Regular and Nonregular Languages

Chapter 8
Languages: Regular or Not?

$\{a^n b^n : n \geq 0\}$ is not.

$\{w \in \{a, b\}^* : \text{every } a \text{ is immediately followed by } b\}$ is regular.

$\{w \in \{a, b\}^* : \text{every } a \text{ has a matching } b \text{ somewhere}\}$ is not

- Showing that a language is regular.
- Showing that a language is not regular.
How Many Regular Languages?

Theorem: There is a countably infinite number of regular languages.

Proof:

- Upper bound on number of regular languages: number of FSMs (or regular expressions).

- Lower bound on number of regular languages:

\[ \{a\}, \{aa\}, \{aaa\}, \{aaaa\}, \{aaaaa\}, \{aaaaaa\}, \ldots \]

are all regular. That set is countably infinite.
Regular And Nonregular Languages?

There is a countably infinite number of regular languages.

There is an uncountably infinite number of languages over any nonempty alphabet $\Sigma$.

So there are *many* more nonregular languages than there are regular ones.
Showing that a Language is Regular

**Theorem:** Every finite language is regular.

**Proof:** If $L$ is the empty set, then it is defined by the regular expression $\emptyset$ and so is regular. If it is any finite language composed of the strings $s_1, s_2, \ldots s_n$ for some positive integer $n$, then it is defined by the regular expression:

$$s_1 \cup s_2 \cup \ldots \cup s_n$$

So it too is regular.
Showing that a Language is Regular

Example:

Let \( L = L_1 \cap L_2 \), where:
\[
L_1 = \{a^n b^n, \ n \geq 0\}, \text{ and}
\]
\[
L_2 = \{b^n a^n, \ n \geq 0\}
\]

\( L = \)
More Examples

- $L_1 = \{ w \in \{0 - 9\}^* : w \text{ is the social security number of the current US president}\}.$

- $L_2 = \{1 \text{ if Santa Claus exists and } 0 \text{ otherwise}\}$

- $L_3 = \{1 \text{ if God exists and } 0 \text{ otherwise}\}$

- $L_4 = \{1 \text{ if there were people in North America before 10,000 BP and } 0 \text{ otherwise}\}$

- $L_5 = \{1 \text{ if there were people in North America before 15,000 BP and } 0 \text{ otherwise}\}$

- $L_6 = \{w \in \{0 - 9\}^+: w \text{ is the decimal representation, no leading 0’ s, of a prime Fermat number}\}$
Prime Fermat Numbers

\[ \Sigma = \{0 - 9\} \]
\[ L = \{w \in \{0 - 9\}^+: w \text{ is the decimal representation, no leading } 0' \text{'s, of a prime Fermat number}\} \]

The Fermat numbers are defined by

\[ F_n = 2^{2^n} + 1, \quad n \geq 0 \]

Example elements of \( L \):

\[ F_0 = 3, \quad F_1 = 5, \quad F_2 = 17, \quad F_3 = 257, \quad F_4 = 65,537 \]

Is \( L \) regular?
Prime Fermat Numbers

It is known that all divisors of $F_n$ are of the form:

$$k \cdot 2^{n+2} + 1$$

for some $k$.

www.prothsearch.net/fermat
Finiteness - Theoretical vs. Practical

Any finite language is regular. The size of the language doesn't matter.

Parity Checking Soc. Sec. # Checking

But, from an implementation point of view, it very well may.

When is an FSM a good way to encode the facts about a language?

FSM’s are good at looking for repeating patterns. They don't bring much to the table when the language is just a set of unrelated strings.
Let $\Sigma = \{12, 13, 21, 23, 31, 32\}$.

Let $L$ be the language of strings that correspond to successful move sequences. The shortest string in $L$ has length $2^{64} - 1$.

There is an FSM that accepts $L$:
Showing That $L$ is Regular

1. Show that $L$ is finite.

2. Exhibit an FSM for $L$.

3. Exhibit a regular expression for $L$.

4. Show that the number of equivalence classes of $\approx_L$ is finite.

5. Exhibit a regular grammar for $L$.

6. Exploit the closure theorems.
Closure Properties of Regular Languages

- Union
- Concatenation
- Kleene star
- Complement
- Intersection
- Difference
- Reverse
- Letter substitution
Closure of Regular Languages Under Complement

\[ M_1 \]
Closure of Regular Languages Under Complement

What about:

On input $bbb$?
Closure of Regular Languages Under Complement

What about:

On input $a$?
A Complement Example
The Complement Procedure

Given: an FSM $M$,

Construct a new machine to accept $\neg L(M)$:

- If necessary, use $ndfsmtodfsm$ to construct $M'$, a deterministic equivalent of $M$.

- Make sure that $M'$ is described completely.

- Swap accepting and nonaccepting states.
Closure of Regular Languages Under Intersection

\[ L_1 \cap L_2 = \]

Write this in terms of operations we have already proved closure for:

- Union
- Concatenation
- Kleene star
- Complementation
Closure of Regular Languages Under Intersection

\[ L_1 \cap L_2 = \neg (\neg L_1 \cup \neg L_2) \]

Write this in terms of operations we have already proved closure for:

- Union
- Concatenation
- Kleene star
- Complementation
Closure of Regular Languages Under Difference

$L_1 - L_2 =$
Closure of Regular Languages Under Difference

$L_1 - L_2 = L_1 \cap \neg L_2$
Letter Substitution

- Let $\Sigma_1$ and $\Sigma_2$ be alphabets.
- Let $\text{sub}$ be any function from $\Sigma_1$ to $\Sigma_2^*$.

Example:

Let: $\Sigma_1 = \{a, b\}$,
$\Sigma_2 = \{0, 1\}$,

$\text{sub}(a) = 0$, and
$\text{sub}(b) = 11$. 
Letter Substitution

$letsub$ is a letter substitution function iff:

$letsub(L_1) = \{w \in \Sigma_2^* : \exists y \in L_1 \text{ and } w = y \text{ except that:}
\begin{align*}
\text{every character } c \text{ of } y \\
\text{is replaced by } sub(c)\}. 
\end{align*}$

Example:

$\begin{align*}
sub(a) &= 0, \text{ and } \\
sub(b) &= 11.
\end{align*}$

Then $letsub(\{a^n b^n, n \geq 0\}) =$
Divide-and-Conquer

Let $L = \{w \in \{a, b\}^* : w$ contains an even number of $a$’s and an odd number of $b$’s and all $a$’s come in runs of three$\}$. 

$L = L_1 \cap L_2$, where:

• $L_1 = \{w \in \{a, b\}^* : w$ contains an even number of $a$’s and an odd number of $b$’s$\}$, and

• $L_2 = \{w \in \{a, b\}^* : all$ $a$’s come in runs of three$\}$
$L_1$ is Regular

- Even $a$'s, even $b$'s
- Odd $a$'s, even $b$'s
$L_2$ is Regular
Don’t Try to Use Closure Backwards

One Closure Theorem:

If $L_1$ and $L_2$ are regular, then so is

$$L = L_1 \cap L_2$$

But if $L$ is regular, what can we say about $L_1$ and $L_2$?

$$L = L_1 \cap L_2$$
Don’t Try to Use Closure Backwards

One Closure Theorem:

If $L_1$ and $L_2$ are regular, then so is

$$L = L_1 \cap L_2$$

But if $L$ is regular, what can we say about $L_1$ and $L_2$?

$$L = L_1 \cap L_2$$

$$ab = ab \cap (a \cup b)^*$$  (they are regular)
Don’t Try to Use Closure Backwards

One Closure Theorem:

If $L_1$ and $L_2$ are regular, then so is

$$L = L_1 \cap L_2$$

But if $L$ is regular, what can we say about $L_1$ and $L_2$?

$$L = L_1 \cap L_2$$

$$ab = ab \cap (a \cup b)^*$$ (they are regular)

$$ab = ab \cap \{a^n b^n, \ n \geq 0\}$$ (they may not be regular)
Don’t Try to Use Closure Backwards

Another Closure Theorem:

If $L_1$ and $L_2$ are regular, then so is

$$L = L_1 \cup L_2$$

But if $L_2$ is not regular, what can we say about $L$?

$$L = L_1 \cap L_2$$
Don’t Try to Use Closure Backwards

Another Closure Theorem:

If $L_1$ and $L_2$ are regular, then so is

$$L = L_1 \cup L_2$$

But if $L_2$ is not regular, what can we say about $L$?

$$L = L_1 \cup L_2$$

$$\{aba^n b^n : n \geq 0\} = \{ab\} \{a^n b^n : n \geq 0\}$$
Don’t Try to Use Closure Backwards

Another Closure Theorem:

If $L_1$ and $L_2$ are regular, then so is

$$L = L_1 \cdot L_2$$

But if $L_2$ is not regular, what can we say about $L$?

$$L = L_1 \cdot L_2$$

$$\{a b a^n b^n : n \geq 0\} = \{a b\} \{a^n b^n : n \geq 0\}$$

$$\{aaa^*\} = \{a^*\} \{a^p : p \text{ is prime}\}$$
Showing that a Language is Not Regular

Every regular language can be accepted by some FSM.

It can only use a finite amount of memory to record essential properties.

Example:
\[
\{a^n b^n, \ n \geq 0\} \text{ is not regular}
\]
Showing that a Language is Not Regular

The only way to generate/accept an infinite language with a finite description is to use:

- Kleene star (in regular expressions), or
- cycles (in automata).

This forces some kind of simple repetitive cycle within the strings.

Example:

\[ ab^*a \text{ generates } \text{aba, abba, abbbba, abbbba, etc.} \]

Example:

\[ \{a^n : n \geq 1 \text{ is a prime number}\} \text{ is not regular.} \]
How Long a String Can Be Accepted?

What is the longest string that a 5-state FSM can accept?
Exploiting the Repetitive Property

If an FSM with \( n \) states accepts any string of length \( \geq n \), how many strings does it accept?

\[
L = \text{bab}^*\text{ab}
\]

\[
\begin{array}{cccccc}
\text{b} & \text{a} & \text{b} & \text{b} & \text{b} & \text{b} & \text{a} & \text{b} \\
x & y & z
\end{array}
\]

\( xy^*z \) must be in \( L \).

So \( L \) includes: \( \text{baab, babab, babbab, babbbbbbbbbab} \)
Theorem – Long Strings

**Theorem:** Let $M = (K, \Sigma, \delta, s, A)$ be any DFSM. If $M$ accepts any string of length $|K|$ or greater, then that string will force $M$ to visit some state more than once (thus traversing at least one loop).

**Proof:** $M$ must start in one of its states. Each time it reads an input character, it visits some state. So, in processing a string of length $n$, $M$ creates a total of $n + 1$ state visits. If $n + 1 > |K|$, then, by the pigeonhole principle, some state must get more than one visit. So, if $n \geq |K|$, then $M$ must visit at least one state more than once.
The Pumping Theorem for Regular Languages

If $L$ is regular, then every long string in $L$ is pumpable.

So, $\exists k \geq 1$

$$(\forall \text{ strings } w \in L, \text{ where } |w| \geq k$$

$$(\exists x, y, z \ (w = xyz, \ |xy| \leq k, \ y \neq \varepsilon, \ \text{and} \ \forall q \geq 0 \ (xy^qz \text{ is in } L))))$$.
Example: \( \{a^m b^n : n \geq 0\} \) is not Regular

\( k \) is the number from the Pumping Theorem.

Choose \( w \) to be \( a^{[k/2]} b^{[k/2]} \) ("long enough").

\[
\begin{array}{cccccccccc}
1 & a & a & a & a & a & \ldots & a & a & a & a & a & 2 \\
& x & | & b & b & b & b & \ldots & b & b & b & b & b \\
\end{array}
\]

We show that there is no \( x, y, z \) with the required properties:
\[
|xy| \leq k, \\
y \neq \varepsilon, \\
\forall \ q \geq 0 \ (xy^qz \text{ is in } L).
\]

Three cases to consider:
- \( y \) falls in region 1:
- \( y \) falls across regions 1 and 2:
- \( y \) falls in region 2:
Example: \( \{a^m b^n: n \geq 0\} \) is not Regular

\( k \) is the number from the Pumping Theorem.

Choose \( w \) to be \( a^{[k/2]}b^{[k/2]} \) (“long enough”).

\[
\begin{array}{c}
1 & 2 \\
\begin{array}{ccccccc}
a & a & a & a & a & \ldots & a \\
& a & a & a & a & a & b \\
\hline
x & y & z
\end{array}
& \begin{array}{ccccccc}
b & b & b & b & \ldots & b & b \\
& b & b & b & b & b & b
\end{array}
\end{array}
\]

While this works, there is an easier way:
Example: \( \{a^m b^n: n \geq 0\} \) is not Regular

Second try:
Choose \( w \) to be \( a^{k}b^{k} \) (We get to choose any \( w \)).

\[
\begin{array}{c|c}
1 & 2 \\
- & - \\
a a a a a a \ldots a a a a a & b b b b b \ldots b b b b b \\
x & y & z \\
\end{array}
\]

We show that there is no \( x, y, z \) with the required properties:
- \( |xy| \leq k \),
- \( y \neq \varepsilon \),
- \( \forall \ q \geq 0 \ (xy^q z \text{ is in } L) \).

Since \( |xy| \leq k \), \( y \) must be in region 1. So \( y = a^p \) for some \( p \geq 1 \).
Let \( q = 2 \), producing:
\[
a^{k+p} b^{k}
\]
which \( \notin L \), since it has more \( a \)'s than \( b \)'s.
A Complete Proof

We prove that $L = \{a^n b^n : n \geq 0\}$ is not regular.

If $L$ were regular, then there would exist some $k$ such that any string $w$ where $|w| \geq k$ must satisfy the conditions of the theorem. Let $w = a^{\lfloor k/2 \rfloor} b^{\lfloor k/2 \rfloor}$. Since $|w| \geq k$, $w$ must satisfy the conditions of the pumping theorem. So, for some $x, y, z$, $w = xyz$, $|xy| \leq k$, $y \neq \varepsilon$, and $\forall q \geq 0$, $xyqz$ is in $L$. We show that no such $x, y, z$ exist. There are 3 cases for where $y$ could occur: We divide $w$ into two regions:

\[
\text{aaaaa.....aaaaaa} \quad | \quad \text{bbbbbb.....bbbbbb}
\]

So $y$ can fall in:

- (1): $y = a^p$ for some $p$. Since $y \neq \varepsilon$, $p$ must be greater than 0. Let $q = 2$. The resulting string is $a^{k+p} b^k$. But this string is not in $L$, since it has more $a$’s than $b$’s.

- (2): $y = b^p$ for some $p$. Since $y \neq \varepsilon$, $p$ must be greater than 0. Let $q = 2$. The resulting string is $a^{k} b^{k+p}$. But this string is not in $L$, since it has more $b$’s than $a$’s.

- (1, 2): $y = a^p b^r$ for some non-zero $p$ and $r$. Let $q = 2$. The resulting string will have interleaved $a$’s and $b$’s, and so is not in $L$.

There exists one long string in $L$ for which no $x, y, z$ exist. So $L$ is not regular.
What You Need to Write

We prove that $L = \{a^n b^n : n \geq 0\}$ is not regular

Let $w = a^{[k/2]}b^{[k/2]}$. (If not completely obvious, as in this case, show that $w$ is in fact in $L$.)

There are 3 cases for where $y$ could occur:

\[
\begin{array}{c|c}
\text{aaaaa.....aaaaaa} & \text{bbbbbb.....bbbbbb} \\
1 & 2 \\
\end{array}
\]

So $y$ can fall in:

- (1): $y = a^p$ for some $p$. Since $y \neq \varepsilon$, $p$ must be greater than 0. Let $q = 2$. The resulting string is $a^{k+p}b^k$. But this string is not in $L$, since it has more $a$'s than $b$'s.

- (2): $y = b^p$ for some $p$. Since $y \neq \varepsilon$, $p$ must be greater than 0. Let $q = 2$. The resulting string is $a^k b^{k+p}$. But this string is not in $L$, since it has more $b$'s than $a$'s.

- (1, 2): $y = a^p b^r$ for some non-zero $p$ and $r$. Let $q = 2$. The resulting string will have interleaved $a$'s and $b$'s, and so is not in $L$.

Thus $L$ is not regular.
Using the Pumping Theorem

If $L$ is regular, then every “long” string in $L$ is pumpable.

To show that $L$ is not regular, we find one that isn’t.

To use the Pumping Theorem to show that a language $L$ is not regular, we must:

1. Choose a string $w$ where $|w| \geq k$. Since we do not know what $k$ is, we must state $w$ in terms of $k$.
2. Divide the possibilities for $y$ into a set of equivalence classes that can be considered together.
3. For each such class of possible $y$ values where $|xy| \leq k$ and $y \neq \varepsilon$:
   - Choose a value for $q$ such that $xy^qz$ is not in $L$. 

Bal = \{w \in \{\}, (\})^* : the parens are balanced\}
\[ \text{PalEven} = \{ww^R : w \in \{a, b\}^*\} \]
\{a^n b^m : n \geq m\}
Choose \( w = \text{aba}^k \text{b}^k \)

What are the choices for \((x, y)\):

- \((\varepsilon, a)\)
- \((\varepsilon, \text{ab})\)
- \((\varepsilon, \text{aba}^+)\)
- \((a, b)\)
- \((a, \text{ba}^+)\)
- \((\text{aba}^*, a^+)\)
A Different Approach to $aba^nb^n$?

Can we argue as follows:

We already know that $\{a^nb^n: n \geq 0\}$ is not regular. So neither is $L$.

Can we defend this argument by appeal to the fact that the regular languages are closed under concatenation:

$$L = \text{ab} \ | \ | \{a^nb^n: n \geq 0\} \ ? \text{ regular} \ | \ | \text{not regular}$$
There is a Closure Theorem that Helps

\( \{aba^n b^n : n \geq 0\} \)
\[ L = \{a^n: n \text{ is prime}\} \]

\[ L = \{w = a^n: n \text{ is prime}\} \]

Let \( w = a^j \), where \( j = \) the next prime number greater than \( k \):

\[
\begin{array}{cccccccccc}
\text{a} & \text{a} & \text{a} & \text{a} & \text{a} & \text{a} & \text{a} & \text{a} & \text{a} & \text{a} \\
\hline
x & y & z
\end{array}
\]

\(|x| + |z| \) is prime.
\(|x| + |y| + |z| \) is prime.
\(|x| + 2|y| + |z| \) is prime.
\(|x| + 3|y| + |z| \) is prime, and so forth.

But the Prime Number Theorem tells us that the primes "spread out", i.e., that the number of primes not exceeding \( x \) is asymptotic to \( x/\ln x \).
Let \( w = a^j \), where \( j \) is the smallest prime number > \( k+1 \).

\( y = a^p \) for some \( p \).

\[ \forall q \geq 0 \ (a^{|x| + |z| + q|y|} \text{ must be in } L). \text{ So } |x| + |z| + q|y| \text{ must be prime.} \]

But suppose that \( q = |x| + |z| \). Then:

\[
|x| + |z| + q|y| = |x| + |z| + (|x| + |z|) \cdot y \\
= (|x| + |z|) \cdot (1 + |y|),
\]

which is non-prime if both factors are greater than 1:
\[ L = \{a^n: n \text{ is prime}\} \]

Let \( w = a^j \), where \( j \) is the smallest prime number \( > k+1 \).

\( y = a^p \) for some \( p \).

\[ \forall q \geq 0 \ (a^{\left|x\right| + \left|z\right| + q\left|y\right|} \text{ must be in } L). \text{ So } \left|x\right| + \left|z\right| + q\cdot \left|y\right| \text{ must be prime.} \]

But suppose that \( q = \left|x\right| + \left|z\right| \). Then:

\[
\left|x\right| + \left|z\right| + q \cdot \left|y\right| = \left|x\right| + \left|z\right| + (\left|x\right| + \left|z\right|) \cdot y \\
= (\left|x\right| + \left|z\right|) \cdot (1 + \left|y\right|),
\]

which is non-prime if both factors are greater than 1:

\[
(\left|x\right| + \left|z\right|) > 1 \text{ because } \left|w\right| > k+1 \text{ and } \left|y\right| \leq k. \\
(1 + \left|y\right|) > 1 \text{ because } \left|y\right| > 0.
\]
Using the Pumping Theorem Effectively

- To choose $w$:
  - Choose a $w$ that is in the part of $L$ that makes it not regular.
  - Choose a $w$ that is only barely in $L$.
  - Choose a $w$ with as homogeneous as possible an initial region of length at least $k$.

- To choose $q$:
  - Try letting $q$ be either 0 or 2.
  - If that doesn’t work, analyze $L$ to see if there is some other specific value that will work.
Using the Closure Properties

The two most useful ones are closure under:

- Intersection
- Complement
Using the Closure Properties

$L = \{ w \in \{a, b\}^*: \#_a(w) = \#_b(w) \}$

If $L$ were regular, then:

$L' = L \cap ________$

would also be regular. But it isn’t.
$L = \{a^i b^j : i, j \geq 0 \text{ and } i \neq j\}$

Try to use the Pumping Theorem by letting $w = a^{k+1}b^k$: 
$L = \{a^i b^j: i, j \geq 0 \text{ and } i \neq j\}$

Try to use the Pumping Theorem by letting $w = a^{k!} b^{k+k!}$.

Then $y = a^p$ for some nonzero $p$.

Let $q = (k!/p) + 1$ (i.e., pump in $(k!/p)$ times).

Note that $(k!/p)$ must be an integer because $p < k$.

The number of $a$’s in the new string is $k + (k!/p)p = k + k!$.

So the new string is $a^{k+k!} b^{k+k!}$, which has equal numbers of $a$’s and $b$’s and so is not in $L$. 
\[ L = \{ a^i b^j : i, j \geq 0 \text{ and } i \neq j \} \]

An easier way:

If \( L \) is regular then so is \( \neg L \). Is it?
\[ L = \{a^i b^j : i, j \geq 0 \text{ and } i \neq j\} \]

An easier way:

If \( L \) is regular then so is \( \neg L \). Is it?

\[ \neg L = A^n B^n \cup \{\text{out of order}\} \]

If \( \neg L \) is regular, then so is \( L' = \neg L \cap a^* b^* \)

\[ = ______________ \]
\[ L = \{a^i b^j c^k : \ i, j, k \geq 0 \text{ and (if } i = 1 \text{ then } j = k)\} \]

Every string in \( L \) of length at least 1 is pumpable.

But is \( L \) regular?
$L = \{a^i b^j c^k : i, j, k \geq 0 \text{ and } (\text{if } i = 1 \text{ then } j = k)\}$

Every string in $L$ of length at least 1 is pumpable.

Rewrite the final condition as: $(i \neq 1) \text{ or } (j = k)$
$L = \{a^ib^jc^k: \ i, j, k \geq 0 \text{ and } (i \neq 1 \text{ or } j = k)\}$

Every string in $L$ of length at least 1 is pumpable:

• If $i = 0$ then: if $j \neq 0$, let $y$ be $b$; otherwise, let $y$ be $c$. Pump in or out. Then $i$ will still be 0 and thus not equal to 1, so the resulting string is in $L$.

• If $i = 1$ then: let $y$ be $a$. Pump in or out. Then $i$ will no longer equal 1, so the resulting string is in $L$.

• If $i = 2$ then: let $y$ be $aa$. Pump in or out. Then $i$ cannot equal 1, so the resulting string is in $L$.

• If $i > 2$ then: let $y = a$. Pump out once or in any number of times. Then $i$ cannot equal 1, so the resulting string is in $L$. 
$L = \{a^i b^j c^k : i, j, k \geq 0 \text{ and } (i \neq 1 \text{ or } j = k)\}$

But the closure theorems help. Suppose we guarantee that $i = 1$. If $L$ is regular, then so is:

$L' = L \cap ab^*c^*$.

$L' = \{a b^j c^k : j, k \geq 0 \text{ and } j = k\}$

Use Pumping to show that $L'$ is not regular:
$L = \{a^i b^j c^k : i, j, k \geq 0 \text{ and } (i \neq 1 \text{ or } j = k)\}$

An Alternative

If $L$ is regular, then so is $L^R$:

$L^R = \{c^k b^j a^i : i, j, k \geq 0 \text{ and } (i \neq 1 \text{ or } j = k)\}$

Use Pumping to show that $L'$ is not regular:
Exploiting Problem-Specific Knowledge

\[ L = \{w \in \{0, 1, 2, 3, 4, 5, 6, 7\}^*: \]

\[ w \text{ is the octal representation of a nonnegative integer that is divisible by 7} \]
Exploiting Problem-Specific Knowledge

Let $\Sigma = \{w, h, q, e, x, r, l\}$.
Let $L = \{w : w \text{ represents a song in 4/4 time}\}.$
Is English Regular?

Is English finite?
In the event that the Purchaser defaults in the payment of any installment of purchase price, taxes, insurance, interest, or the annual charge described elsewhere herein, or shall default in the performance of any other obligations set forth in this Contract, the Seller may: at his option: (a) Declare immediately due and payable the entire unpaid balance of purchase price, with accrued interest, taxes, and annual charge, and demand full payment thereof, and enforce conveyance of the land by termination of the contract or according to the terms hereof, in which case the Purchaser shall also be liable to the Seller for reasonable attorney's fees for services rendered by any attorney on behalf of the Seller, or (b) sell said land and premises or any part thereof at public auction, in such manner, at such time and place, upon such terms and conditions, and upon such public notice as the Seller may deem best for the interest of all concerned, consisting of advertisement in a newspaper of general circulation in the county or city in which the security property is located at least once a week for Three (3) successive weeks or for such period as applicable law may require and, in case of default of any purchaser, to re-sell with such postponement of sale or resale and upon such public notice thereof as the Seller may determine, and upon compliance by the Purchaser with the terms of sale, and upon judicial approval as may be required by law, convey said land and premises in fee simple to and at the cost of the Purchaser, who shall not be liable to see to the application of the purchase money; and from the proceeds of the sale: First to pay all proper costs and charges, including but not limited to court costs, advertising expenses, auctioneer's allowance, the expenses, if any required to correct any irregularity in the title, premium for Seller's bond, auditor's fee, attorney's fee, and all other expenses of sale occurred in and about the protection and execution of this contract, and all moneys advanced for taxes, assessments, insurance, and with interest thereon as provided herein, and all taxes due upon said land and premises at time of sale, and to retain as compensation a commission of five percent (5%) on the amount of said sale or sales; SECOND, to pay the whole amount then remaining unpaid of the principal of said contract, and interest thereon to date of payment, whether the same shall be due or not, it being understood and agreed that upon such sale before maturity of the contract the balance thereof shall be immediately due and payable; THIRD, to pay liens of record against the security property according to their priority of lien and to the extent that funds remaining in the hands of the Seller are available; and LAST, to pay the remainder of said proceeds, if any, to the vendor, his heirs, personal representatives, successors or assigns upon the delivery and surrender to the vendee of possession of the land and premises, less costs and excess of obtaining possession.
Is English Regular?

- The rat ran.
- The rat that the cat saw ran.
- The rat that the cat that the dog chased saw ran.

Let:

\[ A = \{ \text{cat, rat, dog, bird, bug, pony} \} \]
\[ V = \{ \text{ran, saw, chased, flew, sang, frolicked} \} . \]

Let \( L = \text{English} \cap \{ \text{The } A \ (\text{that the } A)^* \ V^* \ V \} \).

\[ L = \{ \text{The } A \ (\text{that the } A)^n \ V^n \ V, \ n \geq 0 \}. \]

Let \( w = \text{The cat (that the rat)}^k \text{ saw}^k \text{ ran}. \)
Poetry
The Pumping Lemma

Any regular language L has a magic number p
And any long-enough word in L has the following property:
Amongst its first p symbols is a segment you can find
Whose repetition or omission leaves x amongst its kind.

So if you find a language L which fails this acid test,
And some long word you pump becomes distinct from all the rest,
By contradiction you have shown that language L is not
A regular guy, resilient to the damage you have wrought.

But if, upon the other hand, x stays within its L,
Then either L is regular, or else you chose not well.
For w is xyz, and y cannot be null,
And y must come before p symbols have been read in full.

As mathematical postscript, an addendum to the wise:
The basic proof we outlined here does certainly generalize.
So there is a pumping lemma for all languages context-free,
Although we do not have the same for those that are r.e.

-- Martin Cohn
Defining Functions from one Language to Another

Let \( firstchars(L) = \{ w : \exists y \in L \)
\[( y = cx, \]
\[ c \in \Sigma_L, \]
\[ x \in \Sigma_L^*, \text{ and} \]
\[ w \in c^* \}\}

Are the regular languages closed under \( firstchars \)?

<table>
<thead>
<tr>
<th>( L )</th>
<th>( firstchars(L) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \emptyset )</td>
<td></td>
</tr>
<tr>
<td>( a^<em>b^</em> )</td>
<td></td>
</tr>
<tr>
<td>( ca^<em>cb^</em> )</td>
<td></td>
</tr>
</tbody>
</table>
Defining Functions from one Language to Another

Let \( \text{chop}(L) = \{ w : \exists x \in L \)
\[
( x = x_1cx_2,
\quad x_1 \in \Sigma_L^*,
\quad x_2 \in \Sigma_L^*,
\quad c \in \Sigma_L,
\quad |x_1| = |x_2|, \text{ and } \)
\quad w = x_1x_2 \} \)

Are the regular languages closed under \( \text{chop} \)?

<table>
<thead>
<tr>
<th>( L )</th>
<th>( \text{chop}(L) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \emptyset )</td>
<td></td>
</tr>
<tr>
<td>( a^<em>b^</em> )</td>
<td></td>
</tr>
<tr>
<td>( a^<em>db^</em> )</td>
<td></td>
</tr>
</tbody>
</table>
Defining Functions from One Language to Another

Let $\text{maxstring}(L) =$
\[ \{ w: w \in L, \forall z \in \Sigma^* (z \neq \varepsilon \rightarrow wz \not\in L) \} \]

Are the regular languages closed under $\text{maxstring}$?

<table>
<thead>
<tr>
<th>$L$</th>
<th>$\text{maxstring}(L)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\emptyset$</td>
<td></td>
</tr>
<tr>
<td>$a^<em>b^</em>$</td>
<td></td>
</tr>
<tr>
<td>$ab^*a$</td>
<td></td>
</tr>
<tr>
<td>$a^*b^*a$</td>
<td></td>
</tr>
</tbody>
</table>
Defining Functions from One Language to Another

Let \( \text{mix}(L) = \{ w: \exists x, y, z \) \\
( x \in L, \\
x = yz, \\
|y| = |z|, \\
w = yz^R \} \).

Are the regular languages closed under \( \text{mix} \)?

<table>
<thead>
<tr>
<th>( L )</th>
<th>( \text{mix}(L) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \emptyset )</td>
<td>( \emptyset )</td>
</tr>
<tr>
<td>( (a \cup b)^* )</td>
<td>( (a \cup b)^* )</td>
</tr>
<tr>
<td>( (ab)^* )</td>
<td>( (ab)^* )</td>
</tr>
<tr>
<td>( (ab)^<em>a(ab)^</em> )</td>
<td>( (ab)^<em>a(ab)^</em> )</td>
</tr>
</tbody>
</table>