CMPU 240 · Theory of Computation

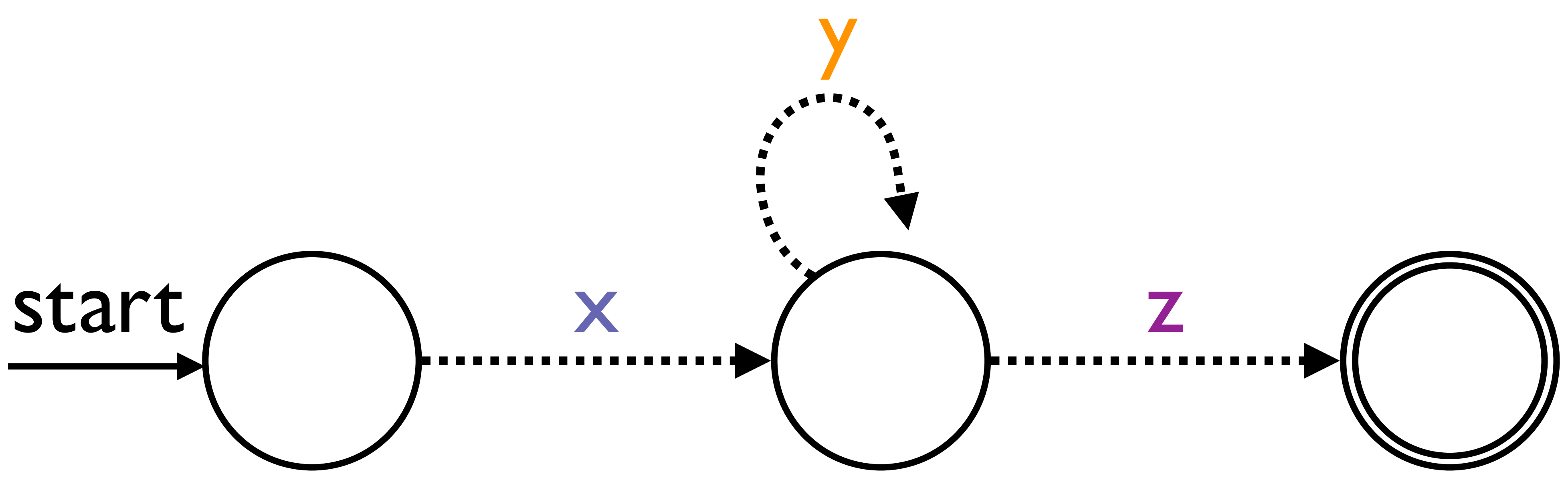
Nonregular Languages

19 February 2026

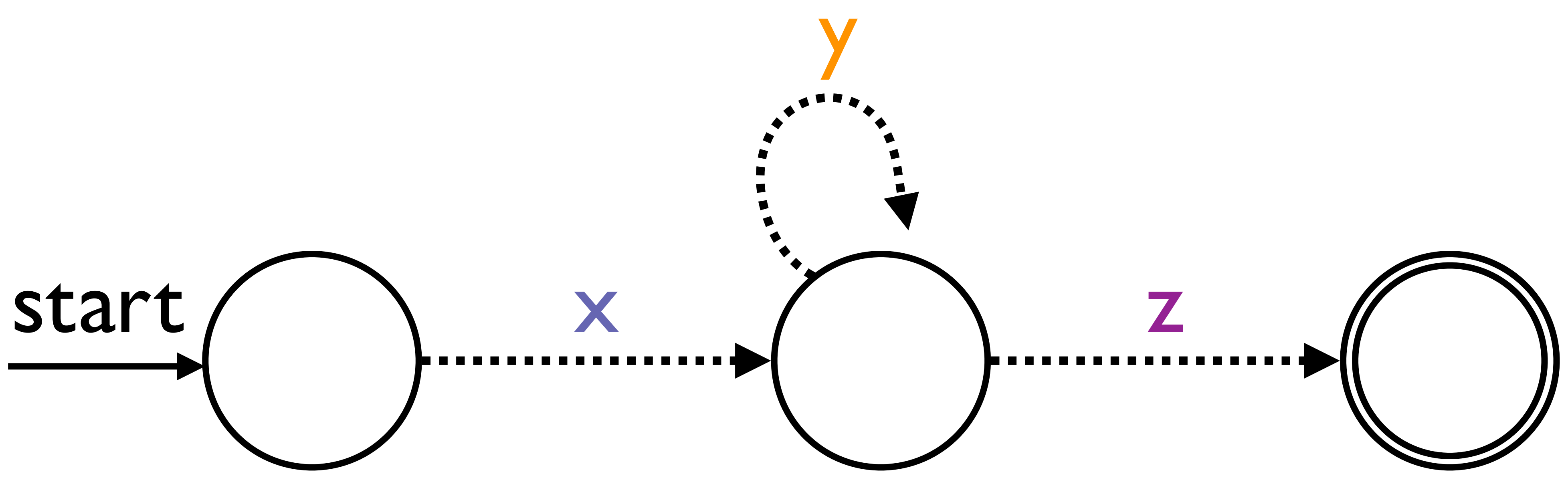
Where are we?

If L is a regular language, there must be a DFA D that recognizes L.

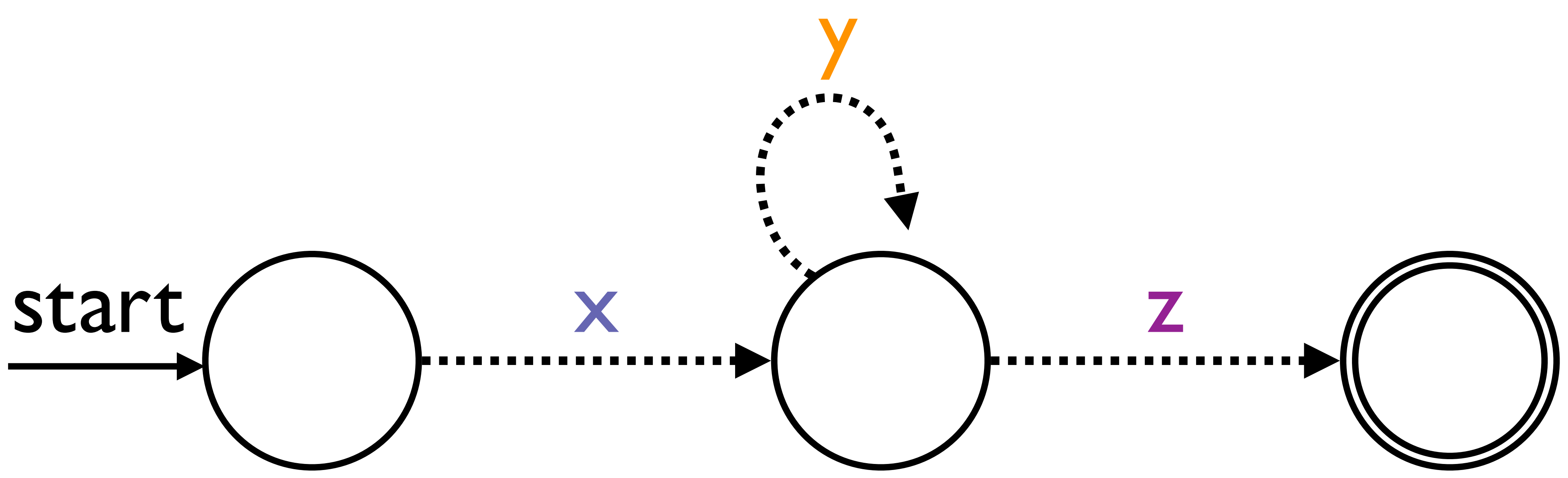
A sufficiently long string $s \in L$ must visit some state in D twice. This means s went through a loop in the DFA D.



xyz

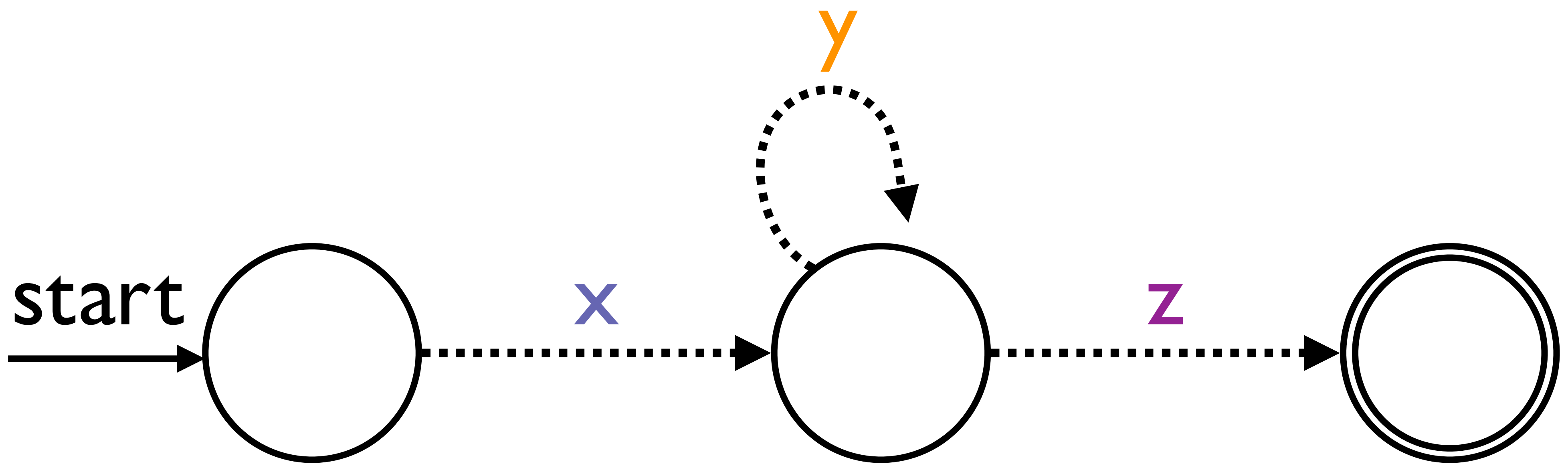


$i = 1$ xy^1z xyz



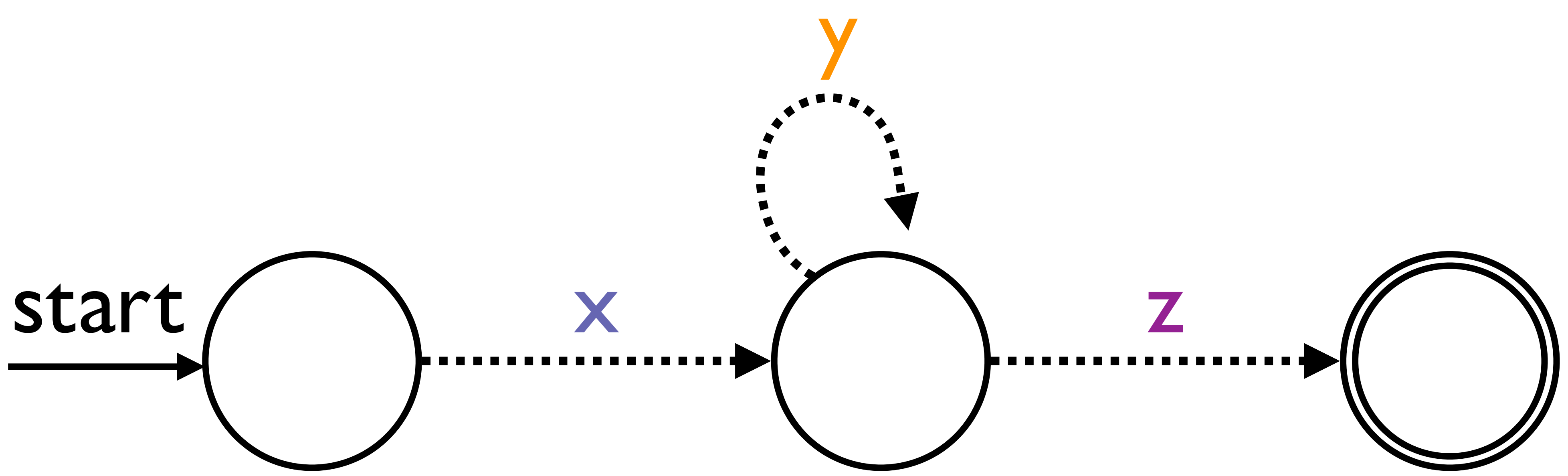
$i = 1$	xy^1z	xyz
$i = 2$	xy^2z	$xyyz$

Pumping up

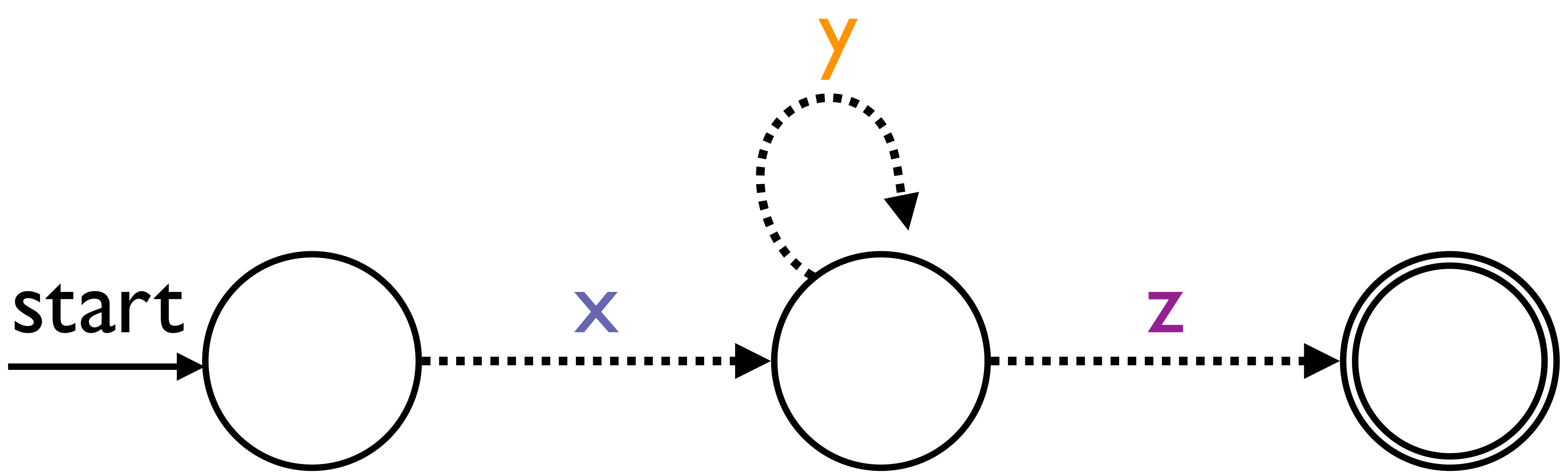


$i = 1$	xy^1z	xyz
$i = 2$	xy^2z	$xyyz$
$i = 3$	xy^3z	$xyyyz$
$\vdots$	$\vdots$	$\vdots$

Pumping up



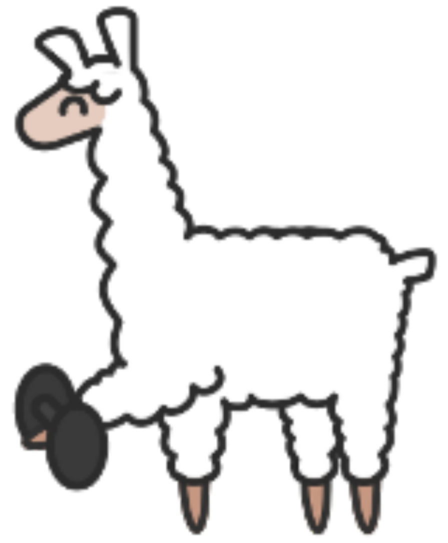
$i = 0$	xy^0z	xz	*Pumping down*
$i = 1$	xy^1z	xyz	
$i = 2$	xy^2z	$xyyz$	*Pumping up*
$i = 3$	xy^3z	$xyyyz$	
⋮	⋮	⋮	



A DFA with a loop (whether that’s a self-loop or a cycle through several states) will always recognize an infinite language, with each trip through the loop permitting a longer string.

This is what the Pumping Lemma describes: How DFAs loop!

The Pumping Lemma



For any regular language L,

there exists a positive integer p such that

for any string $s \in L$ such that $|s| \geq p$,

there exist strings x, y, and z such that

$s = xyz$	*s can be broken into three pieces,*
$\|xy\| \leq p$	*where the first two pieces occur at the start of the string,*
$y \neq \varepsilon$	*the middle part isn't empty, and*
$xy^iz \in L$	*the middle piece can be repeated zero or more times.*

To prove a language is nonregular, we show that no DFA for it is possible:

Any DFA you can imagine recognizing that language must have a loop that leads it to accept strings that aren’t in the language.

THEOREM $B = \{a^n b^n \mid n \in \mathbb{N}_0\}$ is not regular.

PROOF By contradiction; assume B is regular. Let p be the pumping length guaranteed by the Pumping Lemma. Consider the string $s = a^p b^p$. Then $|s| = 2p \geq p$ and $s \in B$, so we can break this string into $s = xyz$, where $|xy| \leq p$ and $y \neq \varepsilon$, and for any $i \in \mathbb{N}_0$, the string $xy^iz \in B$.

Because $|xy| \leq p$ and $|y| > 0$, the string y has to consist only of as.

So, no matter what segment of the string xy covers, pumping to the string xy^2z adds to the number of as, hence there are more as than bs.

There is no way to segment s into xyz that can't be pumped to produce a string that isn't in the language.

Contradiction! Therefore, B is not regular. ■

Pumping Lemma practice: Balanced strings

Example

Consider the alphabet $\Sigma = \{0,1\}$ and the language

BALANCE = {*w* | *w* has an equal number of 1s and 0s}

Example

Consider the alphabet $\Sigma = \{\texttt{0},\texttt{1}\}$ and the language

$$BALANCE = \{w \mid n_{\texttt{0}}(w) = n_{\texttt{1}}(w)\}$$

Example

Consider the alphabet $\Sigma = \{0,1\}$ and the language

$$BALANCE = \{w \mid n_0(w) = n_1(w)\}$$

E.g.,

$01 \in BALANCE$

$110010 \in BALANCE$

$11011 \notin BALANCE$

Is *BALANCE* a regular language?

An incorrect proof

THEOREM *BALANCE* is regular.

PROOF We show that *BALANCE* satisfies the condition of the Pumping Lemma. Let $p = 2$ and consider any string $s \in$ *BALANCE* such that $|s| \geq 2$. Then we can write $s = xyz$ such that $x = z = \varepsilon$ and $y = s$, so $y \neq \varepsilon$. Then for any natural number i, $xy^iz = s^i$, which has the same number of `0`s and `1`s. Since *BALANCE* passes the conditions of the Pumping Lemma, *BALANCE* is regular.

An incorrect proof

THEOREM *BALANCE* is regular.

What? No!

PROOF We show that *BALANCE* satisfies the condition of the Pumping Lemma. Let $p = 2$ and consider any string $s \in BALANCE$ such that $|s| \geq 2$. Then we can write $s = xyz$ such that $x = z = \varepsilon$ and $y = s$, so $y \neq \varepsilon$. Then for any natural number i, $xy^iz = s^i$, which has the same number of `0`s and `1`s. Since *BALANCE* passes the conditions of the Pumping Lemma, *BALANCE* is regular.

*The Pumping Lemma doesn't claim anything about languages that **aren't** regular!*

For every regular language L,

there exists a positive integer p such that

for every string $s \in L$ with $|s| \geq p$,

there exist strings x, y, and z such that

$s = xyz$

$|xy| \leq p$

$y \neq \varepsilon$

$xy^iz \in L$ for all $i \geq 0$

The Pumping Lemma describes a *necessary* condition of regular languages.

If L is regular, L satisfies the conditions of the Pumping Lemma.

But the Pumping Lemma isn't a *sufficient* condition to be a regular language.

If L is *not* regular, it still might pass the conditions of the Pumping Lemma!

An incorrect proof

THEOREM *BALANCE* is regular.

PROOF We show that *BALANCE* satisfies the condition of the Pumping Lemma. Let $p = 2$ and consider any string $s \in$ *BALANCE* such that $|s| \geq 2$. Then we can write $s = xyz$ such that $x = z = \varepsilon$ and $y = w$, so $y \neq \varepsilon$. Then for any natural number i, $xy^iz = s^i$, which has the same number of `0`s and `1`s. Since *BALANCE* passes the conditions of the Pumping Lemma, *BALANCE* is regular.

An incorrect proof

THEOREM *BALANCE* is regular.

PROOF We show that *BALANCE* satisfies the condition of the Pumping Lemma. Let $p = 2$ and consider any string $s \in BALANCE$ such that $|s| \geq 2$. Then we can write $s = xyz$ such that $x = z = \varepsilon$ and $y = w$, so $y \neq \varepsilon$. Then for any natural number i, $xy^iz = s^i$, which has the same number of `0`s and `1`s. Since *BALANCE* passes the conditions of the Pumping Lemma, *BALANCE* is regular.

You don't get to choose p!

An incorrect proof

THEOREM *BALANCE* is regular.

PROOF We show that *BALANCE* satisfies the condition of the Pumping Lemma. Let $p = 2$ and consider any string $s \in$ *BALANCE* such that $|s| \geq 2$. Then we can write $s = xyz$ such that $x = z = \varepsilon$ and $y = w$, so $y \neq \varepsilon$. Then for any natural number i, $xy^iz = s^i$, which has the same number of `0`s and `1`s. Since *BALANCE* passes the conditions of the Pumping Lemma, *BALANCE* is regular.

You don't get to choose p!

But you do need to choose a specific string…

An incorrect proof

THEOREM *BALANCE* is regular.

PROOF We show that *BALANCE* satisfies the condition of the Pumping Lemma. Let $p = 2$ and consider any string $s \in BALANCE$ such that $|s| \geq 2$. Then we can write $s = xyz$ such that $x = z = \varepsilon$ and $y = w$, so $y \neq \varepsilon$. Then for any natural number i, $xy^iz = s^i$, which has the same number of `0`s and `1`s. Since *BALANCE* passes the conditions of the Pumping Lemma, *BALANCE* is regular.

You don't get to choose p!

But you do need to choose a specific string…

And a specific value for i.

Ok – so how do we go about using the Pumping Lemma to write a proof that *BALANCE* really is nonregular?

CLAIM The language *BALANCE* = $\{w \mid n_{0}(w) = n_{1}(w)\}$ is not regular.

Let's choose the string $(01)^p$, using the pumping length p.

We need to show that it's impossible to split this string into xyz where xy^iz is in *BALANCE* for every $i \in \mathbb{N}_0$…

CLAIM The language $BALANCE = \{w \mid n_0(w) = n_1(w)\}$ is not regular.

Let's choose the string $(01)^p$, using the pumping length p.

We need to show that it's impossible to split this string into xyz where xy^iz is in $BALANCE$ for every $i \in \mathbb{N}_0$…

*But it **is** possible!* 😞

CLAIM The language *BALANCE* = $\{w \mid n_0(w) = n_1(w)\}$ is not regular.

Let's choose the string $(01)^p$, using the pumping length p.

We need to show that it's impossible to split this string into xyz where xy^iz is in *BALANCE* for every $i \in \mathbb{N}_0$…

But it ***is*** *possible!* 😞

If $x = \varepsilon$, $y = 01$, and $z = (01)^{p-1}$, then xy^iz is in *BALANCE* for every value of i; the strings we produce by pumping are still balanced.

CLAIM The language *BALANCE* = $\{w \mid n_0(w) = n_1(w)\}$ is not regular.

Let's choose the string $(01)^p$, using the pumping length p.

We need to show that it's impossible to split this string into xyz where xy^iz is in *BALANCE* for every $i \in \mathbb{N}_0$…

*But it **is** possible!* 😞

If $x = \varepsilon$, $y = 01$, and $z = (01)^{p-1}$, then xy^iz is in *BALANCE* for every value of i; the strings we produce by pumping are still balanced.

Are we out of luck? 🙁

When using the Pumping Lemma:

If your string does not succeed, try another!

CLAIM The language *BALANCE* = $\{w \mid n_0(w) = n_1(w)\}$ is not regular.

This time, let's try the string 1^p0^p.

Right?

The Pumping Lemma says that our string has to be divided so that $|xy| \leq p$ and $|y| > 0$

CLAIM The language *BALANCE* = $\{w \mid n_0(w) = n_1(w)\}$ is not regular.

This time, let's try the string 1^p0^p.

Again, we need to show that it's impossible to split this string into xyz where xy^iz is in *BALANCE* for every $i \in \mathbb{N}_0$…

Right?

The Pumping Lemma says that our string has to be divided so that $|xy| \leq p$ and $|y| > 0$

CLAIM The language *BALANCE* = $\{w \mid n_0(w) = n_1(w)\}$ is not regular.

This time, let's try the string 1^p0^p.

Again, we need to show that it's impossible to split this string into *xyz* where xy^iz is in *BALANCE* for every $i \in \mathbb{N}_0$…

But it's still possible! 😞

Right?

The Pumping Lemma says that our string has to be divided so that $|xy| \leq p$ and $|y| > 0$

CLAIM The language *BALANCE* = $\{w \mid n_0(w) = n_1(w)\}$ is not regular.

This time, let's try the string 1^p0^p.

Again, we need to show that it's impossible to split this string into *xyz* where xy^iz is in *BALANCE* for every $i \in \mathbb{N}_0$…

But it's still possible! 😞

If x and z are ε and y is 1^n0^n,
then xy^iz always has an equal number of 0s and 1s

Right?

The Pumping Lemma says that our string has to be divided so that $|xy| \leq p$ and $|y| > 0$

CLAIM The language *BALANCE* = $\{w \mid n_0(w) = n_1(w)\}$ is not regular.

This time, let's try the string 1^p0^p.

Again, we need to show that it's impossible to split this string into *xyz* where xy^iz is in *BALANCE* for every $i \in \mathbb{N}_0$…

But it's still possible! 😞

If *x* and *z* are ε and *y* is 1^n0^n, then xy^iz always has an equal number of 0s and 1s

Right?

The Pumping Lemma says that our string has to be divided so that $|xy| \leq p$ and $|y| > 0$

CLAIM The language *BALANCE* = $\{w \mid n_0(w) = n_1(w)\}$ is not regular.

This time, let's try the string 1^p0^p.

Again, we need to show that it's impossible to split this string into *xyz* where xy^iz is in *BALANCE* for every $i \in \mathbb{N}_0$…

Since $|xy| \leq p$, then *y* must consist only of 1s, so $xyyz \notin$ *BALANCE*.

Contradiction! We win! 🎉

For the full proof, see Example 1.74 in Sipser.

Remember

You only need to find *one* string for which the Pumping Lemma does not hold to prove a language is not regular.

But you must show that for *any* decomposition of that string into *xyz* the Pumping Lemma holds

This sometimes means considering several different cases.

Closure properties revisited

Recall: Closure properties

Certain operations on regular languages are guaranteed to produce regular languages.

These *closure properties* can be used to prove a language is regular.

We've seen the regular languages are closed under the ***regular operations***, i.e., those used to construct regular expressions:

Union: $L_1 \cup L_2$

Concatenation: $L_1 L_2$

Kleene star: L_1^*

We've seen the regular languages are closed under the *regular operations*, i.e., those used to construct regular expressions:

Union: $L_1 \cup L_2$

Concatenation: $L_1 L_2$

Kleene star: L_1^*

And that they're also closed under these set operations:

Complement: $\overline{L_1}$

Intersection: $L_1 \cap L_2$

Difference: $L_1 - L_2$

These closure properties can also be used to prove a language *isn't* regular!

To show that a language L is nonregular using closure properties, we do a *proof by contradiction:*

Assume L is regular.

Combine L with known regular languages using operations the regular languages are closed under.

If you produce a known nonregular language, then the assumption was wrong and L is nonregular.

Intersection

CLAIM $L = \{w \in \{\mathtt{a}, \mathtt{b}\}^* \mid n_{\mathtt{a}}(w) = n_{\mathtt{b}}(w)\}$ is nonregular.

PROOF SKETCH Assume L is regular. We know **a*b*** is a regular language because we can write it as a regular expression. Because the regular languages are closed under intersection,

$$L \cap \mathbf{a^*b^*} = \{\mathtt{a}^n\mathtt{b}^n \mid n \in \mathbb{N}_0\}$$

must be regular. However, $\{\mathtt{a}^n\mathtt{b}^n \mid n \in \mathbb{N}_0\}$ is easily proved nonregular using the Pumping Lemma – as we did last class!

Therefore, L must be nonregular.

Caveat: Direction matters!

Closure for union ($\cup$) says:

If L_1 and L_2 are regular, then $L = L_1 \cup L_2$ is regular.

What happens if we know L is a regular language?
Does that mean that L_1 and L_2 are regular also?

Closure for union (∪) says:

If L_1 and L_2 are regular, then $L = L_1 \cup L_2$ is regular.

What happens if we know L is a regular language?
Does that mean that L_1 and L_2 are regular also?

Maybe – but it's not guaranteed!

Example

We know $\mathbf{a}^+$ is regular.

Consider two cases for L_1 and L_2:

$\mathbf{a}^+ = \{a^n \mid n > 0 \text{ and } n \text{ is prime}\} \cup$
$\{a^n \mid n > 0 \text{ and } n \text{ is not prime}\}$

$\mathbf{a}^+ = L_1 \cup L_2$

Example

We know $\mathbf{a}^+$ is regular.

Consider two cases for L_1 and L_2:

$$\mathbf{a}^+ = \{a^n \mid n > 0 \text{ and } n \text{ is prime}\} \cup \{a^n \mid n > 0 \text{ and } n \text{ is not prime}\}$$

$$\mathbf{a}^+ = L_1 \cup L_2$$

Neither L_1 nor L_2 is regular!

Example

We know **a**$^+$ is regular.

Consider two cases for L_1 and L_2:

a$^+$ = $\{a^n \mid n > 0$ and n is prime$\}$ ∪
$\{a^n \mid n > 0$ and n is not prime$\}$

a$^+$ = $L_1 \cup L_2$

Neither L_1 nor L_2 is regular!

a$^+$ = $\{a^n \mid n > 0$ and n is even$\}$ ∪
$\{a^n \mid n > 0$ and n is odd$\}$

a$^+$ = $L_1 \cup L_2$

Example

We know $\mathbf{a}^+$ is regular.

Consider two cases for L_1 and L_2:

$\mathbf{a}^+ = \{a^n \mid n > 0 \text{ and } n \text{ is prime}\} \cup$
$\{a^n \mid n > 0 \text{ and } n \text{ is not prime}\}$

$\mathbf{a}^+ = L_1 \cup L_2$

Neither L_1 nor L_2 is regular!

$\mathbf{a}^+ = \{a^n \mid n > 0 \text{ and } n \text{ is even}\} \cup$
$\{a^n \mid n > 0 \text{ and } n \text{ is odd}\}$

$\mathbf{a}^+ = L_1 \cup L_2$

Both L_1 and L_2 are regular!

Where are we now?

We’ve ended up where we are by trying to answer the question “what problems can you solve with a computer?”

We defined a computer to be a DFA, which means that the problems we can solve are precisely the regular languages.

We’ve discovered several equivalent ways to think about regular languages (DFAs, NFAs, and regular expressions).

We now have a powerful intuition for these languages:

> DFAs are finite-memory computers, and regular languages correspond to the problems solvable with finite memory.

Using the Pumping Lemma, we've shown that there are languages that are *not* regular!

Does that mean these languages aren't computable?

Next

What does computation look like with unbounded memory?

What problems can you solve with unbounded-memory computers?

Acknowledgments

This lecture incorporates material from:

David Chiang, University of Notre Dame

Keith Schwarz, Stanford University

Michael Sipser, *Introduction to the Theory of Computation*