Nonregular Languages

The Class of Regular Languages

Theorem: The following are all equivalent:

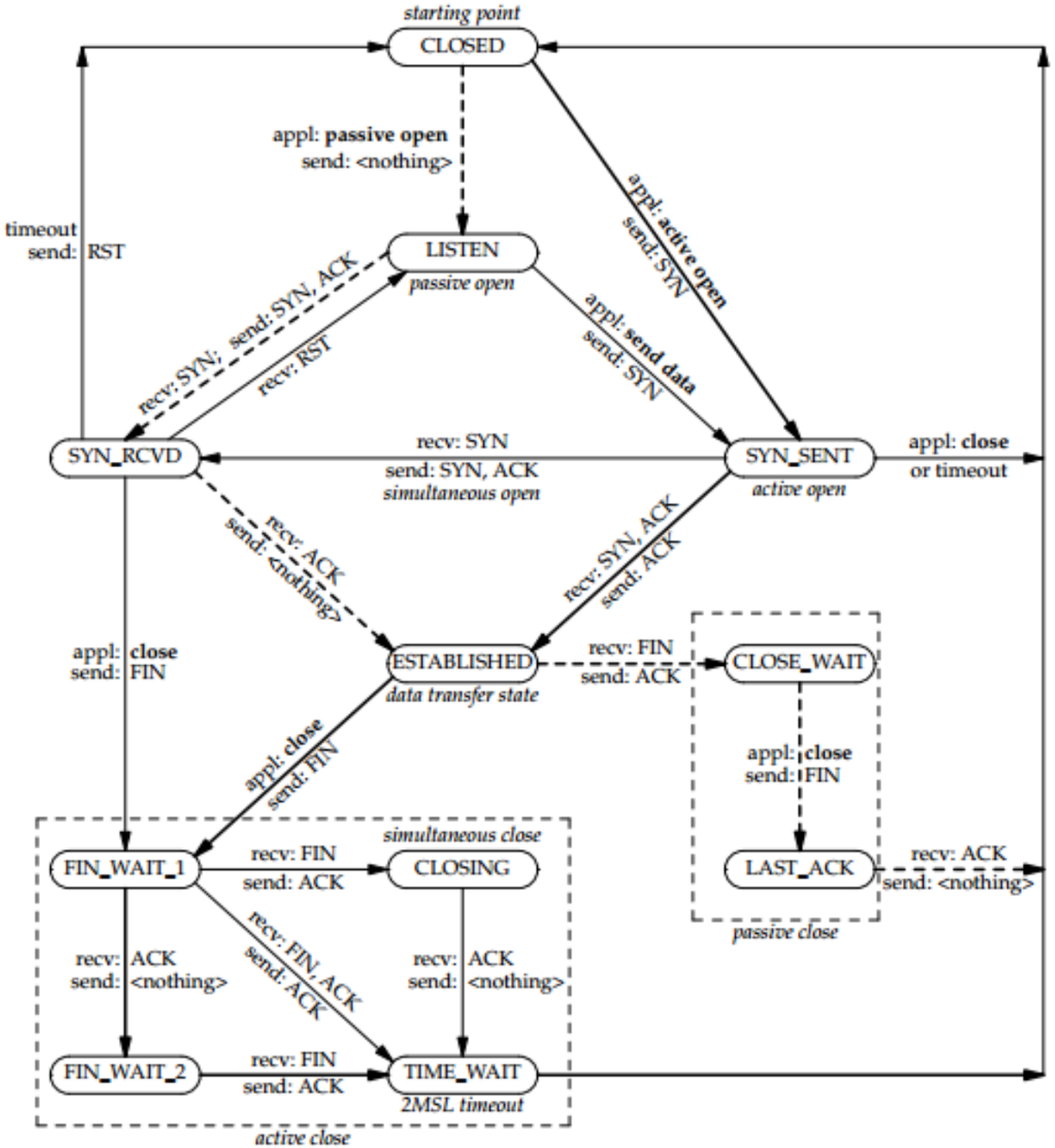
- L is a regular language.
- There is a **DFA** D such that $\mathscr{L}(D) = L$.
- There is an **NFA** N such that $\mathscr{L}(N) = L$.
- There is a **regular expression** R such that $\mathscr{L}(R) = L$.

Why does this matter?

Buttons as Finite-State Machines:

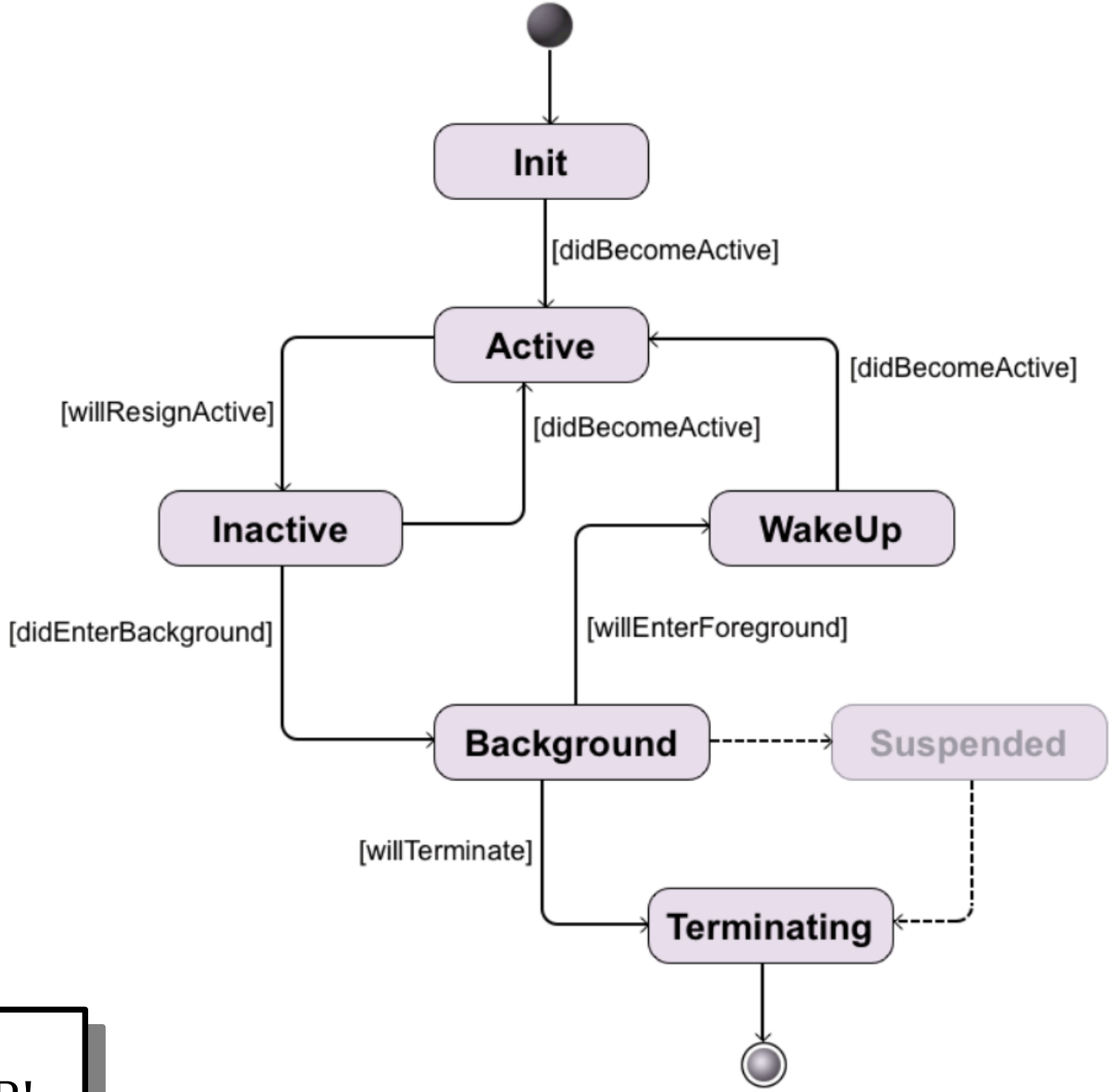
http://cs103.stanford.edu/tools/button-fsm/

Take CS148!



Take CS144!

http://www.tti.unipa.it/~gneglia/ip_networks06/slides/TCPIP_State_Transition_Diagram.pdf



Take CS193P!

https://blog.codecentric.de/en/2016/07/handling-ios-app-states-state-machine/

Computers as Finite Automata

- My computer has 12GB of RAM and about 150GB of hard disk space.
- That's a total of 162GB of memory, which is 1,391,569,403,904 bits.
- There are “only” $2^{1,391,569,403,904}$ possible configurations of the memory in my computer.
- You could in principle build a DFA representing my computer, where there's one symbol per type of input the computer can receive.

A Powerful Intuition

- ***Regular languages correspond to problems that can be solved with finite memory.***
  - At each point in time, we only need to store one of finitely many pieces of information.
- Nonregular languages, in a sense, correspond to problems that cannot be solved with finite memory.
- Since every computer ever built has finite memory, in a sense, nonregular languages correspond to problems that cannot be solved by physical computers!

Finding Nonregular Languages

Finding Nonregular Languages

- To prove that a language **is regular**, we can just find a DFA, NFA, or regex for it.
- To prove that a language **is not regular**, we need to prove that there are **no possible** DFAs, NFAs, or regexes for it.
  - ***Claim:*** We can actually just prove that there's no DFA for it. Why is this?
- ***This sort of argument will be challenging!***

A Simple Language

- Let $\Sigma = \{\mathbf{a}, \mathbf{b}\}$ and consider the following language:

$$E = \{\mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N}\}$$

- E is the language of all strings of n **a**'s followed by n **b**'s:

{ ε, **ab**, **aabb**, **aaabbb**, **aaaabbbb**, … }

A Simple Language

$$E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$$

How many of the following are regular expressions for the language E defined above?

a*b*

(ab)*

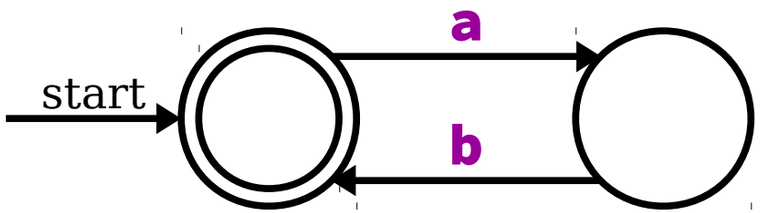
ε ∪ ab ∪ a²b² ∪ a³b³

Another Attempt

- Let’s try to design an NFA for

$$E = \{ \mathbf{a}^n \mathbf{b}^n \mid n \in \mathbb{N} \}.$$

- Does this machine work?

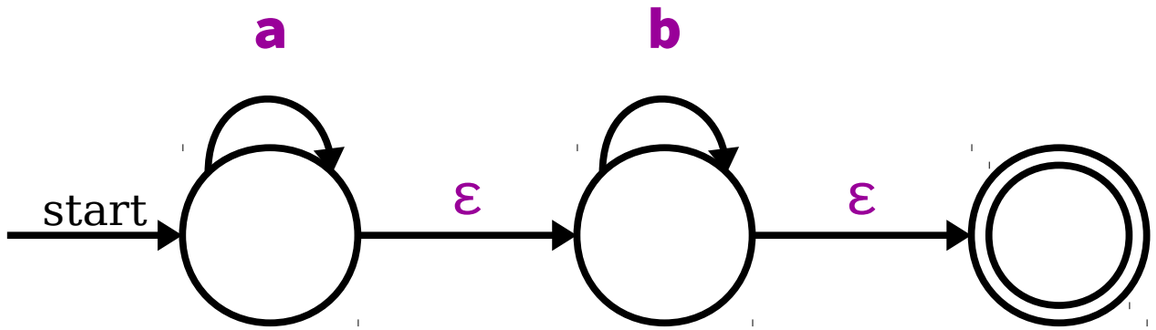


Another Attempt

- Let’s try to design an NFA for

$$E = \{ \mathbf{a}^n \mathbf{b}^n \mid n \in \mathbb{N} \}.$$

- How about this one?

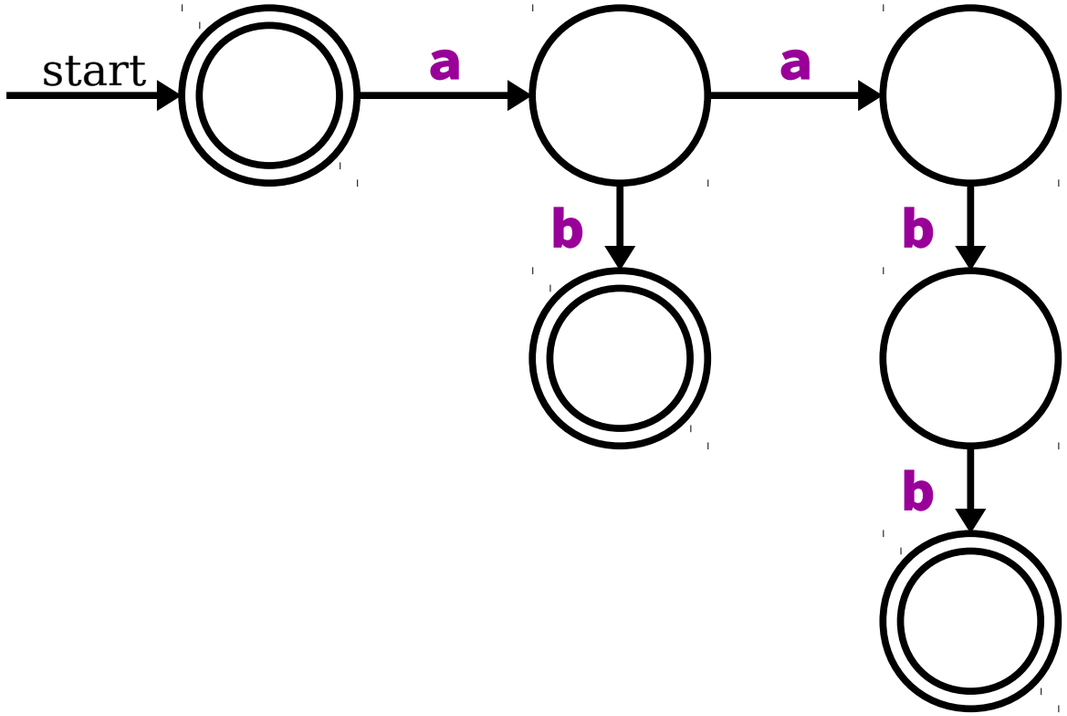


Another Attempt

- Let’s try to design an NFA for

$$E = \{ \mathbf{a}^n \mathbf{b}^n \mid n \in \mathbb{N} \}.$$

- What about this?

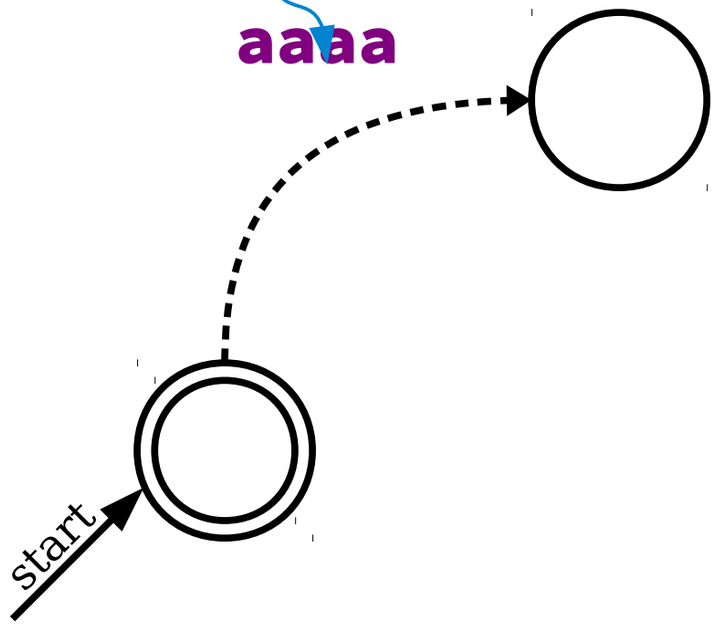


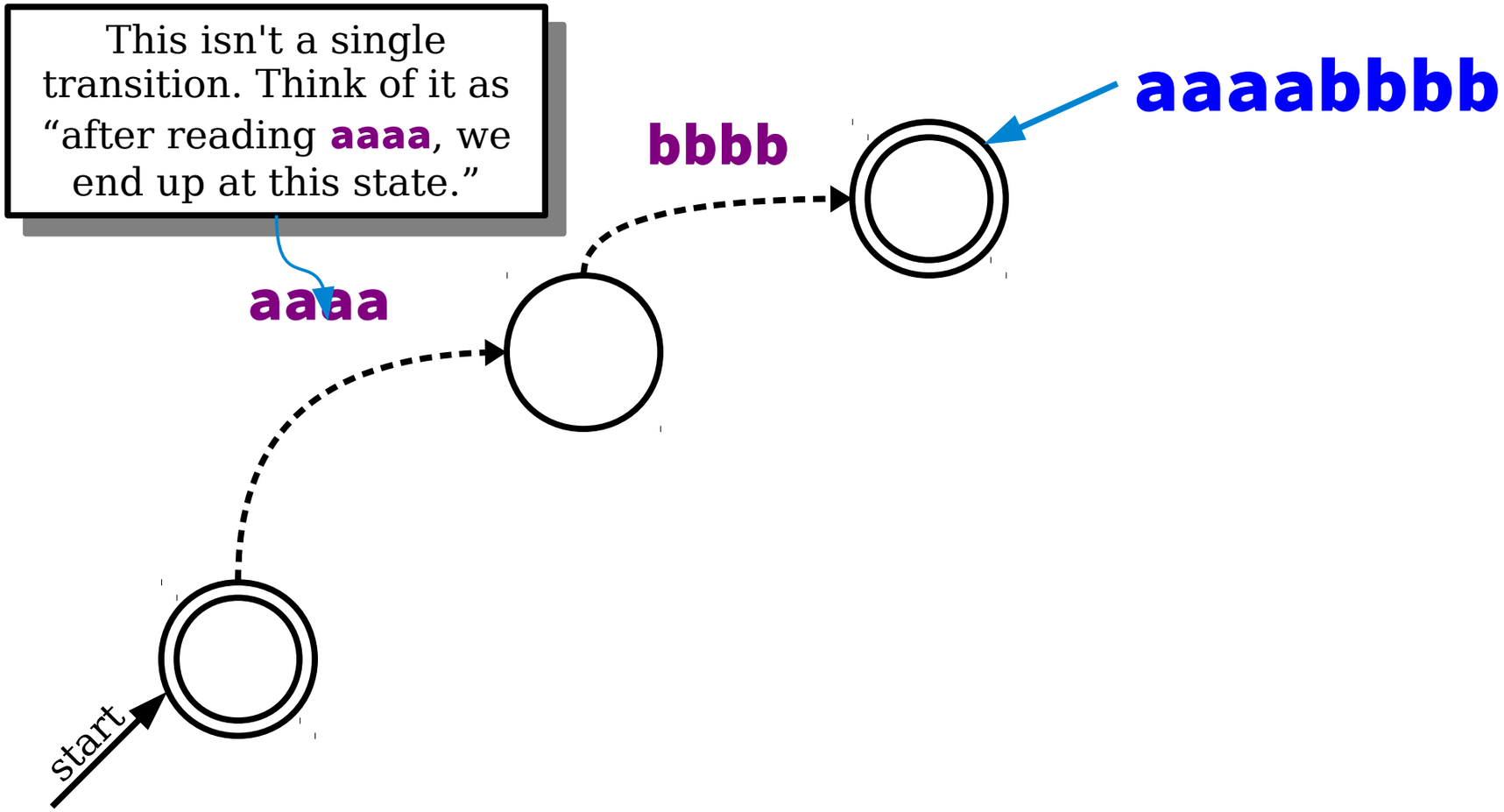
We seem to be running into some trouble.
Why is that?

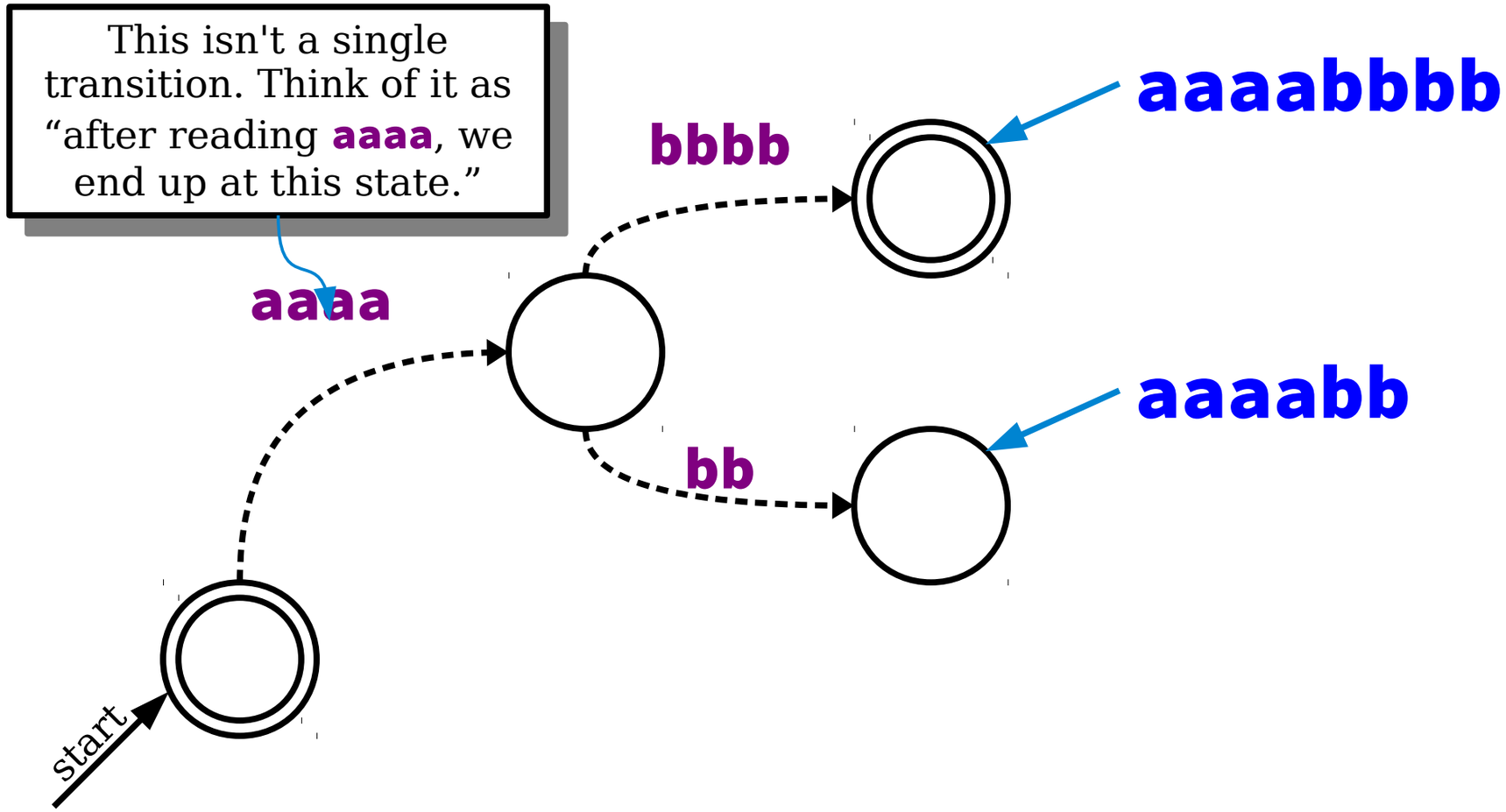
Let's imagine what a DFA for the language $\{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ would have to look like.

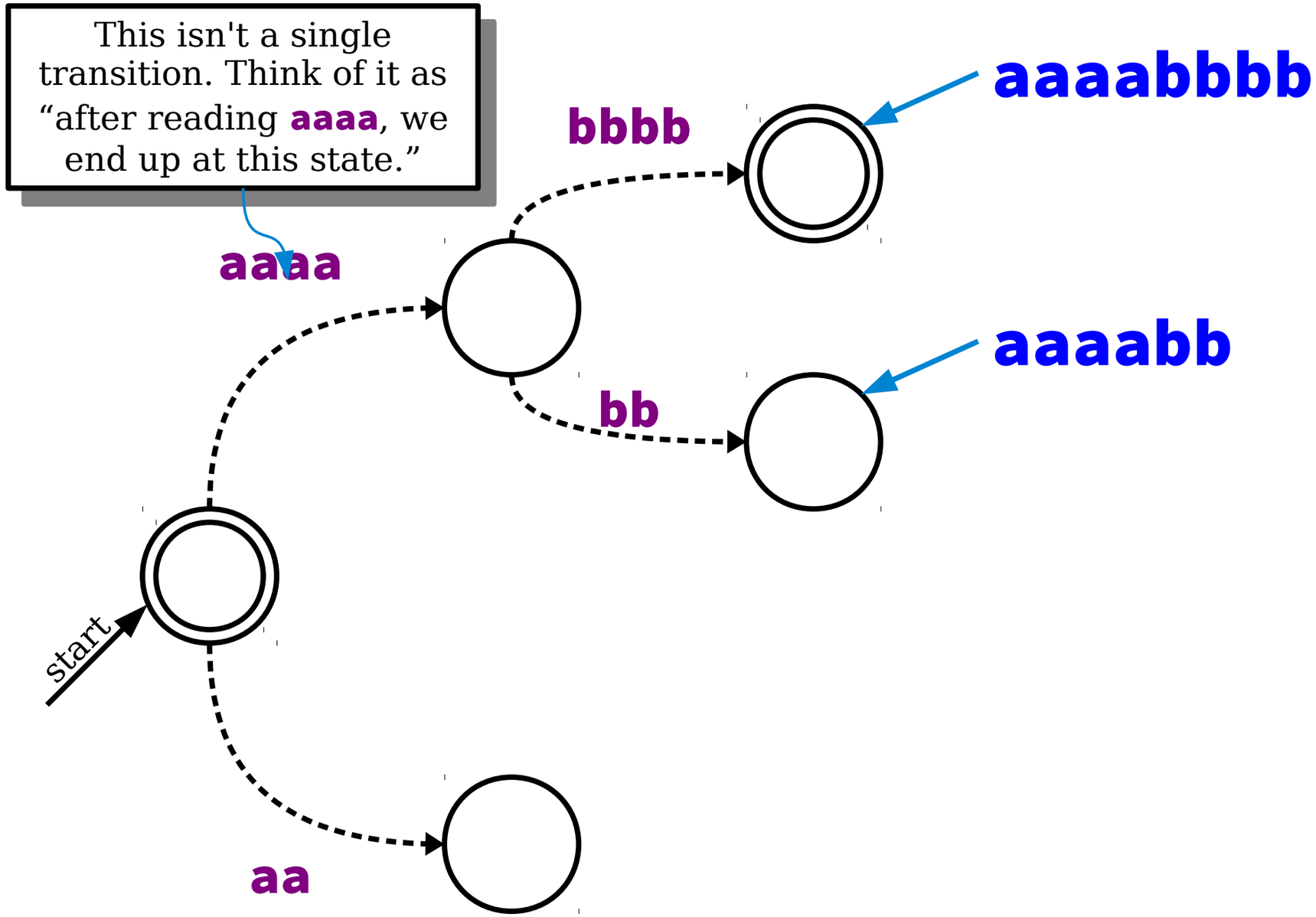
Can we say anything about it?

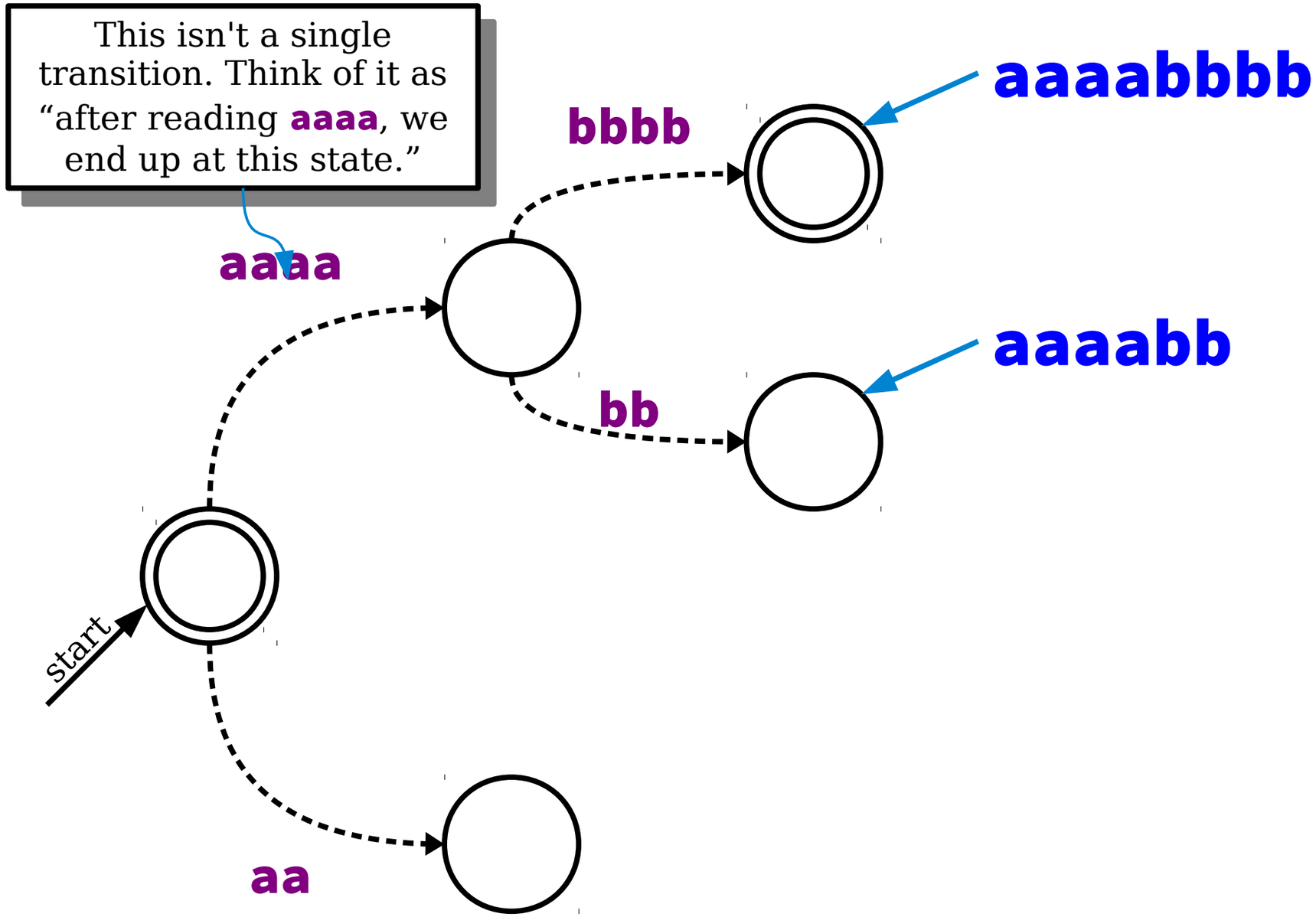
This isn't a single transition. Think of it as “after reading **aaaa**, we end up at this state.”

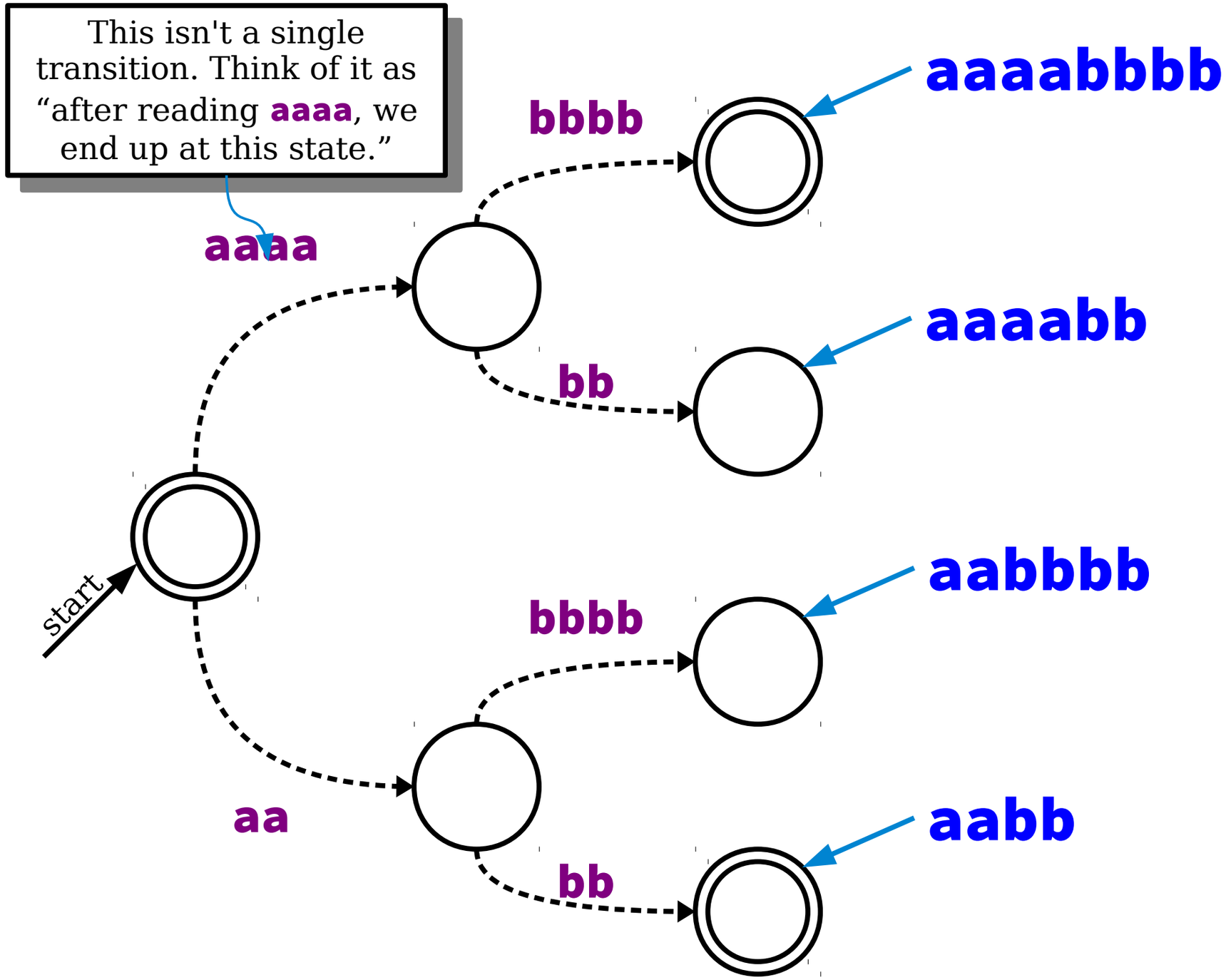


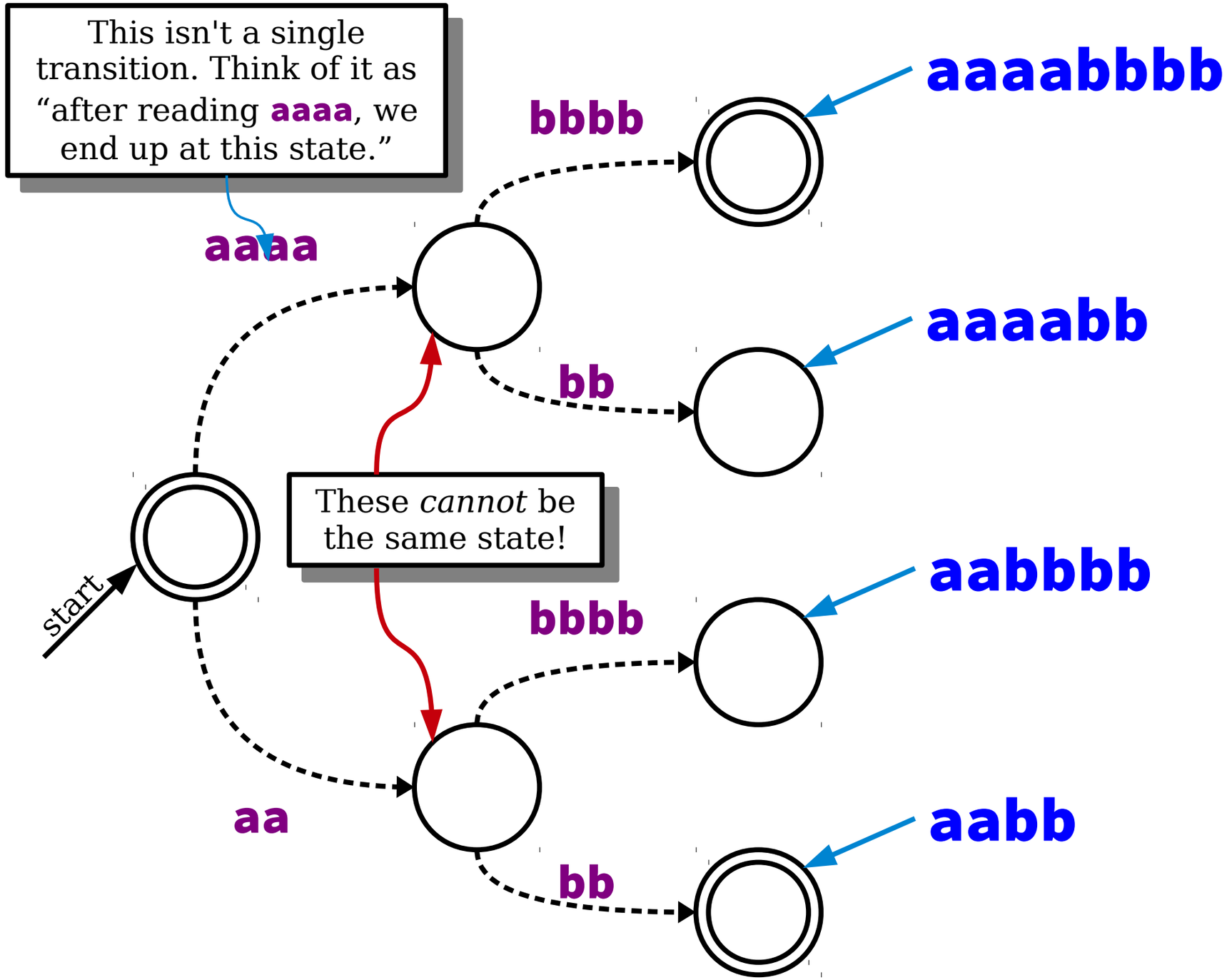

This isn't a single transition. Think of it as "after reading aaaa, we end up at this state."
aaaa
bbbb
aaaabbbb
start


This isn't a single transition. Think of it as "after reading aaaa, we end up at this state."
aaaa
bbbb
bb
aaaabbbb
aaaabb
start

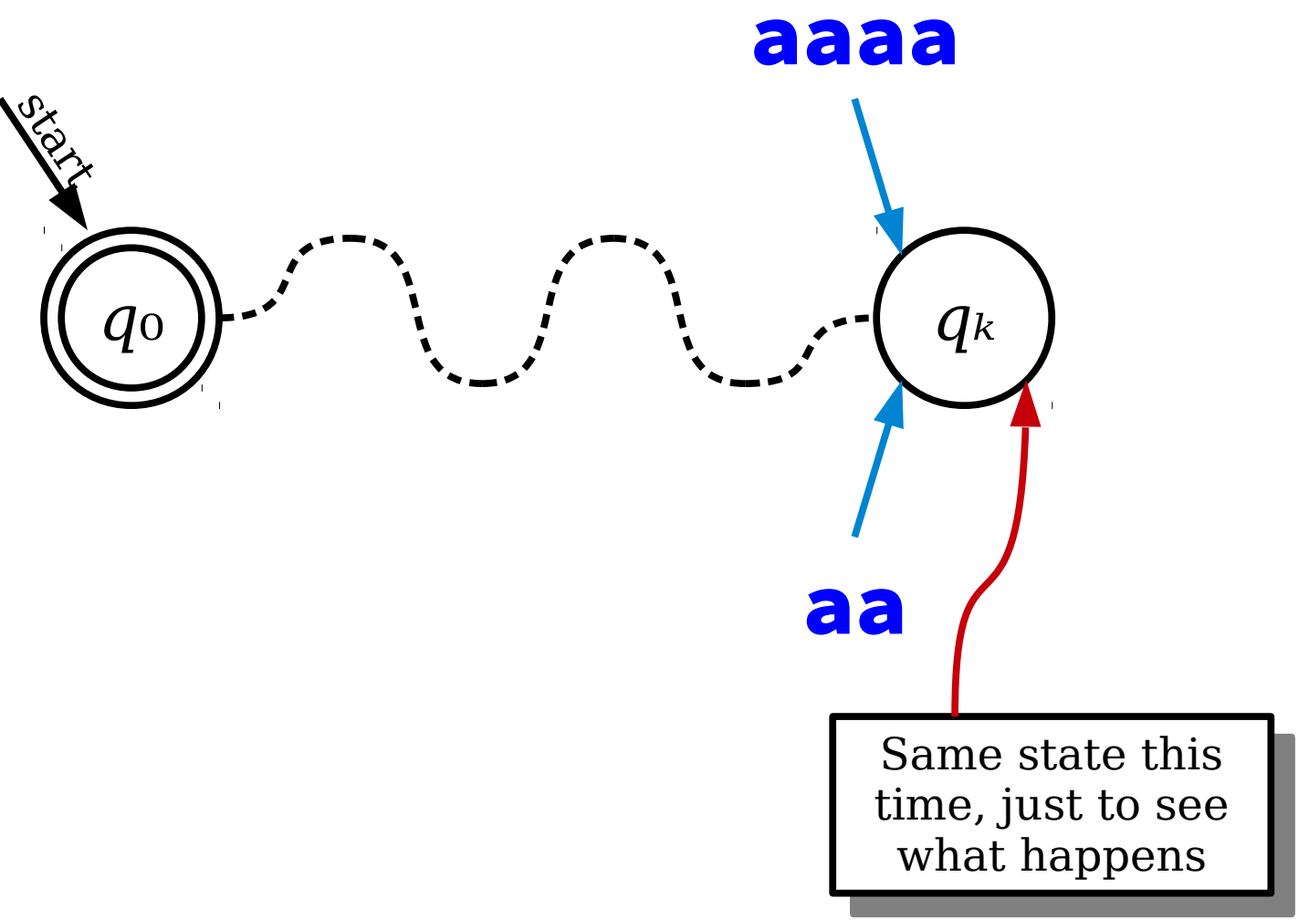

This isn't a single transition. Think of it as "after reading aaaa, we end up at this state."
aaaa
bbbb
aaaabbbb
bb
aaaabb
start
aa


This isn't a single transition. Think of it as "after reading aaaa, we end up at this state."
aaaa
bbbb
aaaabbbb
bb
aaaabb
start
aa

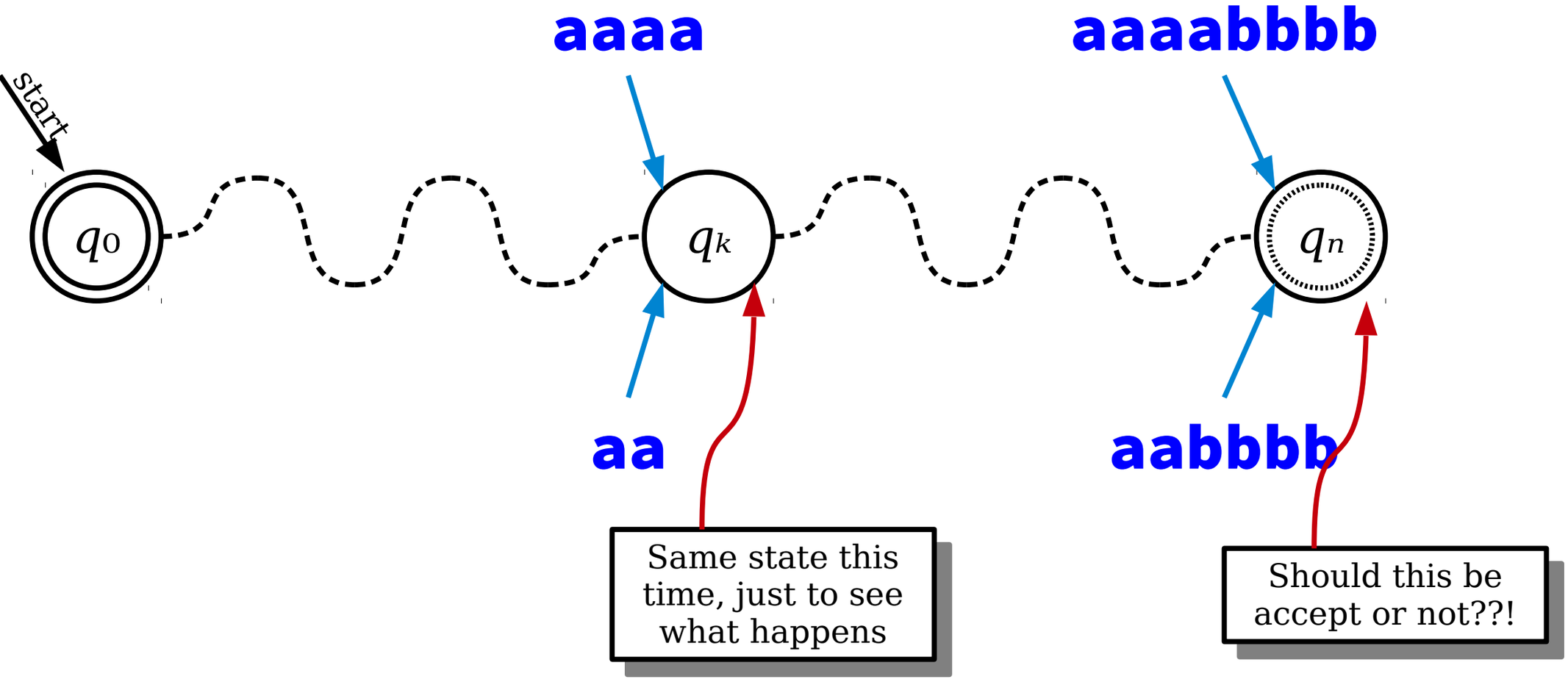

This isn't a single transition. Think of it as "after reading aaaa, we end up at this state."
aaaa
bbbb
aaaabbbb
bb
aaaabb
start
aa
bbbb
aabbbb
bb
aabb


This isn't a single transition. Think of it as "after reading aaaa, we end up at this state."
aaaa
bbbb
aaaabbbb
aaaabb
bb
These *cannot* be the same state!
start
aabbbb
bbbb
aa
aabb
bb

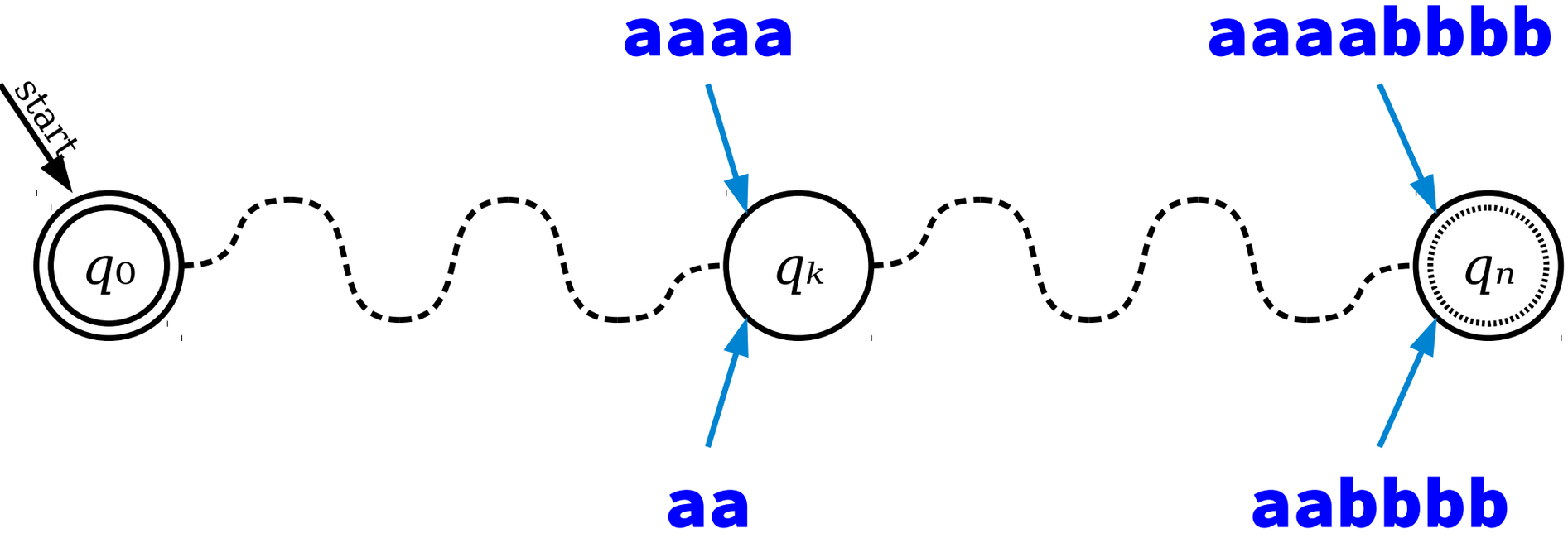
Let’s say they *were* the same state, then what?



Let’s say they *were* the same state, then what?



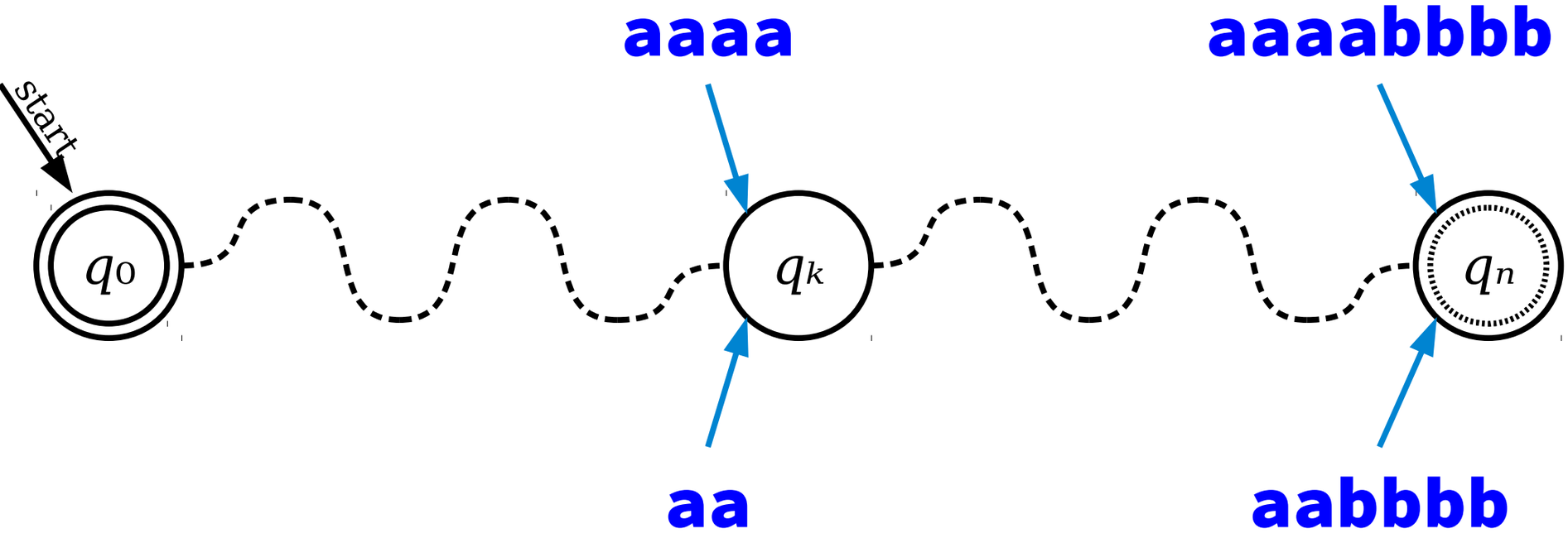
A Different Perspective



What happens if q_n is...

...an accepting state?

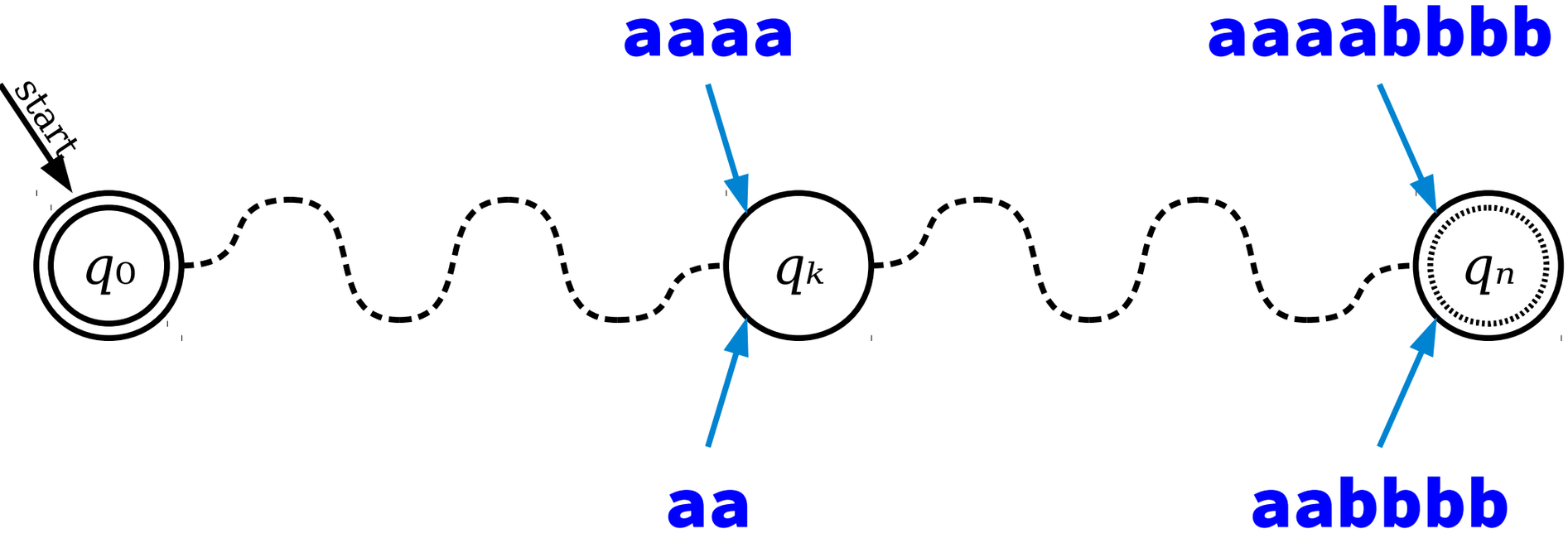
A Different Perspective



What happens if q_n is...

...an accepting state? We accept **aabbbb** $\notin E$!

A Different Perspective

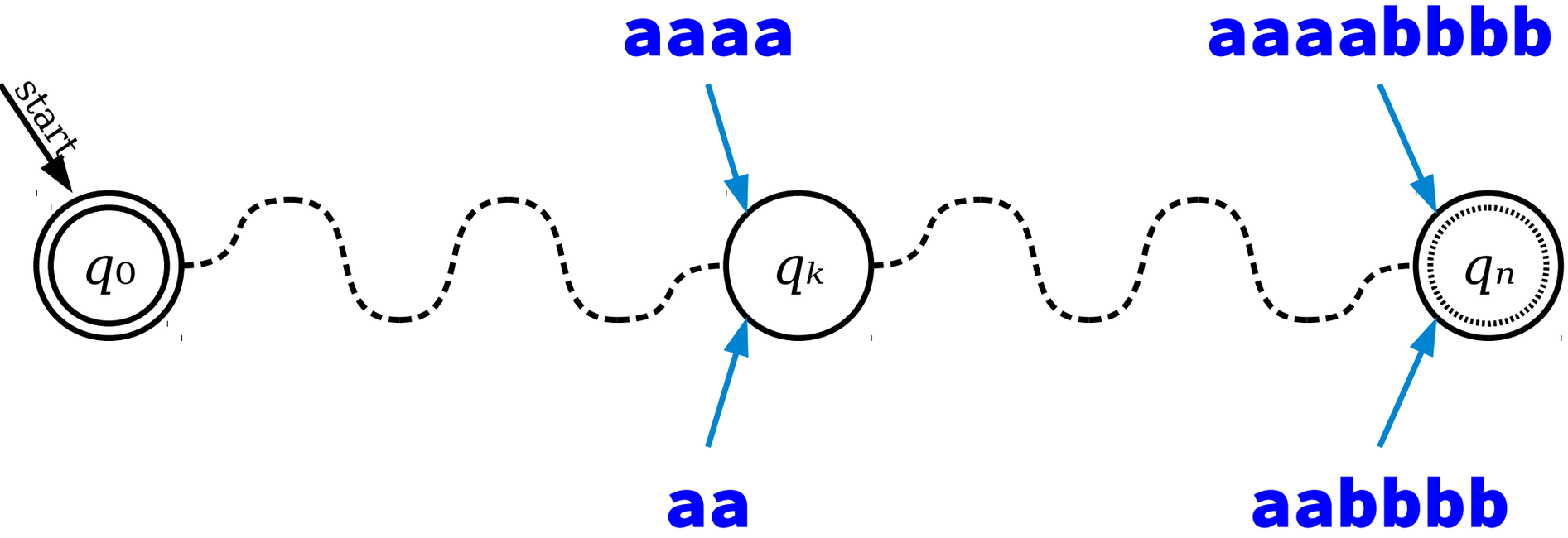


What happens if q_n is...

...an accepting state? We accept **aabbbb** $\notin E$!

...a rejecting state?

A Different Perspective



What happens if q_n is...

...an accepting state? We accept **aabbbb** $\notin E$!

...a rejecting state? We reject **aaaabbbb** $\in E$!

What's Going On?

- As you just saw, the strings $\mathbf{a}^4$ and $\mathbf{a}^2$ can't end up in the same state in *any* DFA for $E = \{\mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N}\}$.
- Two proof routes:
  - *Direct:* The states you reach for $\mathbf{a}^4$ and $\mathbf{a}^2$ have to behave differently when reading $\mathbf{b}^4$ – in one case it should lead to an accept state, in the other it should lead to a reject state. Therefore, they must be different states.
  - *Contradiction:* Suppose you do end up in the same state. Then $\mathbf{a}^4\mathbf{b}^4$ and $\mathbf{a}^2\mathbf{b}^4$ end up in the same state, so we either reject $\mathbf{a}^4\mathbf{b}^4$ (oops) or accept $\mathbf{a}^2\mathbf{b}^4$ (oops).
- ***Powerful intuition:*** Any DFA for E must keep $\mathbf{a}^4$ and $\mathbf{a}^2$ separated. It needs to remember something fundamentally different after reading those strings.

This idea – that two strings shouldn't end up in the same DFA state – is fundamental to discovering nonregular languages.

Let's go formalize this!

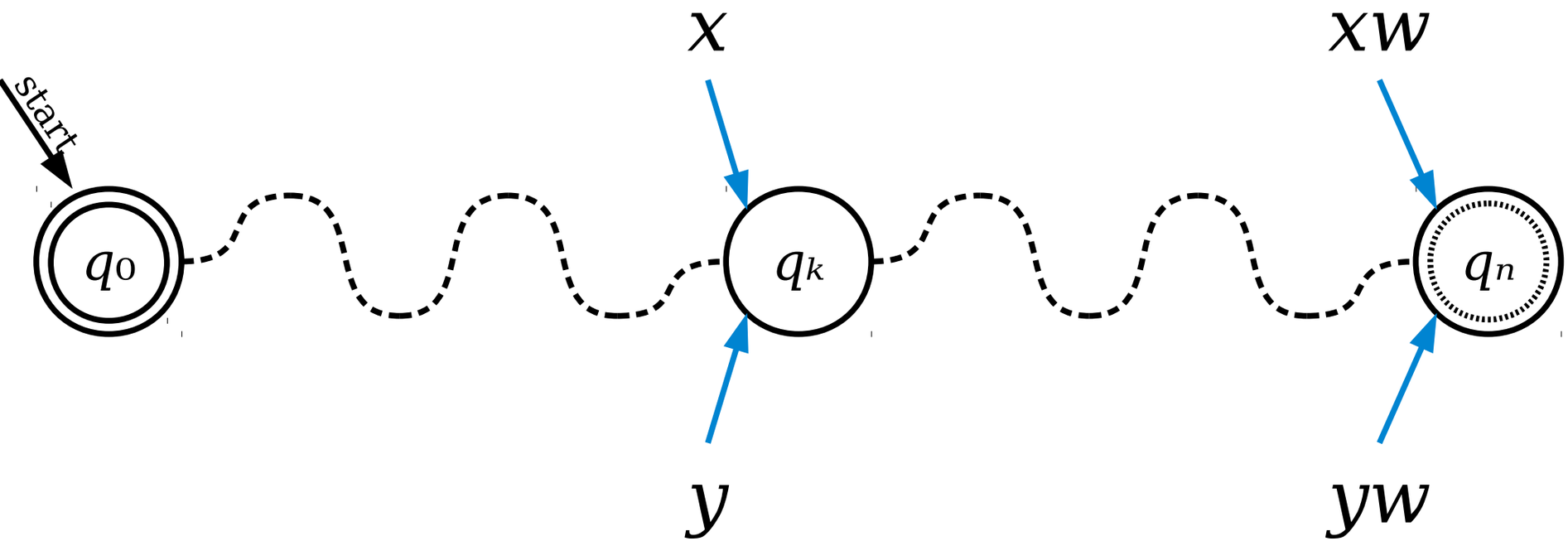
Distinguishability

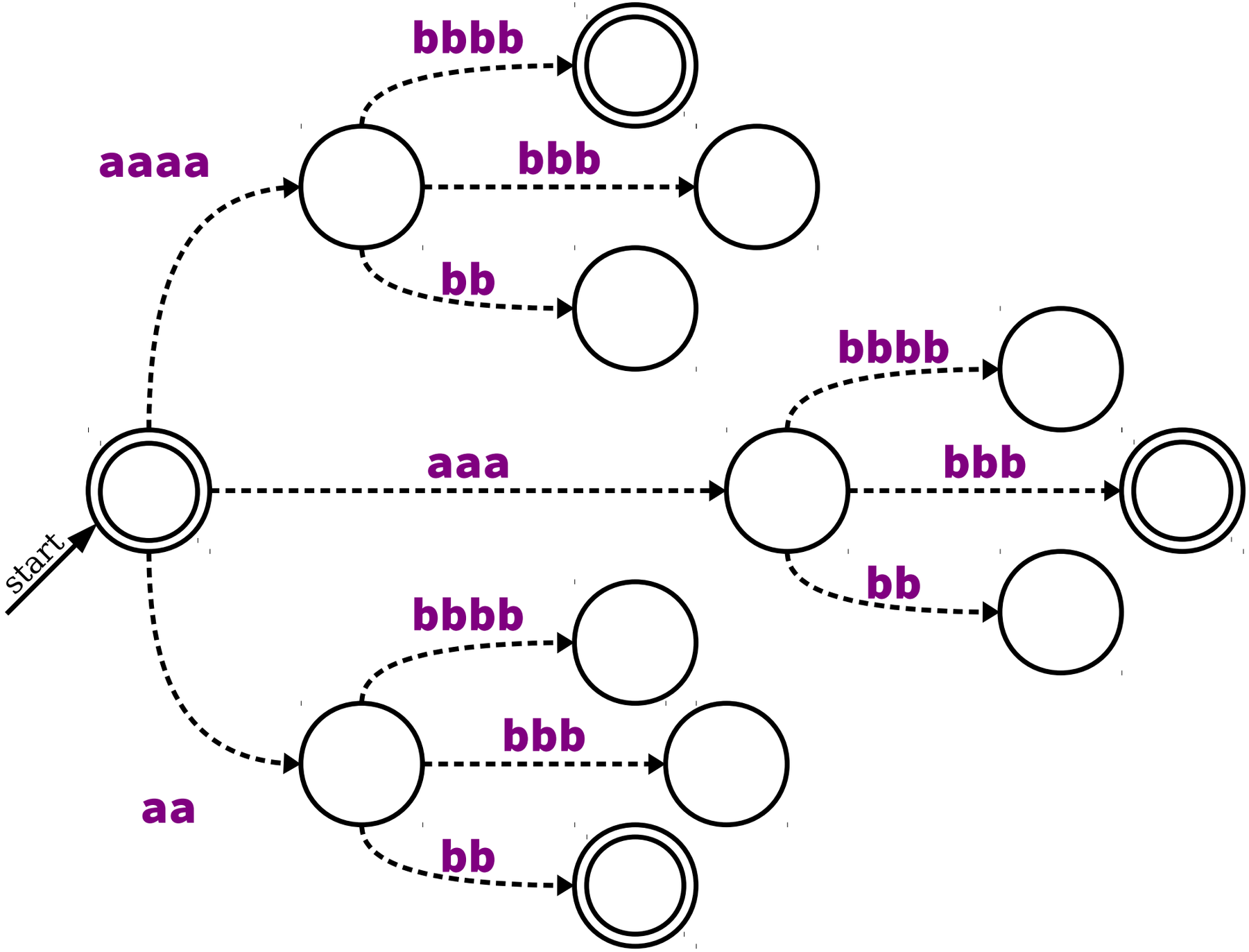
- Let L be an arbitrary language over Σ.
- Two strings $x \in \Sigma^*$ and $y \in \Sigma^*$ are called ***distinguishable relative to L*** if there is a string $w \in \Sigma^*$ such that exactly one of xw and yw is in L.
- We denote this by writing $\boldsymbol{x \not\equiv_L y}$.
- In our previous example, we saw that $\mathbf{a}^2 \not\equiv_E \mathbf{a}^4$.
  - Try appending $\mathbf{b}^4$ to both of them.
- Formally, we say that $x \not\equiv_L y$ if the following is true:

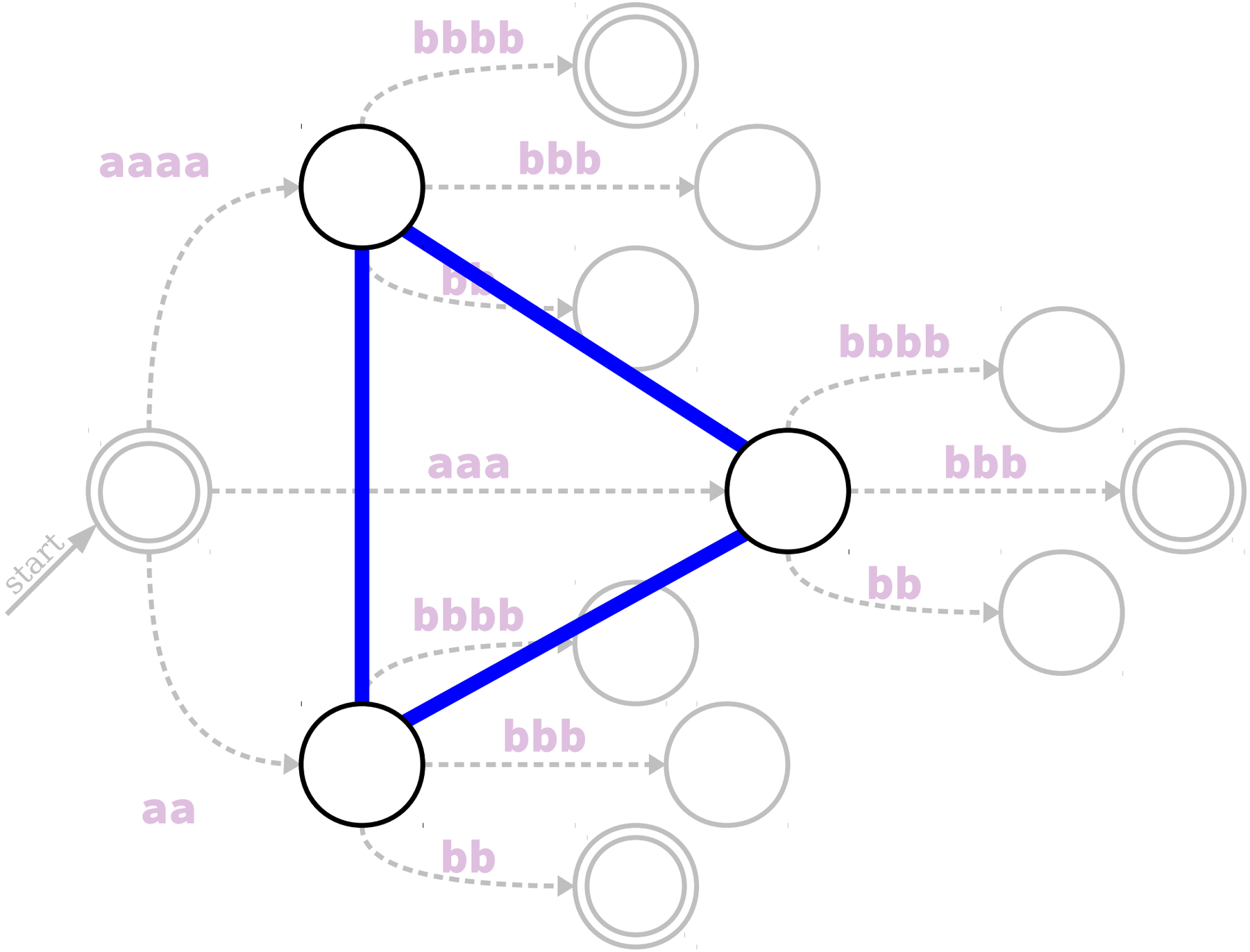
$$\boldsymbol{\exists w \in \Sigma^*.\ (xw \in L \leftrightarrow yw \notin L)}$$

Distinguishability

- ***Theorem:*** Let L be an arbitrary language over Σ. Let $x \in \Sigma^*$ and $y \in \Sigma^*$ be strings where $x \not\equiv_L y$. Then if D is ***any*** DFA for L, then D must end in different states when run on inputs x and y.
- ***Proof sketch:***




bbbb
aaaa
bbb
bb
bbbb
aaa
bbb
bb
start
bbbb
bbb
aa
bb


bbbb
aaaa
bbb
bb
bbbb
aaa
bbb
start
bb
bbbb
bbb
aa
bb

Distinguishability

- Let's focus on this language for now:

$$E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$$

Lemma: If m, $n \in \mathbb{N}$ and $m \neq n$, then $\mathbf{a}^m \not\equiv_E \mathbf{a}^n$.

Proof: Let $\mathbf{a}^m$ and $\mathbf{a}^n$ be strings where $m \neq n$. Then $\mathbf{a}^m\mathbf{b}^m \in E$ and $\mathbf{a}^n\mathbf{b}^m \notin E$. Therefore, we see that $\mathbf{a}^m \not\equiv_E \mathbf{a}^n$, as required. ■

A Bad Combination

- Suppose there is a DFA D for the language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$.
- We know the following:
  - Any two strings of the form $\mathbf{a}^m$ and $\mathbf{a}^n$, where $m \neq n$, cannot end in the same state when run through D.
  - There are infinitely many strings of the form $\mathbf{a}^m$.
  - However, there are only *finitely many* states they can end up in, since D is a deterministic ***finite*** automaton!
- What happens if we put these pieces together?

Theorem: The language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ is not regular.

Theorem: The language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: Suppose for the sake of contradiction that E is regular.

Theorem: The language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: Suppose for the sake of contradiction that E is regular.

Let D be a DFA for E, and let k be the number of states in D.

Theorem: The language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: Suppose for the sake of contradiction that E is regular.

Let D be a DFA for E, and let k be the number of states in D. Consider the strings $\mathbf{a}^0$, $\mathbf{a}^1$, $\mathbf{a}^2$, …, $\mathbf{a}^k$.

Theorem: The language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: Suppose for the sake of contradiction that E is regular. Let D be a DFA for E, and let k be the number of states in D. Consider the strings $\mathbf{a}^0$, $\mathbf{a}^1$, $\mathbf{a}^2$, …, $\mathbf{a}^k$. This is a collection of $k+1$ strings and there are only k states in D.

Theorem: The language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: Suppose for the sake of contradiction that E is regular. Let D be a DFA for E, and let k be the number of states in D. Consider the strings $\mathbf{a}^0$, $\mathbf{a}^1$, $\mathbf{a}^2$, …, $\mathbf{a}^k$. This is a collection of $k+1$ strings and there are only k states in D. Therefore, by the pigeonhole principle, there must be two distinct strings $\mathbf{a}^m$ and $\mathbf{a}^n$ that end in the same state when run through D.

Theorem: The language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: Suppose for the sake of contradiction that E is regular. Let D be a DFA for E, and let k be the number of states in D. Consider the strings $\mathbf{a}^0$, $\mathbf{a}^1$, $\mathbf{a}^2$, …, $\mathbf{a}^k$. This is a collection of $k+1$ strings and there are only k states in D. Therefore, by the pigeonhole principle, there must be two distinct strings $\mathbf{a}^m$ and $\mathbf{a}^n$ that end in the same state when run through D.

Our lemma tells us that $\mathbf{a}^m \not\equiv_E \mathbf{a}^n$, so by our earlier theorem we know that $\mathbf{a}^m$ and $\mathbf{a}^n$ cannot end in the same state when run through D.

Theorem: The language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: Suppose for the sake of contradiction that E is regular. Let D be a DFA for E, and let k be the number of states in D. Consider the strings $\mathbf{a}^0$, $\mathbf{a}^1$, $\mathbf{a}^2$, …, $\mathbf{a}^k$. This is a collection of $k+1$ strings and there are only k states in D. Therefore, by the pigeonhole principle, there must be two distinct strings $\mathbf{a}^m$ and $\mathbf{a}^n$ that end in the same state when run through D.

Our lemma tells us that $\mathbf{a}^m \not\equiv_E \mathbf{a}^n$, so by our earlier theorem we know that $\mathbf{a}^m$ and $\mathbf{a}^n$ cannot end in the same state when run through D. But this is impossible, since we know that $\mathbf{a}^m$ and $\mathbf{a}^n$ do end in the same state when run through D.

We have reached a contradiction, so our assumption must have been wrong.

Theorem: The language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: Suppose for the sake of contradiction that E is regular. Let D be a DFA for E, and let k be the number of states in D. Consider the strings $\mathbf{a}^0$, $\mathbf{a}^1$, $\mathbf{a}^2$, …, $\mathbf{a}^k$. This is a collection of $k+1$ strings and there are only k states in D. Therefore, by the pigeonhole principle, there must be two distinct strings $\mathbf{a}^m$ and $\mathbf{a}^n$ that end in the same state when run through D.

Our lemma tells us that $\mathbf{a}^m \not\equiv_E \mathbf{a}^n$, so by our earlier theorem we know that $\mathbf{a}^m$ and $\mathbf{a}^n$ cannot end in the same state when run through D. But this is impossible, since we know that $\mathbf{a}^m$ and $\mathbf{a}^n$ do end in the same state when run through D.

We have reached a contradiction, so our assumption must have been wrong. Therefore, E is not regular.

Theorem: The language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: Suppose for the sake of contradiction that E is regular. Let D be a DFA for E, and let k be the number of states in D. Consider the strings $\mathbf{a}^0$, $\mathbf{a}^1$, $\mathbf{a}^2$, …, $\mathbf{a}^k$. This is a collection of $k+1$ strings and there are only k states in D. Therefore, by the pigeonhole principle, there must be two distinct strings $\mathbf{a}^m$ and $\mathbf{a}^n$ that end in the same state when run through D.

Our lemma tells us that $\mathbf{a}^m \not\equiv_E \mathbf{a}^n$, so by our earlier theorem we know that $\mathbf{a}^m$ and $\mathbf{a}^n$ cannot end in the same state when run through D. But this is impossible, since we know that $\mathbf{a}^m$ and $\mathbf{a}^n$ do end in the same state when run through D.

We have reached a contradiction, so our assumption must have been wrong. Therefore, E is not regular. ■

Theorem: The language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: Suppose for the sake of contradiction that E is regular. Let D be a DFA for E, and let k be the number of states in D. Consider the strings $\mathbf{a}^0$, $\mathbf{a}^1$, $\mathbf{a}^2$, …, $\mathbf{a}^k$. This is a collection of $k+1$ strings and there are only k states in D. Therefore, by the pigeonhole principle, there must be two distinct strings $\mathbf{a}^m$ and $\mathbf{a}^n$ that end in the same state when run through D.

Our lemma tells us that $\mathbf{a}^m \not\equiv_E \mathbf{a}^n$, so by our earlier theorem we know that $\mathbf{a}^m$ and $\mathbf{a}^n$ cannot end in the same state when run through D. But this is impossible, since we know that $\mathbf{a}^m$ and $\mathbf{a}^n$ do end in the same state when run through D.

We have reached a contradiction, so our assumption must have been wrong. Therefore, E is not regular. ■

We're going to see a simpler proof of this result later on once we've built more machinery. If (hypothetically speaking) you want to prove something like this in the future, we'd recommend not using this proof as a template.

What Just Happened?

- ***We've just hit the limit of finite-memory computation.***
- To build a DFA for $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$, we need to have different memory configurations (states) for all possible strings of the form $\mathbf{a}^n$.
- There's no way to do this with finitely many possible states!

Where We're Going

- We just used the idea of *distinguishability* to show that no possible DFA can exist for some language.
- This technique turns out to be pretty powerful.
- We're going to see one more example of this technique in action, then generalize it to an extremely powerful theorem for finding nonregular languages.

Theorem: The language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ is not a regular language.

Proof: Suppose for the sake of contradiction that E is regular. Let D be a DFA for E and let k be the number of states in D.

Consider the strings $\mathbf{a}^0$, $\mathbf{a}^1$, $\mathbf{a}^2$, …, $\mathbf{a}^k$. This is a collection of $k+1$ strings and there are only k states in D. Therefore, by the pigeonhole principle, there must be two distinct strings $\mathbf{a}^m$ and $\mathbf{a}^n$ that end in the same state when run through D.

Our lemma tells us that $\mathbf{a}^m \not\equiv_E \mathbf{a}^n$. By our earlier theorem we know that $\mathbf{a}^m$ and $\mathbf{a}^n$ cannot end in the same state when run through D. But this is impossible, since we know that $\mathbf{a}^m$ and $\mathbf{a}^n$ do end in the same state when run through D.

We have reached a contradiction, so our assumption must have been wrong. Therefore, E is not regular. ■

Theorem: The language L = **[fill in the blank]** is not a regular language.

Proof: Suppose for the sake of contradiction that L is regular. Let D be a DFA for L and let k be the number of states in D.

Consider **[some *k*+1 specific strings.]** This is a collection of $k+1$ strings and there are only k states in D. Therefore, by the pigeonhole principle, there must be two distinct strings x and y that end in the same state when run through D.

[Somehow we know] that $x \not\equiv_L y$. By our earlier theorem we know that x and y cannot end in the same state when run through D. But this is impossible, since we know that x and y must end in the same state when run through D.

We have reached a contradiction, so our assumption must have been wrong. Therefore, L is not regular. ■

Theorem: The language L = **[fill in the blank]** is not a regular language.

Proof: Suppose for the sake of contradiction that L is regular. Let D be a DFA for L and let k be the number of states in D.

Consider **[some *k*+1 specific strings.]** This is a collection of $k+1$ strings and there are only k states in D. Therefore, by the pigeonhole principle, there must be two distinct strings x and y that end in the same state when run through D.

[Somehow we know] that $x \not\equiv_L y$. By our earlier theorem we know that x and y cannot end in the same state when run through D. But this is impossible, since we know that x and y must end in the same state when run through D.

We have reached a contradiction, so our assumption must have been wrong. Therefore, L is not regular. ■

Theorem: The language L = **[** regular language.

Proof: Suppose for the sake of contradiction that L is regular. Let D be a DFA for L and let k be the number of states in D.

For any number of states k, we need a way to find k+1 strings so that two of them get into the same state...

Consider **[some *k*+1 specific strings.]** This is a collection of k+1 strings and there are only k states in D. Therefore, by the pigeonhole principle, there must be two distinct strings x and y that end in the same state when run through D.

[Somehow we know] that $x \not\equiv_L y$. By our earlier theorem we know that x and y cannot end in the same state when run through D. But this is impossible, since we know that x and y must end in the same state when run through D.

We have reached a contradiction, so our assumption must have been wrong. Therefore, L is not regular. ■

Theorem: The language L = **[** regular language.

Proof: Suppose for the sake of contradiction that L is regular. Let D be a DFA for L and let k be the number of states in D.

Consider **[some *k*+1 specific strings.]** This is a collection of $k+1$ strings and there are only k states in D. Therefore, by the pigeonhole principle, there must be two distinct strings x and y that end in the same state when run through D.

[Somehow we know] that $x \not\equiv_L y$. By our earlier theorem we know that x and y cannot end in the same state when run through D. But this is impossible, since we know that x and y must end in the same state when

We have reached a contradiction, so have been wrong. Therefore, L is

For any number of states k, we need a way to find $k+1$ strings so that two of them get into the same state...

... and all those strings need to be distinguishable so that we get a contradiction.

Distinguishing Sets

- Let L be a language over Σ.
- A ***distinguishing set*** for L is a set $S \subseteq \Sigma^*$ where the following is true:

$$\forall x \in S.\ \forall y \in S.\ (x \neq y \rightarrow x \not\equiv_L y)$$

Distinguishing Sets

- Let L be a language over Σ.
- A ***distinguishing set*** for L is a set $S \subseteq \Sigma^*$ where the following is true:

$$\forall x \in S.\ \forall y \in S.\ (x \neq y \rightarrow x \not\equiv_L y)$$

If you pick any two strings in S that aren't equal to one another...

... then they're distinguishable relative to L.

Distinguishing Sets

- Let L be a language over Σ.
- A ***distinguishing set*** for L is a set $S \subseteq \Sigma^*$ where the following is true:

$$\forall x \in S.\ \forall y \in S.\ (x \neq y \rightarrow x \not\equiv_L y)$$

- As an example, here's a distinguishing set for $E = \{\ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N}\ \}$:

$$S = \{\ \mathbf{a}^n \mid n \in \mathbb{N}\ \}$$

Theorem (Myhill-Nerode): If L is a language and S is a distinguishing set for L that contains **infinitely many strings**, then L is not regular.

Intuition: that infinite set of distinguishing strings would require the number of states in the DFA to be infinite, and DFAs only have a finite number of states.

Proof: Let L be an arbitrary language over Σ and let S be a distinguishing set for L that contains infinitely many strings. We will show that L is not regular.

Suppose for the sake of contradiction that L is regular. This means that there must be some DFA D for L. Let k be the number of states in D. Since there are infinitely many strings in S, we can choose $k+1$ distinct strings from S and consider what happens when we run D on all of those strings. Because there are only k states in D and we've chosen $k+1$ strings from S, by the pigeonhole principle we know that at least two strings from S must end in the same state in D. Choose any two such strings and call them x and y.

Because $x \neq y$ and S is a distinguishing set for L, we know that $x \not\equiv_L y$. Our earlier theorem therefore tells us that when we run D on inputs x and y, they must end up in different states. But this is impossible – we chose x and y precisely because they end in the same state when run through D.

We have reached a contradiction, so our assumption must have been wrong. Thus L is not a regular language. ■

Using the Myhill-Nerode Theorem

Theorem: The language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ is not regular.

Theorem: The language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ is not regular.

Proof:

Theorem: The language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: Let $S = \{ \mathbf{a}^n \mid n \in \mathbb{N} \}$.

Theorem: The language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: Let $S = \{ \mathbf{a}^n \mid n \in \mathbb{N} \}$. We will prove that S is infinite and that S is a distinguishing set for E.

Theorem: The language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: Let $S = \{ \mathbf{a}^n \mid n \in \mathbb{N} \}$. We will prove that S is infinite and that S is a distinguishing set for E.

To see that S is infinite, note that S contains one string for each natural number.

Theorem: The language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: Let $S = \{ \mathbf{a}^n \mid n \in \mathbb{N} \}$. We will prove that S is infinite and that S is a distinguishing set for E.

To see that S is infinite, note that S contains one string for each natural number.

To see that S is a distinguishing set for E, consider any strings strings $\mathbf{a}^m$, $\mathbf{a}^n \in S$ where $m \neq n$.

Theorem: The language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: Let $S = \{ \mathbf{a}^n \mid n \in \mathbb{N} \}$. We will prove that S is infinite and that S is a distinguishing set for E.

To see that S is infinite, note that S contains one string for each natural number.

To see that S is a distinguishing set for E, consider any strings strings $\mathbf{a}^m$, $\mathbf{a}^n \in S$ where $m \neq n$. Note that $\mathbf{a}^m\mathbf{b}^m \in E$ and that $\mathbf{a}^n\mathbf{b}^m \notin E$.

Theorem: The language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: Let $S = \{ \mathbf{a}^n \mid n \in \mathbb{N} \}$. We will prove that S is infinite and that S is a distinguishing set for E.

To see that S is infinite, note that S contains one string for each natural number.

To see that S is a distinguishing set for E, consider any strings strings $\mathbf{a}^m$, $\mathbf{a}^n \in S$ where $m \neq n$. Note that $\mathbf{a}^m\mathbf{b}^m \in E$ and that $\mathbf{a}^n\mathbf{b}^m \notin E$. Therefore, we see that $\mathbf{a}^m \not\equiv_E \mathbf{a}^n$, as required.

Theorem: The language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: Let $S = \{ \mathbf{a}^n \mid n \in \mathbb{N} \}$. We will prove that S is infinite and that S is a distinguishing set for E.

To see that S is infinite, note that S contains one string for each natural number.

To see that S is a distinguishing set for E, consider any strings strings $\mathbf{a}^m$, $\mathbf{a}^n \in S$ where $m \neq n$. Note that $\mathbf{a}^m\mathbf{b}^m \in E$ and that $\mathbf{a}^n\mathbf{b}^m \notin E$. Therefore, we see that $\mathbf{a}^m \not\equiv_E \mathbf{a}^n$, as required.

Since S is infinite and is a distinguishing set for E, by the Myhill-Nerode theorem we see that E is not regular.

Theorem: The language $E = \{ \mathbf{a}^n\mathbf{b}^n \mid n \in \mathbb{N} \}$ is not regular.

Proof: Let $S = \{ \mathbf{a}^n \mid n \in \mathbb{N} \}$. We will prove that S is infinite and that S is a distinguishing set for E.

To see that S is infinite, note that S contains one string for each natural number.

To see that S is a distinguishing set for E, consider any strings strings $\mathbf{a}^m$, $\mathbf{a}^n \in S$ where $m \neq n$. Note that $\mathbf{a}^m\mathbf{b}^m \in E$ and that $\mathbf{a}^n\mathbf{b}^m \notin E$. Therefore, we see that $\mathbf{a}^m \not\equiv_E \mathbf{a}^n$, as required.

Since S is infinite and is a distinguishing set for E, by the Myhill-Nerode theorem we see that E is not regular. ■

Another Language

- Consider the following language L over the alphabet $\Sigma = \{\mathbf{a}, \mathbf{b}, \stackrel{?}{=}\}$:

$$EQ = \{\ w \stackrel{?}{=} w \mid w \in \{\mathbf{a}, \mathbf{b}\}^*\}$$

- EQ is the language all strings consisting of the same string of **a**'s and **b**'s twice, with a $\stackrel{?}{=}$ symbol in-between.
- Examples:

$\mathbf{ab} \stackrel{?}{=} \mathbf{ab} \in EQ$ $\quad$ $\mathbf{bbb} \stackrel{?}{=} \mathbf{bbb} \in EQ$ $\quad$ $\stackrel{?}{=} \in EQ$

$\mathbf{ab} \stackrel{?}{=} \mathbf{ba} \notin EQ$ $\quad$ $\mathbf{bbb} \stackrel{?}{=} \mathbf{aaa} \notin EQ$ $\quad$ $\mathbf{b} \stackrel{?}{=} \notin EQ$

Another Language

$$EQ = \{ w \stackrel{?}{=} w \mid w \in \{\mathbf{a}, \mathbf{b}\}^* \}$$

- This language corresponds to the following problem:

Given strings x, $y \in \{\mathbf{a}, \mathbf{b}\}^*$, does $x = y$?

- We can think of things this way because

$$x = y \quad \text{if and only if} \quad x \stackrel{?}{=} y \in EQ.$$

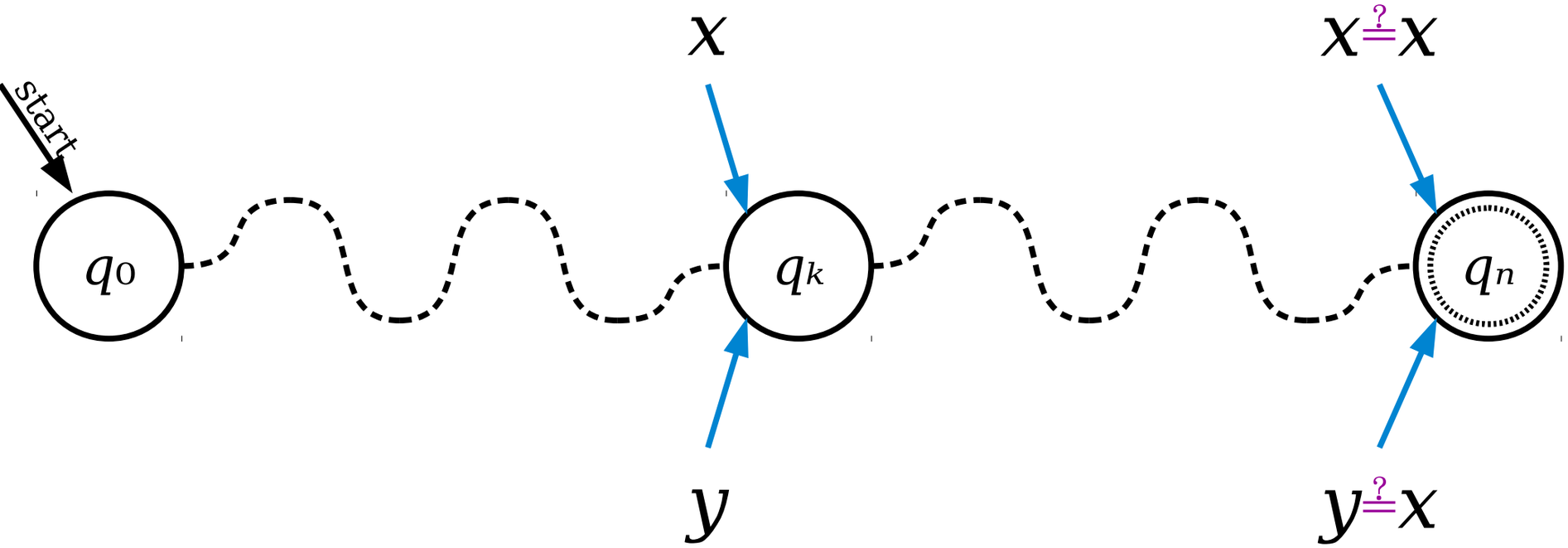
- Is this language regular?

The Intuition

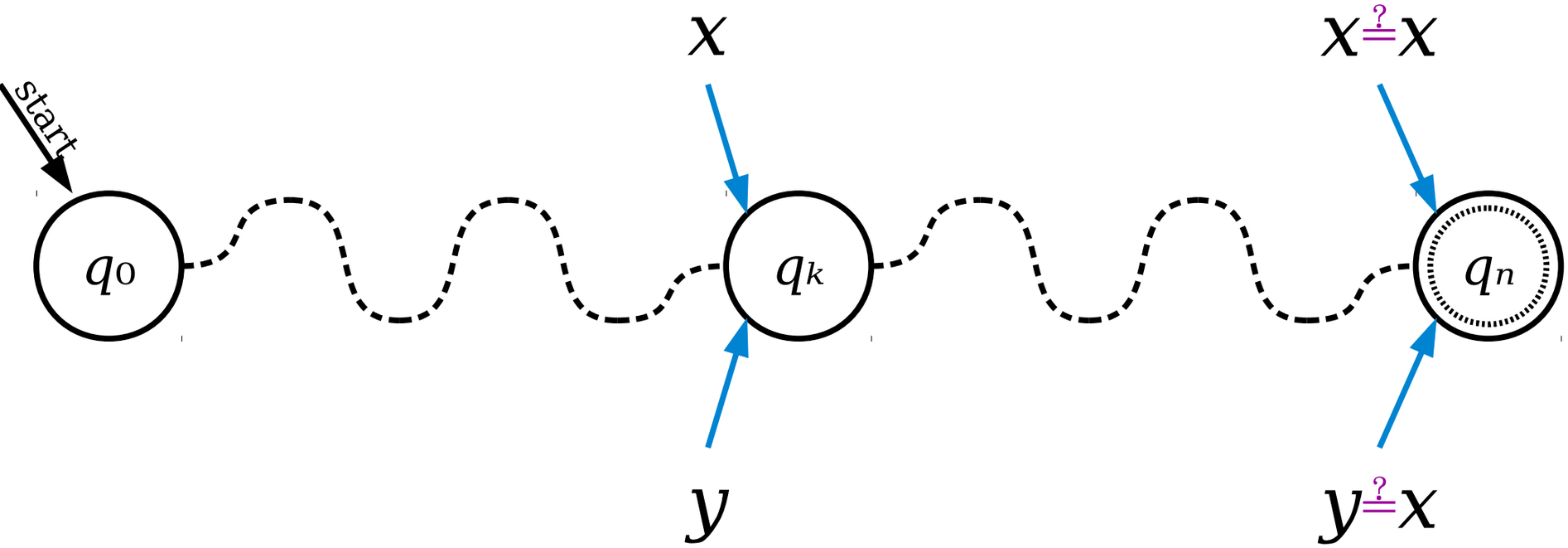
$$EQ = \{ w \stackrel{?}{=} w \mid w \in \{\mathbf{a}, \mathbf{b}\}^* \}$$

- Intuitively, any machine for EQ has to be able to remember the contents of everything to the left of the $\stackrel{?}{=}$ so that it can match them against the contents of the string to the right of the $\stackrel{?}{=}$.
- There are infinitely many possible strings we can see, but we only have finite memory to store which string we saw.
- That's a problem... can we formalize this?

The Intuition



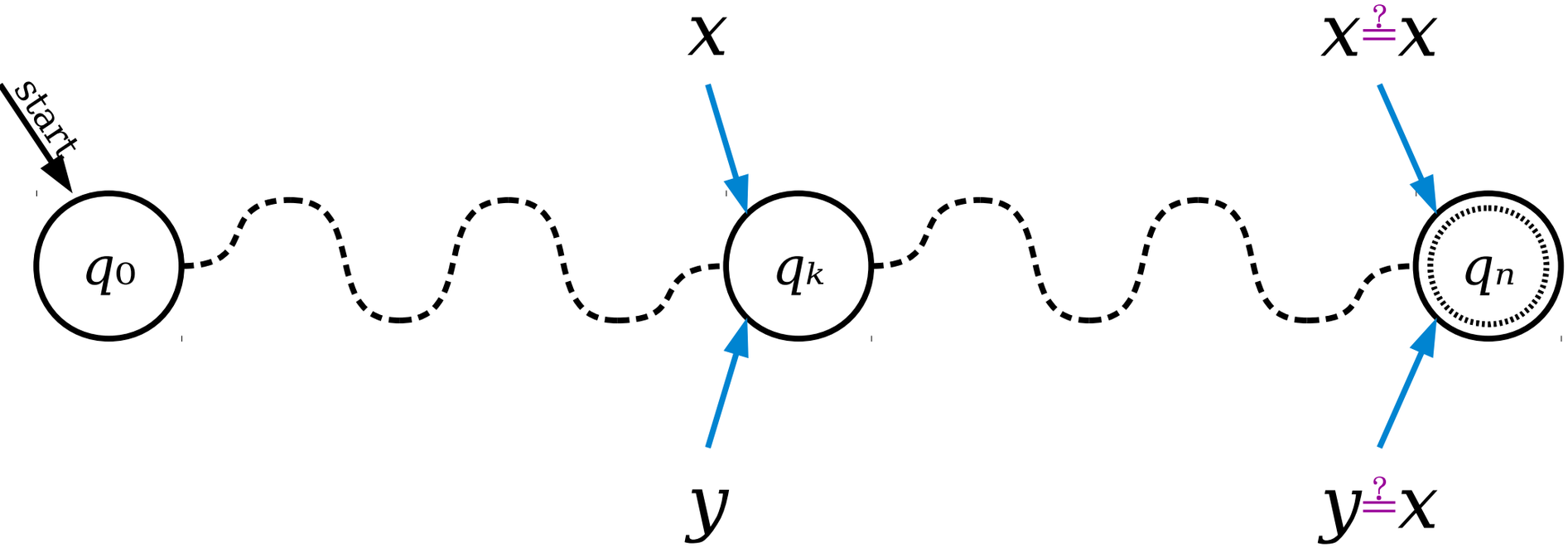
The Intuition



What happens if q_n is...

...an accepting state?
...a rejecting state?

The Intuition

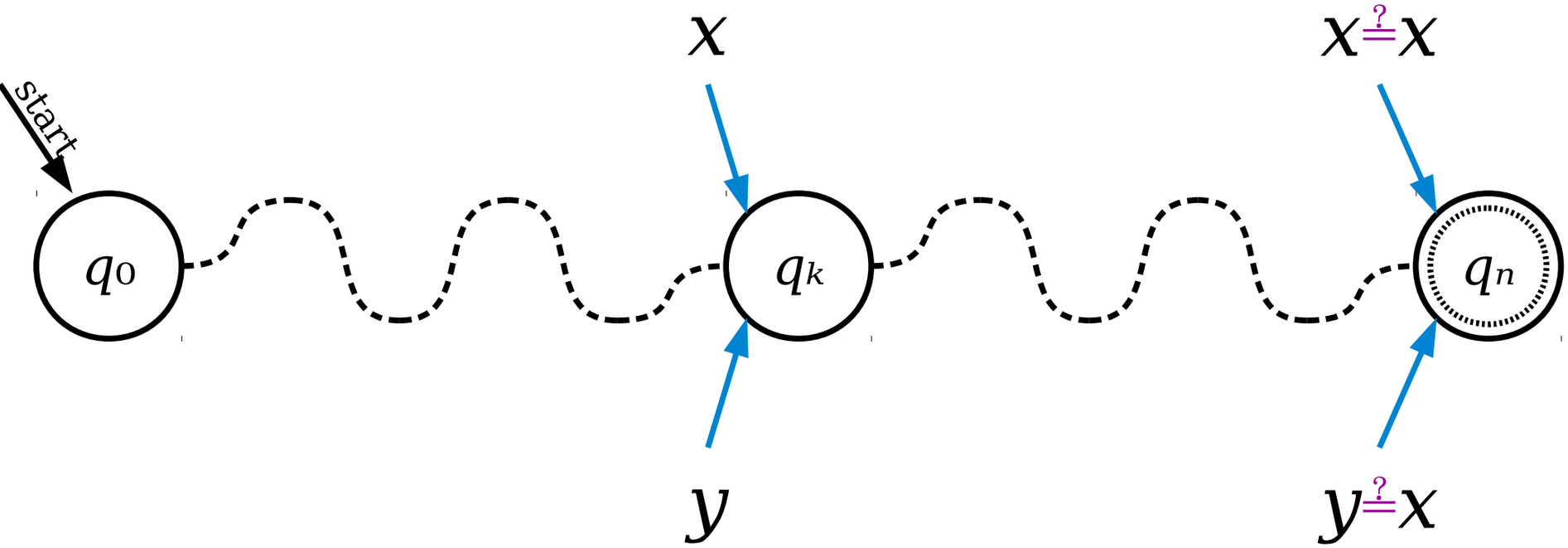


What happens if q_n is...

...an accepting state? We accept $y \stackrel{?}{=} x \notin EQ$!

...a rejecting state?

The Intuition



What happens if q_n is...

...an accepting state? We accept $y \stackrel{?}{=} x \notin EQ$!

...a rejecting state? We reject $x \stackrel{?}{=} x \in EQ$!

Theorem: The language $EQ = \{ w \stackrel{?}{=} w \mid w \in \{\mathbf{a}, \mathbf{b}\}^*\}$ is not regular.

Theorem: The language $EQ = \{ w \stackrel{?}{=} w \mid w \in \{\mathbf{a}, \mathbf{b}\}^*\}$ is not regular.

Proof:

Theorem: The language $EQ = \{ w \stackrel{?}{=} w \mid w \in \{\mathbf{a}, \mathbf{b}\}^*\}$ is not regular.

Proof: Let $S = \{\mathbf{a}, \mathbf{b}\}^*$.

Theorem: The language $EQ = \{ w \stackrel{?}{=} w \mid w \in \{\mathtt{a}, \mathtt{b}\}^*\}$ is not regular.

Proof: Let $S = \{\mathtt{a}, \mathtt{b}\}^*$. We will prove that S is infinite and that S is a distinguishing set for EQ.

Theorem: The language $EQ = \{ w \stackrel{?}{=} w \mid w \in \{\mathtt{a}, \mathtt{b}\}^*\}$ is not regular.

Proof: Let $S = \{\mathtt{a}, \mathtt{b}\}^*$. We will prove that S is infinite and that S is a distinguishing set for EQ.

To see that S is infinite, note that, for any $n \in \mathbb{N}$, we have $\mathtt{a}^n \in S$.

Theorem: The language $EQ = \{ w \stackrel{?}{=} w \mid w \in \{\mathtt{a}, \mathtt{b}\}^* \}$ is not regular.

Proof: Let $S = \{\mathtt{a}, \mathtt{b}\}^*$. We will prove that S is infinite and that S is a distinguishing set for EQ.

To see that S is infinite, note that, for any $n \in \mathbb{N}$, we have $\mathtt{a}^n \in S$. Therefore, S contains at least one string for each natural number, so S is infinite.

Theorem: The language $EQ = \{ w \stackrel{?}{=} w \mid w \in \{\mathtt{a}, \mathtt{b}\}^*\}$ is not regular.

Proof: Let $S = \{\mathtt{a}, \mathtt{b}\}^*$. We will prove that S is infinite and that S is a distinguishing set for EQ.

To see that S is infinite, note that, for any $n \in \mathbb{N}$, we have $\mathtt{a}^n \in S$. Therefore, S contains at least one string for each natural number, so S is infinite.

To see that S is a distinguishing set for EQ, consider any strings $x, y \in S$ where $x \neq y$.

Theorem: The language $EQ = \{ w \stackrel{?}{=} w \mid w \in \{\mathbf{a}, \mathbf{b}\}^*\}$ is not regular.

Proof: Let $S = \{\mathbf{a}, \mathbf{b}\}^*$. We will prove that S is infinite and that S is a distinguishing set for EQ.

To see that S is infinite, note that, for any $n \in \mathbb{N}$, we have $\mathbf{a}^n \in S$. Therefore, S contains at least one string for each natural number, so S is infinite.

To see that S is a distinguishing set for EQ, consider any strings $x, y \in S$ where $x \neq y$. Then $x \stackrel{?}{=} x \in EQ$ and $y \stackrel{?}{=} x \notin EQ$.

Theorem: The language $EQ = \{ w \stackrel{?}{=} w \mid w \in \{\mathbf{a}, \mathbf{b}\}^*\}$ is not regular.

Proof: Let $S = \{\mathbf{a}, \mathbf{b}\}^*$. We will prove that S is infinite and that S is a distinguishing set for EQ.

To see that S is infinite, note that, for any $n \in \mathbb{N}$, we have $\mathbf{a}^n \in S$. Therefore, S contains at least one string for each natural number, so S is infinite.

To see that S is a distinguishing set for EQ, consider any strings $x, y \in S$ where $x \not\equiv_{EQ} y$. Then $x \stackrel{?}{=} x \in EQ$ and $y \stackrel{?}{=} x \notin EQ$. Therefore, $x \not\equiv \quad y$, as required.

Theorem: The language $EQ = \{ w \stackrel{?}{=} w \mid w \in \{\mathbf{a}, \mathbf{b}\}^*\}$ is not regular.

Proof: Let $S = \{\mathbf{a}, \mathbf{b}\}^*$. We will prove that S is infinite and that S is a distinguishing set for EQ.

To see that S is infinite, note that, for any $n \in \mathbb{N}$, we have $\mathbf{a}^n \in S$. Therefore, S contains at least one string for each natural number, so S is infinite.

To see that S is a distinguishing set for EQ, consider any strings $x, y \in S$ where $x \neq_{EQ} y$. Then $x \stackrel{?}{=} x \in EQ$ and $y \stackrel{?}{=} x \notin EQ$. Therefore, $x \not\equiv \quad y$, as required.

Since S is infinite and a distinguishing set for EQ, by the Myhill-Nerode theorem we see that EQ is not regular, as required.

Theorem: The language $EQ = \{ w \stackrel{?}{=} w \mid w \in \{\mathbf{a}, \mathbf{b}\}^*\}$ is not regular.

Proof: Let $S = \{\mathbf{a}, \mathbf{b}\}^*$. We will prove that S is infinite and that S is a distinguishing set for EQ.

To see that S is infinite, note that, for any $n \in \mathbb{N}$, we have $\mathbf{a}^n \in S$. Therefore, S contains at least one string for each natural number, so S is infinite.

To see that S is a distinguishing set for EQ, consider any strings $x, y \in S$ where $x \neq_{EQ} y$. Then $x \stackrel{?}{=} x \in EQ$ and $y \stackrel{?}{=} x \notin EQ$. Therefore, $x \not\equiv \quad y$, as required.

Since S is infinite and a distinguishing set for EQ, by the Myhill-Nerode theorem we see that EQ is not regular, as required. ■

Approaching Myhill-Nerode

- The challenge in using the Myhill-Nerode theorem is finding the right set of strings.
- ***General intuition:***
  - Start by thinking about what information a computer “must” remember in order to answer correctly.
  - Choose a group of strings that all require different information.
  - Prove that you have infinitely many strings an that the group of strings is a distinguishing set.

Palindromes

Let $\Sigma = \{$ **A**, **N** $\}$.

Consider the language

$$L = \{ w \in \Sigma^* \mid w \text{ is a palindrome} \}$$

Palindromes

Let $\Sigma = \{$ **A**, **N** $\}$.

Consider the language

$$L = \{ w \in \Sigma^* \mid w \text{ is a palindrome} \}$$

N $\in L$

ANANA $\in L$

AAAA $\in L$

AN $\notin L$

NNNNAN $\notin L$

NAAA $\notin L$

Palindromes

Let $\Sigma = \{$ **A**, **N** $\}$.

Consider the language

$$L = \{ w \in \Sigma^* \mid w \text{ is a palindrome} \}$$

3a) Which of the following sets would be a suitable choice to show that L is non-regular? For choices that don't work, make sure you can explain why not.

i. { **A**, **AA**, **AAA**, **AAAA**, **AAAAA** }

ii. $\{ \mathbf{A}^n \mid n \in \mathbb{N} \}$

iii. $\{ \mathbf{A}^n\mathbf{N} \mid n \in \mathbb{N} \}$

iv. $\{ \mathbf{A}^n\mathbf{N}^n \mid n \in \mathbb{N} \}$

As a reminder, a distinguishing set $S \subseteq \Sigma^*$ is a set such that:

$$\forall x \in S.\ \forall y \in S.\ (x \neq y \rightarrow \exists w \in \Sigma^*.\ (xw \in L \leftrightarrow yw \notin L))$$

Submit on Gradescope!

$$L = \{\ w \in \Sigma^* \mid w \text{ is a palindrome}\ \}$$

(i) $S = \{$ **A**, **AA**, **AAA**, **AAAA**, **AAAAA** $\}$

As a reminder, a distinguishing set $S \subseteq \Sigma^*$ is a set such that:

$$\forall x \in S.\ \forall y \in S.\ (x \neq y \rightarrow \exists w \in \Sigma^*.\ (xw \in L \leftrightarrow yw \notin L))$$

$$L = \{\ w \in \Sigma^* \mid w \text{ is a palindrome}\ \}$$

(ii)
$$S = \{\ \mathbf{A}^n \mid n \in \mathbb{N}\ \}$$

As a reminder, a distinguishing set $S \subseteq \Sigma^*$ is a set such that:

$$\forall x \in S.\ \forall y \in S.\ (x \neq y \rightarrow \exists w \in \Sigma^*.\ (xw \in L \leftrightarrow yw \notin L))$$

$$L = \{\ w \in \Sigma^* \mid w \text{ is a palindrome}\ \}$$

(iii)
$$S = \{\ \mathbf{A}^n\mathbf{N}\ \mid n \in \mathbb{N}\ \}$$

As a reminder, a distinguishing set $S \subseteq \Sigma^*$ is a set such that:

$$\forall x \in S.\ \forall y \in S.\ (x \neq y \rightarrow \exists w \in \Sigma^*.\ (xw \in L \leftrightarrow yw \notin L))$$

$$L = \{\ w \in \Sigma^* \mid w \text{ is a palindrome}\ \}$$

(iv)
$$S = \{\ \mathbf{A}^n\mathbf{N}^n \mid n \in \mathbb{N}\ \}$$

As a reminder, a distinguishing set $S \subseteq \Sigma^*$ is a set such that:

$$\forall x \in S.\ \forall y \in S.\ (x \neq y \rightarrow \exists w \in \Sigma^*.\ (xw \in L \leftrightarrow yw \notin L))$$

Palindromes

Let $\Sigma = \{ \mathtt{A}, \mathtt{N} \}$.

Consider the language

$$L = \{ w \in \Sigma^* \mid w \text{ is a palindrome} \}$$

3b) Let's say we choose S to be $\{ \mathtt{A}^n \mid n \in \mathbb{N} \}$. Identify all the errors in the following *incorrect* way of proving that S is a distinguishing set.

- Consider any two strings $\mathtt{A}^n, \mathtt{A}^m \in S$ where $m \neq n$. Then $\mathtt{A}^n\mathtt{A}^n \in L$ but $\mathtt{A}^{n+1}\mathtt{A}^n \notin L$.

Submit on Gradescope!

$$L = \{\ w \in \Sigma^* \mid w \text{ is a palindrome}\ \}$$

$$S = \{\ \mathbf{A}^n \mid n \in \mathbb{N}\ \}$$

Consider any two strings $\mathbf{A}^n$, $\mathbf{A}^m \in S$ where $m \neq n$.
Then $\mathbf{A}^n\mathbf{A}^n \in L$ but $\mathbf{A}^{n+1}\mathbf{A}^n \notin L$.

As a reminder, a distinguishing set $S \subseteq \Sigma^*$ is a set such that:

$$\forall x \in S.\ \forall y \in S.\ (x \neq y \to \exists w \in \Sigma^*.\ (xw \in L \leftrightarrow yw \notin L))$$

Theorem: The language $L = \{ w \in \Sigma^* \mid w \text{ is a palindrome} \}$ is non-regular.

Proof: Let $S = \{ \mathbf{A}^n \mid n \in \mathbb{N} \}$. We will prove that S is infinite and that S is a distinguishing set for L.

To see that S is infinite, note that S contains one string for each natural number.

To see that S is a distinguishing set for L, consider any strings $\mathbf{A}^m$, $\mathbf{A}^n \in S$ where $m \neq n$. Note that $\mathbf{A}^m\mathbf{N}\mathbf{A}^m \in L$ but $\mathbf{A}^n\mathbf{N}\mathbf{A}^m \notin L$. Therefore, we see that $\mathbf{A}^m \not\equiv_L \mathbf{A}^n$, as required.

Since S is infinite and is a distinguishing set for L, by the Myhill-Nerode theorem we see that L is not regular. ■

Tying Everything Together

- One of the intuitions we hope you develop for DFAs is to have each state in a DFA represent some key piece of information the automaton has to remember.
- If you only need to remember one of finitely many pieces of information, that gives you a DFA.
  - This can be made rigorous! Take CS154 for details.
- If you need to remember one of infinitely many pieces of information, you can use the Myhill-Nerode theorem to prove that the language has no DFA.

Where We Stand

Where We're Going

- What does computation look like with unbounded memory?
- What problems can you solve with unbounded-memory computers?
- What does it even mean to “solve” such a problem?
- And how do we know the answers to any of these questions?

Next Time

- ***Context-Free Languages***
  - Context-Free Grammars
  - Generating Languages from Scratch