So far, we have defined "regularity" of languages
- accepted by DFAs.

We have also developed notions equivalent to regularity:
- accepted by NFAs
- denoted by regular expressions (last lecture).

We have learned tools to show regularity in earlier classes:
- closure properties:
  set-theoretic operations: $\cup$, $\cap$, $\setminus$, $\neg$
  language operations: $\cdot$, $\ast$, $^r$

Have we done problem to answer a fundamental question: show that a language $L$ is not regular?

Yes, we have - Homework 2, Problem 3

Show that there does not exist any DFA that accepts the language $L = \{ a^n b^n a^{2n} \mid n \geq 1 \}$.

- Via a contradiction argument with Pigeonhole Principle.
Now, we develop below a necessary condition for regularity - Pumping Lemma for Regular Languages.

And, we apply the lemma to disprove regularity (i.e., show that a language is not regular).

We'll state the Pumping Lemma in First-Order form (our usual quantified boolean statement), the framework in setting up the contradiction argument (to prove non-regularity) and then few examples.

Rationale for the Pumping Lemma will be given later.

Pumping Lemma ([Sip12] Theorem 1.70)

\[
\forall L \text{ regular language } \forall \Sigma \text{ over alphabet } \Sigma, \exists n \text{ positive integer} \quad \forall x \in L, |x| \geq n \Rightarrow \exists u, v, w \in \Sigma^* (z = uvw) \land |v| \geq 1) \land \forall i \geq 0 \forall w \in L (i \geq 0) \quad (i \geq 0) \quad (i \geq 0)
\]

(Recall the usual structures of & A- and E-statements)

\[
\forall \ldots (\text{see } \Rightarrow \text{see}) \quad \exists \ldots (\text{see } \wedge \text{see})
\]
Repeat the above with annotations that help the semantics of the Pumping Lemma:

A regular \( L \) is a necessary condition for a regular \( L \) that is lengthy (compared to the Pumping Lemma constant \( n \))

The following part is a necessary condition:

Imagine an adversary giving us a pumping lemma constant \( n \geq 1 \) (but we do not know its value)

Then, there exist \( u, v, w \) such that:

\( z = uvw \) is decomposed

\( |uv| \leq n \) : \( uv \) is short (relative to \( n \))

\( |v| \geq 1 \) : \( v \) is not empty

\( \forall i \geq 0 \) : \( uv^i w \) are also in \( L \)

\( i = 0 \) : \( uv^0 w = uvw \) (unpumped)

\( i \geq 2 \) : \( uv^i w = uvvw^i w \) (pumped)
Note that the Pumping Lemma is stated as a necessary condition of a regular language. How is it applied to show the non-regularity of a language $L$?

1. For a given language $L = \{x \in \Sigma^* \mid \cdots \}$, what should be our "educated guess" on the regularity of $L$? Regular or not regular?

From extensive example languages and insights learned (also, remember FAs are "finite-state" machines) finite!

2. Assume that we have educated guess that $L$ is not regular.

We prove its non-regularity by following a contradiction argument as follows.

Suppose that $L = \{x \in \Sigma^* \mid \cdots \}$ were regular.

Now, we follow the Pumping Lemma on $L$ (as $L$ was supposed to be regular), we would enjoy the truth of the necessary outcome:


that would be:
\[ \exists \epsilon \geq 1 \quad \forall z \in L \left( \begin{array}{c}
|z| \geq n \\
\left( \frac{z=uv^n}{V_{uv^n} \leq n} \wedge \exists u \exists v \exists w (|uv^n| = 1) \right)
\end{array} \right) \]

So, as we would enjoy/follow the necessary outcome,

we let \( n \geq 1 \) be the Pumping Lemma Constant

(remember, it is "existential" - given by adversary
and we do NOT know its precise value).

Then, we want to show that:
\[ \forall z \in L \left( \begin{array}{c}
|z| \geq n \\
\left( \frac{z=uv^n}{V_{uv^n} \leq n} \wedge \exists u \exists v \exists w (|uv^n| = 1) \right)
\end{array} \right) \]

is NOT TRUE (remember, we are in the process of developing a contradiction)

That is, at this point, we want to argue/show that
\[ \neg \left( \forall z \in L \left( \begin{array}{c}
|z| \geq n \\
\left( \frac{z=uv^n}{V_{uv^n} \leq n} \wedge \exists u \exists v \exists w (|uv^n| = 1) \right)
\end{array} \right) \right) \]

is TRUE!

\[ \bigcirc \]

Now, can you rewrite statement \( \bigcirc \)?
Recall what we learned in Weeks 1/2 on Common Logical Equivalences for Propositional Logic and First-Order Logic:

\[ \neg (\forall \ldots) \equiv \exists (\neg \ldots) \]
\[ \neg (\exists \ldots) \equiv \forall (\neg \ldots) \]
\[ p \rightarrow q \equiv \neg p \lor q \]
\[ \neg (p \lor q) \equiv \neg p \lor \neg q \equiv p \rightarrow \neg q \]

The above statement (1) is logically equivalent to:

\[ \exists z \in L \left( |z| \geq n \land \forall u, v, w \left( \left( z = uvw \right) \land \left( |u| \leq n \right) \land \left( |v| \geq 1 \right) \right) \Rightarrow \exists i \in 0..|w| \left( z \downarrow i \neq L \right) \right) \]

(2)

Want to argue/show that this is TRUE!

3. Now, continue to argue that statement (2) is true.

So, we need to find (existential!) one \( z \in L \) that is long (i.e., \(|z| \geq n\)).

Note: Do not commit the "rookie" mistake! As the Pumping Lemma constant \( n \) is given by the adversary, its value is unknown - existential, so, when we try to find \( z \in L \), any such candidate \( z \in L \) must be long, \(|z| \geq n\). So, do NOT try specific \( z \) like \( z = aabb \) — as \(|z| = 2 \neq n \)
4. So, we try to find \( z = \ldots \in L \) with \( |z| = \infty \geq n \).

Then, we need to show that
\[
\forall u, v, w \left( z = u v w \wedge |u v| \leq n \wedge |v| \geq 1 \Rightarrow \exists i \geq 0 \ u^i v^i w \notin L \right).
\]

That is, we need to consider ALL possible \( u, v, w \)

such that \( z = u v w \)

\( |u v| \leq n \)

\( |v| \geq 1 \)

and for each such combination \( u, v, w \),
we find (existential) \( i \geq 0 \)

\( i \) may be 0, may be 2, 3, \ldots

\( u v^i w \notin L \)

(note that, "i=1" will not work since \( uv^1 w = z \in L \))

\( \forall u, v, w, \ z = u v w \wedge (|u v| \leq n \wedge |v| \geq 1) \)

\( u = a, v = b, w = \ldots \)

find \( i \geq 0 \)

(may be \( i = 0 \))

\( u v^w = u w \notin L \)

\( u = a, v = bb, w = \ldots \)

find \( i \geq 0 \)

(may be \( i = 3 \))

\( u v^3 w = u v v v w \notin L \)
Really understand the "logical game" described above. Our textbook applies the same logics in setting up a contradiction argument. The details are given above to illustrate the important understanding of the logics behind.

Now, we consider a few examples.

Example 1. \[ L = \{ a^i b^i | i \geq 0 \} \]
\[ = \{ \varepsilon, ab, aabb, aaabbb, \ldots \} \]

Our educated guess is that \( L \) is not regular.

(any FA seems to require to remember the number of As — cannot be achieved by "finite-state"

- This is not a proof, simply an educated guess.

We prove that \( L \) is not regular by Pumping Lemma (for regularity).

Notice that the following "framework" for setting up a contradiction argument — detailed in pages 4, 5, 6, and 7 — this notes.
Suppose that \( L = \{ a^i b^i | i \geq 0 \} \) were regular.

Let \( n \geq 1 \) be the Pumppi Lemma constant.

We consider \( z = a^n b^n \in L \)

with \( |z| = n + n \geq n \).

Exercise: You may want to try another candidate \( z = a^{\frac{n}{2}} b^{\frac{n}{2}} \in L \)

with \( |z| = \frac{n}{2} + \frac{n}{2} \geq n \).

No! As we do not know the value of \( n \geq 1 \), \( \frac{n}{2} \) is not necessarily integral!

How about another candidate \( z = a^{\lfloor \frac{n}{2} \rfloor} b^{\lfloor \frac{n}{2} \rfloor} \in L \)

with \( |z| = \lfloor \frac{n}{2} \rfloor + \lfloor \frac{n}{2} \rfloor = 2 \lfloor \frac{n}{2} \rfloor \geq n \).

Try this candidate \( z \) as exercise.

Now, back to our candidate \( z = a^n b^n \in L \) with \( |z| = n + n \geq n \).

Then, we need to consider:

\[ \forall u, v, w \text{ such that } z = uvw \text{ and } |vw| \geq 1, \]

and for each combination of \( u, v, w \), find \( i \geq 0 \) \( uv^i w \notin L \).
A last-resort method to consider

\[ z = uvw \quad \text{such that} \quad \left\{ \begin{array}{l}
\forall u, v, w \quad \text{such that} \quad z = uvw \\
\wedge (uv| \leq n) \\
\wedge (v| \geq 1)
\end{array} \right. \]

consider all possible "sliding u, v, w" decomposing w

\[ z = \underbrace{uvw} \]

\[ (uv| \leq n) \quad \wedge (v| \geq 1) \]

"slide u, v, w from left to right to generate all cases"

All these possibilities can be organized into following cases
(by paying attention to the "boundary" between
\( a^n \) and \( b^n \) — that is, minding the symbols composing \( u \) and \( v \)

Case 1:

\[ \overbrace{\underbrace{a^n}_{u}}^{\text{and}} \underbrace{b^n}_{v} \]

\[ \left\{ \begin{array}{l}
u = a^{|uv|} \quad \text{with} \quad |uv| + |v| \leq n \\
v = a^{|v|} \quad \text{and} \quad |v| \geq 1 \\
w = a^{n-|uv|-|v|} b^n
\end{array} \right. \]

Now, we try \( i = 0 \):

\( uv^0w = uvw = a^{uv} a^{n-|uv|-|v|} b^n \)

\( = a^{uv} a^{n-|uv|-|v|} b^n \)

\( \neq L \)

since \( n-|uv| \neq n \)

\( \forall |v| \geq 1 \)

( unpumping \( v \) to
discard at least
one \( a \) )
Case 2

\[ \begin{array}{c|c|c}
1 & a^n & b^n \\
\hline
u & & v \\
\end{array} \]

So \( u \) is within \( a^n \) and \( v \) crosses the "mid-line".

\[ u = a^{1u} \quad \text{and} \quad v = a^{n-1u} b^\beta \]

As we are considering "\( v \) crosses ..."

so \( n-1u \geq 1 \)

and \( \beta \geq 1 \).

This consideration is NOT possible

since we must have \( |uv| \leq n \) \( (\text{here, } |uv|=n+\beta > n) \) as \( \beta \geq 1 \)

and \( |v| \geq 1 \) \( (\text{satisfied}) \)

Remaining Case 3

\[ \begin{array}{c|c|c}
1 &  & \\
\hline
u & & v \\
\end{array} \]

Not possible

So, combining Cases 1, 2, and 3

(only Case 1 is possible)

for Case 1, \( \exists i (=0) \ u v^i w \not \in L \).

Such contradiction shows that \( L \) is not regular.
How about we use the candidate string 
\[ z = a^{\frac{n}{2}} b^{\frac{n}{2}} \in L \text{ with } 121 = \frac{n}{2} + \frac{n}{2} \geq n. \]

Proceed as above, we consider 
\[ u, v, w \] such that 
\[ z = uvw \text{ with } |uv| \leq n \text{ and } |v| \geq 1 \]
and for each such combination \( u, v, w \), find a \( \bar{x} = 0 \) with \( uv^i w \notin L \).

Consider the following cases for all such \( u, v, w \):

Case 1:

\[ \begin{array}{c|c}
\frac{n}{2} & \frac{n}{2} \\
\hline
a & b
\end{array} \]

\[ u = a^{ul_1}, \quad v = a^{vl_1}, \quad w = a^{n-1ul_1-1vl_1}b^n \]

Then, check that, for \( \bar{x} = 0 \) (unpumped)

\[ uv^0w = uvw = a^{ul_1}a^{n-1ul_1-1vl_1}b^n = a^{n-1vl_1}b^n \notin L \]

as \( |vl_1| \geq 1 \)

Case 2:

\[ \begin{array}{c|c}
\frac{n}{2} & \frac{n}{2} \\
\hline
a & b
\end{array} \]

\[ u = a^{ul_1}, \quad v = a^{\beta-1ul_1}b^\beta, \quad w = b^{\frac{n}{2}-\beta} \]

with \( 1ul_1 + \frac{n}{2} - 1ul_1 + \beta \leq n \) (POSSIBLE here).

\( \bar{v} \) crosses \( \bar{x} \) at midpoint boundary.
Be careful for this case!

\[ \frac{a}{b} \quad \frac{a}{b} \]

\[ u \quad u \quad u \]

If we try \( i = 0 \) (unpumping to discard symbols of \( v \)),

\[ uv^0w = uw = a \quad b \]

MAY still be in \( L \)

(so, no contradiction!)

Think of the scenario that \( v \) may be

\[ v = a^3 b^3 \]

\( uv^0w = uw \)

lost 3 \( a \)'s and 3 \( b \)'s

but the resulting string is still in \( L \)!

But, we are "fortunate".

We try \( i = 2 \):

\[ uv^2w =uvw = a \quad a \quad b \quad a \quad b \quad b \]

remember \( \beta > 1 \)

so we have \( b \)'s preceding \( a \)s

and this string is \( \not\in L \)

as desired
Case 3

\[
\begin{array}{c|c}
\alpha & 0 \\
\hline
a & b \\
\hline
u & v
\end{array}
\]

\[
u = a^{\frac{n}{2}} b^{-1} \\
v = b^{1 \nu_1} \\
w = b^{\frac{n}{2}-\alpha-1 \nu_1}
\]

u = a^{\frac{n}{2}} b^{-1} \alpha

\frac{n}{2} + \alpha + 1 \nu_1 \leq n \quad \text{(Possible)}

| \nu_1 | \geq 1

What is \( i \) for this case?

Unpumping to discard \( v \) (and i'th symbol - \( b \))

\[
i = 0: \ uv^0w = uW = a^{\frac{n}{2}} b^{-1} b^{\frac{n}{2}-\alpha-1 \nu_1}
\]

Combining Cases 1, 2, and 3, for each case,

\[
\exists i > 0, \ uv^iw \notin L
\]

More examples next class.