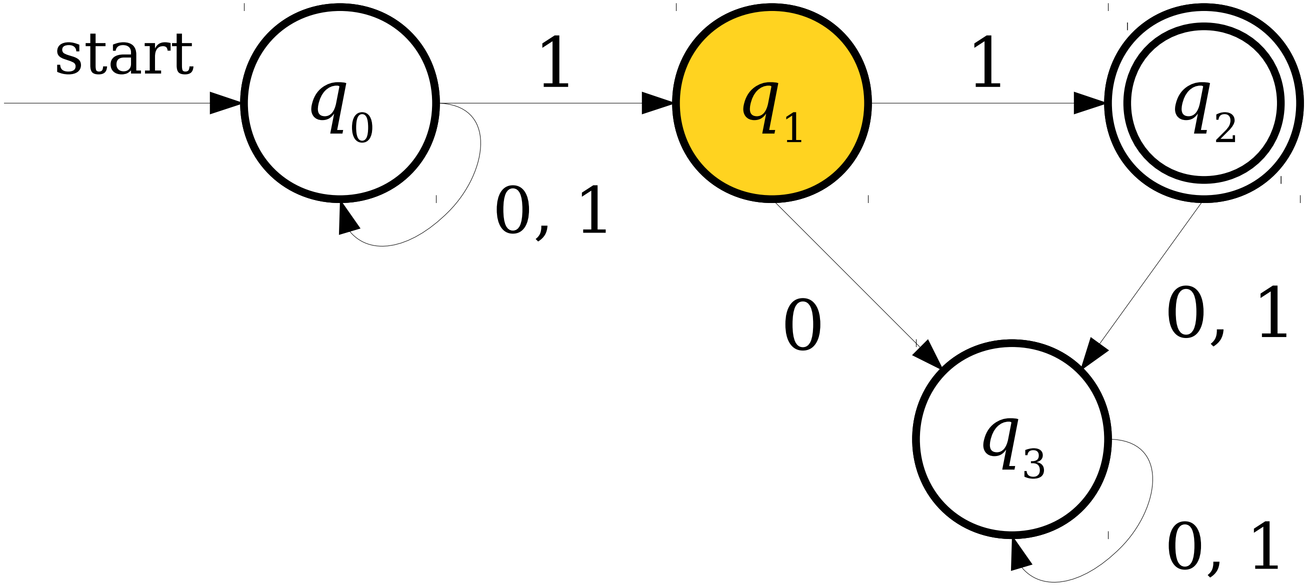
A Simple NFA



A Simple NFA

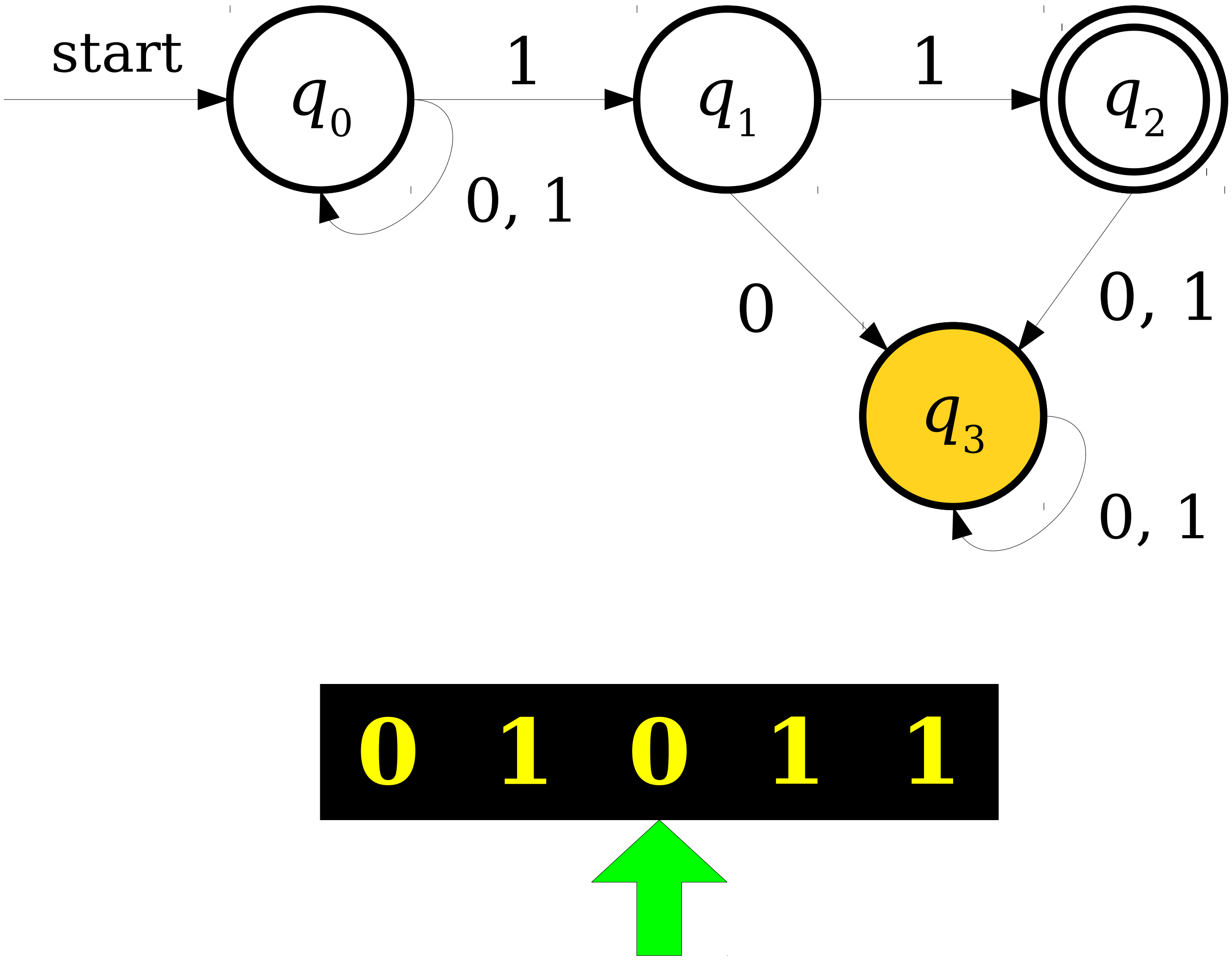


A Simple NFA

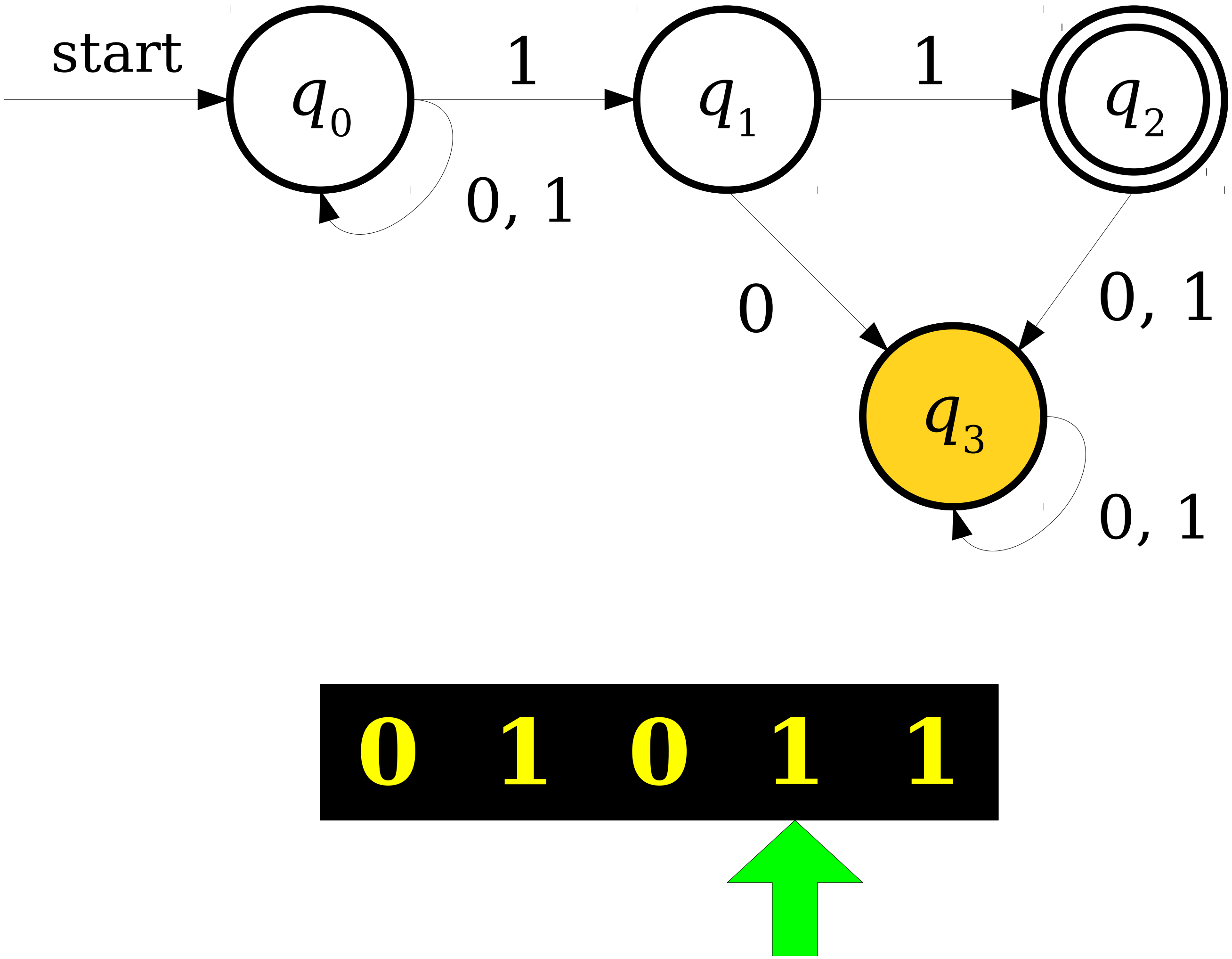


0 1 0 1 1

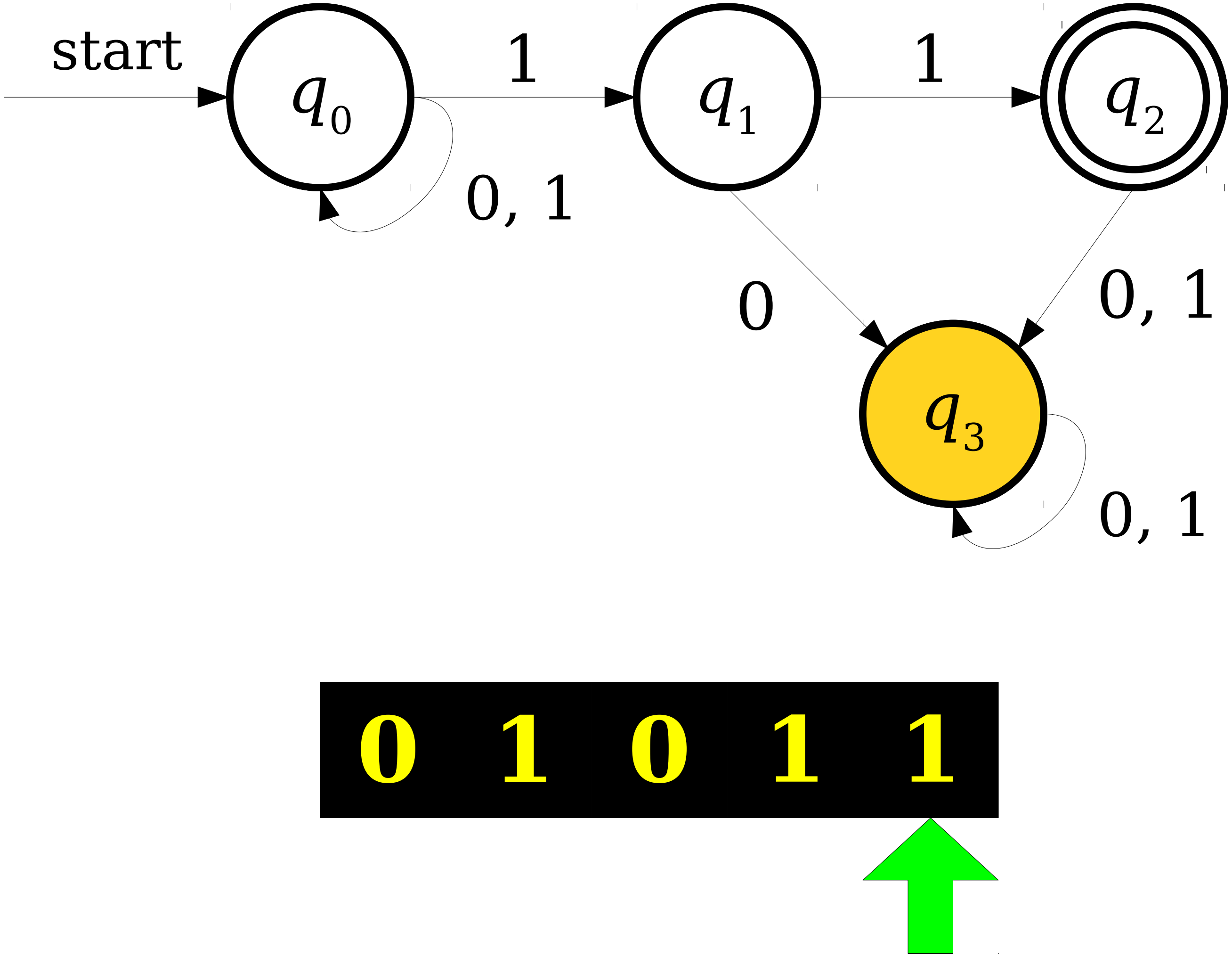
A Simple NFA



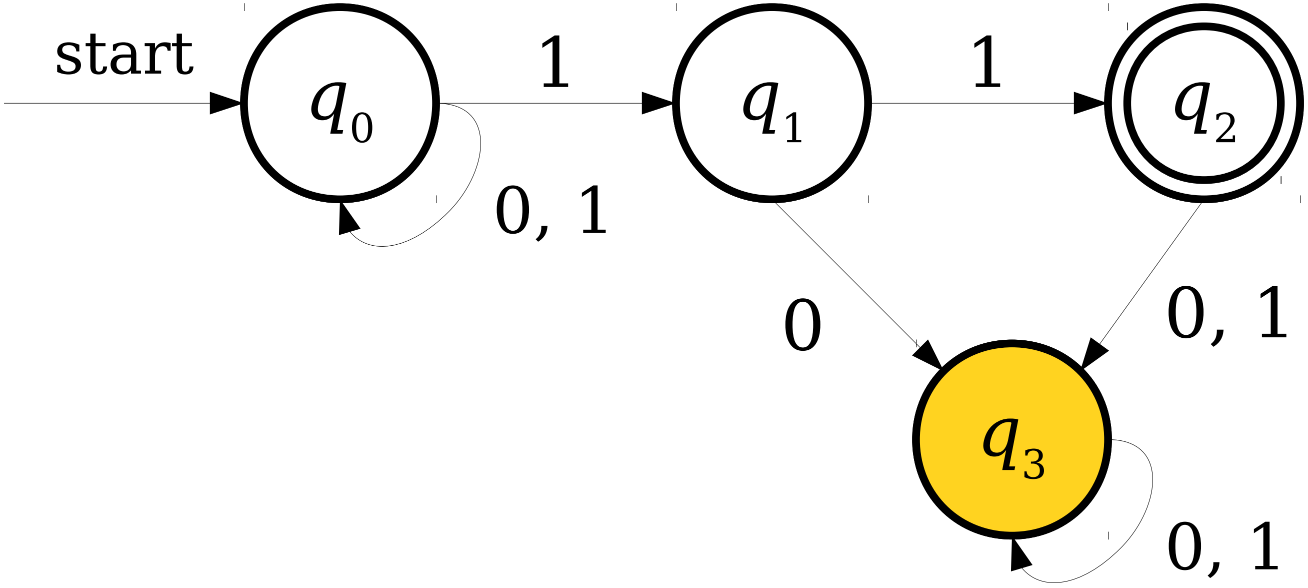
A Simple NFA



A Simple NFA

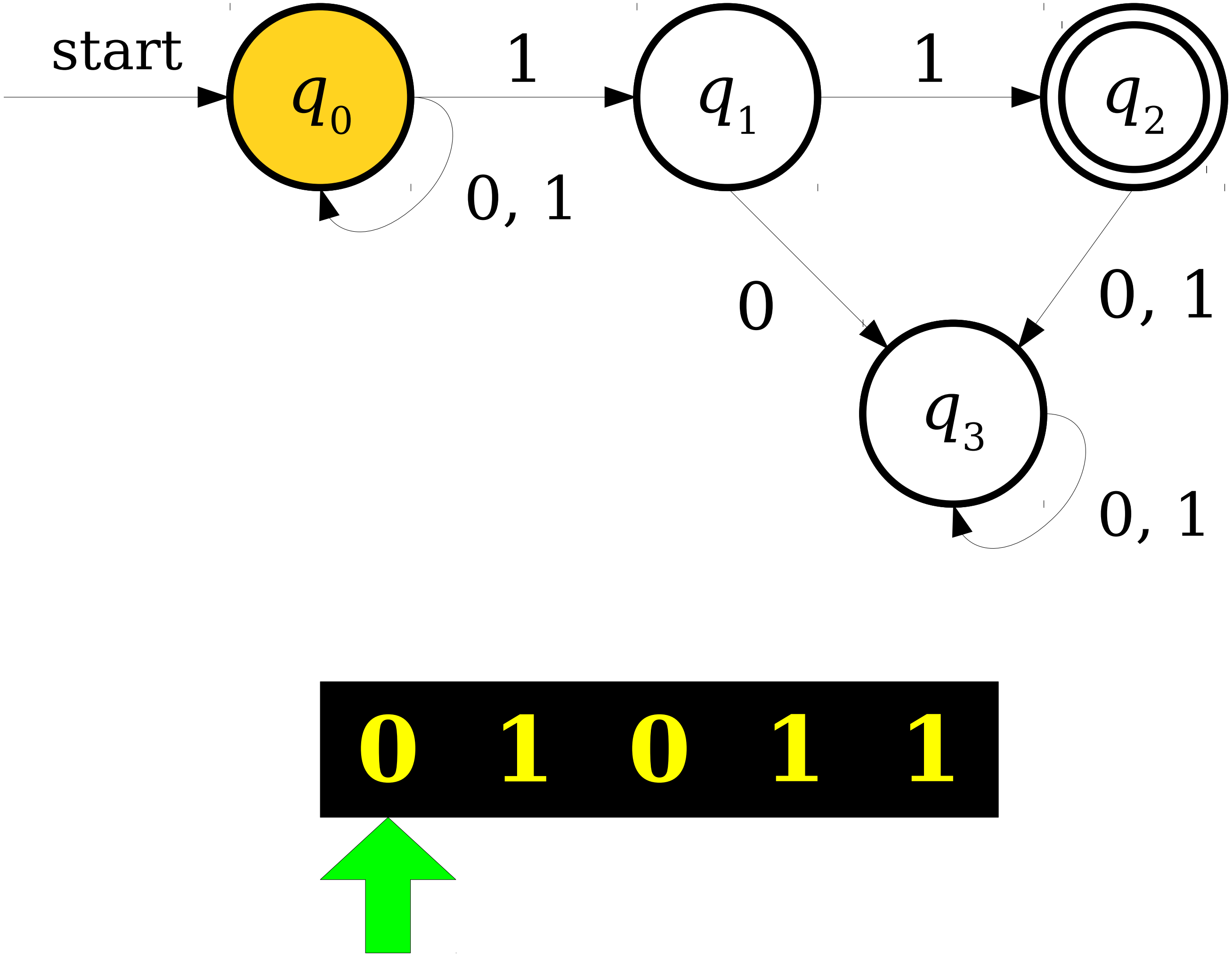


A Simple NFA

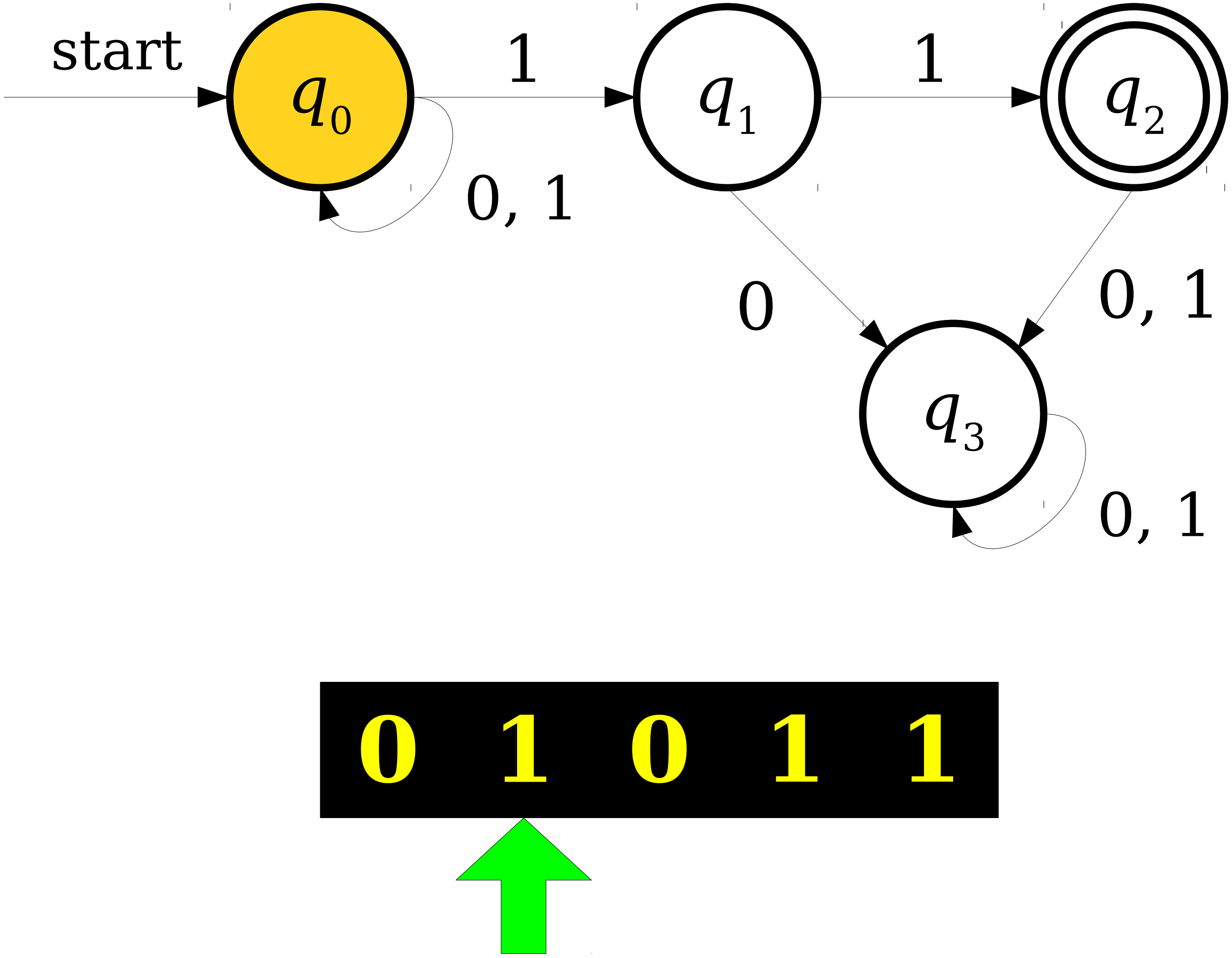


0 1 0 1 1

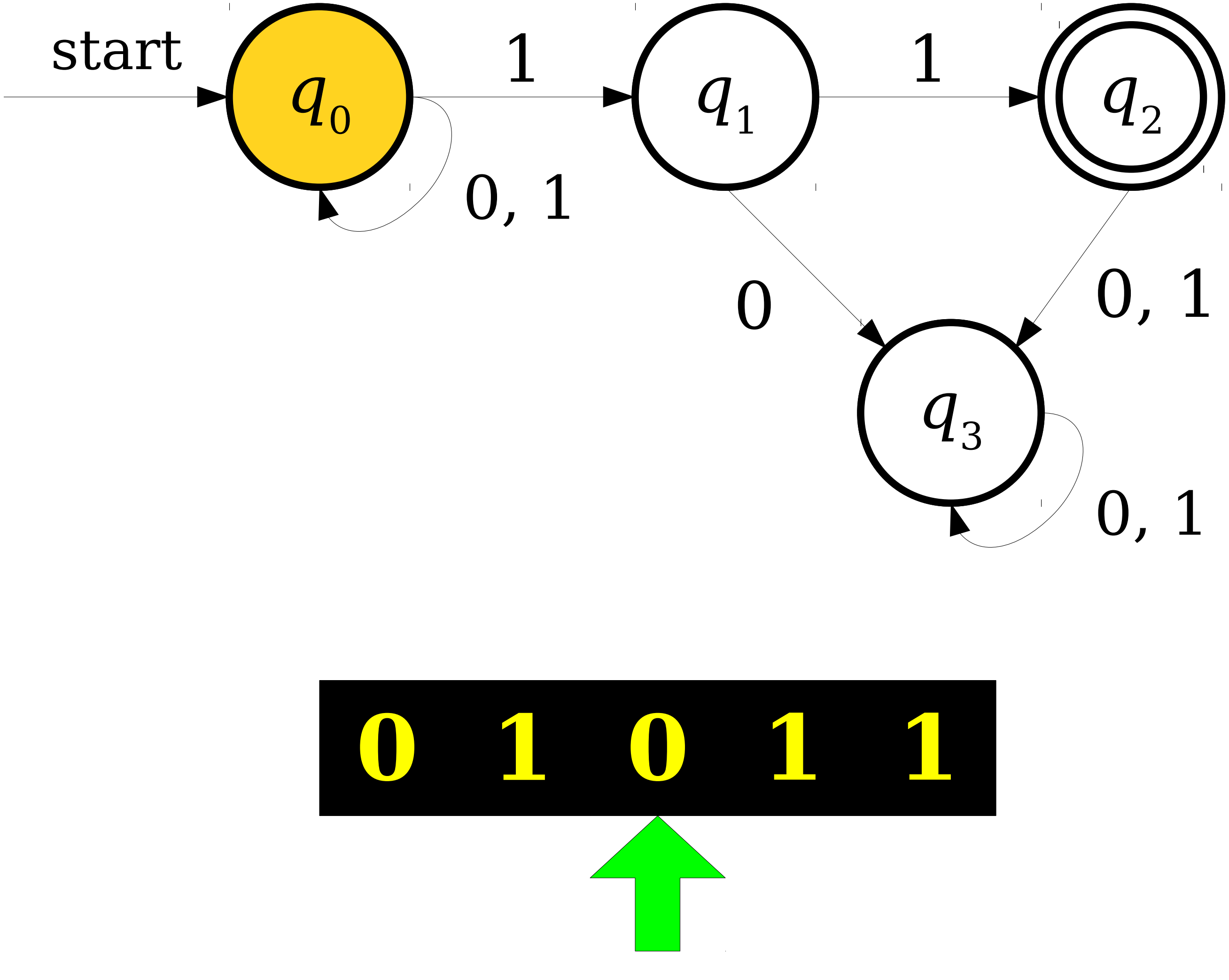
A Simple NFA



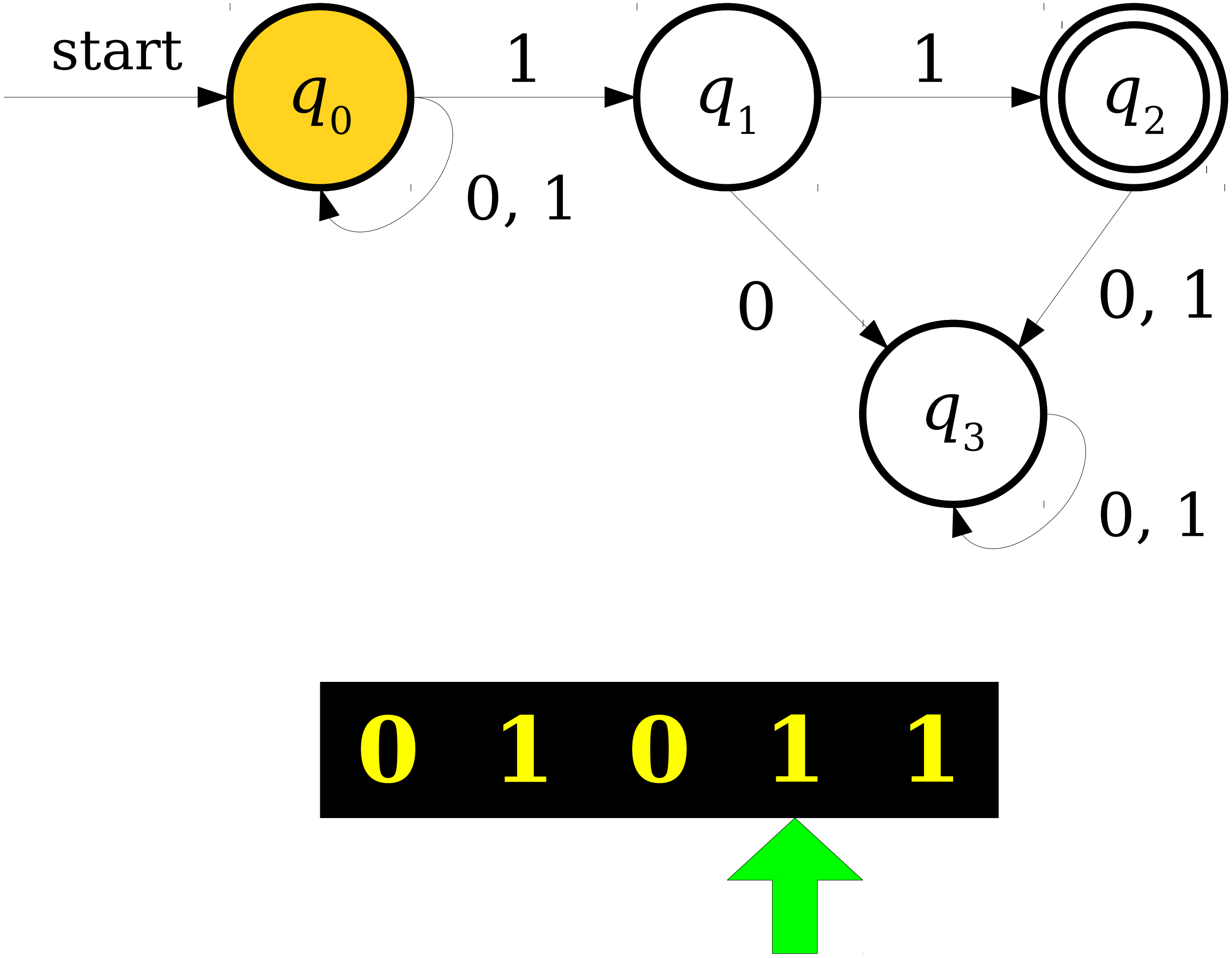
A Simple NFA



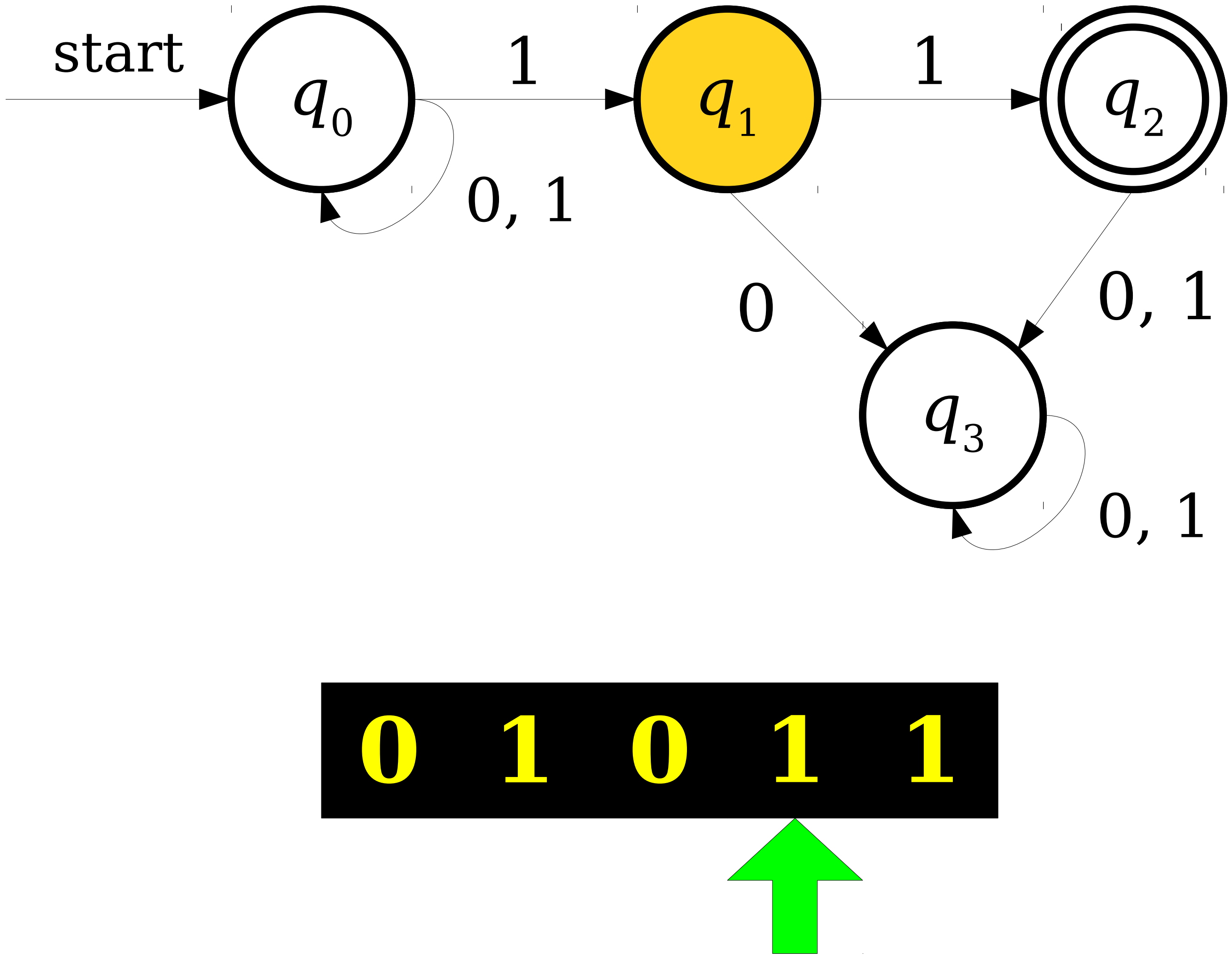
A Simple NFA



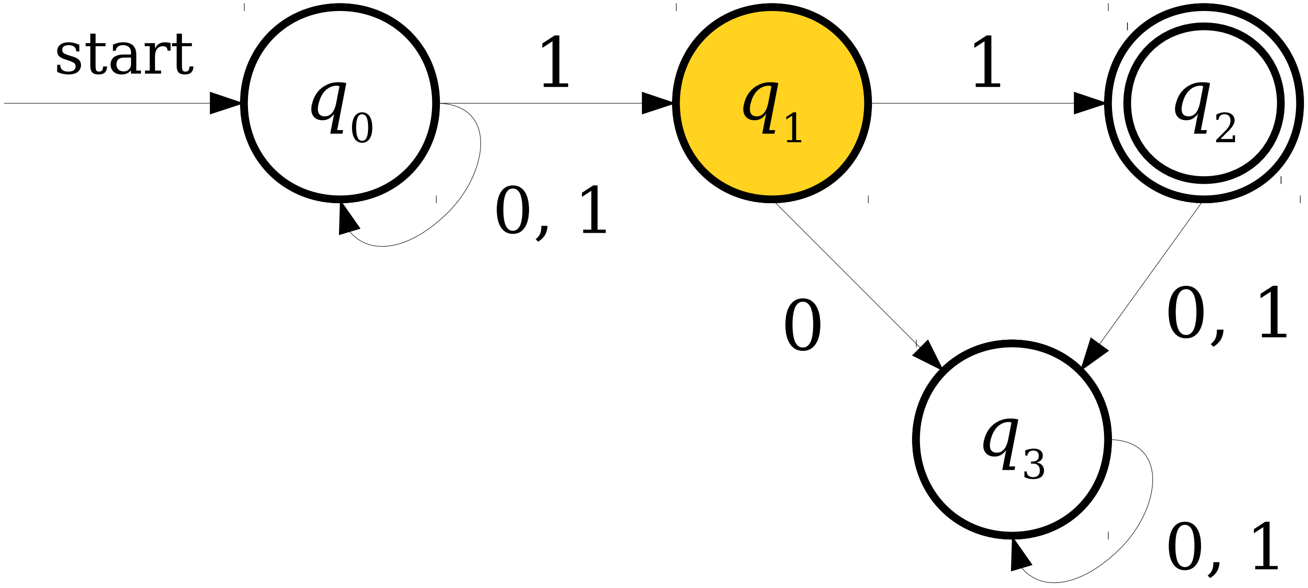
A Simple NFA

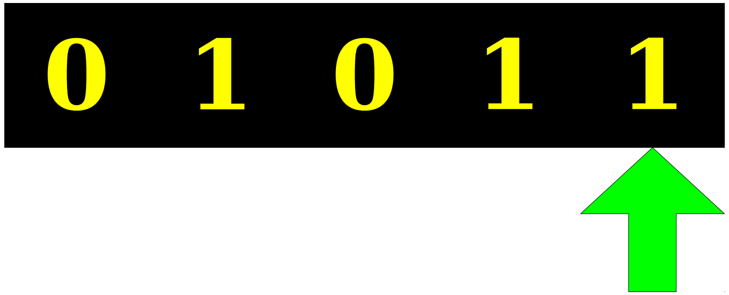


A Simple NFA

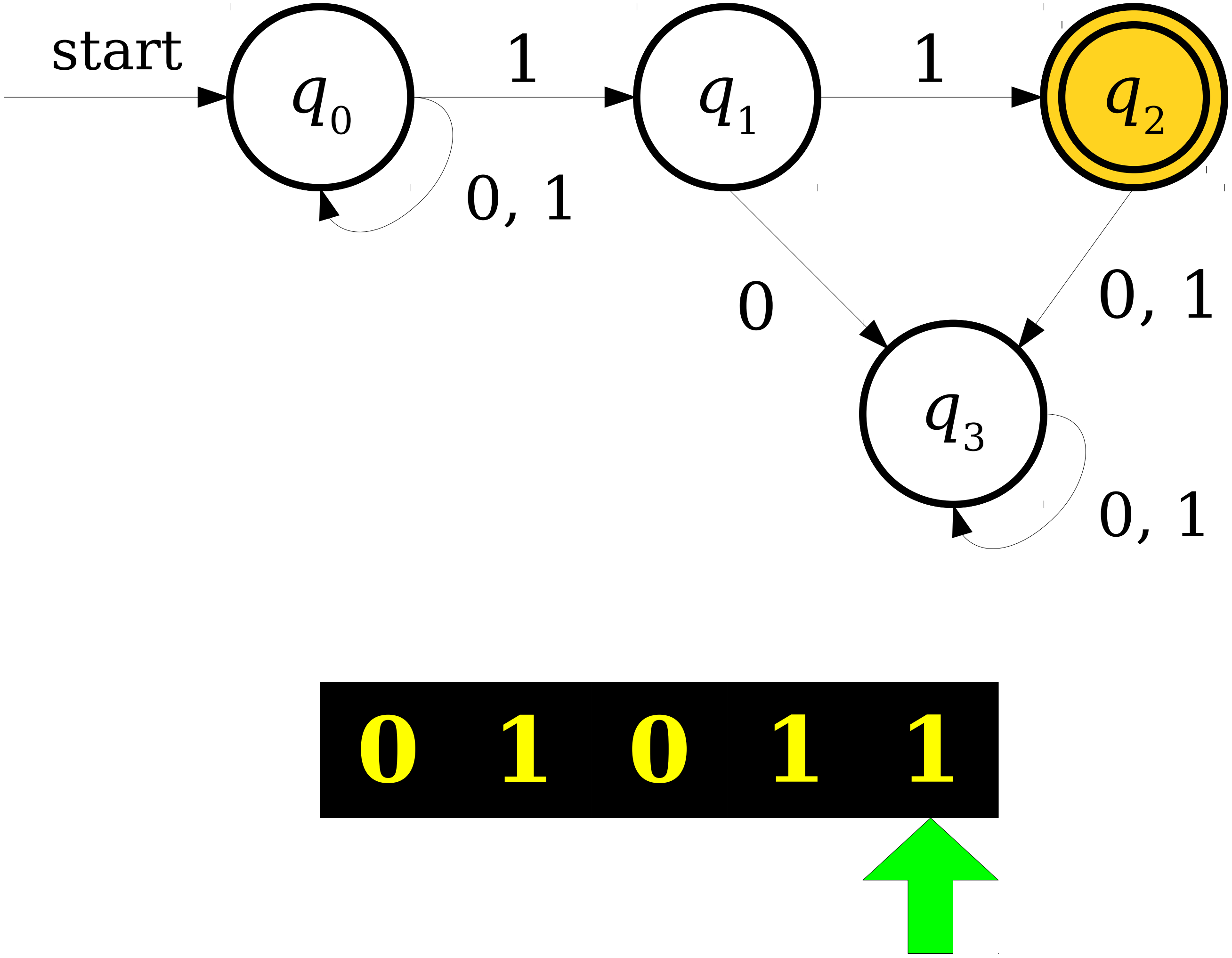


A Simple NFA

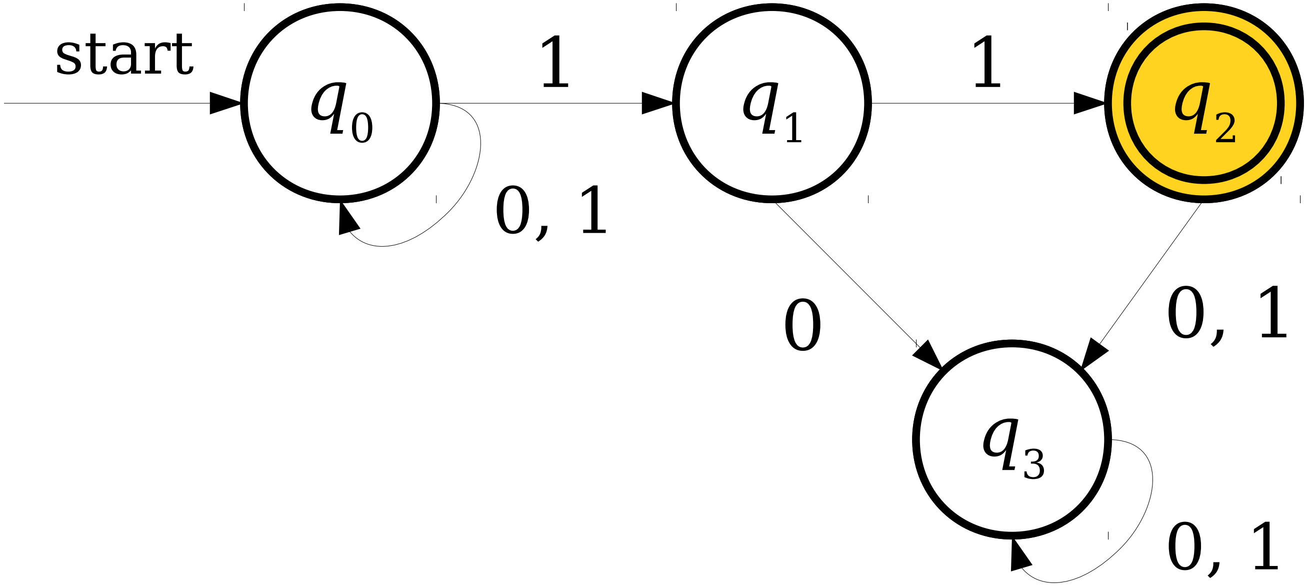




A Simple NFA

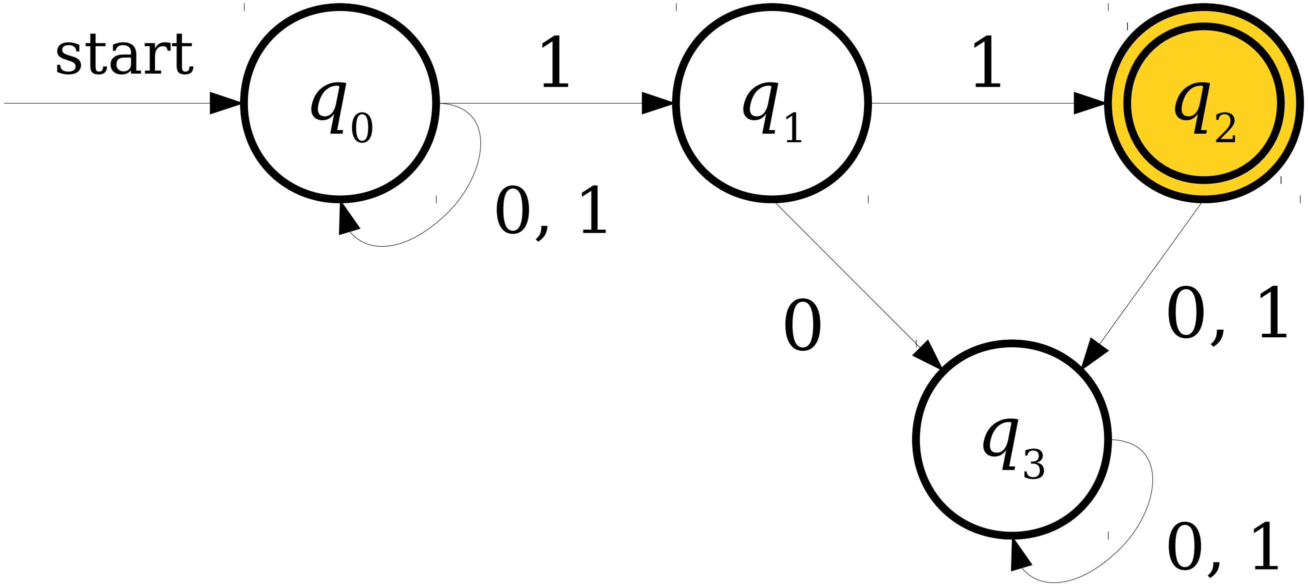


A Simple NFA



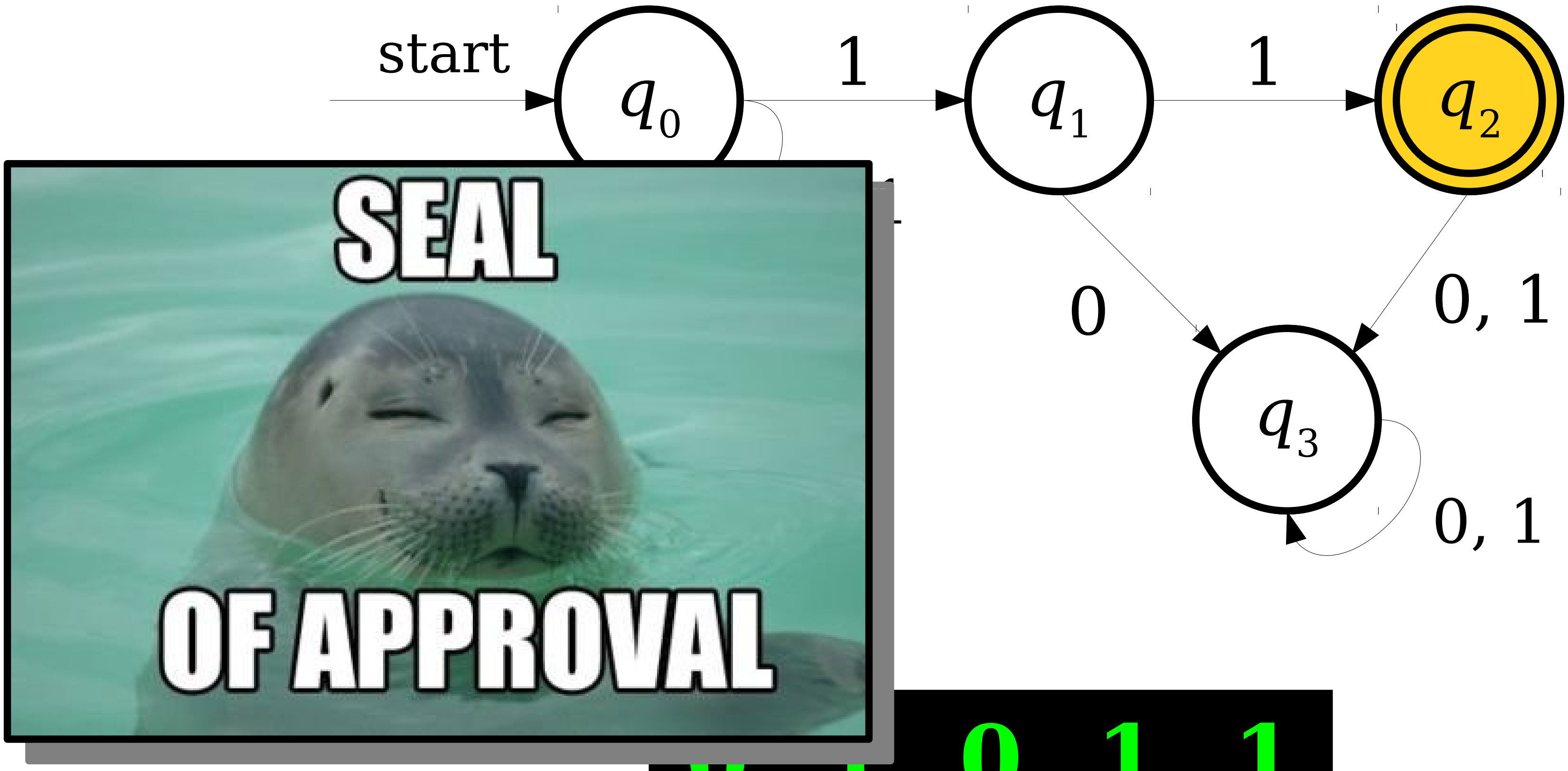
0 1 0 1 1

A Simple NFA



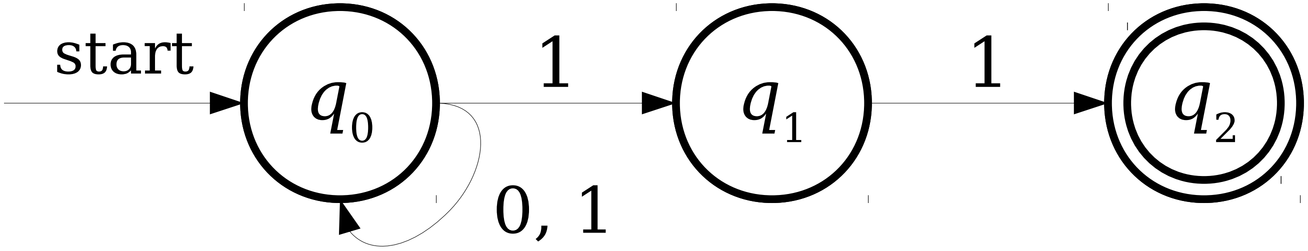
0 1 0 1 1

A Simple NFA

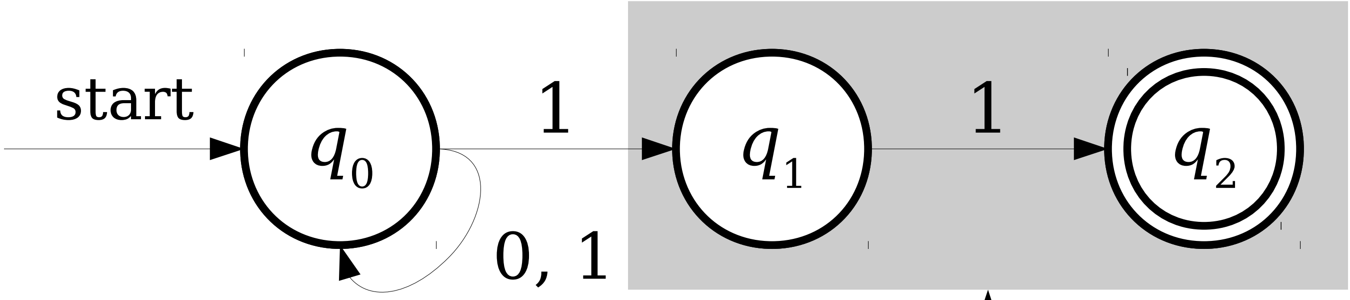


0 1 0 1 1

A More Complex NFA

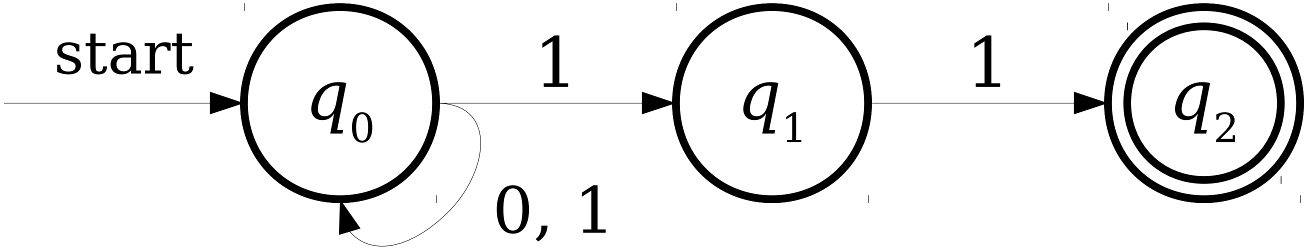


A More Complex NFA



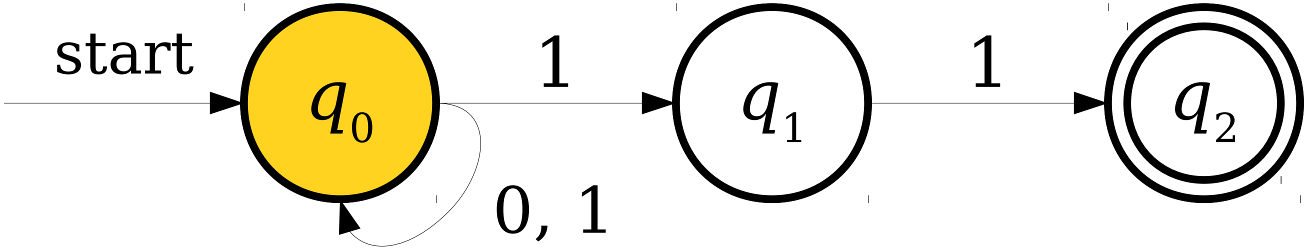
If a NFA needs to make a transition when no transition exists, the automaton dies and that particular path does not accept.

A More Complex NFA



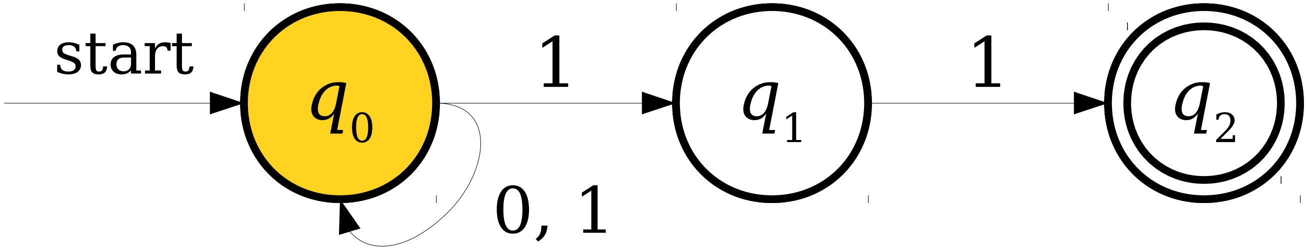
0 1 0 1 1

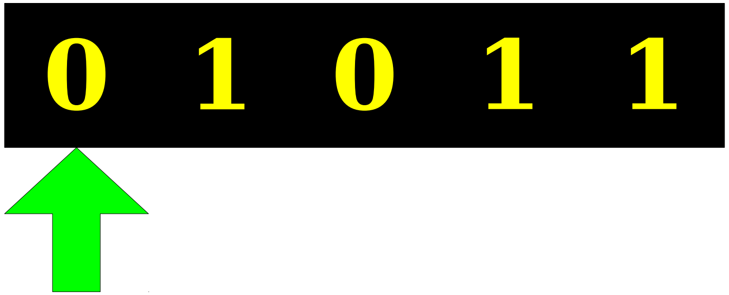
A More Complex NFA



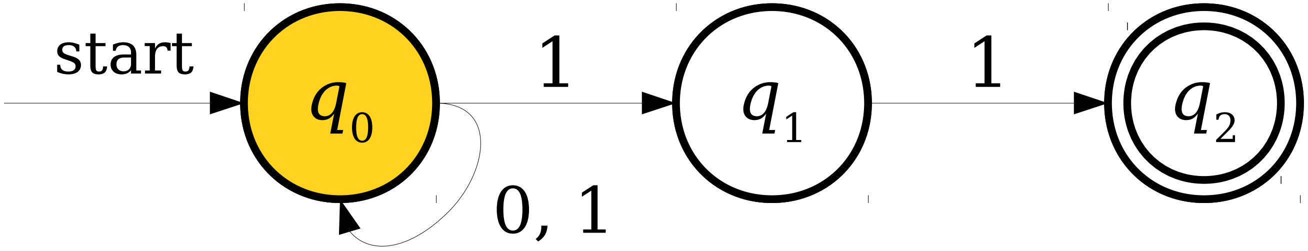
0 1 0 1 1

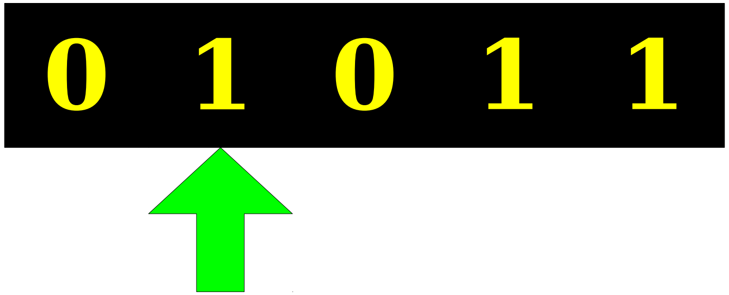
A More Complex NFA



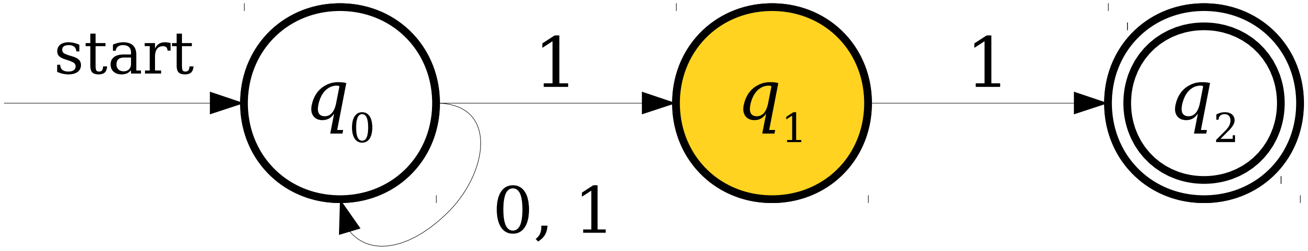


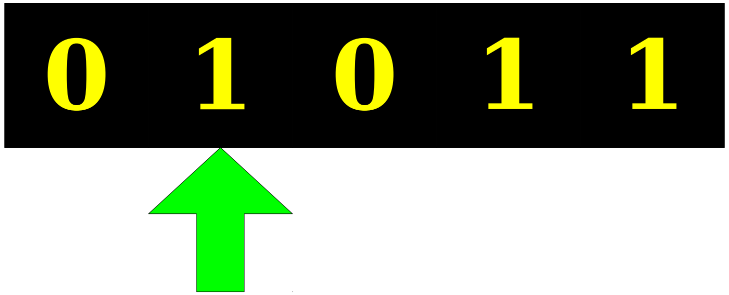
A More Complex NFA





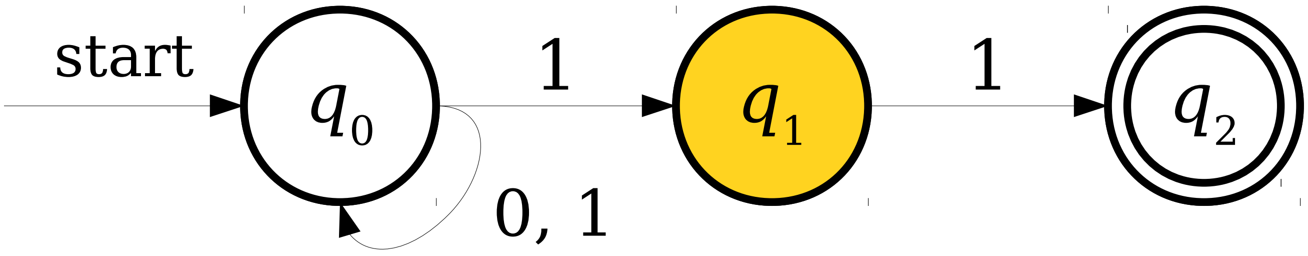
A More Complex NFA

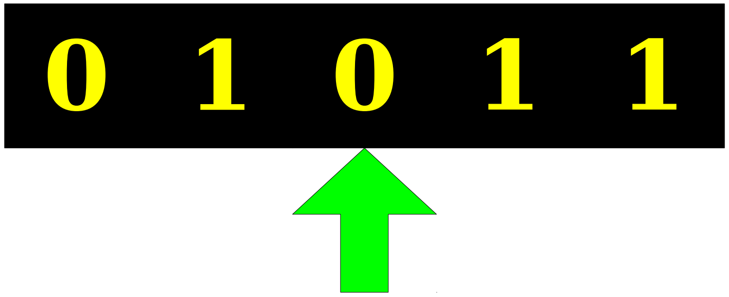




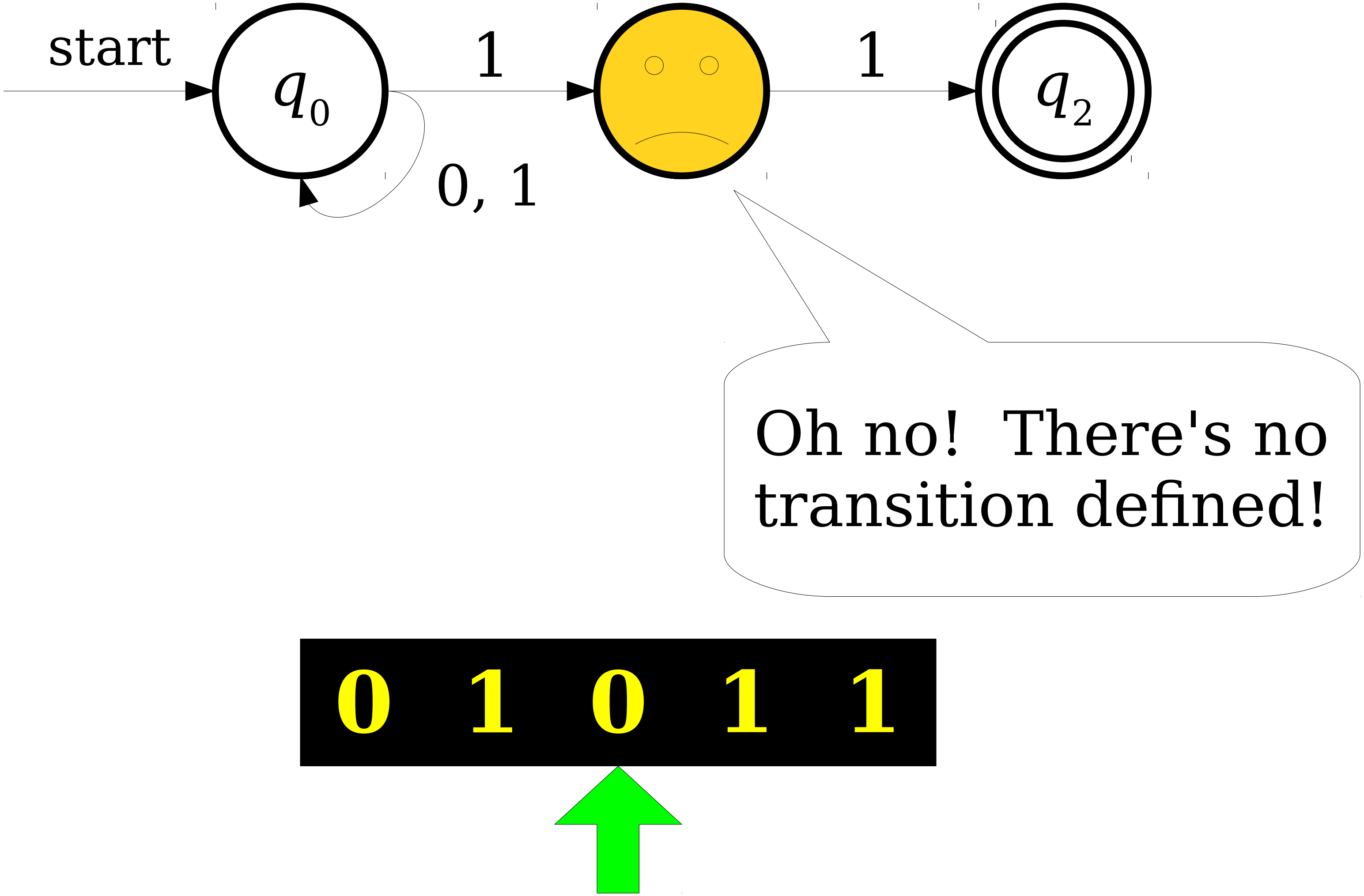
0 1 0 1 1

A More Complex NFA





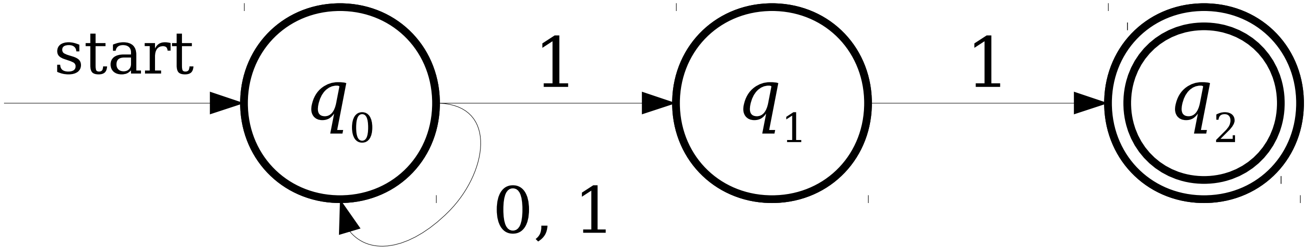
A More Complex NFA

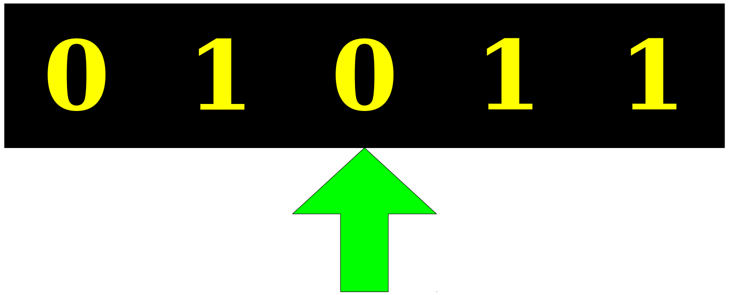


Oh no! There's no transition defined!

0 1 0 1 1

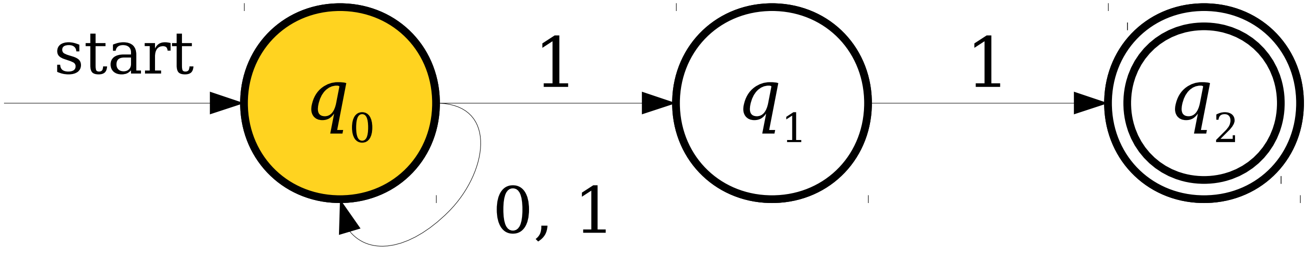
A More Complex NFA

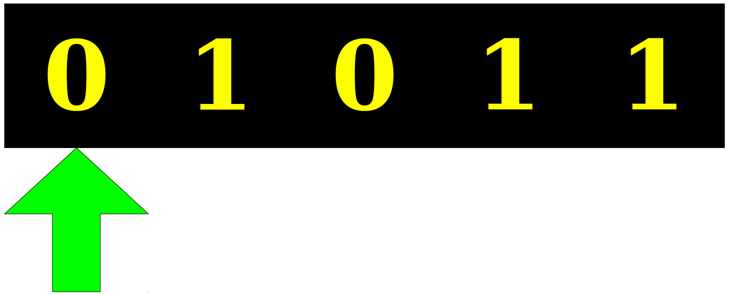




0 1 0 1 1

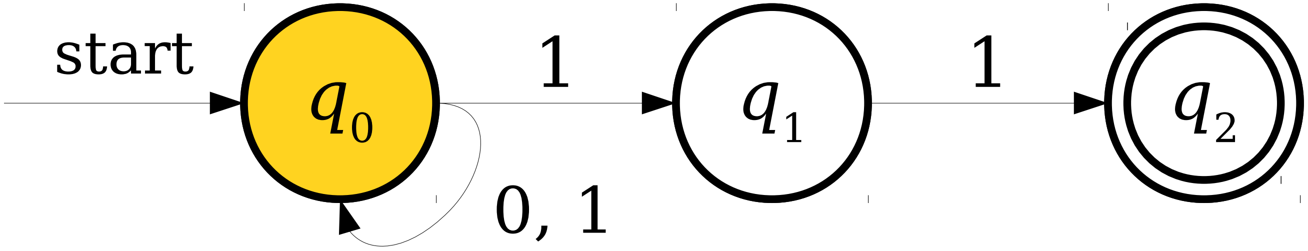
A More Complex NFA

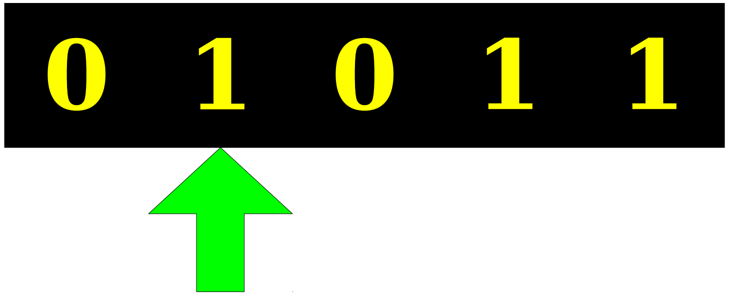




0 1 0 1 1

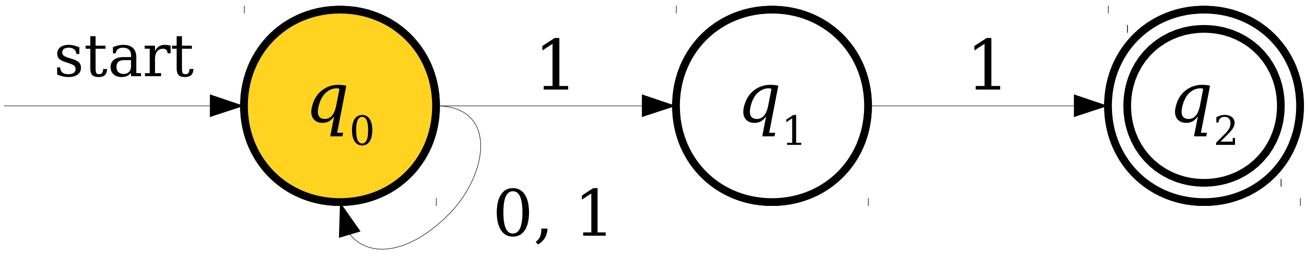
A More Complex NFA

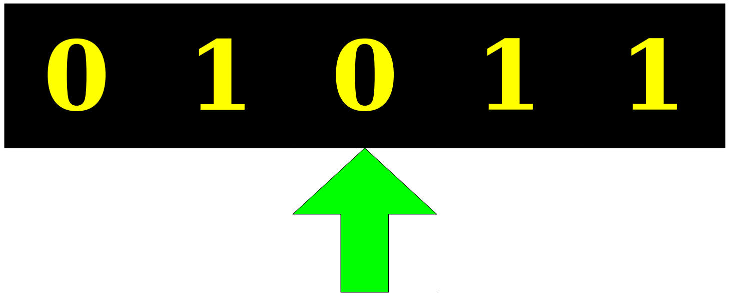




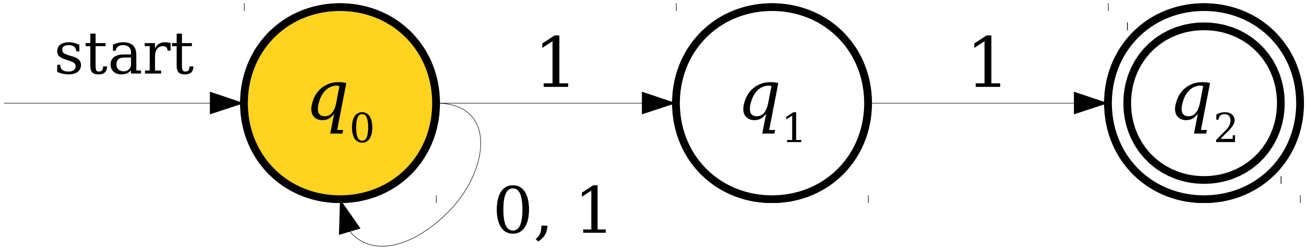
0 1 0 1 1

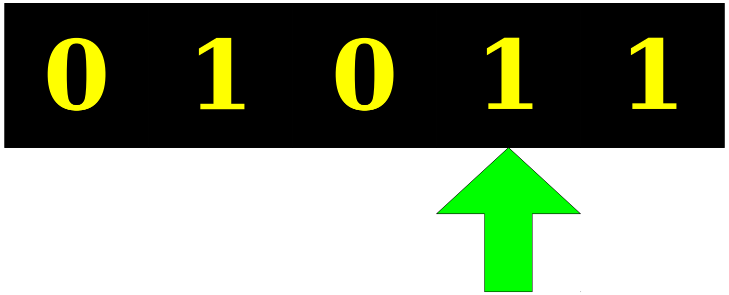
A More Complex NFA





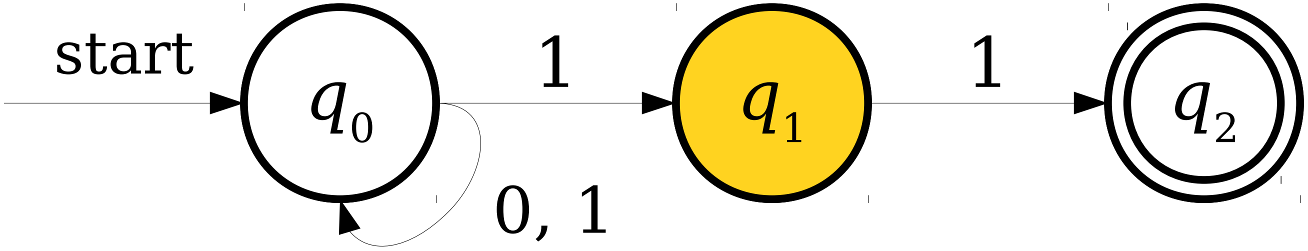
A More Complex NFA

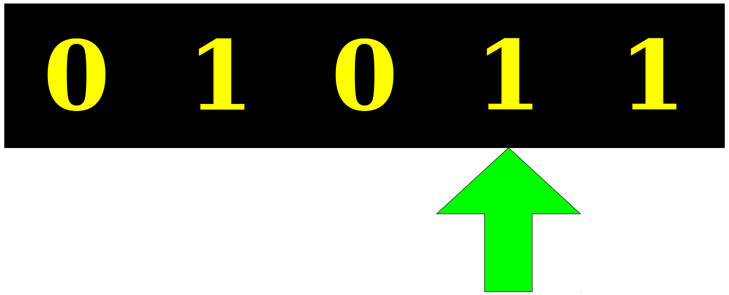




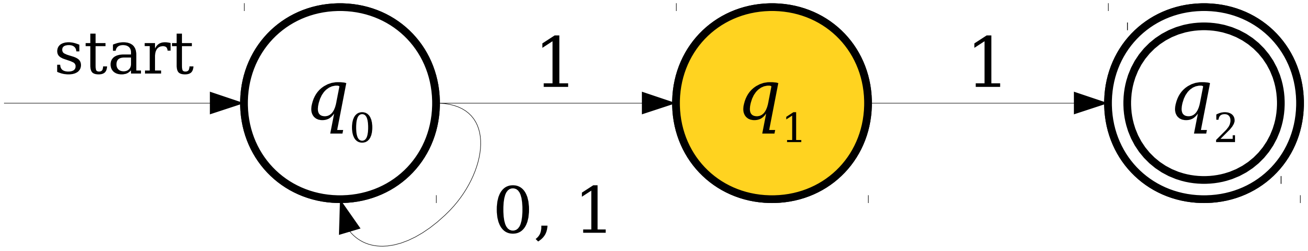
0 1 0 1 1

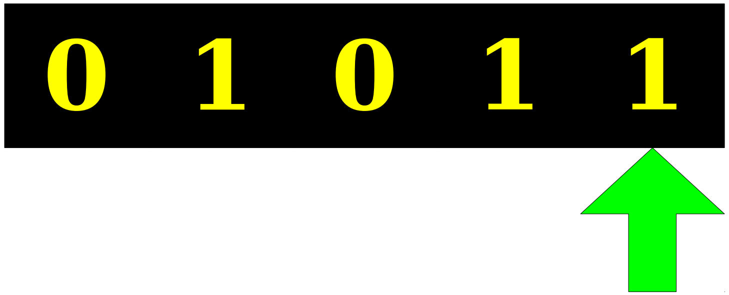
A More Complex NFA



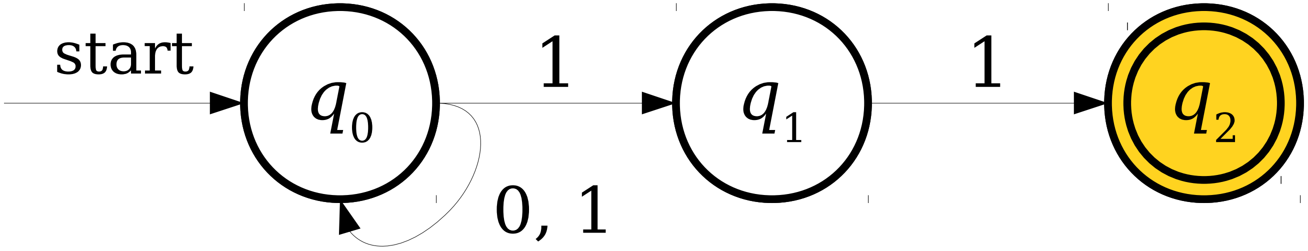


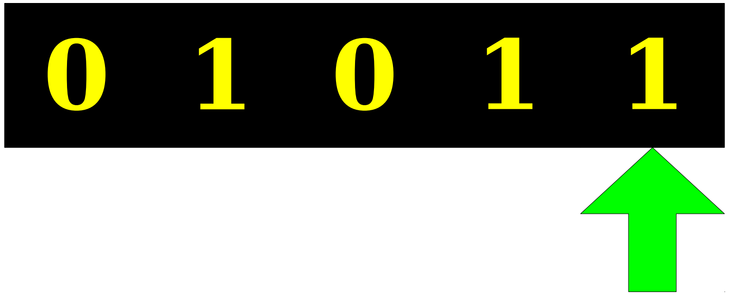
A More Complex NFA





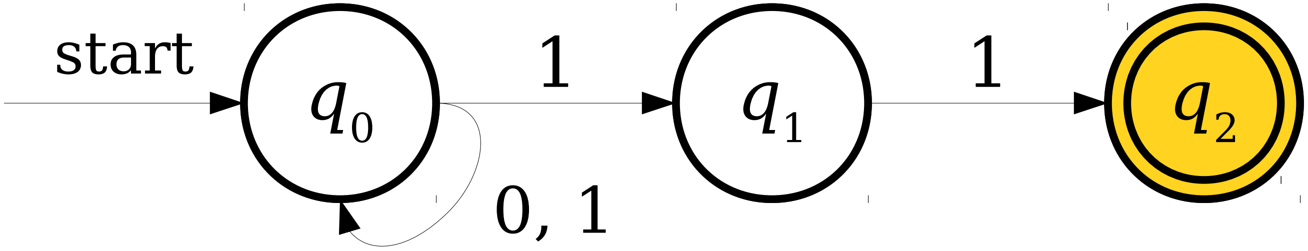
A More Complex NFA





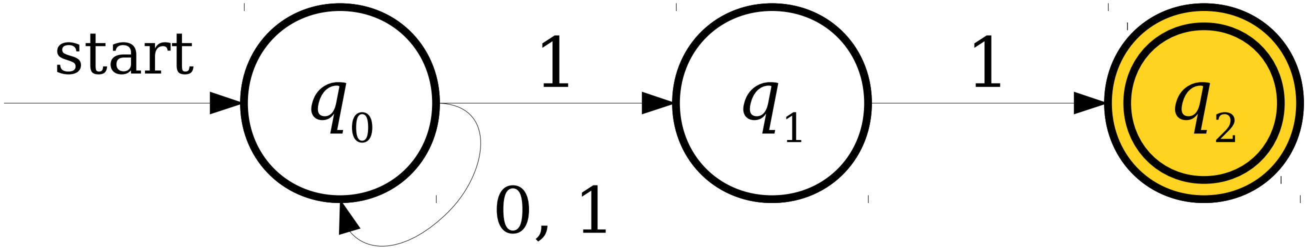
0 1 0 1 1

A More Complex NFA



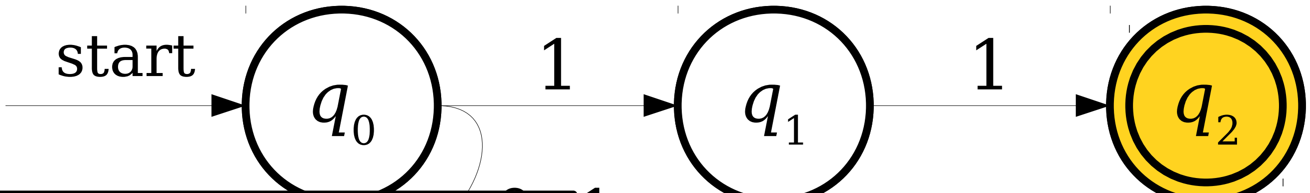
0 1 0 1 1

A More Complex NFA

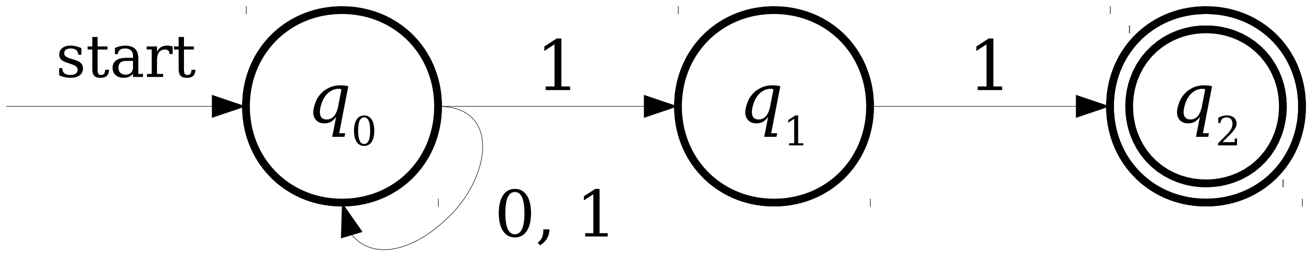


0 1 0 1 1

A More Complex NFA



0 1 0 1 1



As with DFAs, the language of an NFA N is the set of strings that N accepts:

$$\mathscr{L}(N) = \{\ w \in \Sigma^* \mid N \text{ accepts } w\ \}.$$

What is the language of the NFA shown above?

A. { **01011** }
B. { $w \in \{$**0**, **1**$\}^* \mid w$ contains at least two **1**s }
C. { $w \in \{$**0**, **1**$\}^* \mid w$ ends with **11** }
D. { $w \in \{$**0**, **1**$\}^* \mid w$ ends with **1** }
E. None of these, or two or more of these.

Answer at **PollEv.com/cs103** or
text **CS103** to **22333** once to join, then ***A***, …, or ***E***.

NFA Acceptance

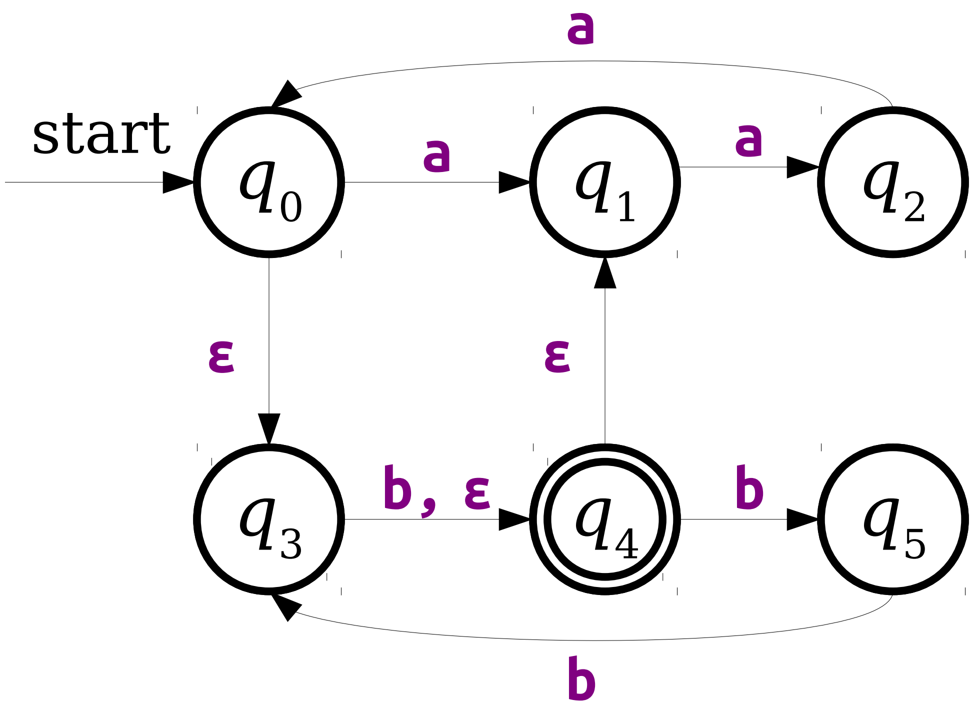
- An NFA *N* accepts a string *w* if there is some series of choices that lead to an accepting state.
- Consequently, an NFA *N* rejects a string *w* if *no possible* series of choices lead it into an accepting state.
- It's easier to show that an NFA does accept something than to show that it doesn't.

ε-Transitions

- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.

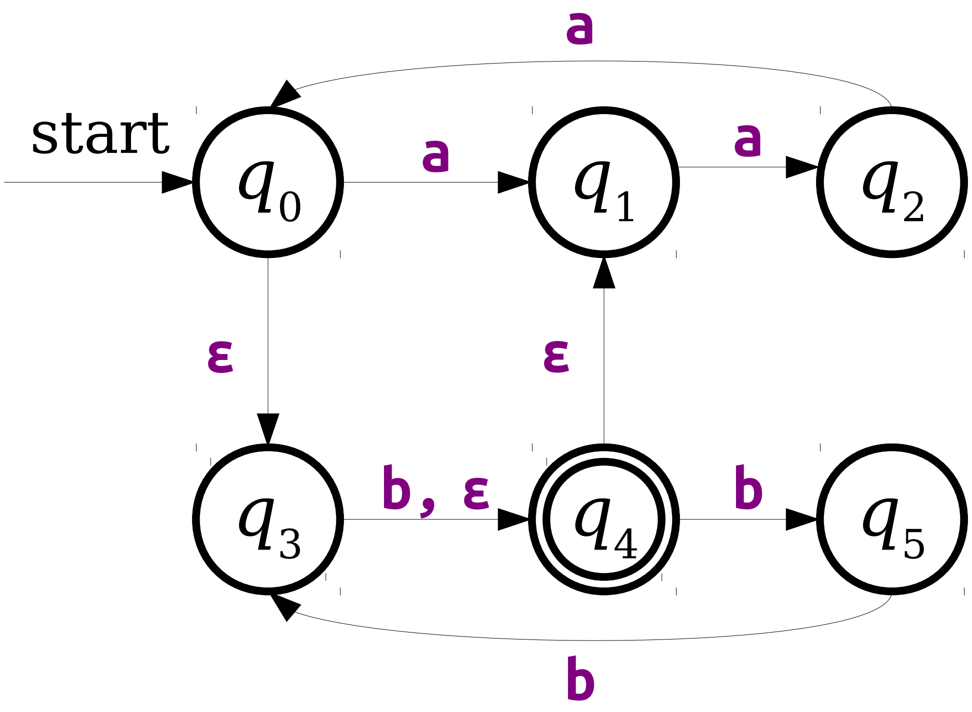
ε-Transitions

- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.



ε-Transitions

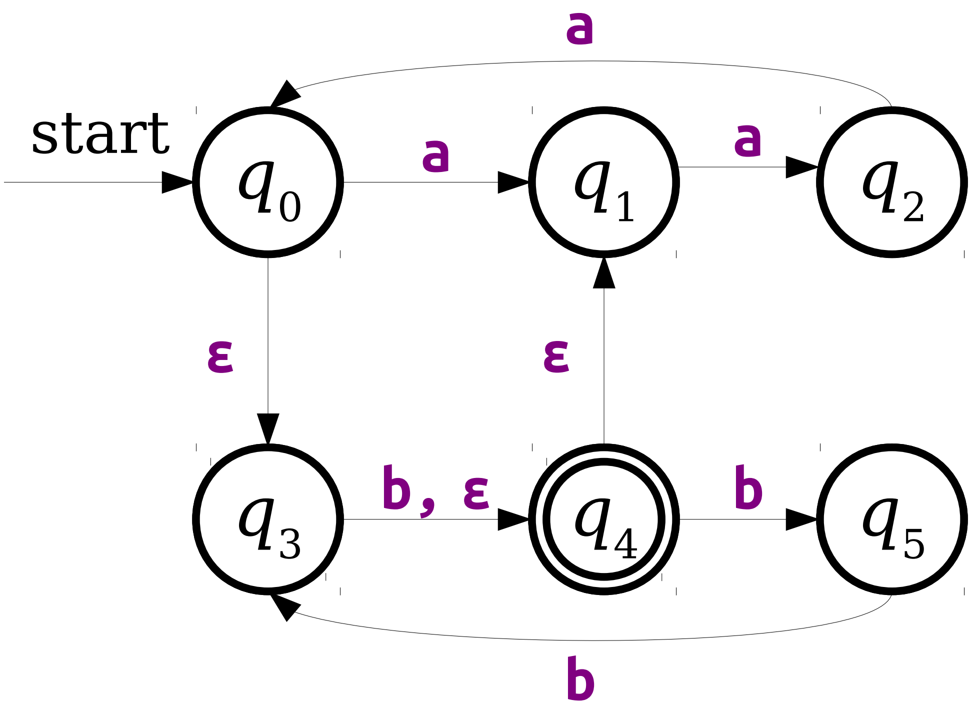
- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.

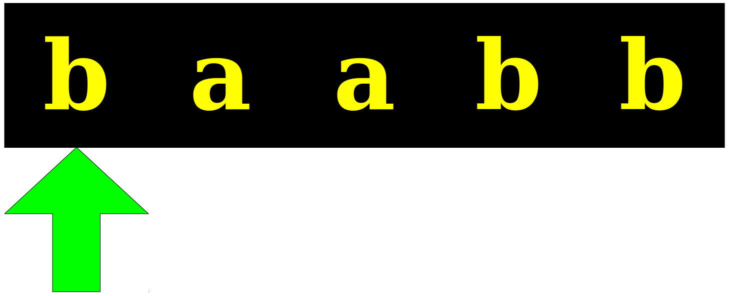


b a a b b

ε-Transitions

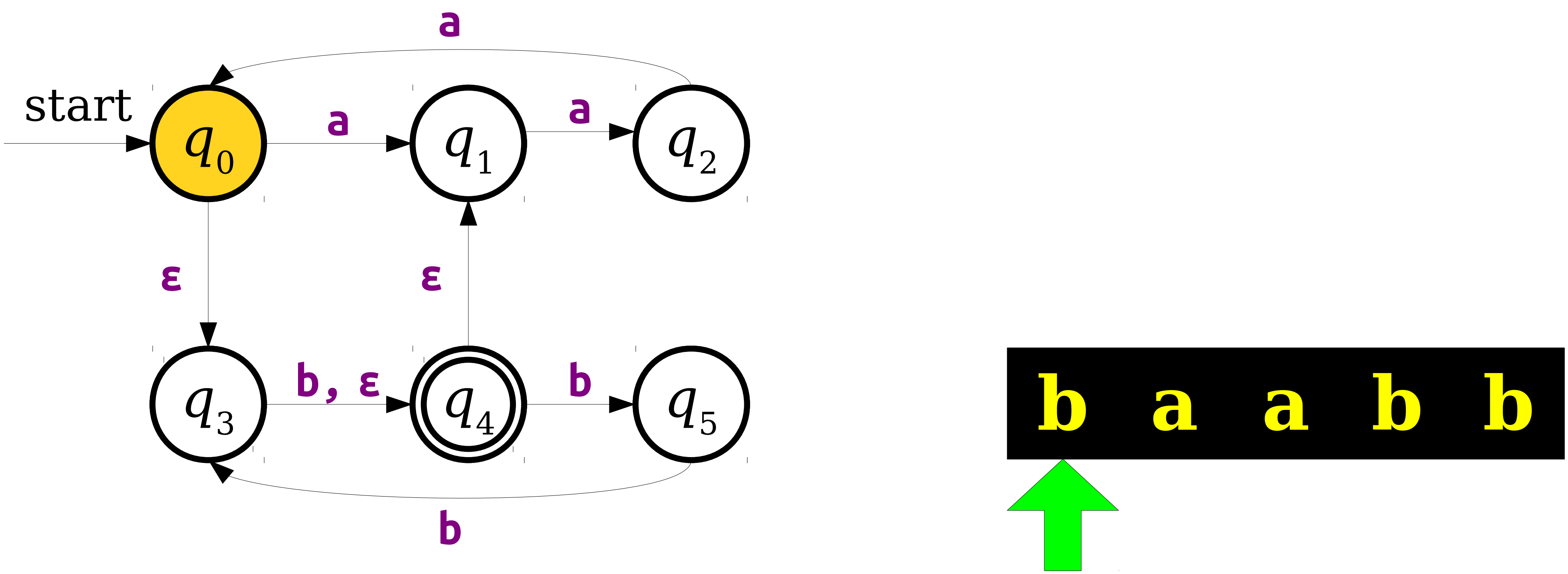
- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.





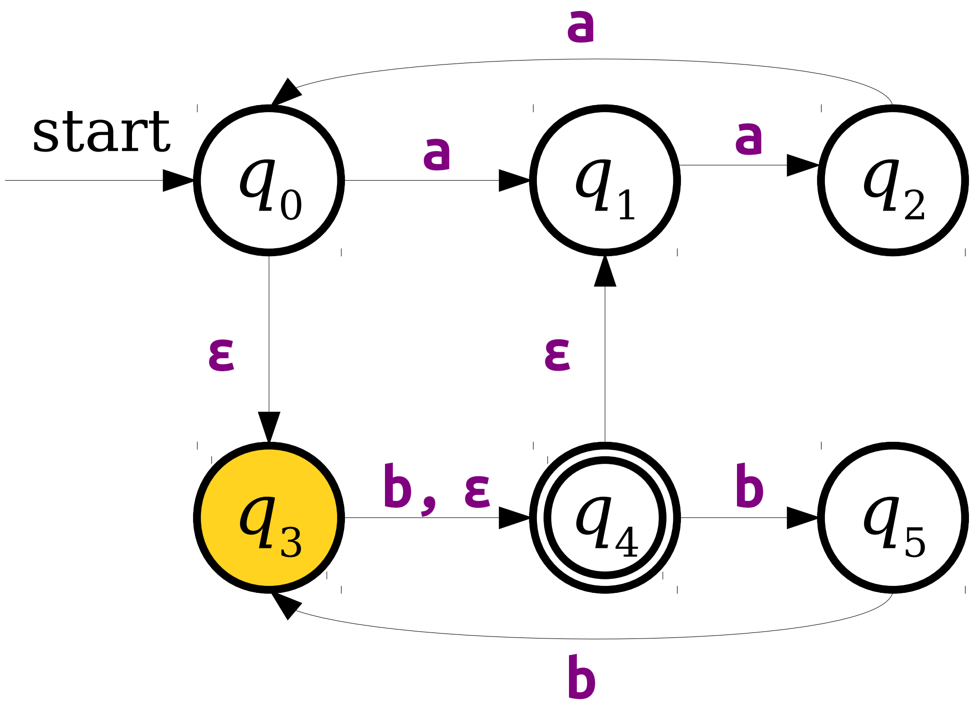
ε-Transitions

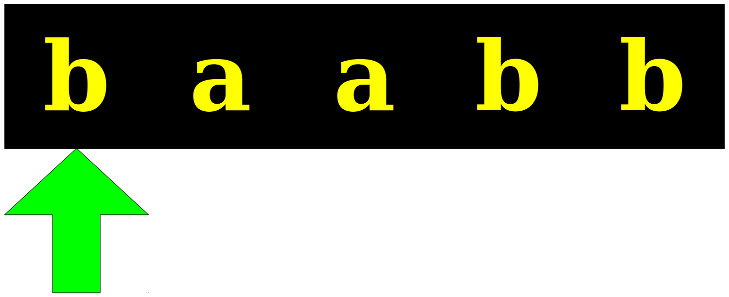
- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.



ε-Transitions

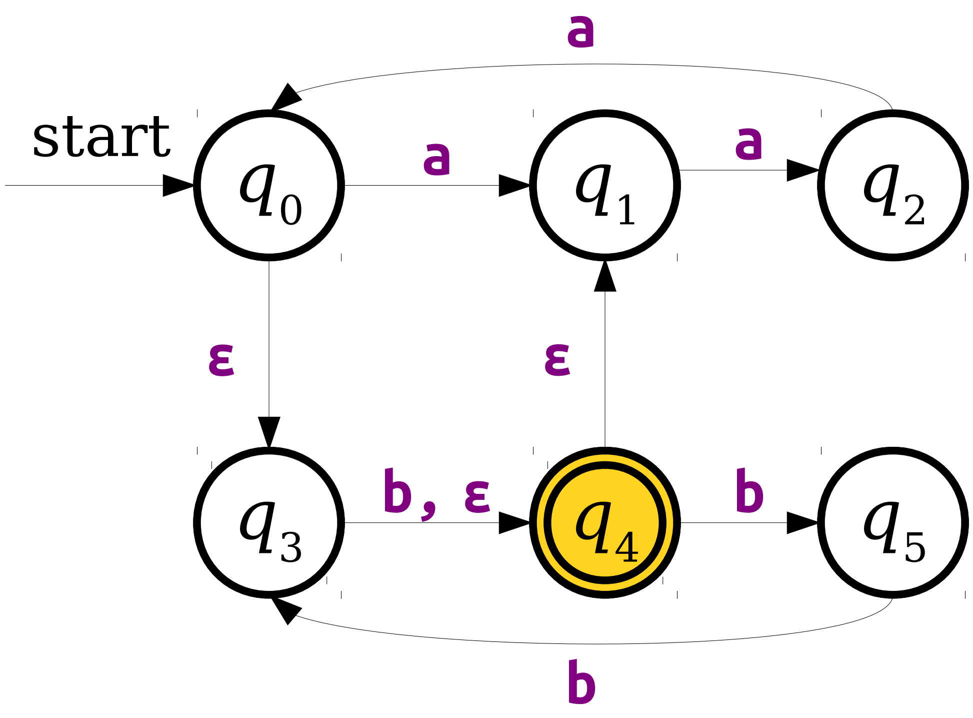
- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.

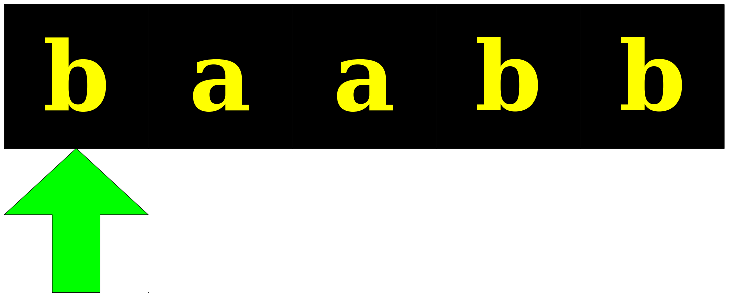




ε-Transitions

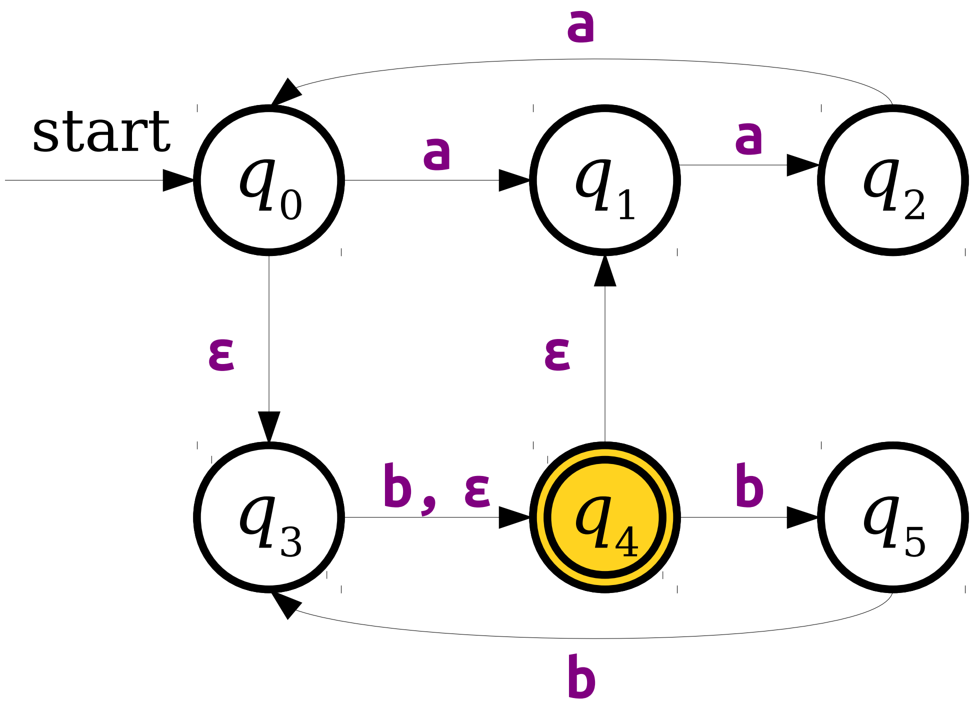
- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.

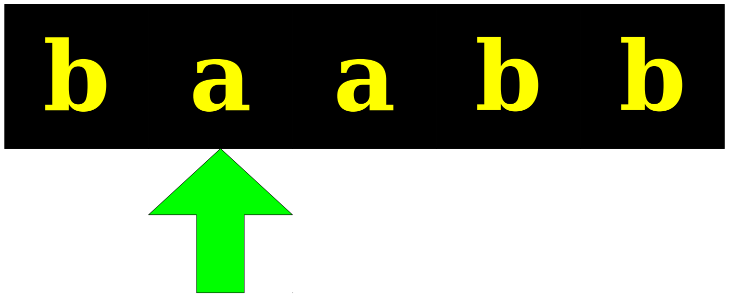



b a a b b

ε-Transitions

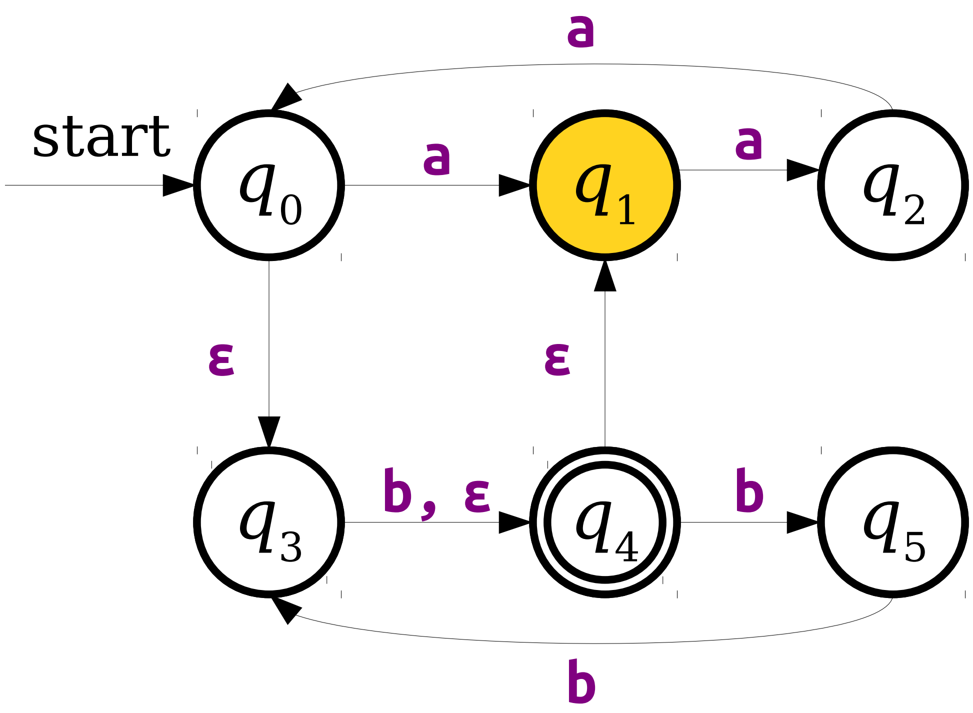
- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.

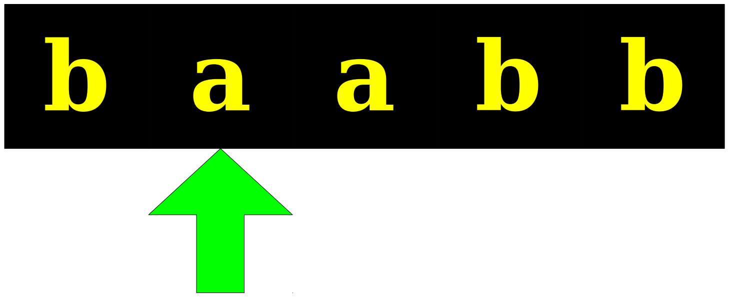




ε-Transitions

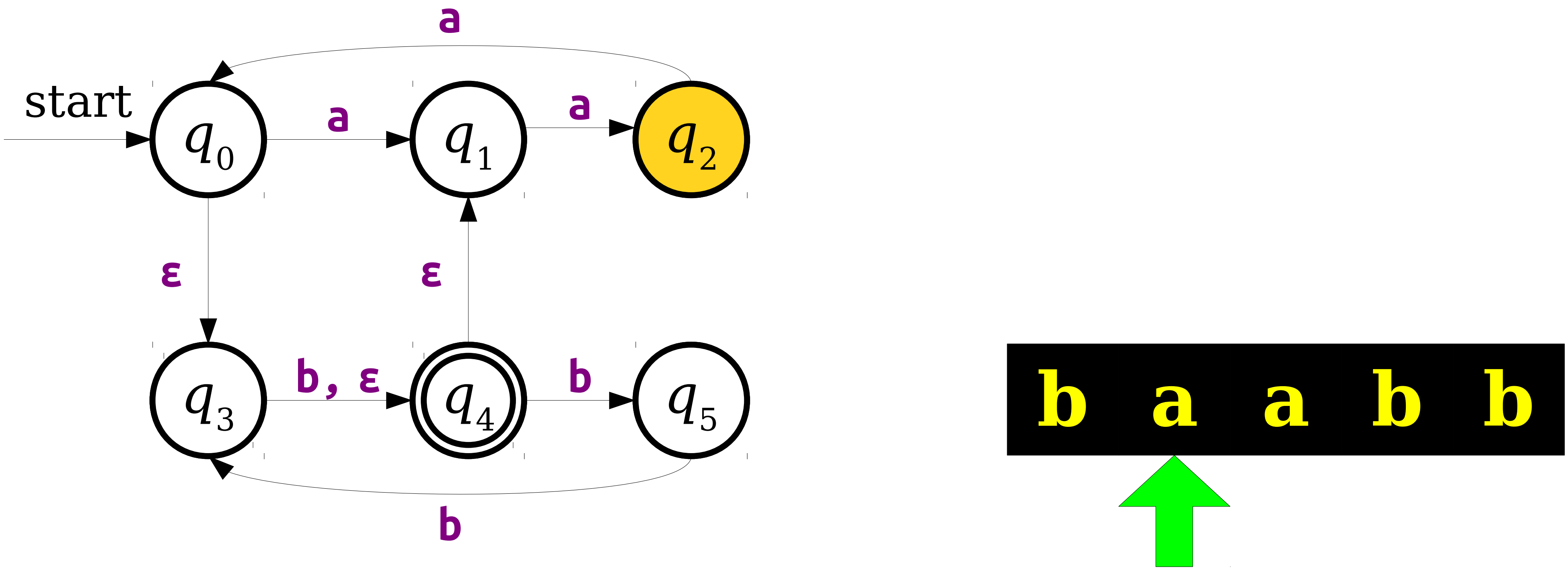
- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.





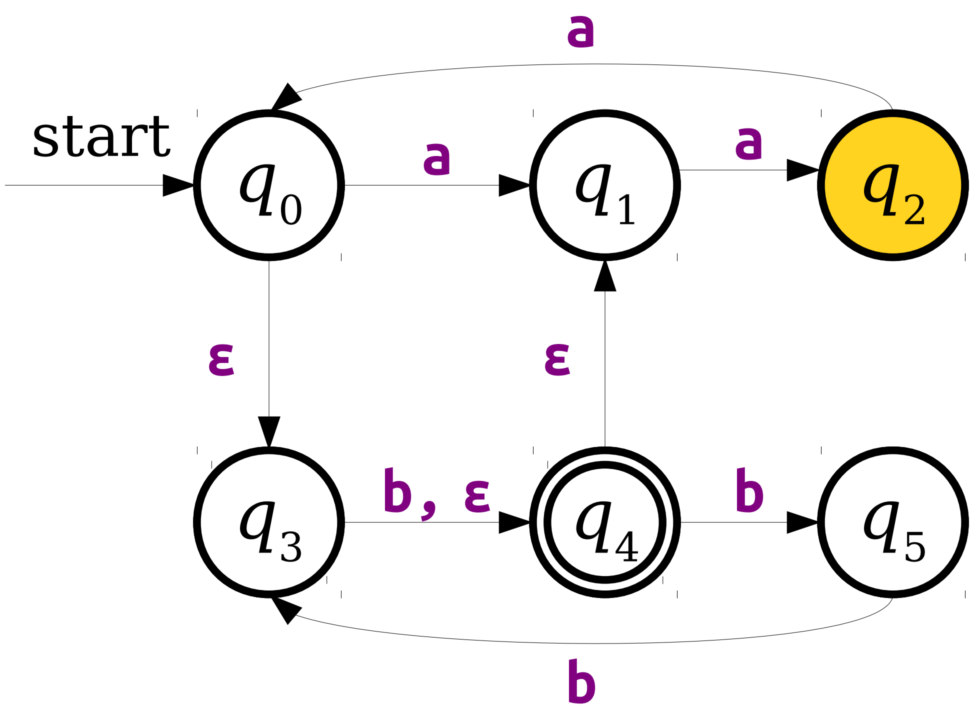
ε-Transitions

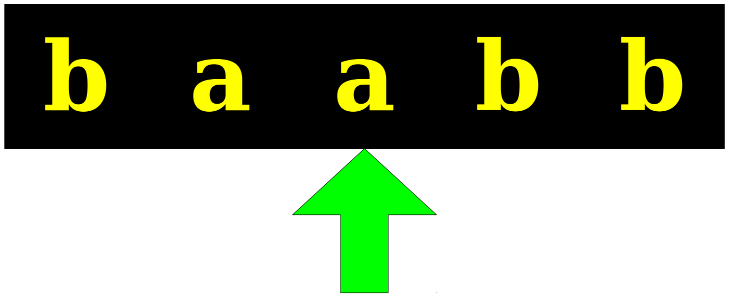
- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.



ε-Transitions

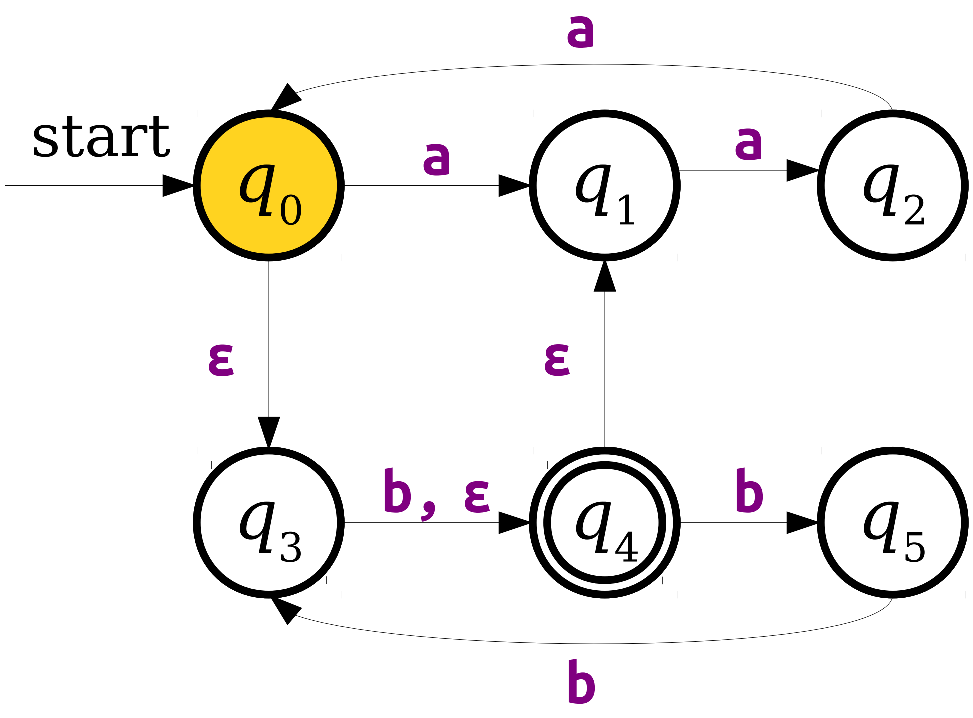
- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.

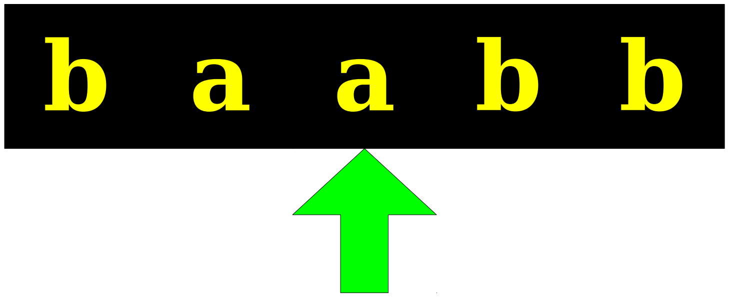




ε-Transitions

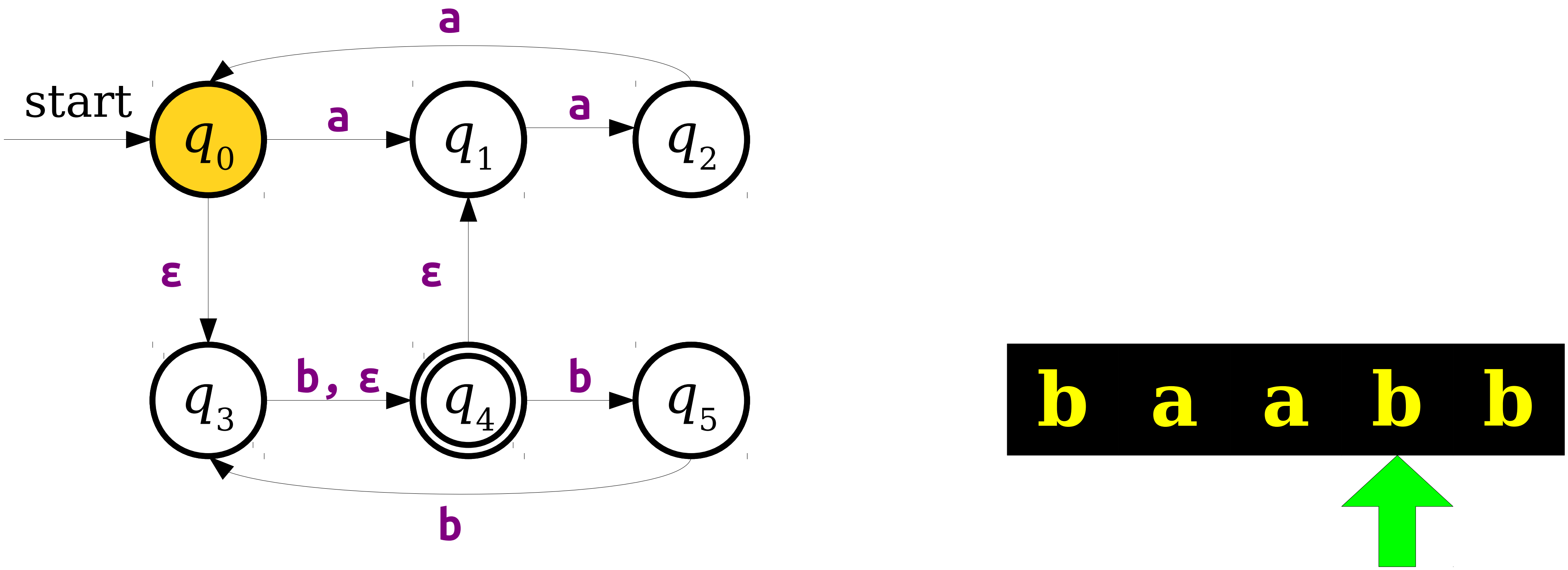
- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.





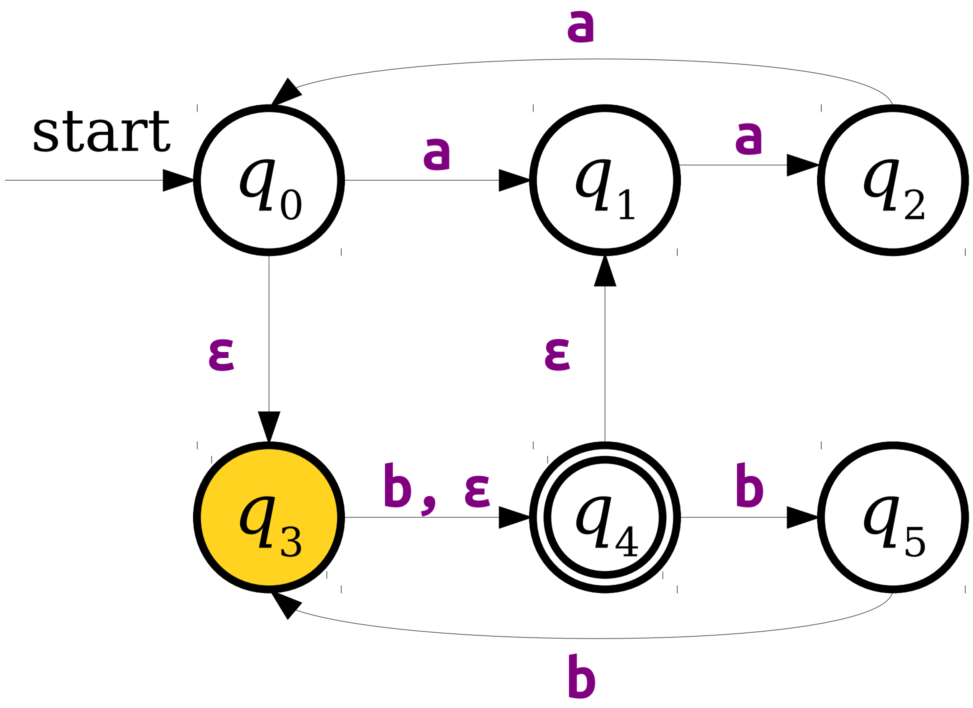
ε-Transitions

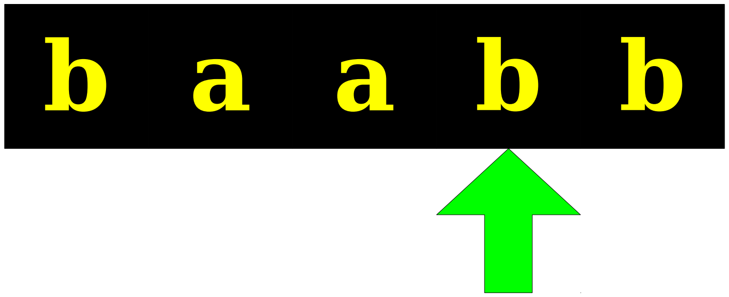
- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.



ε-Transitions

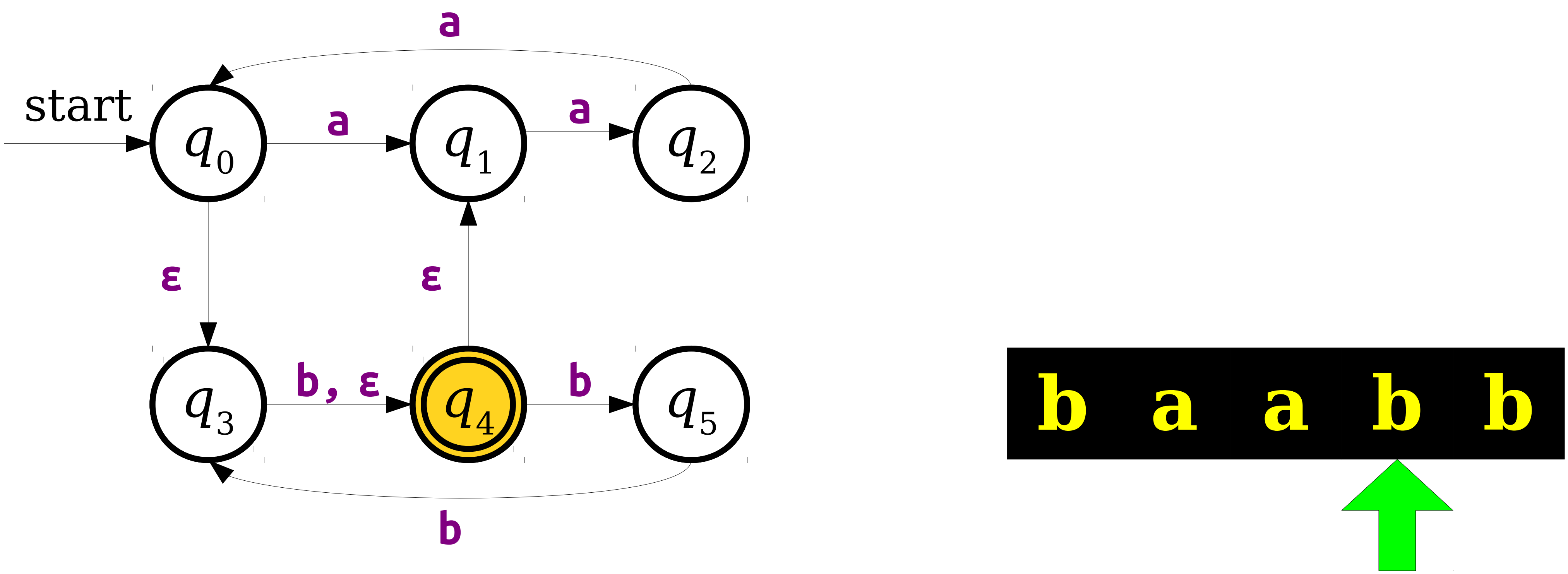
- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.





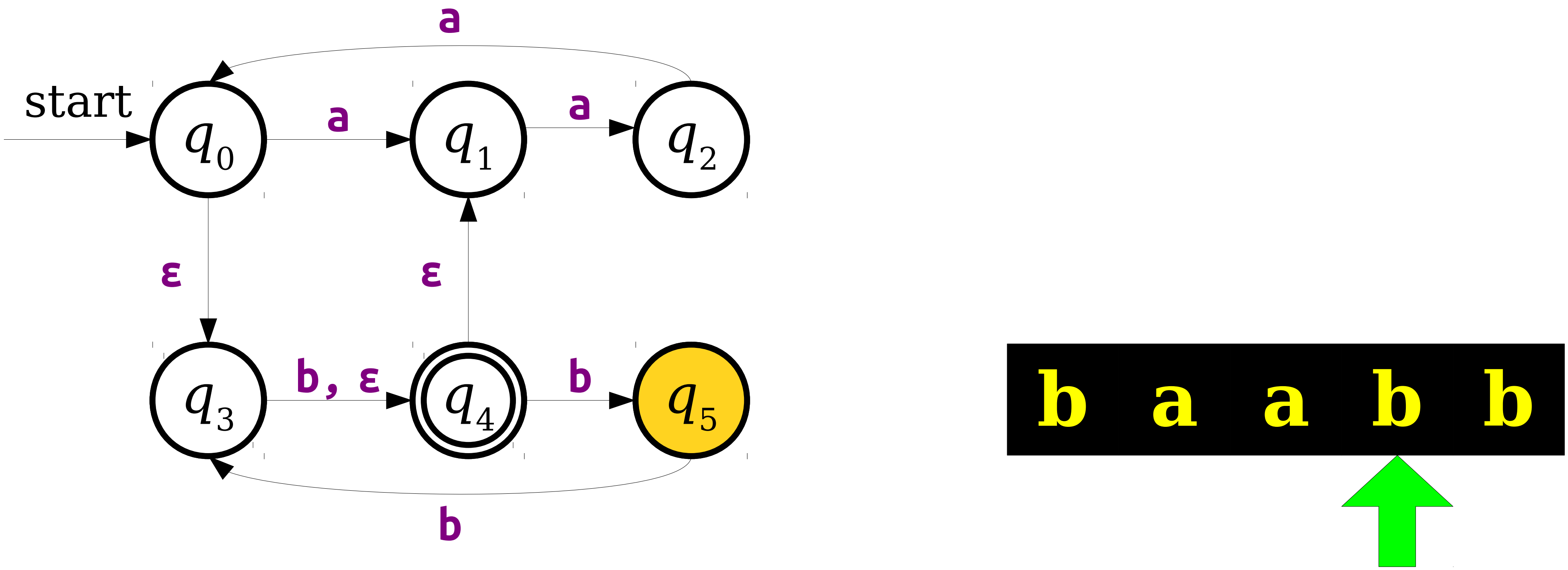
ε-Transitions

- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.



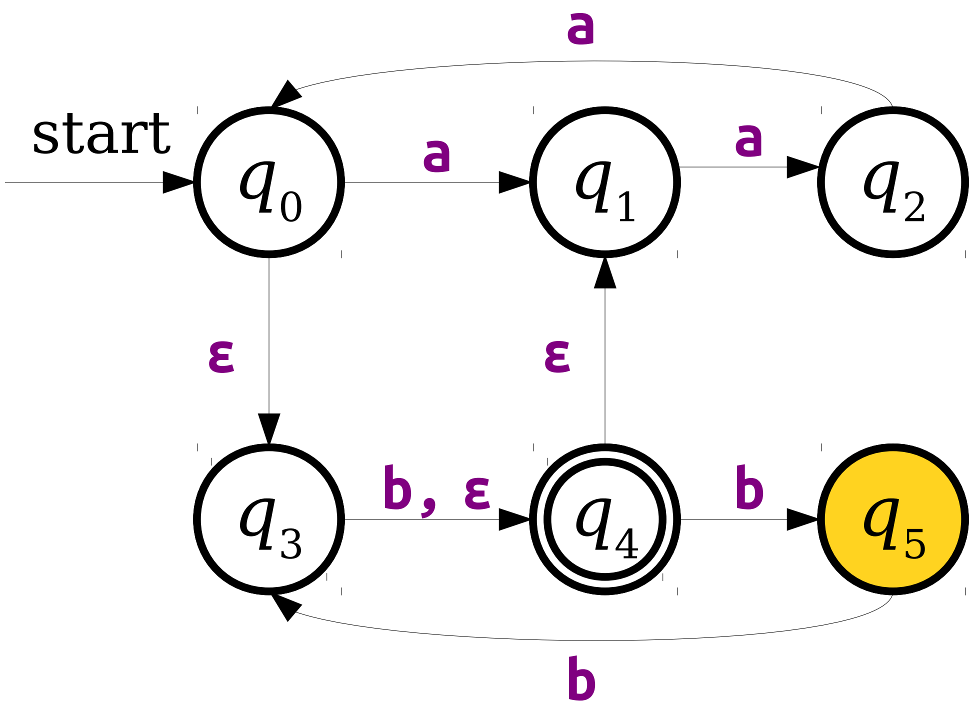
ε-Transitions

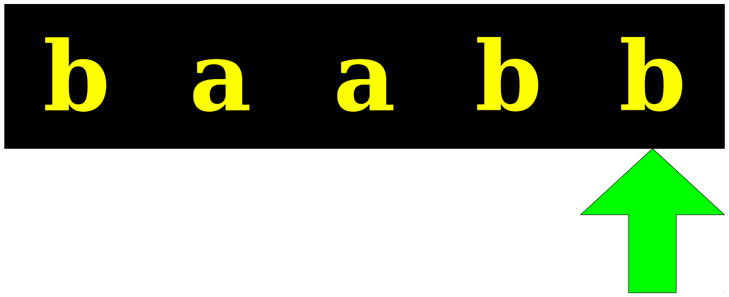
- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.



ε-Transitions

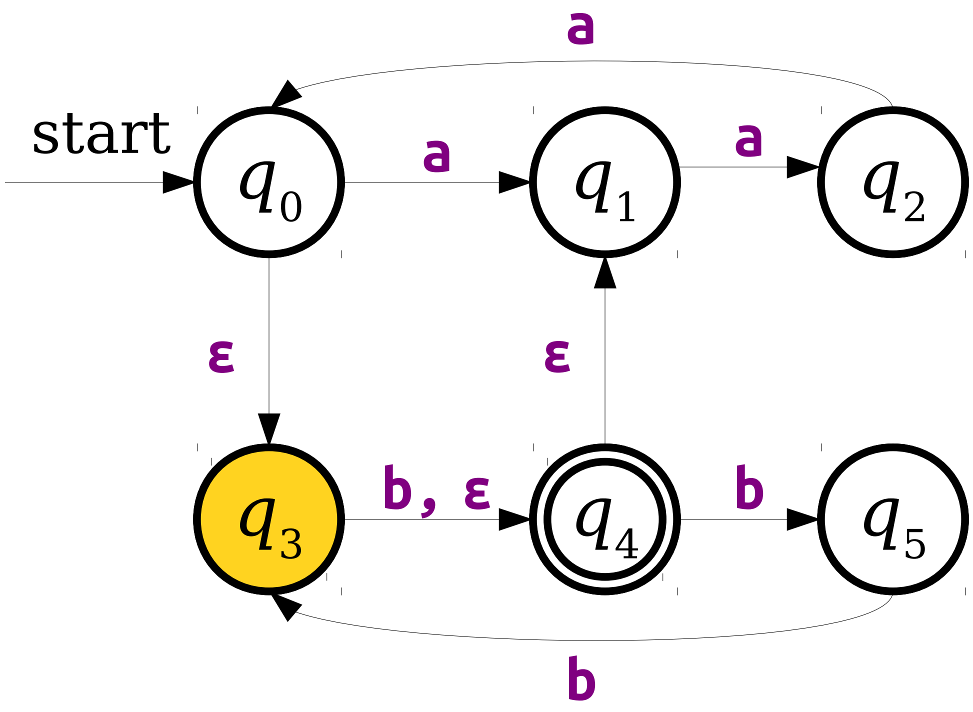
- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.

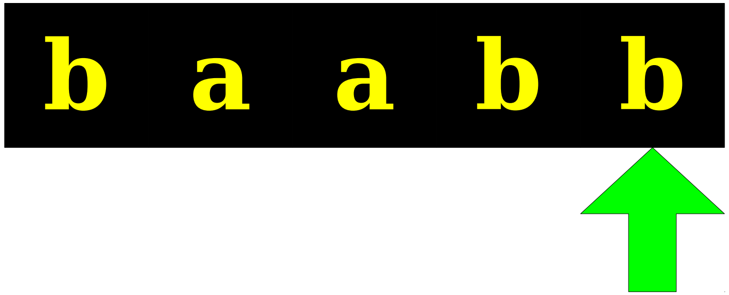




ε-Transitions

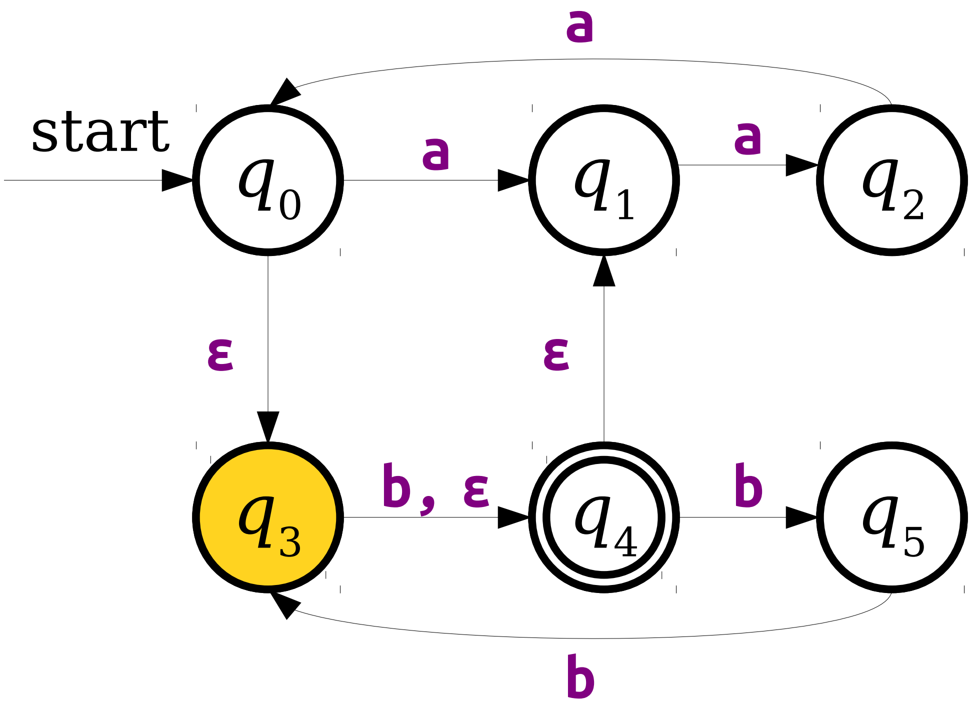
- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.





ε-Transitions

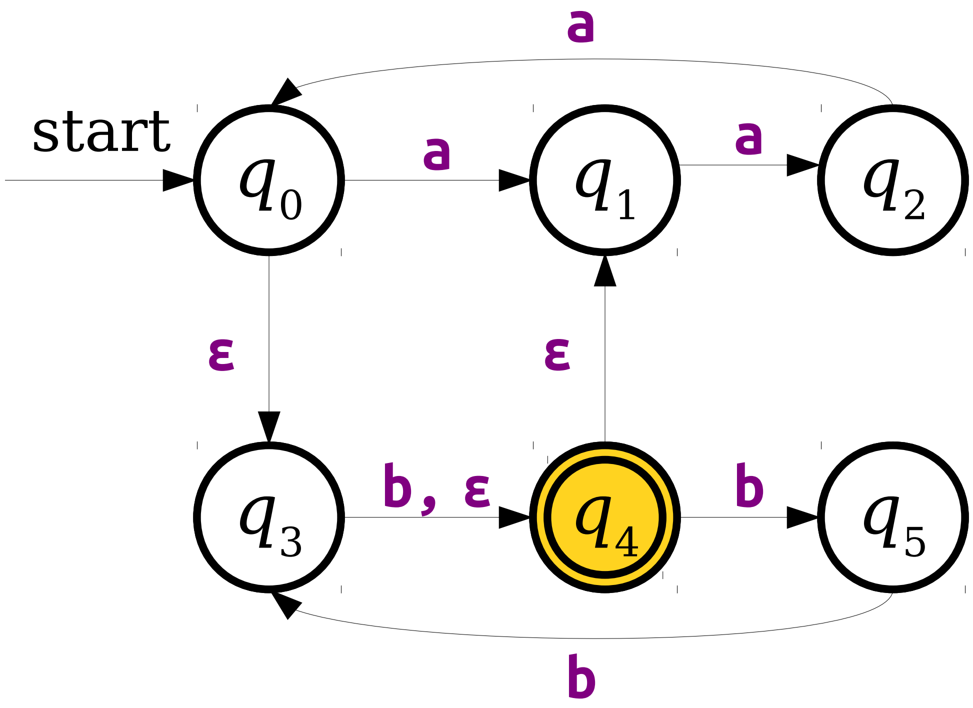
- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.



b a a b b

ε-Transitions

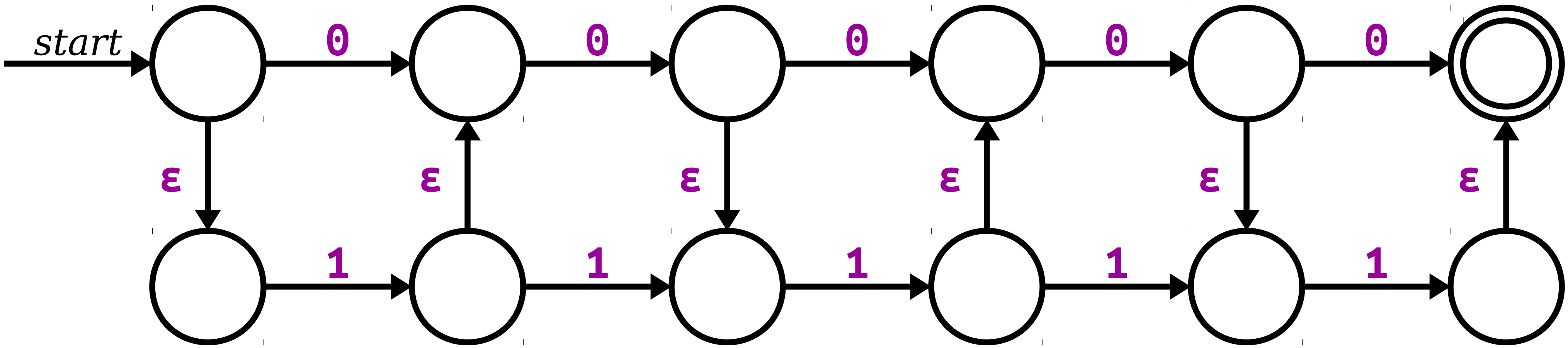
- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.



b a a b b

ε-Transitions

- NFAs have a special type of transition called the **ε-transition**.
- An NFA may follow any number of ε-transitions at any time without consuming any input.
- NFAs are not *required* to follow ε-transitions. It's simply another option at the machine's disposal.



Suppose we run the above NFA on the string **10110**. How many of the following statements are true?

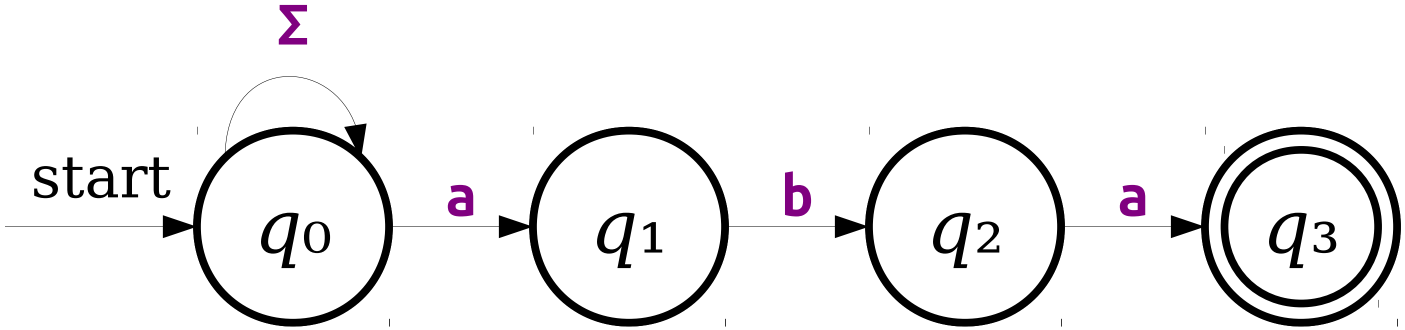
- There is at least one computation that finishes in an accepting state.
- There is at least one computation that finishes in a rejecting state.
- There is at least one computation that dies.
- This NFA accepts **10110**.
- This NFA rejects **10110**.

Answer at **PollEv.com/cs103** or text **CS103** to **22333** once to join, then **a number**.

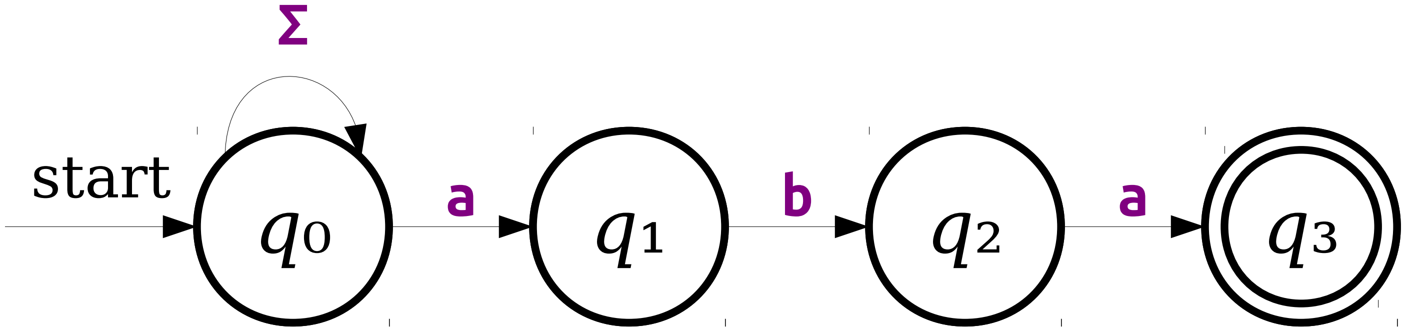
Intuiting Nondeterminism

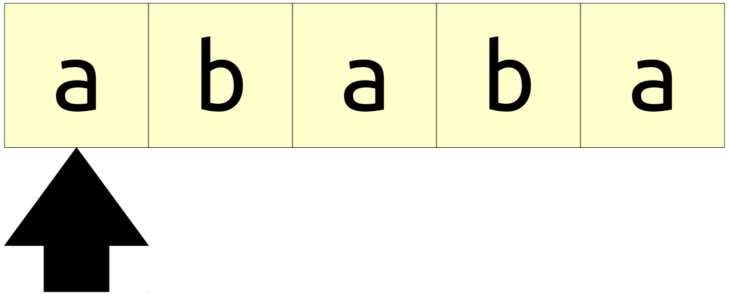
- Nondeterministic machines are a serious departure from physical computers. How can we build up an intuition for them?
- There are two particularly useful frameworks for interpreting nondeterminism:
  - ***Perfect guessing***
  - ***Massive parallelism***

Perfect Guessing

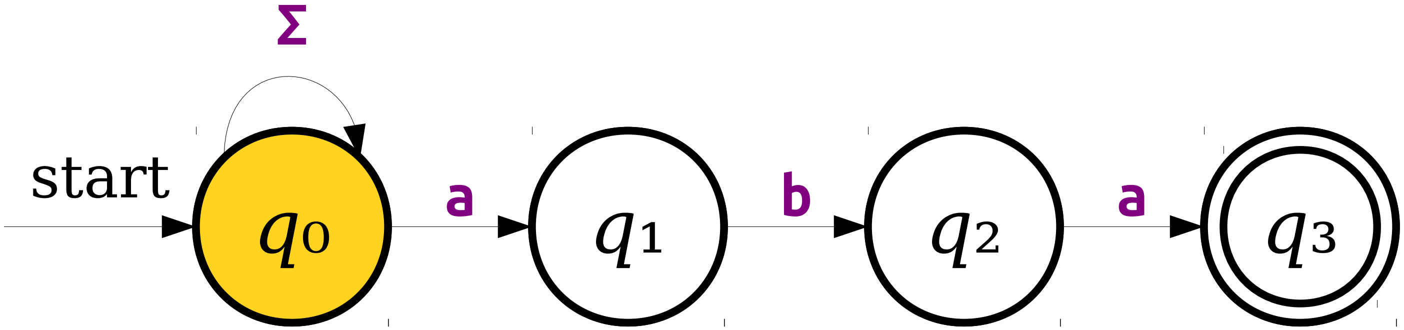


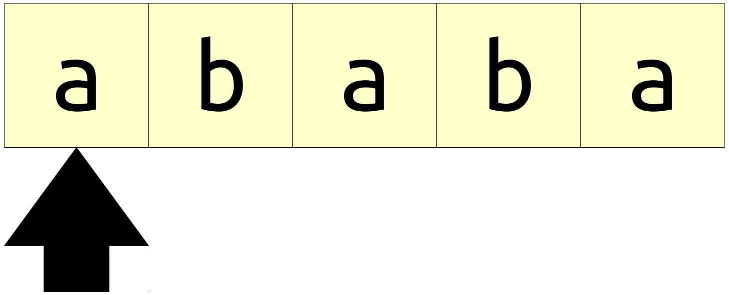
Perfect Guessing



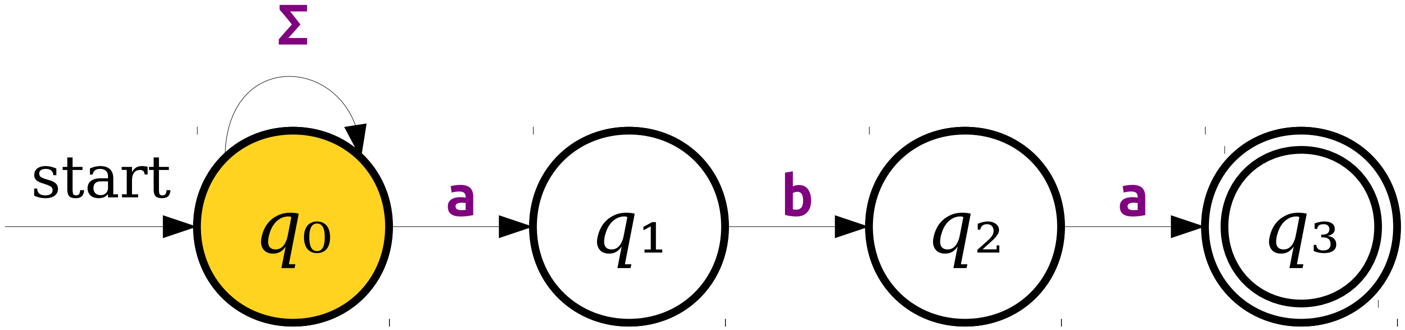


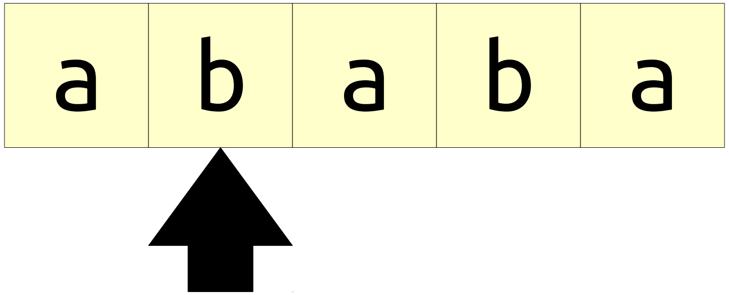
Perfect Guessing





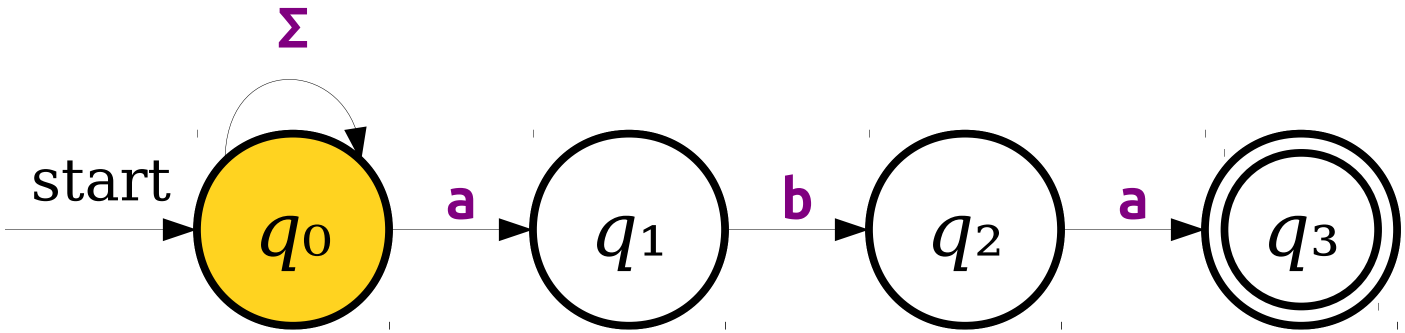
Perfect Guessing

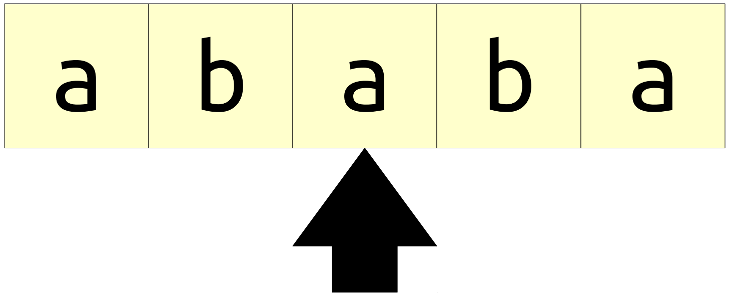




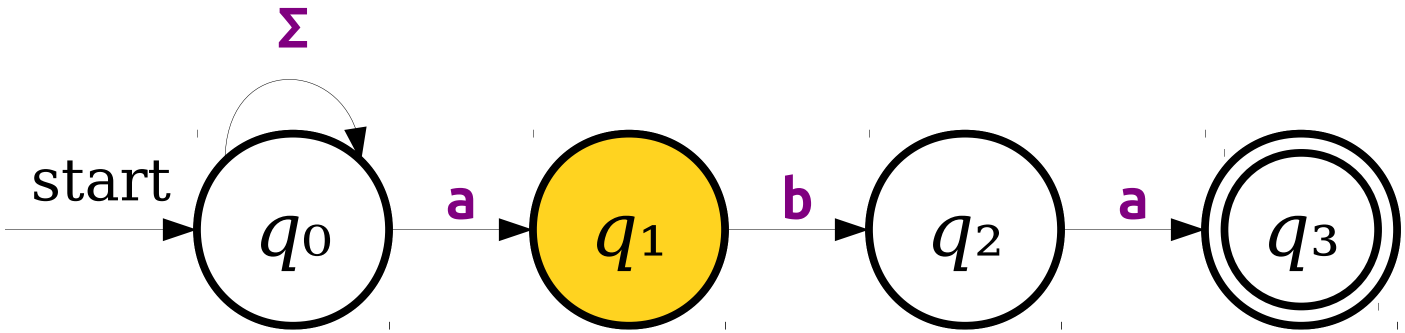
a	b	a	b	a

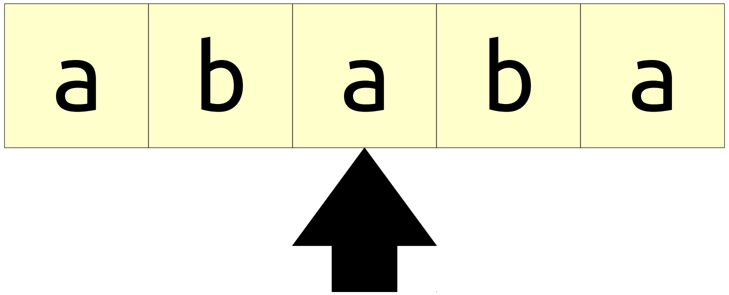
Perfect Guessing



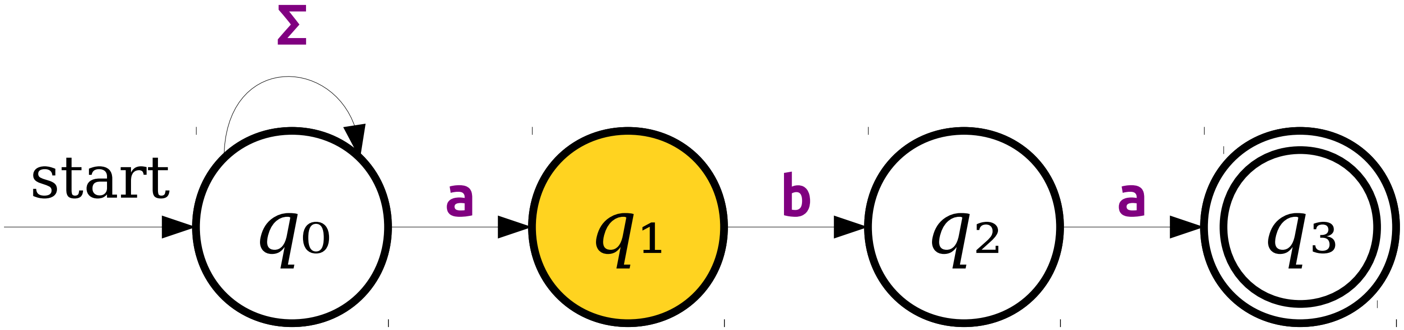


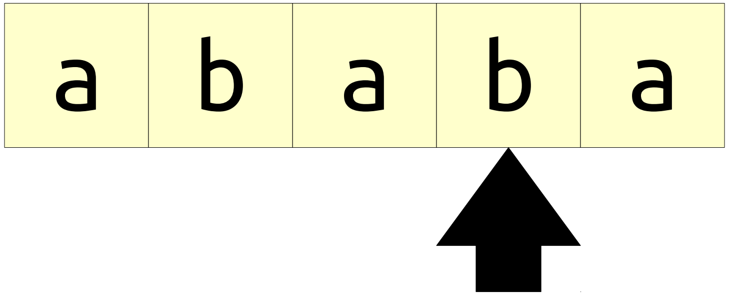
Perfect Guessing



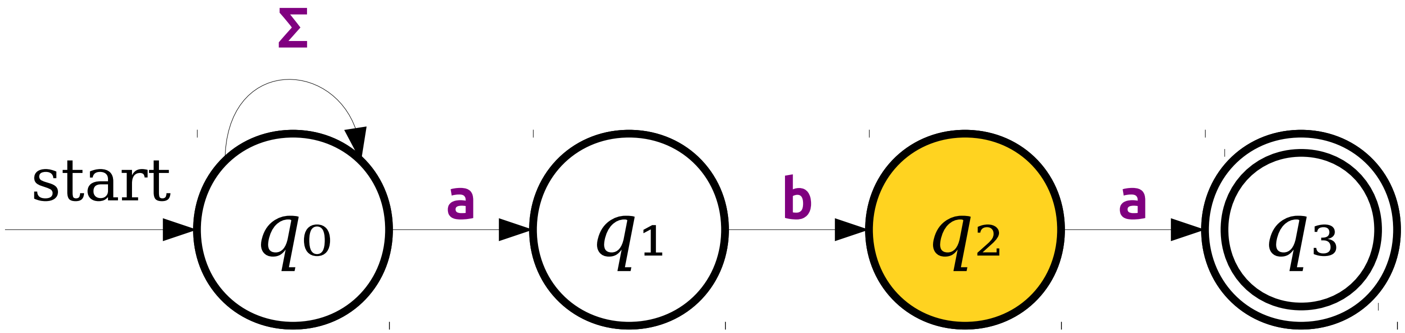


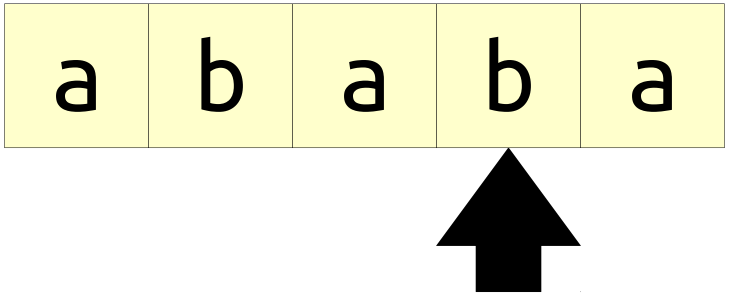
Perfect Guessing



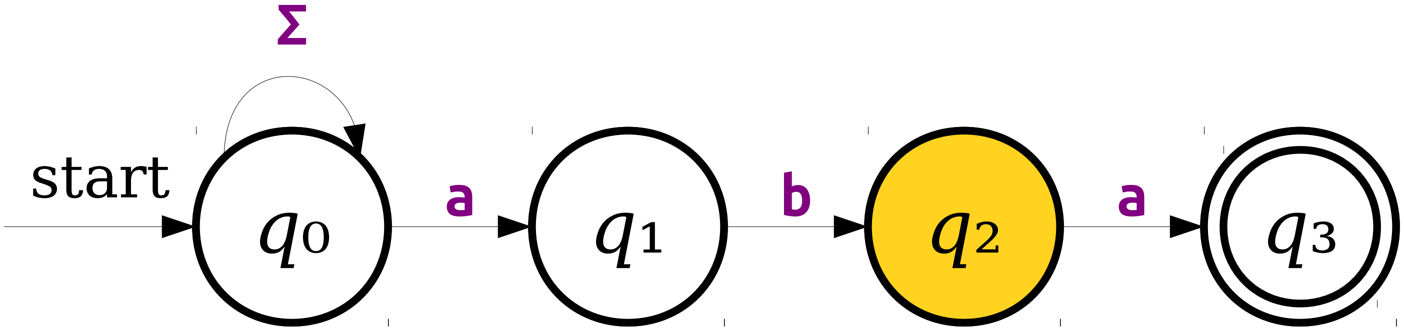


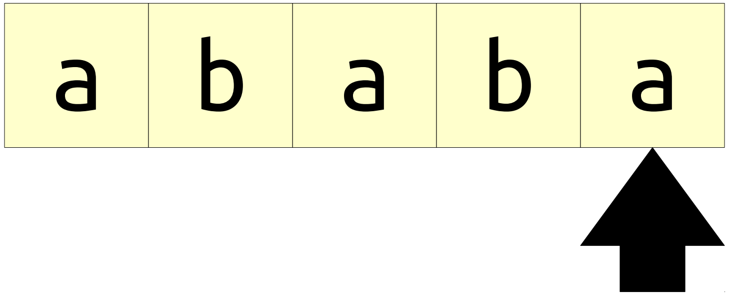
Perfect Guessing



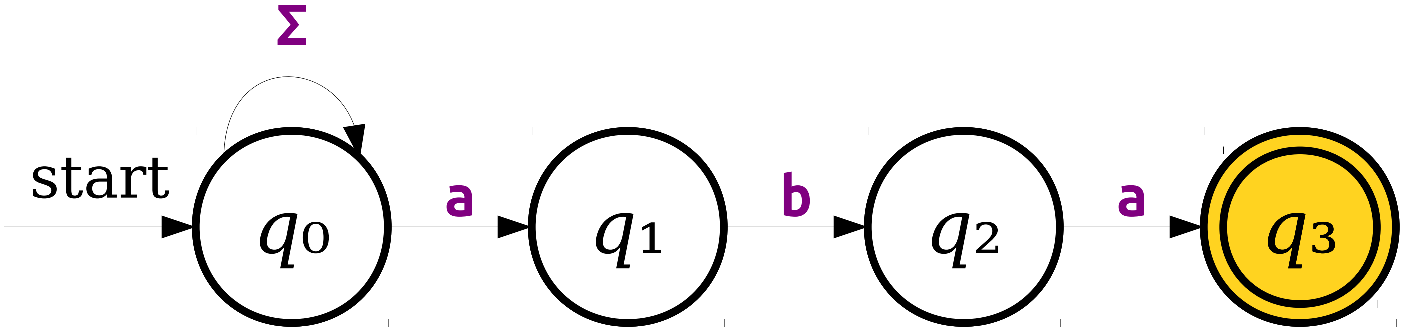


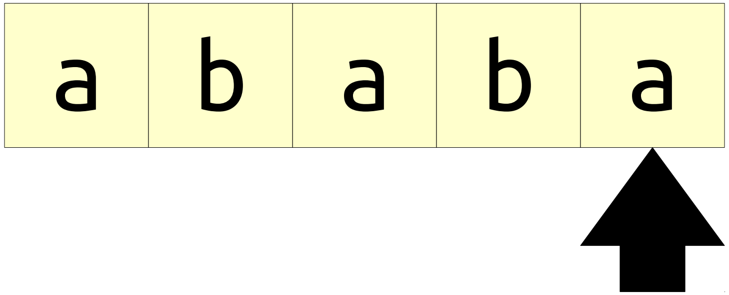
Perfect Guessing





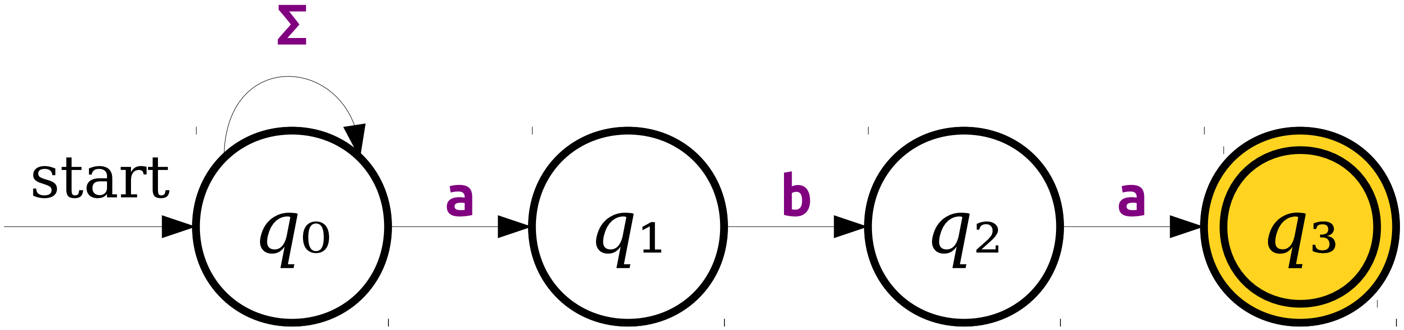
Perfect Guessing





a	b	a	b	a

Perfect Guessing

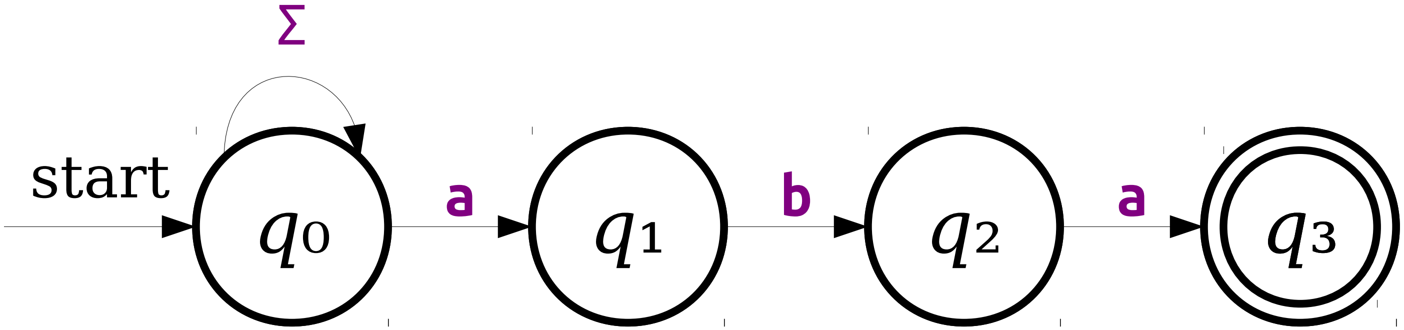


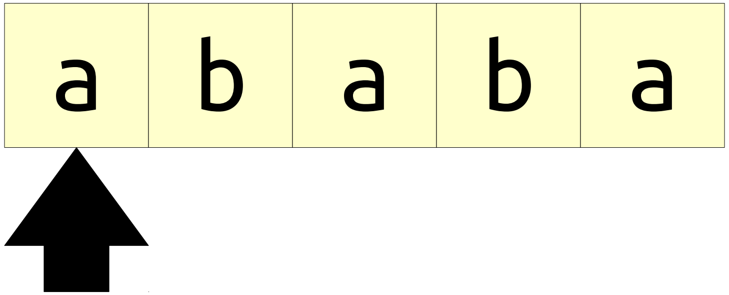
a	b	a	b	a

Perfect Guessing

- We can view nondeterministic machines as having ***Magic Superpowers*** that enable them to guess choices that lead to an accepting state.
  - If there is at least one choice that leads to an accepting state, the machine will guess it.
  - If there are no choices, the machine guesses any one of the wrong guesses.
- No known physical analog for this style of computation – this is totally new!

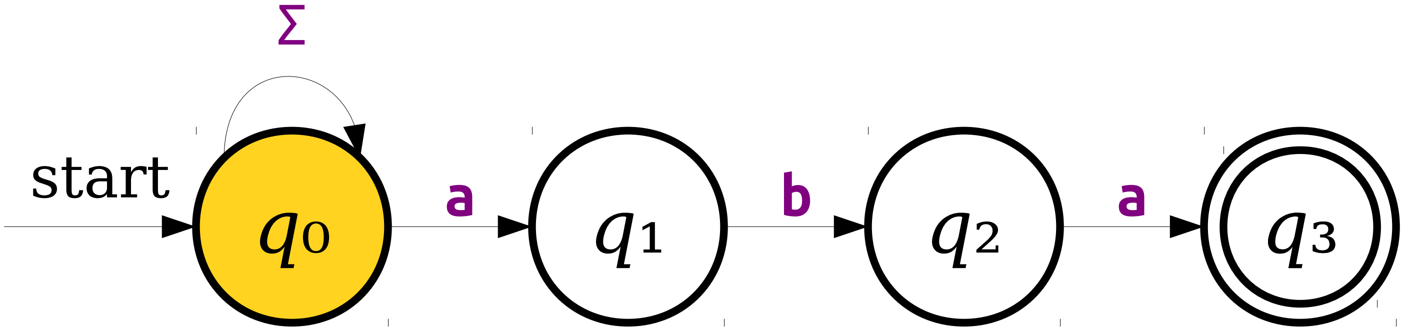
Massive Parallelism

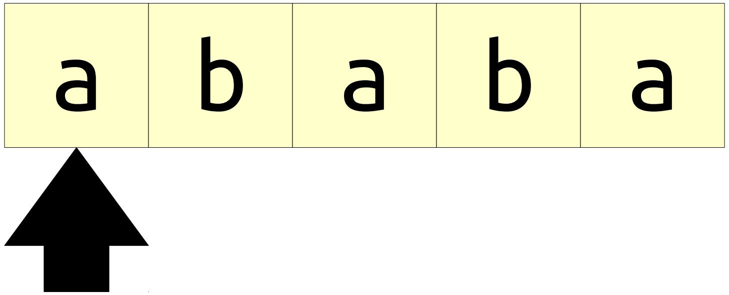




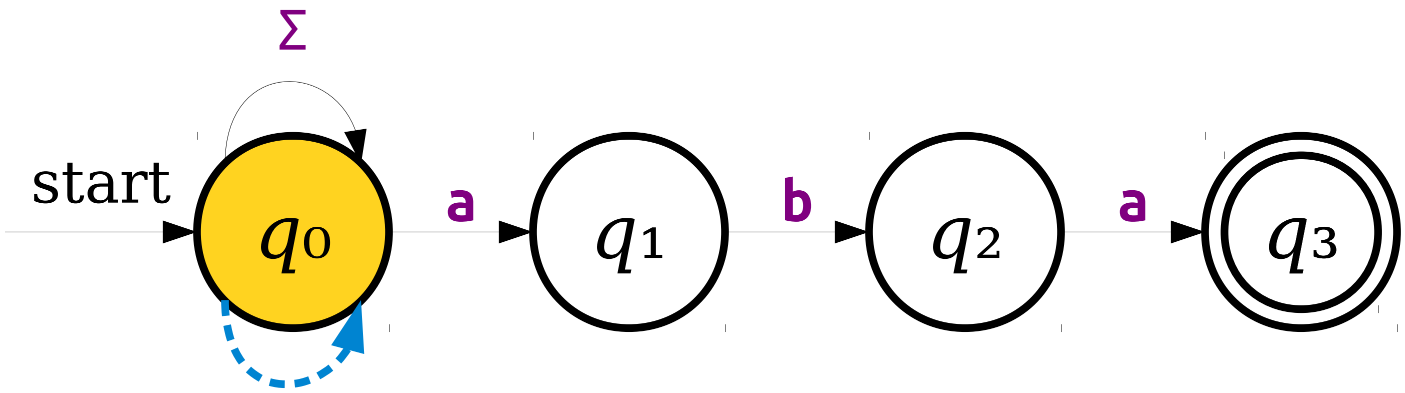
a	b	a	b	a

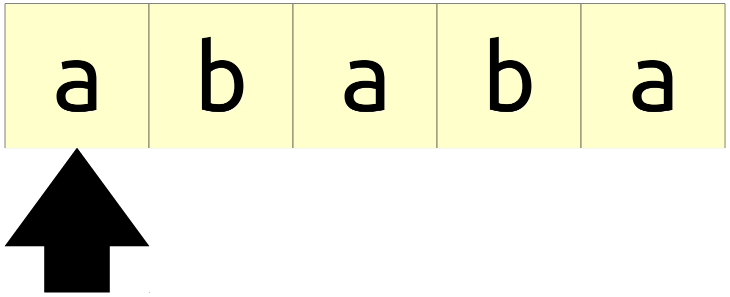
Massive Parallelism



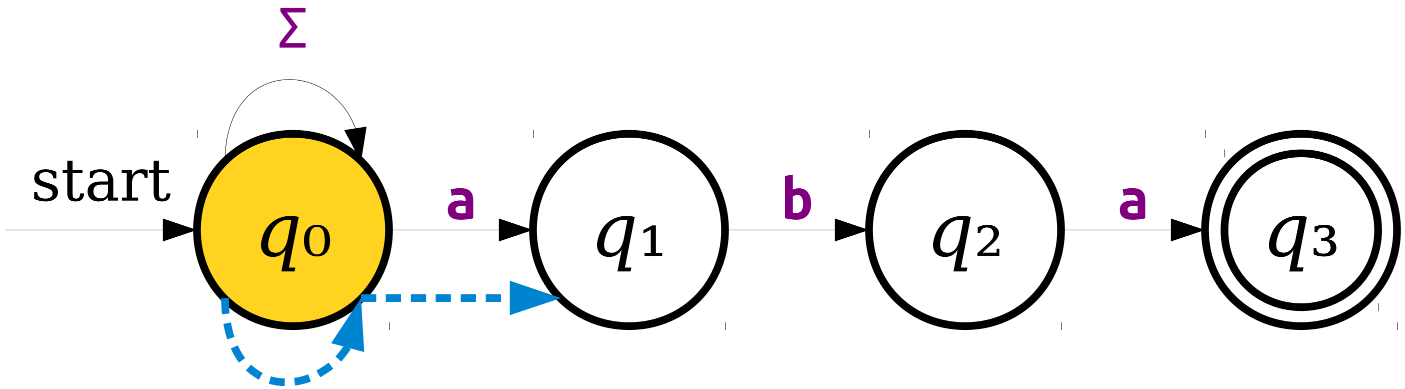


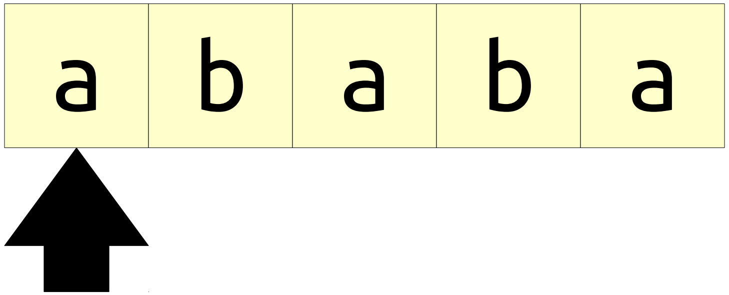
Massive Parallelism



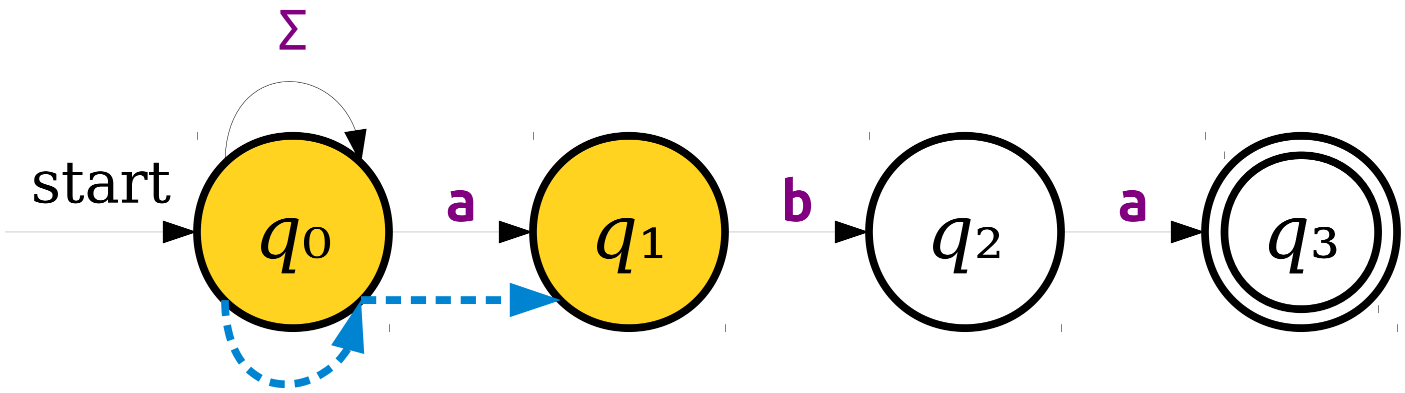


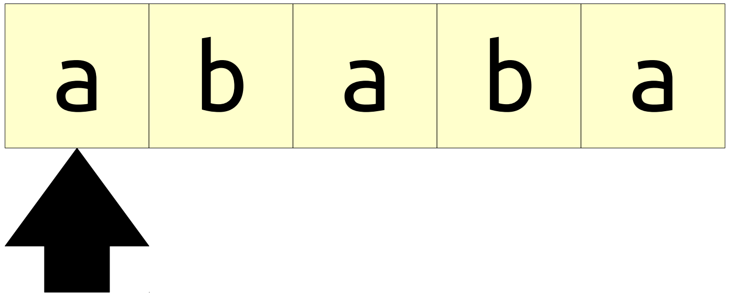
Massive Parallelism



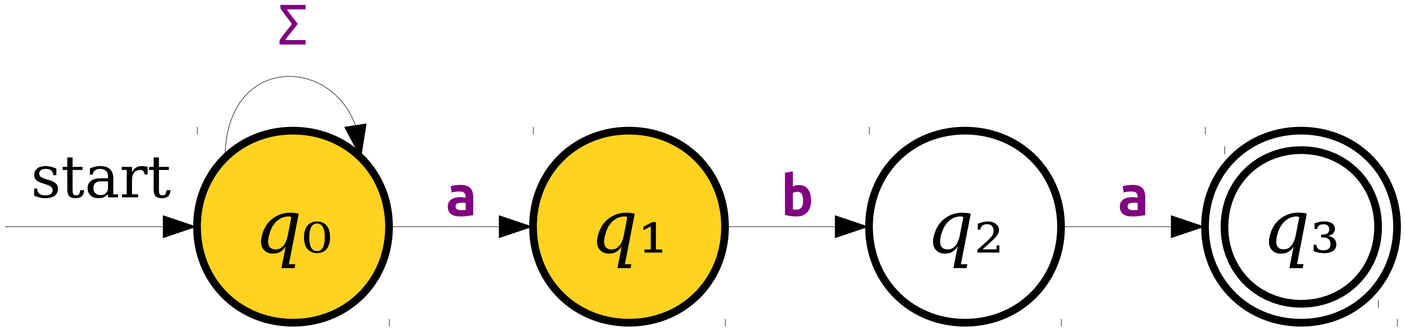


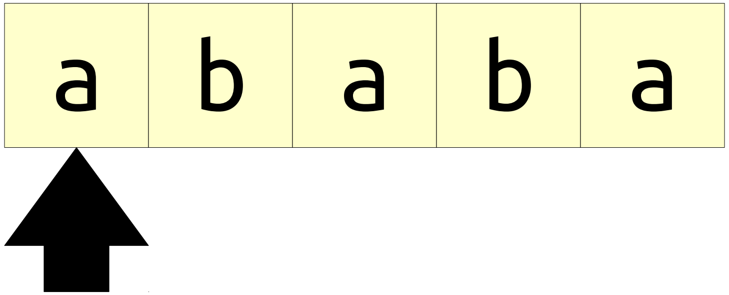
Massive Parallelism





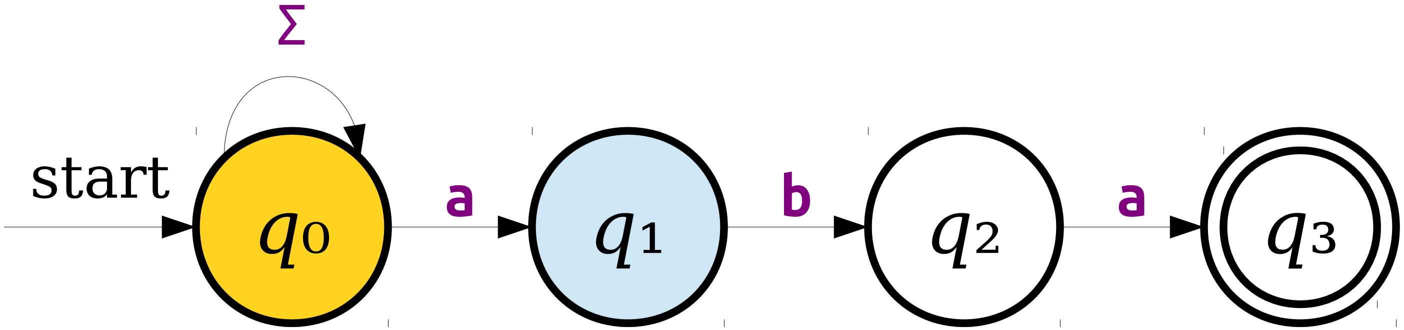
Massive Parallelism

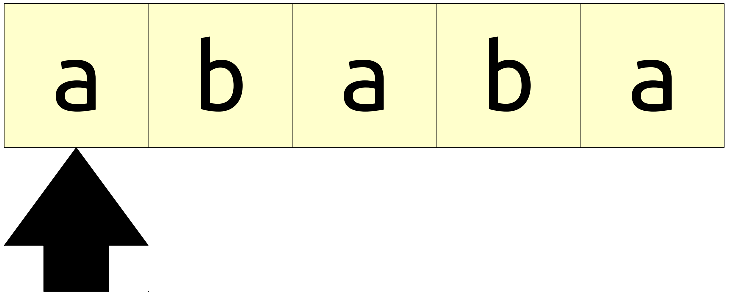




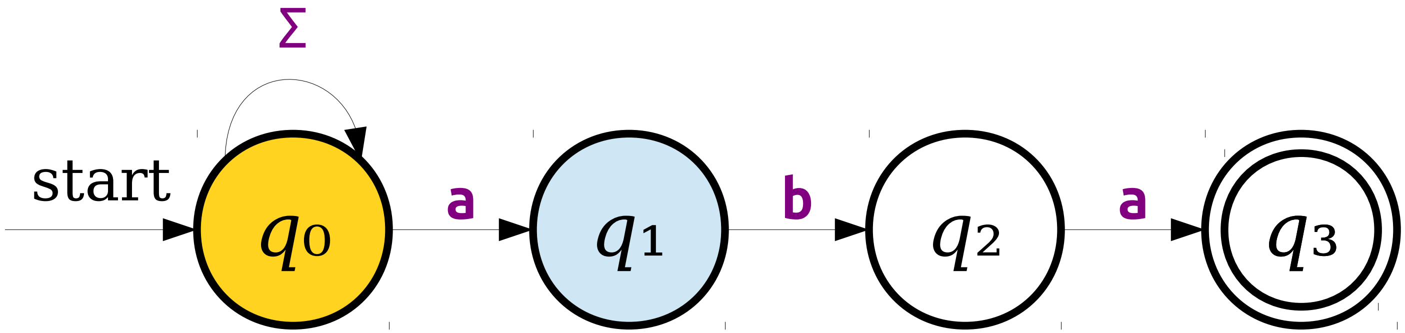
a	b	a	b	a

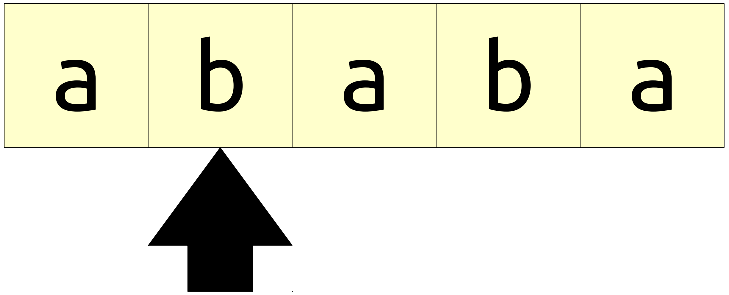
Massive Parallelism



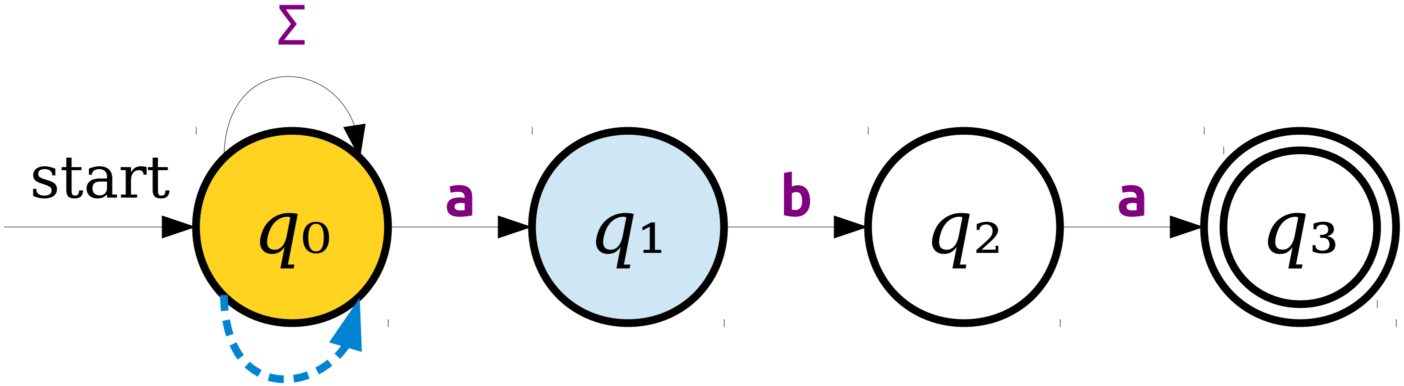


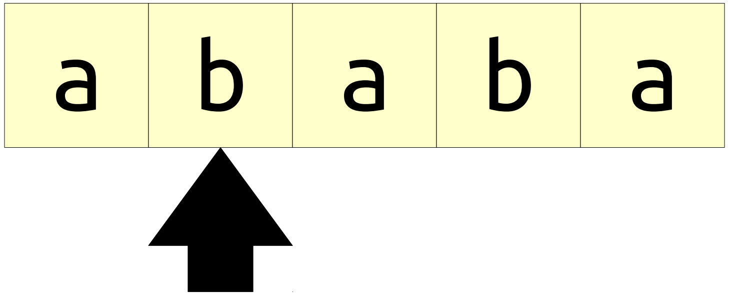
Massive Parallelism





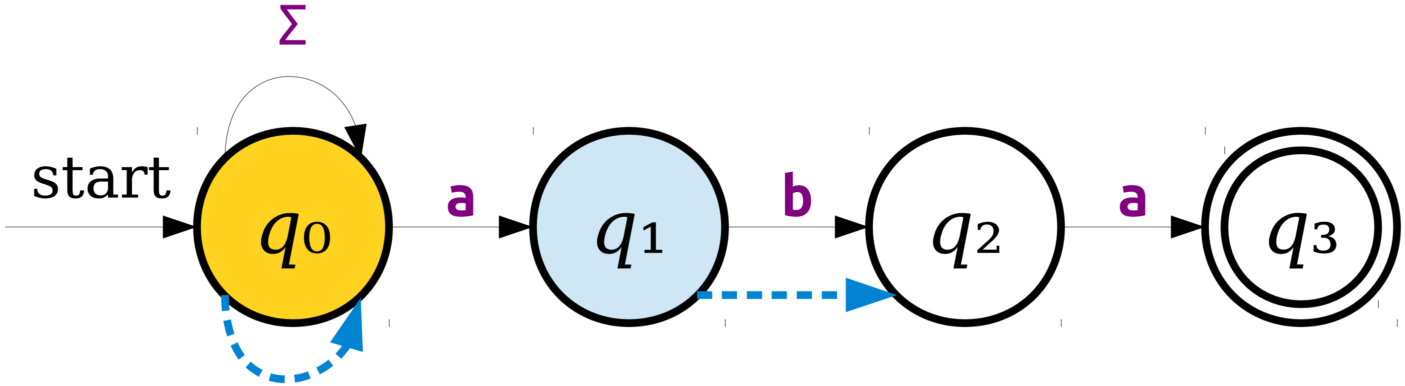
Massive Parallelism

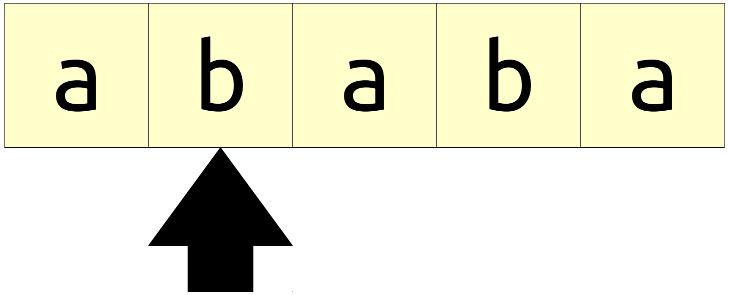




a	b	a	b	a

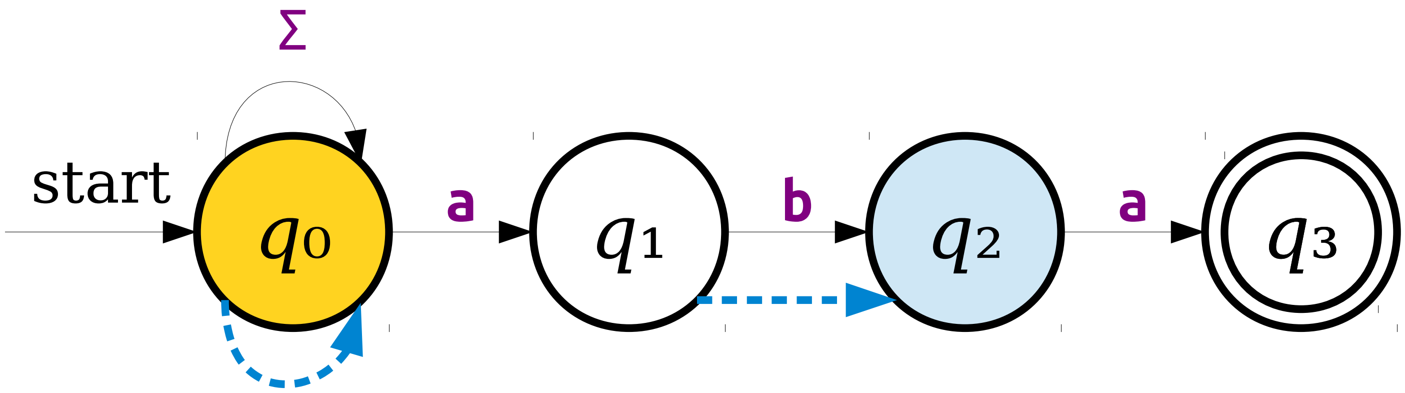
Massive Parallelism

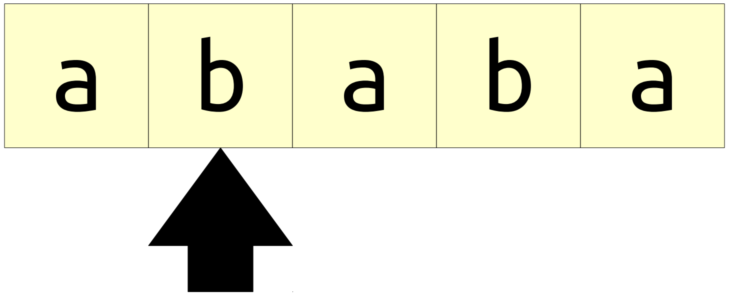




a	b	a	b	a

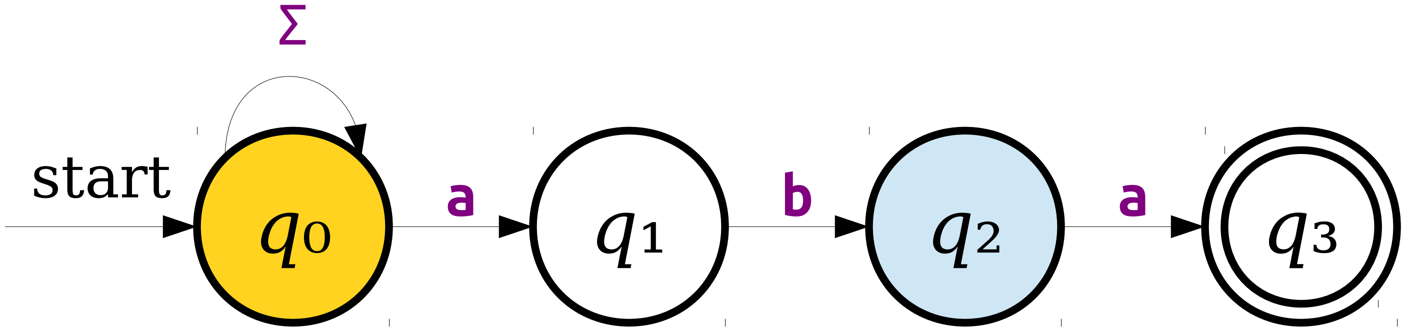
Massive Parallelism

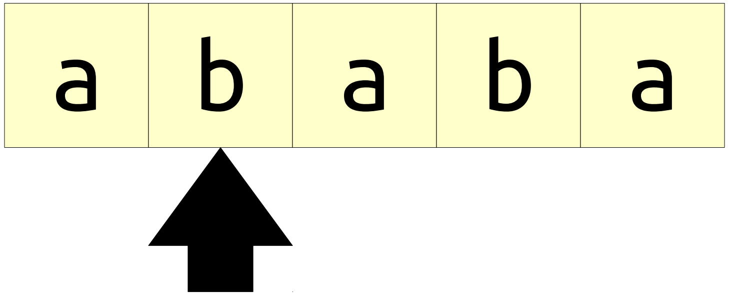




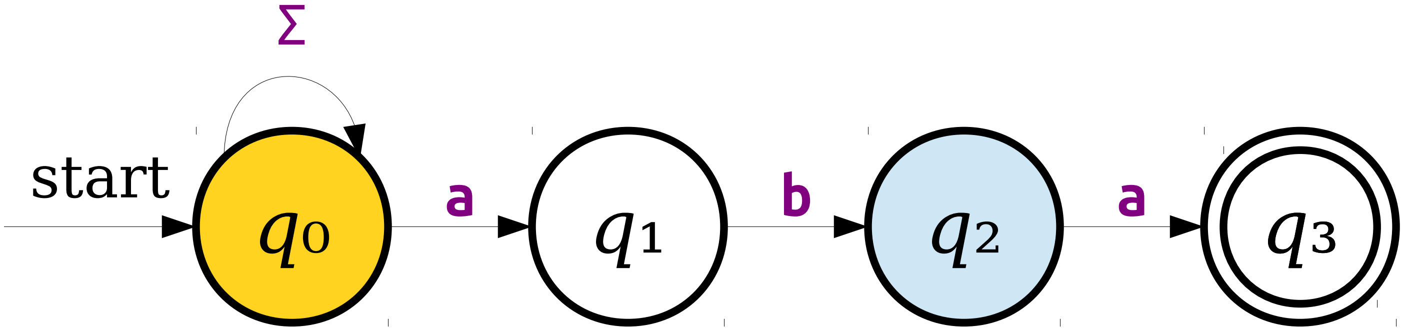
a	b	a	b	a

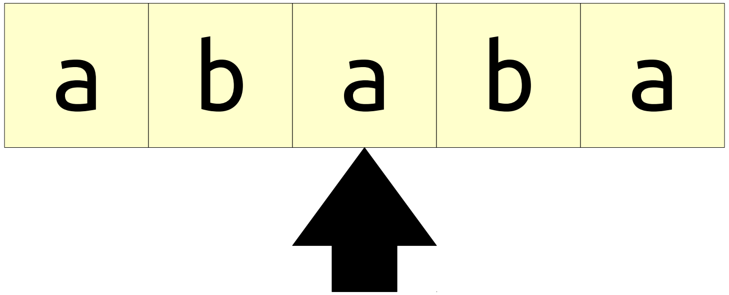
Massive Parallelism



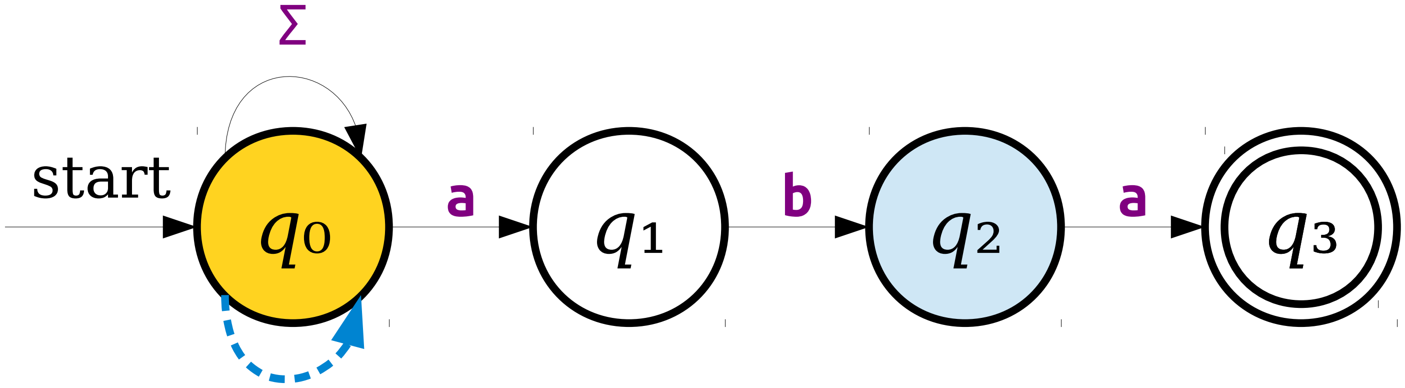


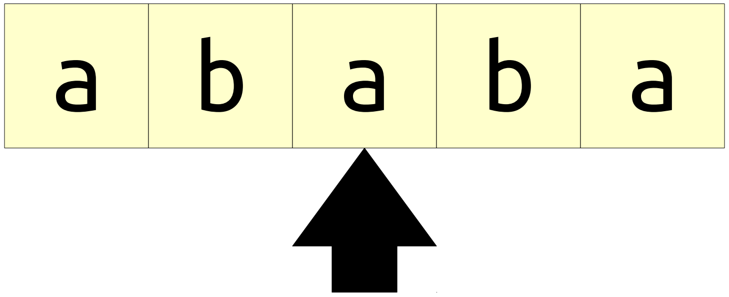
Massive Parallelism



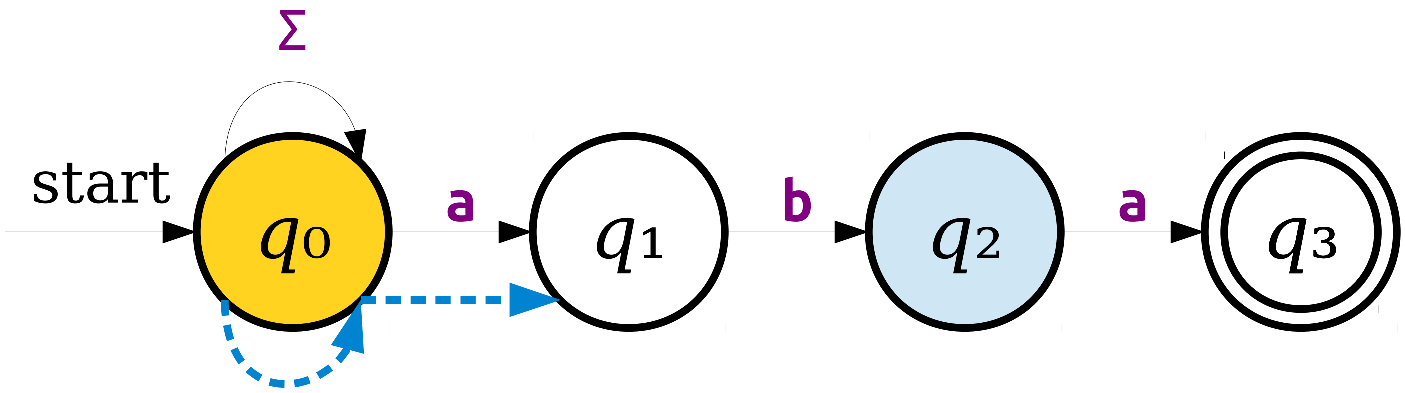


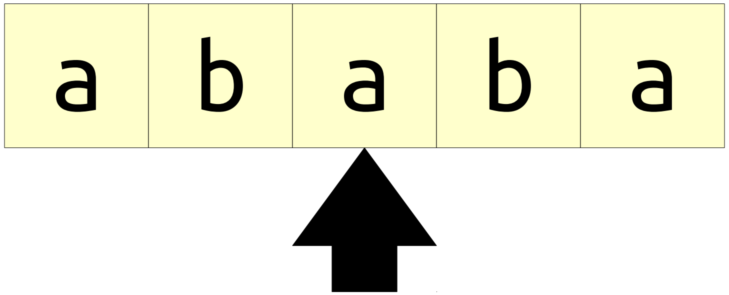
Massive Parallelism





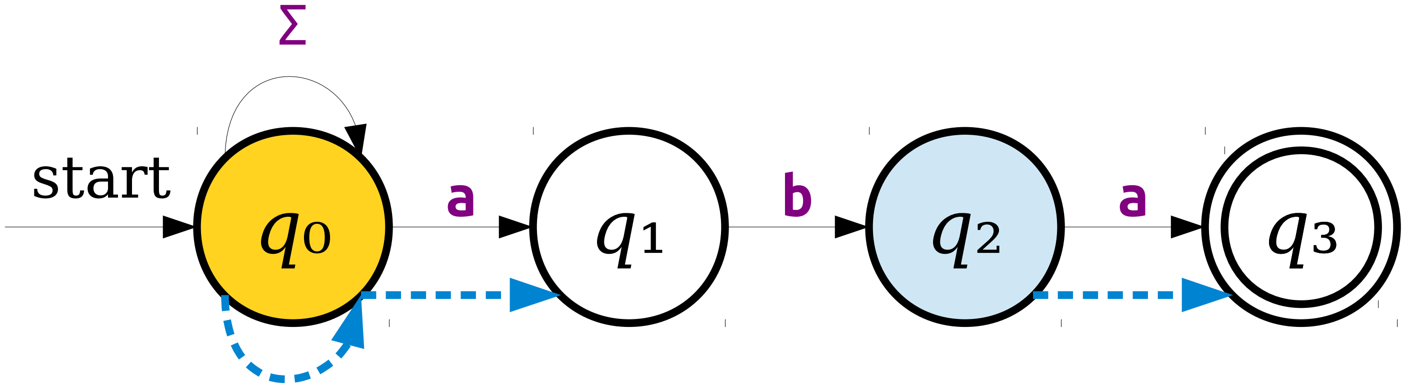
Massive Parallelism

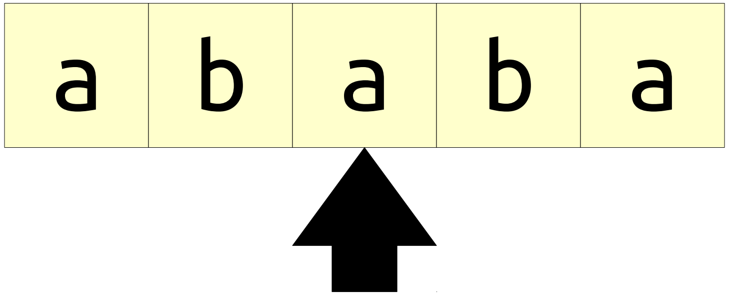




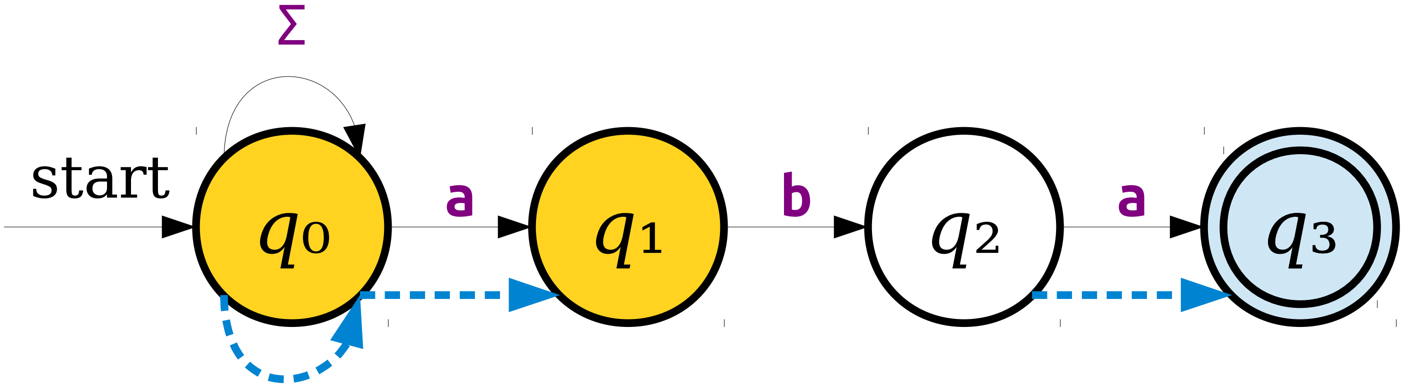
a	b	a	b	a

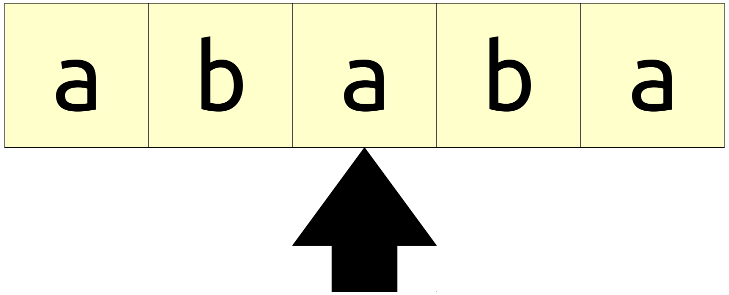
Massive Parallelism





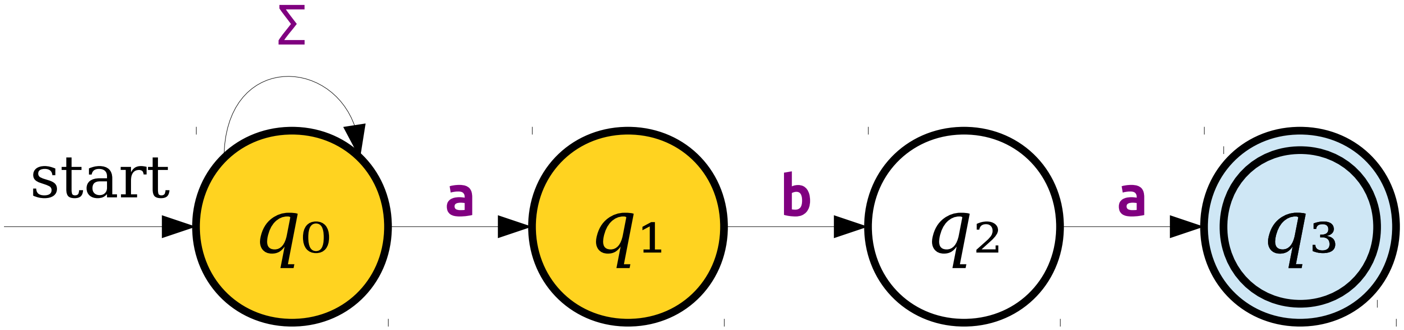
Massive Parallelism

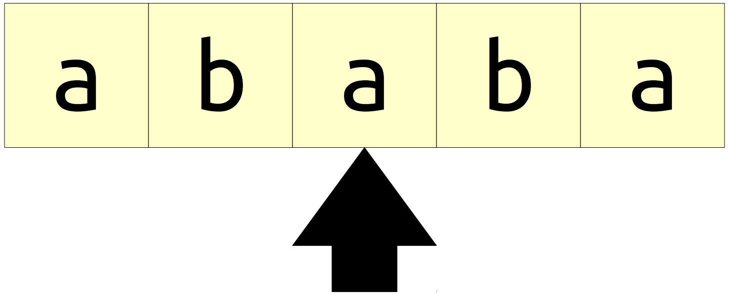




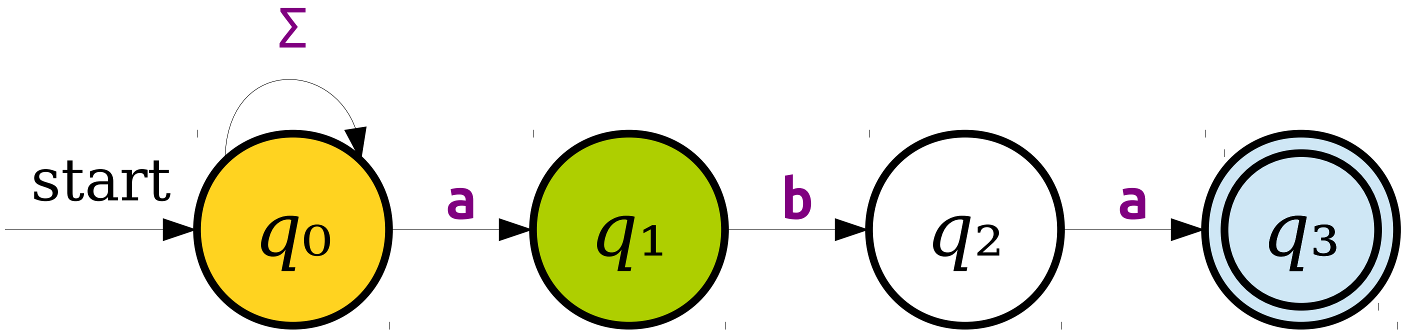
a	b	a	b	a

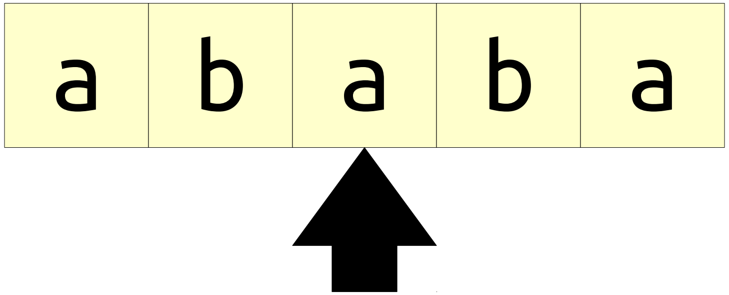
Massive Parallelism



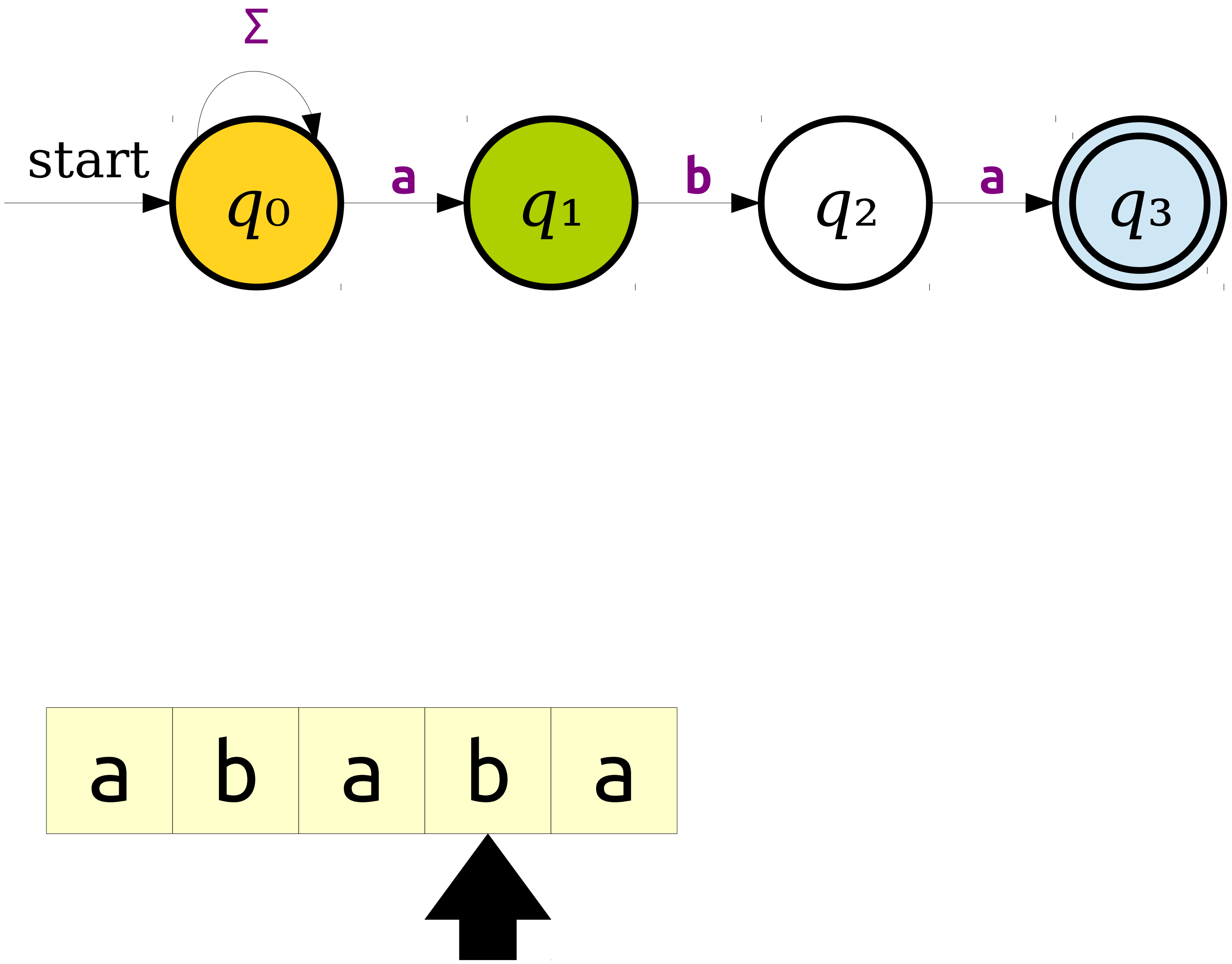


Massive Parallelism

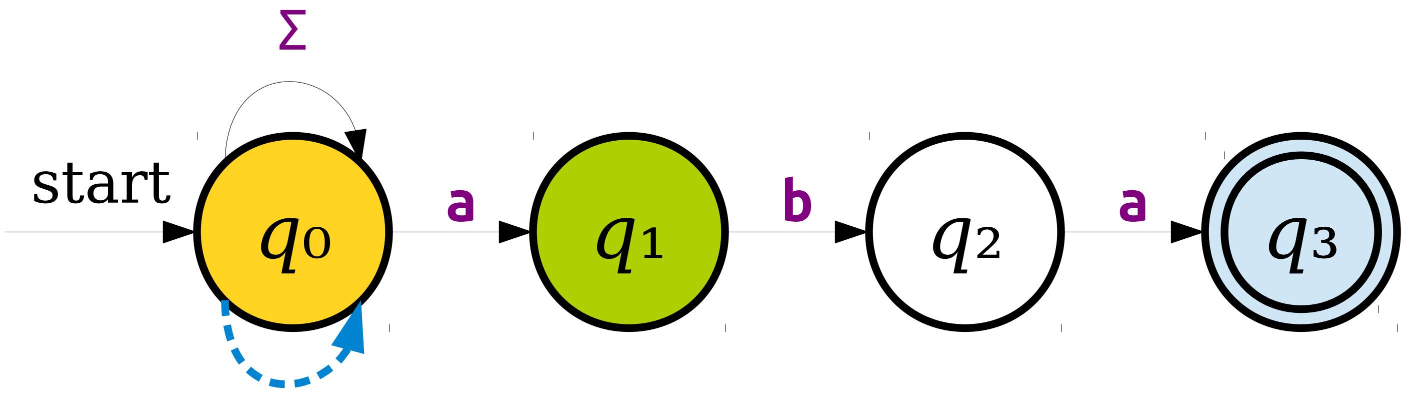


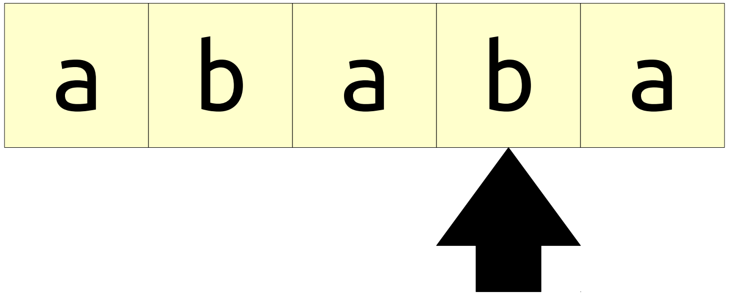


Massive Parallelism



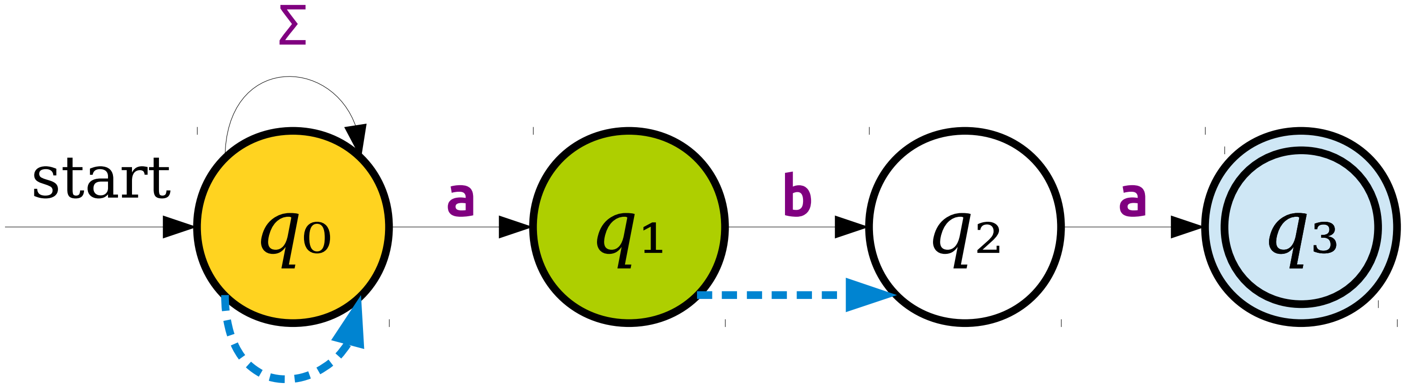
Massive Parallelism

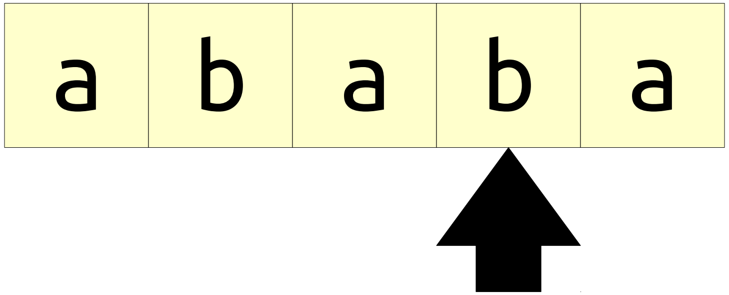




a	b	a	b	a

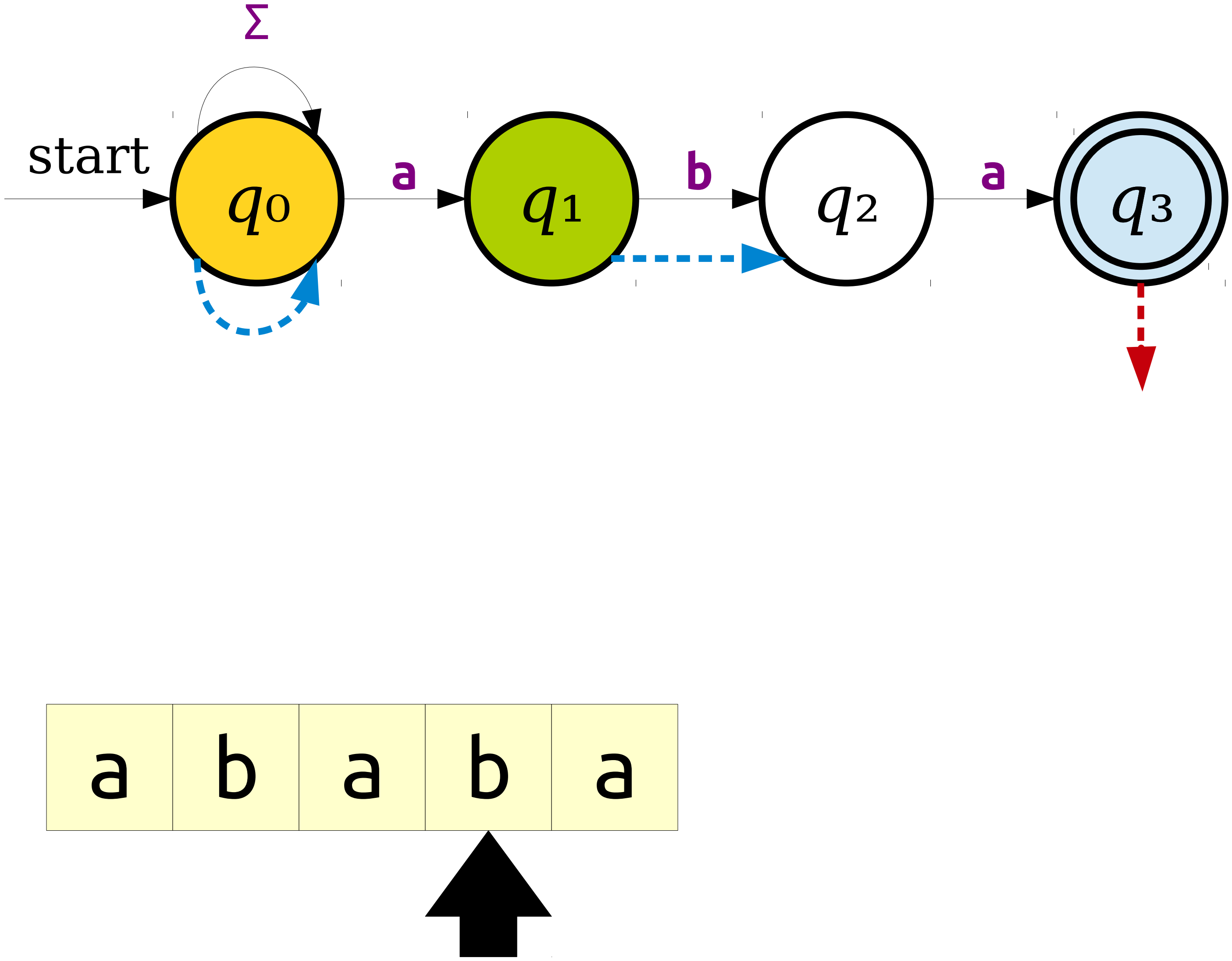
Massive Parallelism



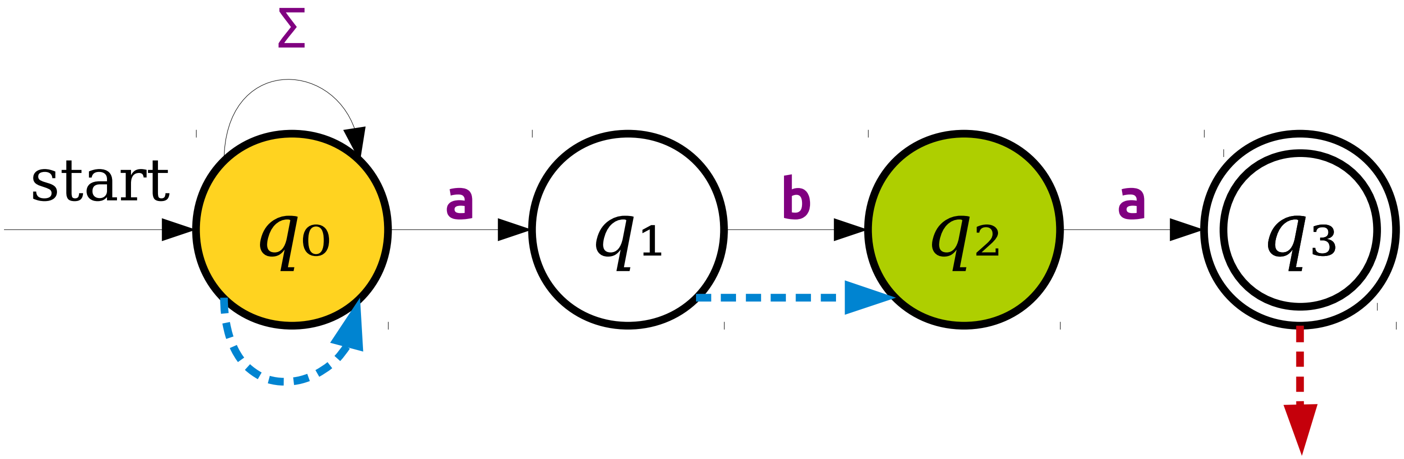


a	b	a	b	a

Massive Parallelism

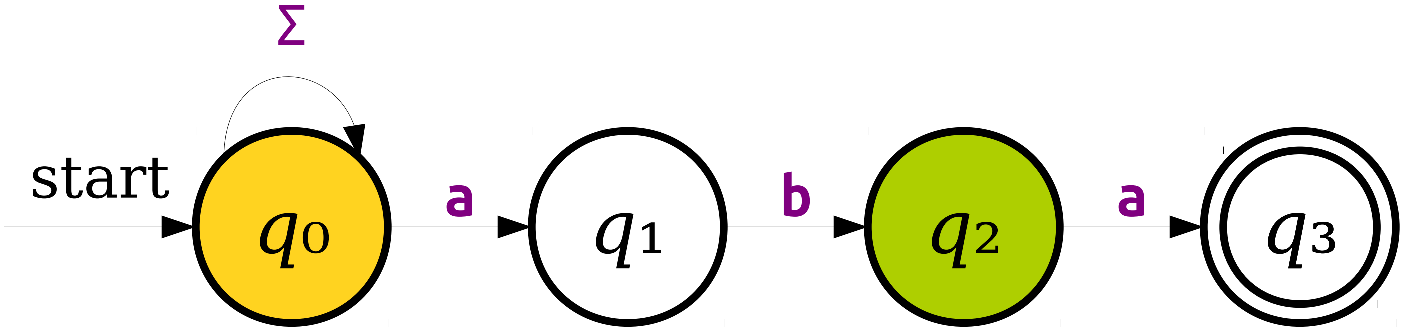


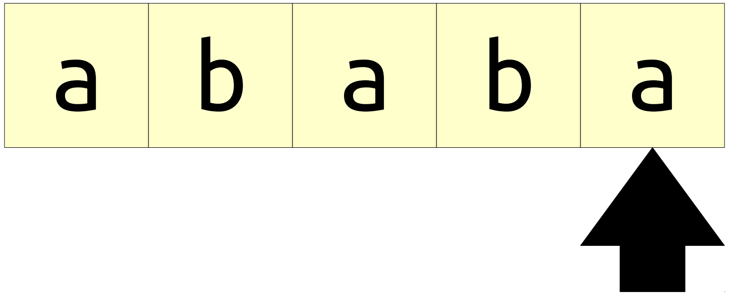
Massive Parallelism



a	b	a	b	a

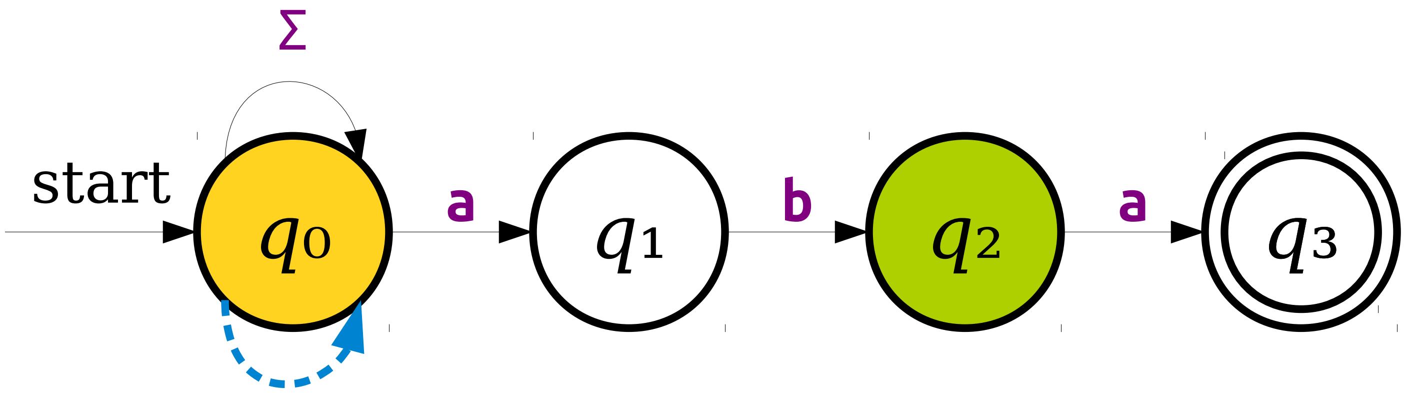
Massive Parallelism

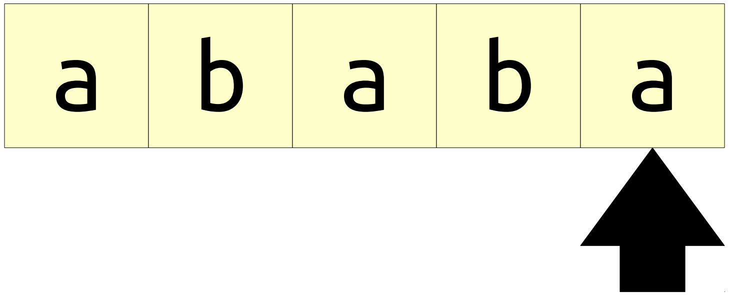




a	b	a	b	a

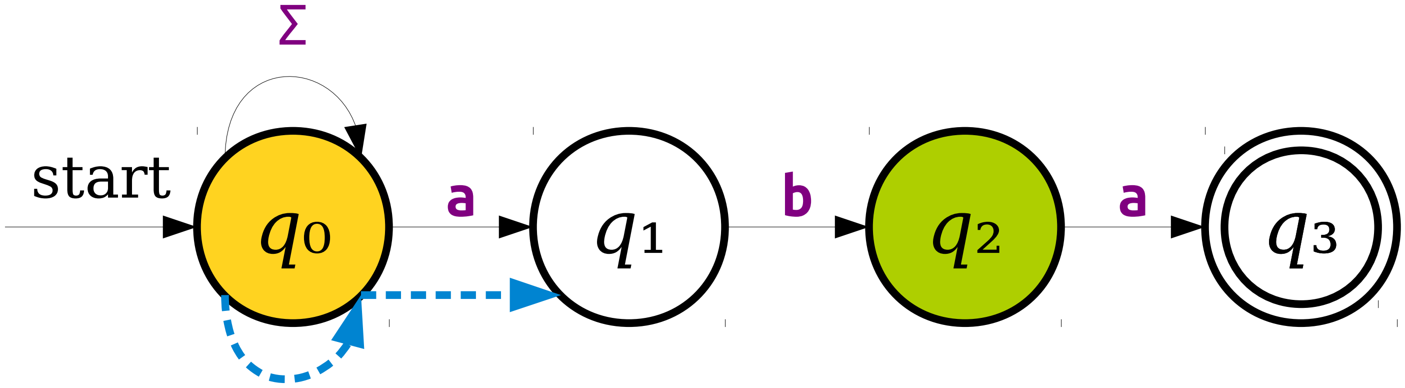
Massive Parallelism

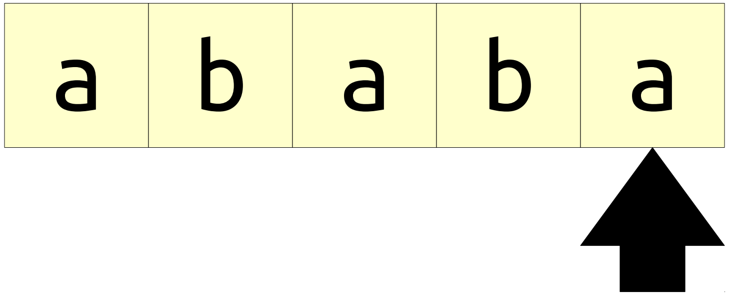




a	b	a	b	a

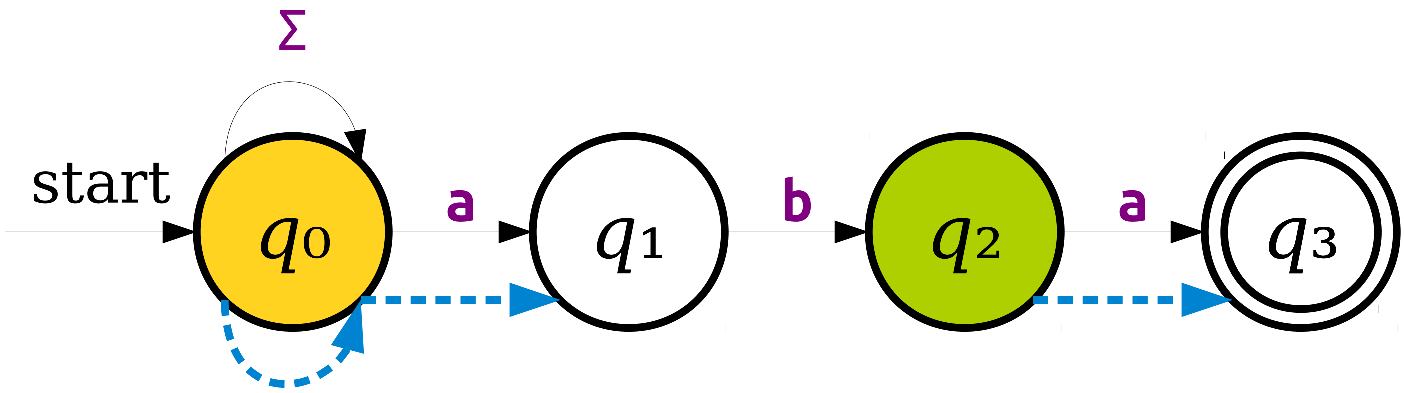
Massive Parallelism

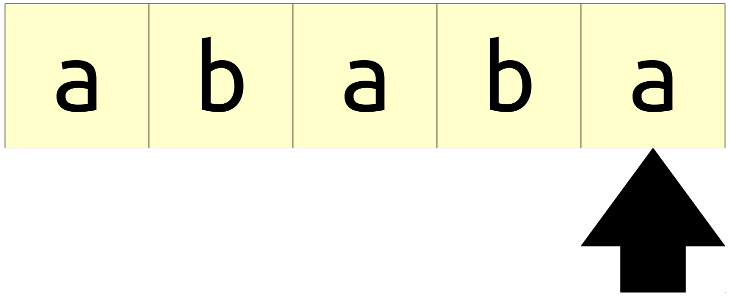




a	b	a	b	a

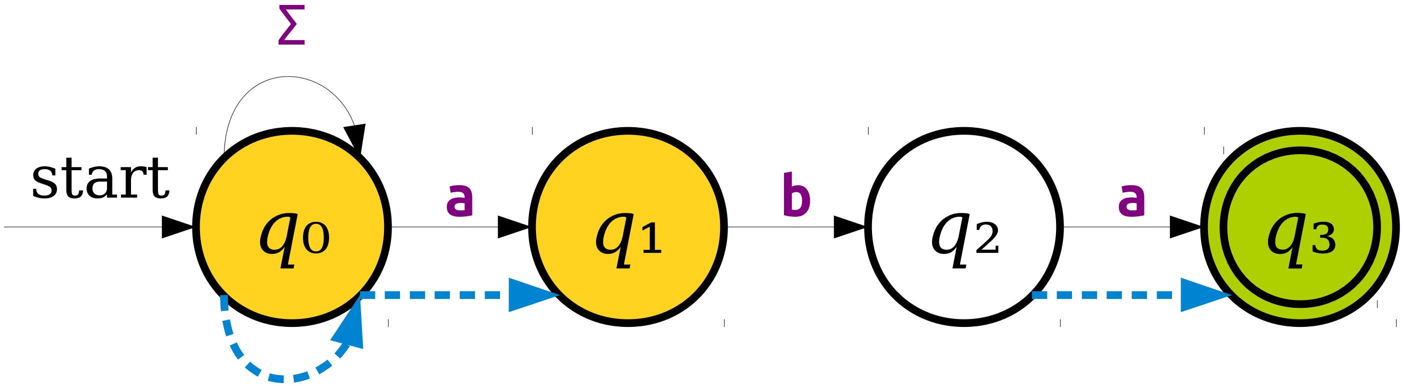
Massive Parallelism





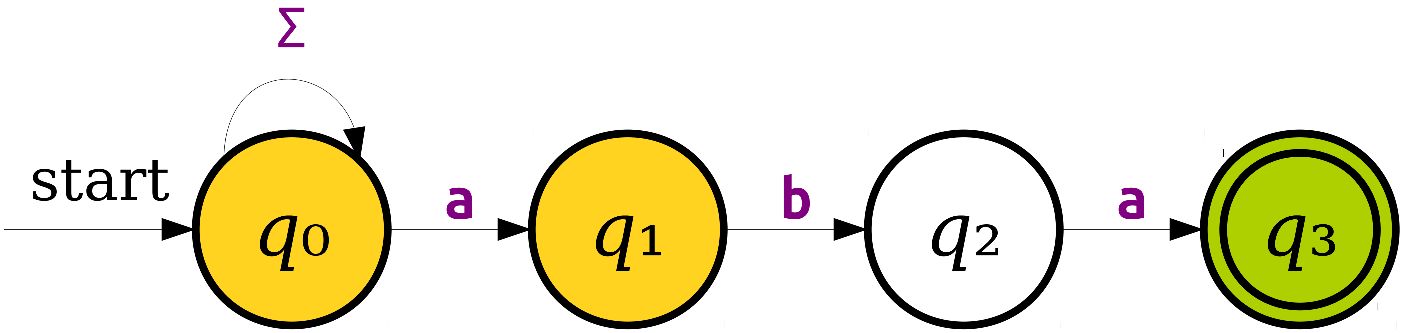
a	b	a	b	a

Massive Parallelism



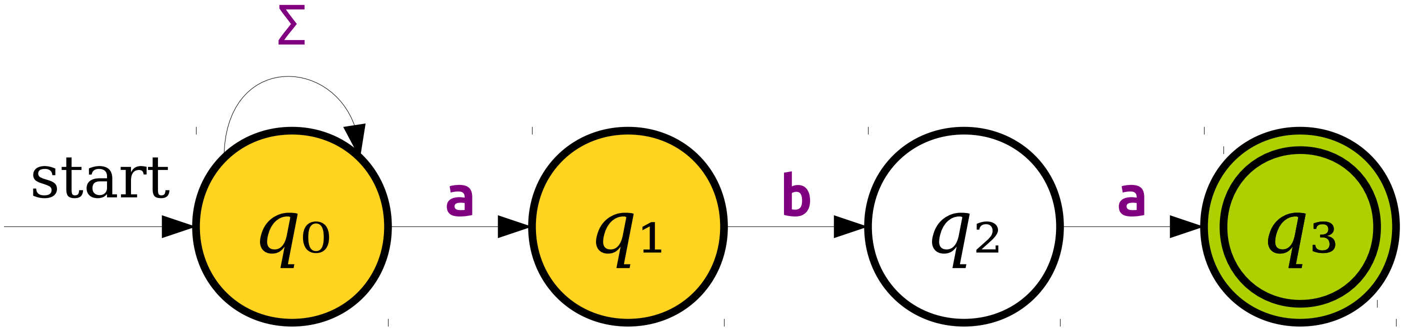
a	b	a	b	a

Massive Parallelism



a	b	a	b	a

Massive Parallelism



a	b	a	b	a

We're in at least one accepting state, so there's some path that gets us to an accepting state.

Therefore, we accept!

Massive Parallelism

- An NFA can be thought of as a DFA that can be in many states at once.
- At each point in time, when the NFA needs to follow a transition, it tries all the options at the same time.
- (Here's a rigorous explanation about how this works; read this on your own time).
    - Start off in the set of all states formed by taking the start state and including each state that can be reached by zero or more ε-transitions.
    - When you read a symbol `a` in a set of states *S*:
        - Form the set *S'* of states that can be reached by following a single `a` transition from some state in *S*.
        - Your new set of states is the set of states in *S'*, plus the states reachable from *S'* by following zero or more ε-transitions.

So What?

- Each intuition of nondeterminism is useful in a different setting:
  - Perfect guessing is a great way to think about how to design a machine.
  - Massive parallelism is a great way to test machines – and has nice theoretical implications.
- Nondeterministic machines may not be feasible, but they give a great basis for interesting questions:
  - Can any problem that can be solved by a nondeterministic machine be solved by a deterministic machine?
  - Can any problem that can be solved by a nondeterministic machine be solved ***efficiently*** by a deterministic machine?
- The answers vary from automaton to automaton.

Designing NFAs

Designing NFAs

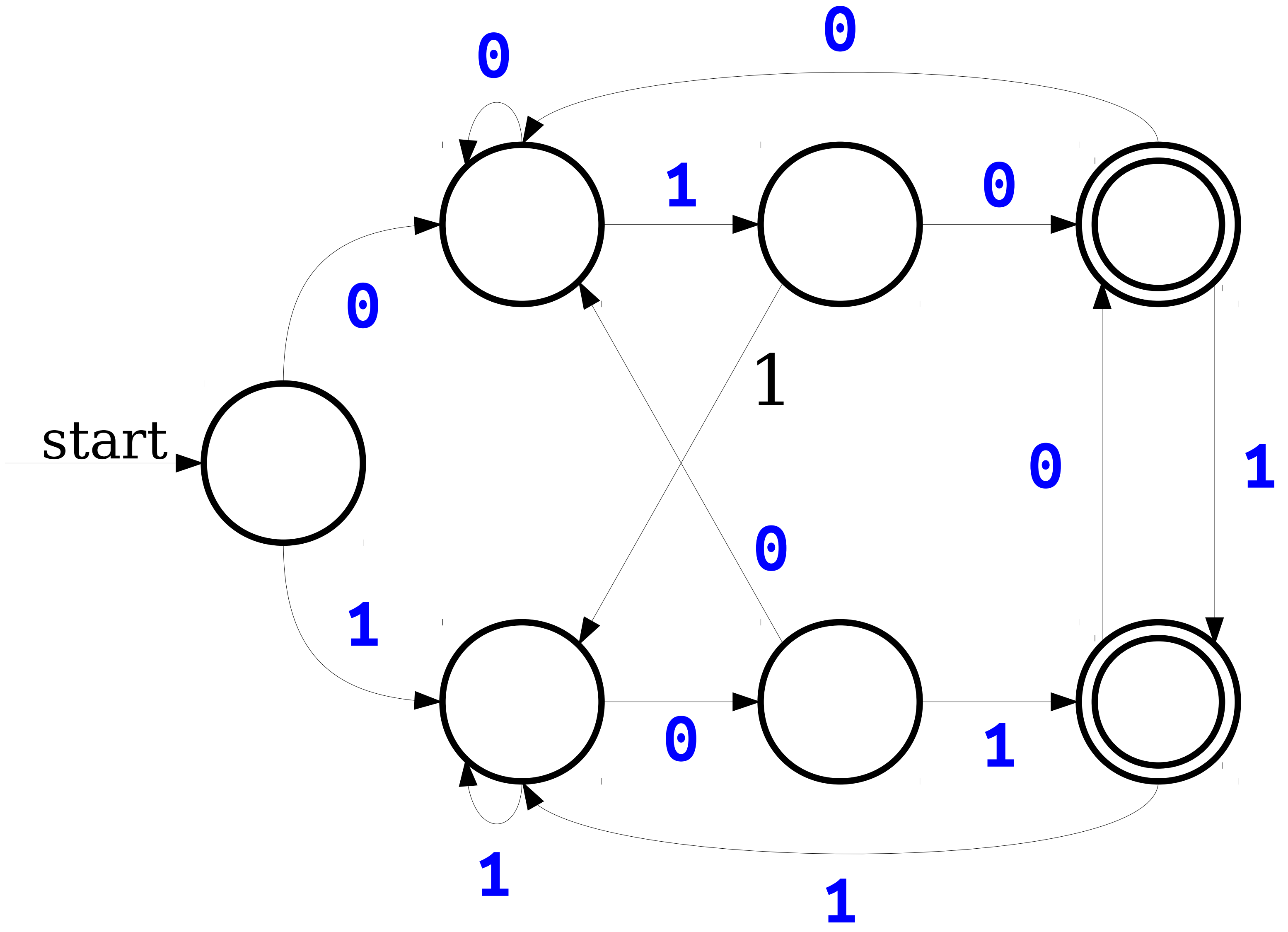
- When designing NFAs, *embrace the nondeterminism!*
- Good model: ***Guess-and-check***:
  - Is there some information that you'd really like to have? Have the machine *nondeterministically guess* that information.
  - Then, have the machine *deterministically check* that the choice was correct.
- The *guess* phase corresponds to trying lots of different options.
- The *check* phase corresponds to filtering out bad guesses or wrong options.

Guess-and-Check

$L = \{\ w \in \{\mathtt{0}, \mathtt{1}\}^* \mid w \text{ ends in } \mathtt{010} \text{ or } \mathtt{101}\ \}$

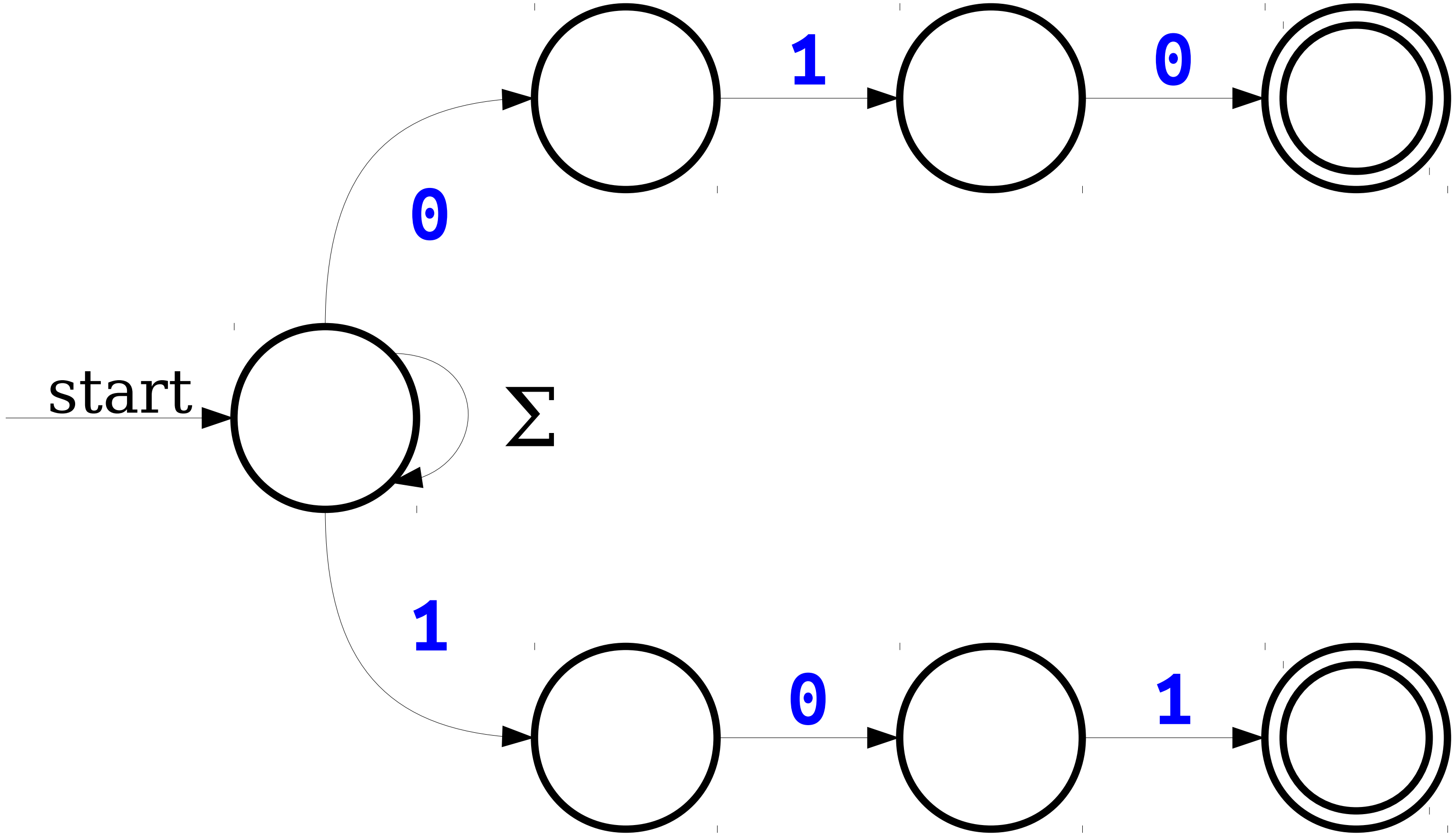
Guess-and-Check

$L = \{ w \in \{0, 1\}^* \mid w \text{ ends in } 010 \text{ or } 101 \}$



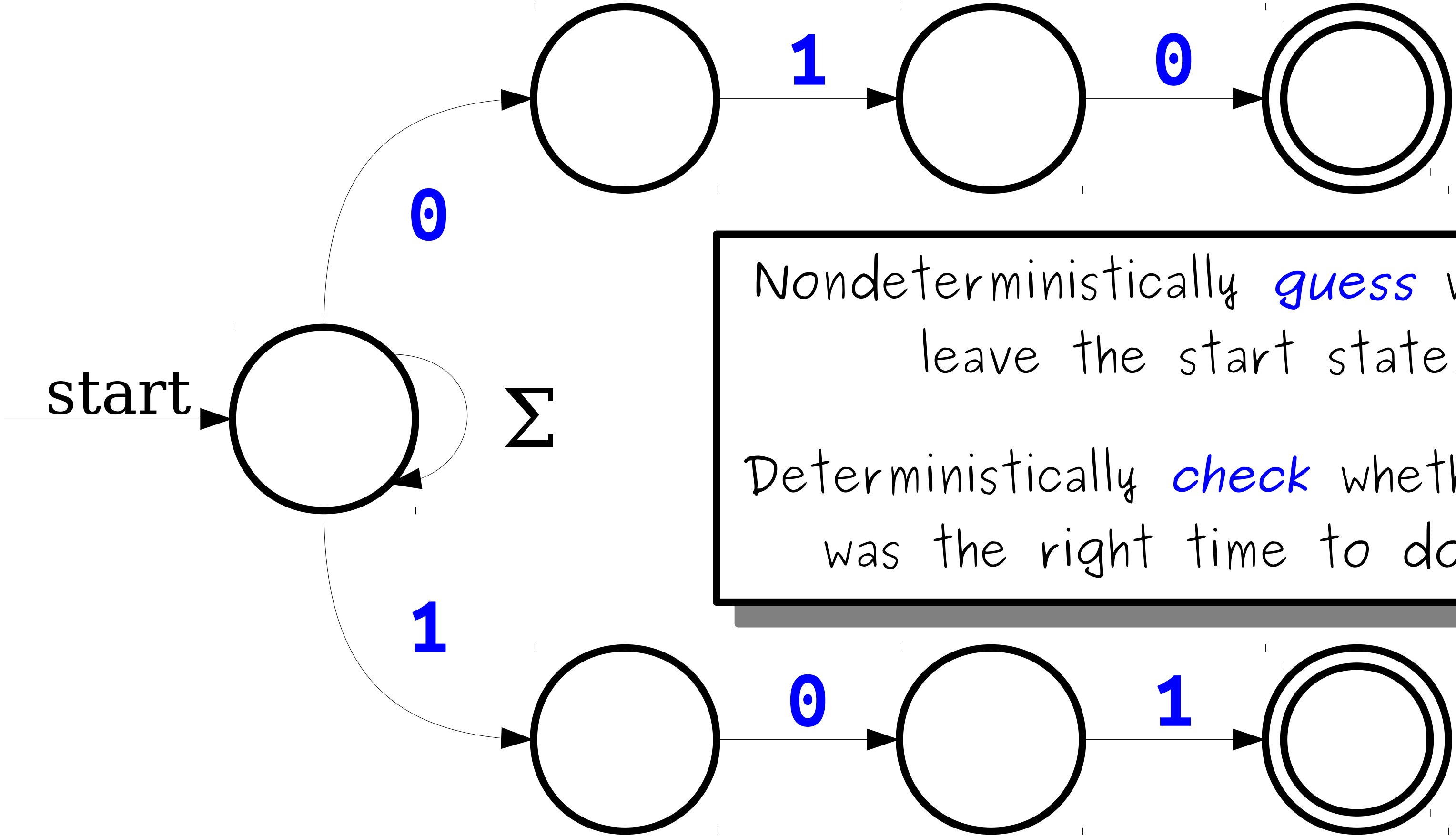
Guess-and-Check

$L = \{\ w \in \{\texttt{0}, \texttt{1}\}^* \mid w \text{ ends in } \texttt{010} \text{ or } \texttt{101}\ \}$



Guess-and-Check

$L = \{ w \in \{\mathtt{0}, \mathtt{1}\}^* \mid w \text{ ends in } \mathtt{010} \text{ or } \mathtt{101} \}$



Nondeterministically *guess* when to leave the start state.

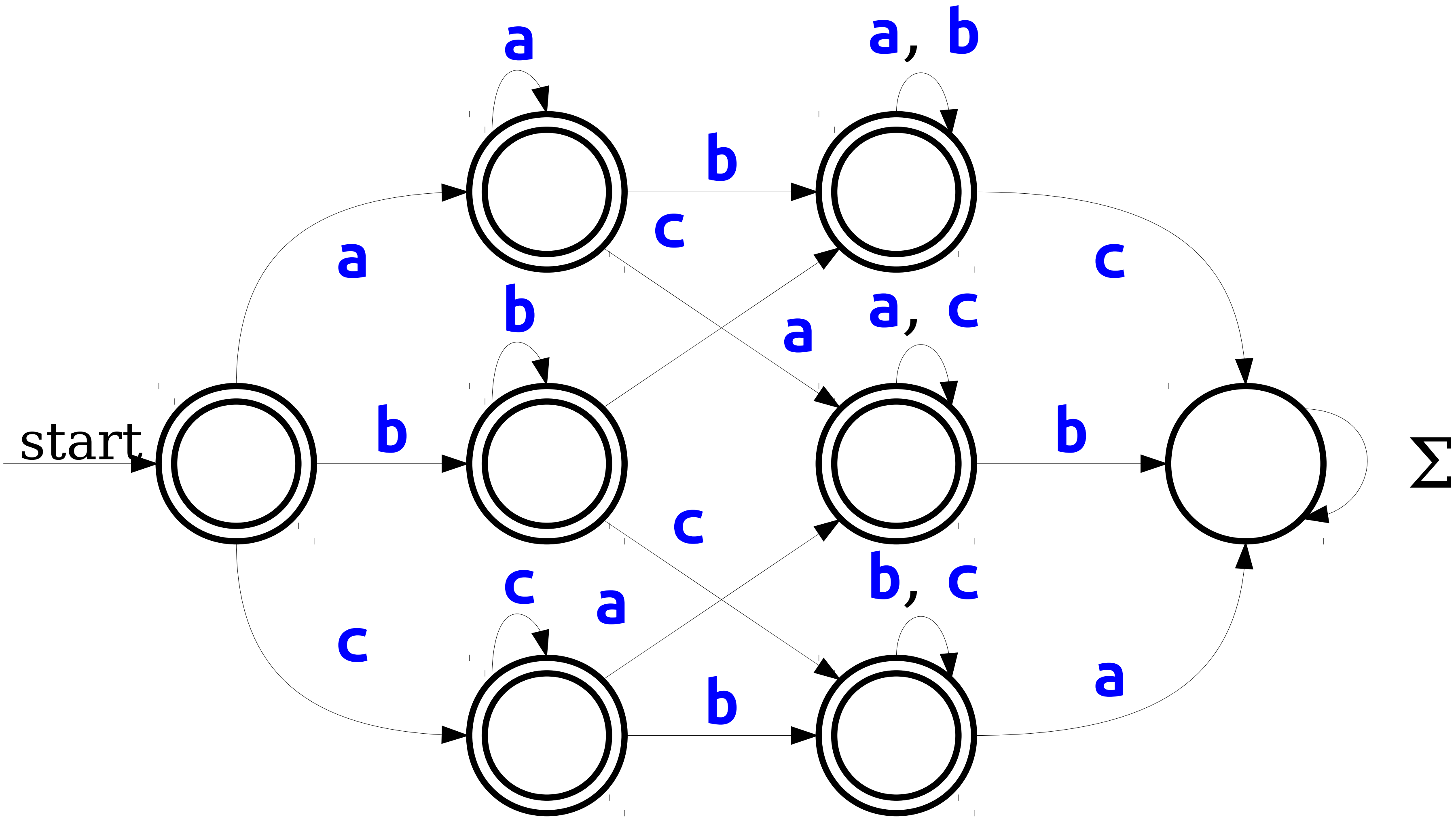
Deterministically *check* whether that was the right time to do so.

Guess-and-Check

$L = \{\ w \in \{\texttt{a}, \texttt{b}, \texttt{c}\}^* \mid \text{at least one of } \texttt{a}, \texttt{b}, \text{ or } \texttt{c} \text{ is not in } w\ \}$

Guess-and-Check

$L = \{\ w \in \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}^* \mid \text{at least one of } \mathtt{a}, \mathtt{b}, \text{ or } \mathtt{c} \text{ is not in } w\ \}$

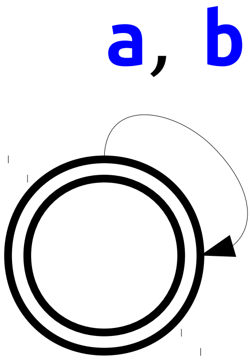


Guess-and-Check

$L = \{\ w \in \{$**a**, **b**, **c**$\}^* \mid$ at least one of **a**, **b**, or **c** is not in w $\}$

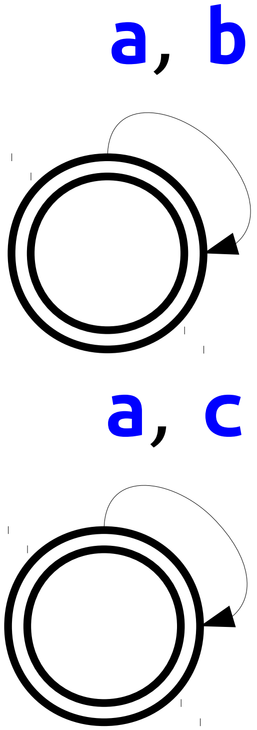
Guess-and-Check

$L = \{\ w \in \{\texttt{a}, \texttt{b}, \texttt{c}\}^* \mid \text{at least one of } \texttt{a}, \texttt{b}, \text{ or } \texttt{c} \text{ is not in } w\ \}$



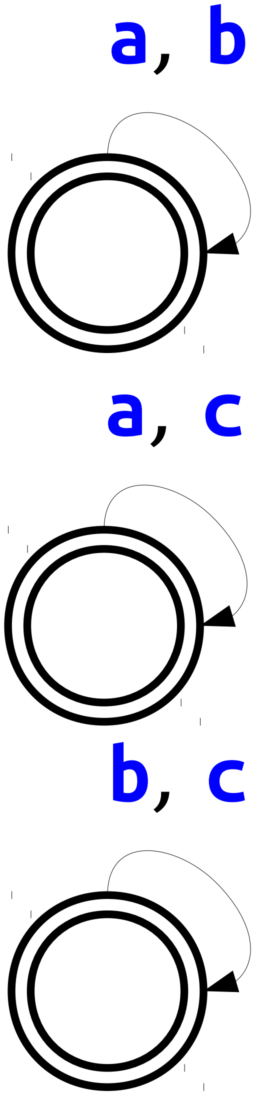
Guess-and-Check

$L = \{ w \in \{\texttt{a}, \texttt{b}, \texttt{c}\}^* \mid$ at least one of **a**, **b**, or **c** is not in w $\}$



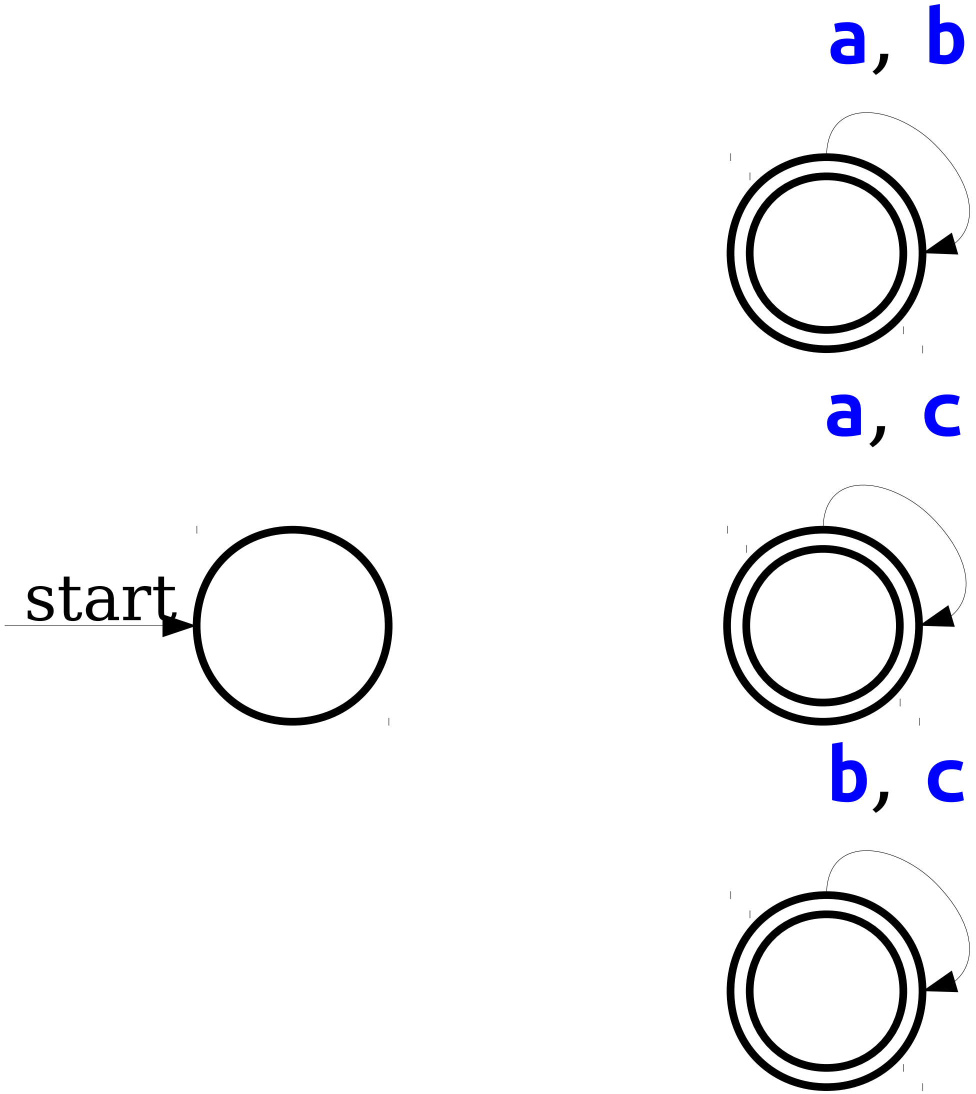
Guess-and-Check

$L = \{ w \in \{\texttt{a}, \texttt{b}, \texttt{c}\}^* \mid \text{at least one of } \texttt{a}, \texttt{b}, \text{ or } \texttt{c} \text{ is not in } w \}$



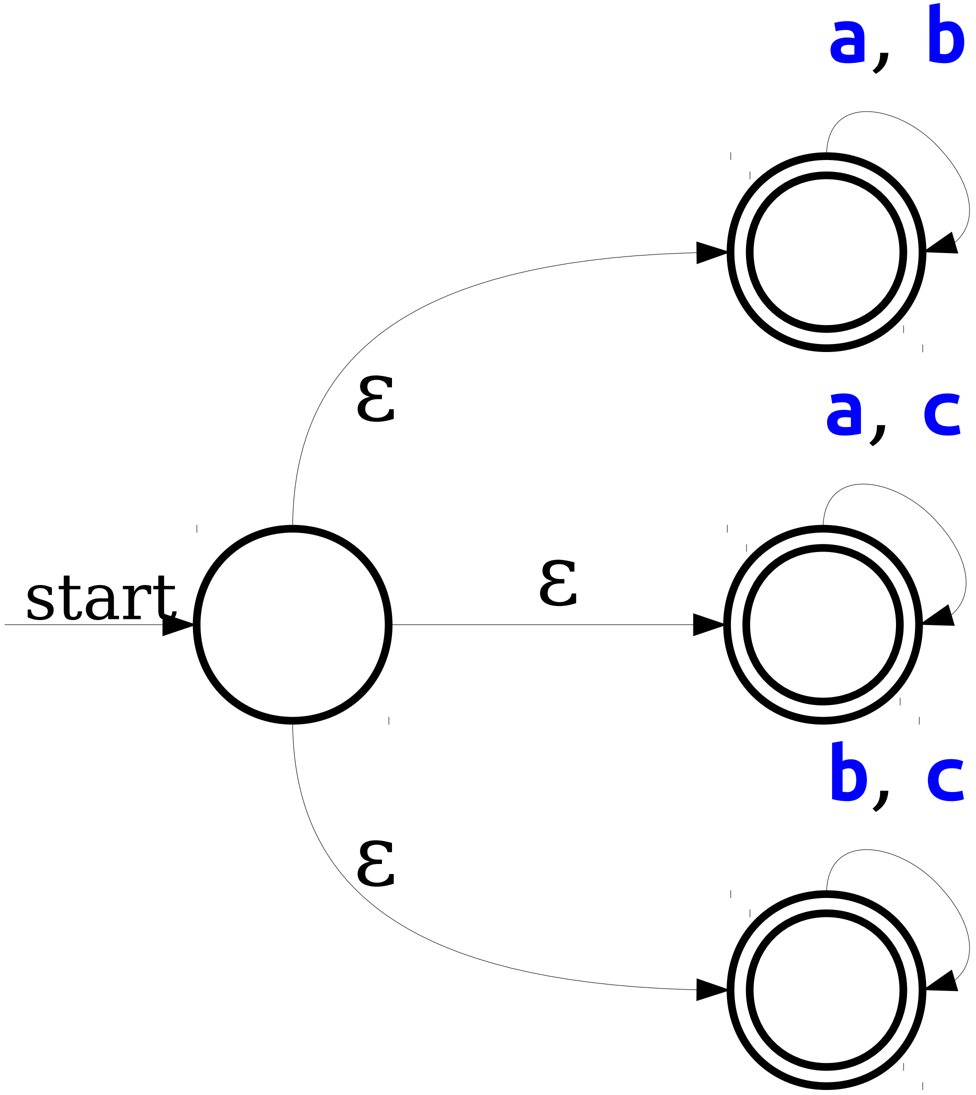
Guess-and-Check

$L = \{\ w \in \{\texttt{a}, \texttt{b}, \texttt{c}\}^* \mid \text{at least one of } \texttt{a}, \texttt{b}, \text{ or } \texttt{c} \text{ is not in } w\ \}$



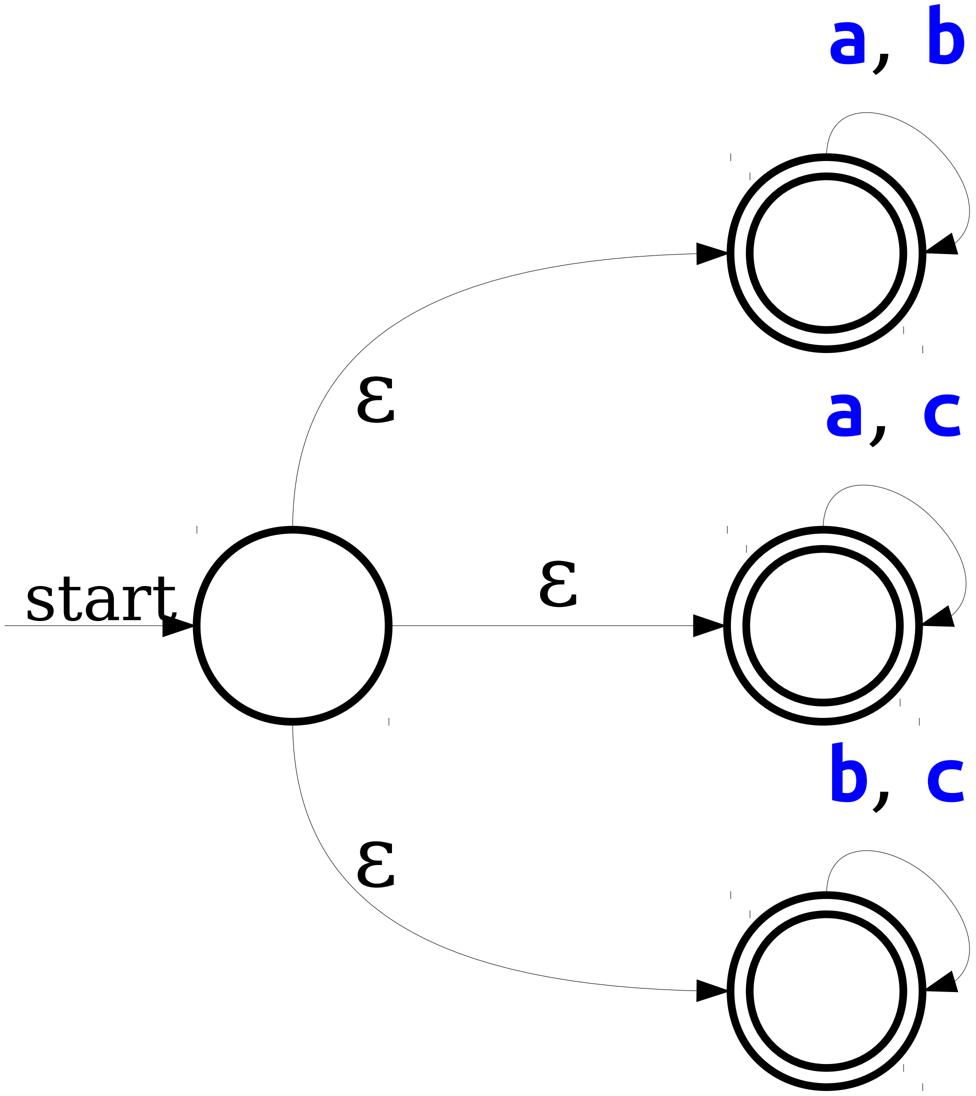
Guess-and-Check

$L = \{ w \in \{\texttt{a}, \texttt{b}, \texttt{c}\}^* \mid$ at least one of **a**, **b**, or **c** is not in w $\}$



Guess-and-Check

$L = \{\ w \in \{$**a**, **b**, **c**$\}^* \mid$ at least one of **a**, **b**, or **c** is not in $w\ \}$



Nondeterministically *guess* which character is missing.

Deterministically *check* whether that character is indeed missing.

Just how powerful are NFAs?