Today we will learn...

- Void
- All
- Power
- Kleene star
- Language equivalence
The void language
Void

- The language that rejects all strings.
Void

The language that rejects all strings.

**Definition** Void $w := \text{False}$.

**Correction properties**

1. Show every word is rejected by Void
The all language
All

Language that accepts all strings
All

Language that accepts all strings

Definition \textit{All} (w:\text{word}) := True.

Correction properties

1. Show that any word is accepted by \textit{All}. 
### Exercises

Solve the following exercises

1. \( L_1 \cup \{ \epsilon \} = \)

<table>
<thead>
<tr>
<th>Coq</th>
<th>Notation</th>
<th>Math</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nil</td>
<td>{\epsilon}</td>
<td></td>
</tr>
<tr>
<td>Char c</td>
<td>c</td>
<td>{c}</td>
</tr>
<tr>
<td>Union L1 L2</td>
<td>L1 \cup L2</td>
<td>( L_1 \cup L_2 )</td>
</tr>
<tr>
<td>App L1 L2</td>
<td>L1 &gt;&gt; L2</td>
<td>( L_1 \cdot L_2 )</td>
</tr>
<tr>
<td>Void</td>
<td>\emptyset</td>
<td>( \Sigma^* )</td>
</tr>
</tbody>
</table>

- \( L_1 = \{[0], [1], [2]\} \)
- \( L_2 = \{[3], [4]\} \)
Solve the following exercises

1. $L_1 \cup \{\epsilon\} = \{[0], [1], [2], \epsilon\}$

2. $L_1 \cup L_2 = \ldots$

- $L_1 = \{[0], [1], [2]\}$
- $L_2 = \{[3], [4]\}$
Exercises

<table>
<thead>
<tr>
<th>Coq</th>
<th>Notation</th>
<th>Math</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nil</td>
<td>${\epsilon}$</td>
<td></td>
</tr>
<tr>
<td>Char $c$</td>
<td>$c$</td>
<td>${c}$</td>
</tr>
<tr>
<td>Union $L_1$ $L_2$</td>
<td>$L_1 \cup L_2$</td>
<td>$L_1 \cup L_2$</td>
</tr>
<tr>
<td>App $L_1$ $L_2$</td>
<td>$L_1 \cdot L_2$</td>
<td>$L_1 \cdot L_2$</td>
</tr>
<tr>
<td>Void</td>
<td>$\emptyset$</td>
<td>$\Sigma^*$</td>
</tr>
<tr>
<td>All</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Solve the following exercises

1. $L_1 \cup \{\epsilon\} = \{[0], [1], [2], \epsilon\}$
2. $L_1 \cup L_2 = \{[0], [1], [2], [3], [4]\}$
3. $L_1 \cdot L_2 =$

- $L_1 = \{[0], [1], [2]\}$
- $L_2 = \{[3], [4]\}$
Solve the following exercises

1. \( L_1 \cup \{\epsilon\} = \{[0], [1], [2], \epsilon\} \)
2. \( L_1 \cup L_2 = \{[0], [1], [2], [3], [4]\} \)
3. \( L_1 \cdot L_2 = \{[0, 3], [0, 4], [1, 3], [1, 4], [2, 4]\} \)
4. \( L_2 \cdot \{\epsilon\} = \)

- \( L_1 = \{[0], [1], [2]\} \)
- \( L_2 = \{[3], [4]\} \)
Solve the following exercises

1. \( L_1 \cup \{\epsilon\} = \{[0], [1], [2], \epsilon\} \)
2. \( L_1 \cup L_2 = \{[0], [1], [2], [3], [4]\} \)
3. \( L_1 \cdot L_2 = \{[0, 3], [0, 4], [1, 3], [1, 4], [2, 4]\} \)
4. \( L_2 \cdot \{\epsilon\} = L_2 \)
5. \( L_1 \cup \Sigma^* = \)
Exercises

<table>
<thead>
<tr>
<th>Coq</th>
<th>Notation</th>
<th>Math</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nil</td>
<td>{\epsilon}</td>
<td>${\epsilon}$</td>
</tr>
<tr>
<td>Char c</td>
<td>c</td>
<td>${c}$</td>
</tr>
<tr>
<td>Union L1 L2</td>
<td>L1 U L2</td>
<td>$L_1 \cup L_2$</td>
</tr>
<tr>
<td>App L1 L2</td>
<td>L1 $\cdot$ L2</td>
<td>$L_1 \cdot L_2$</td>
</tr>
<tr>
<td>Void</td>
<td>$\emptyset$</td>
<td>$\Sigma^*$</td>
</tr>
<tr>
<td>All</td>
<td>$\Sigma^*$</td>
<td>$\Sigma^*$</td>
</tr>
</tbody>
</table>

- $L_1 = \{[0], [1], [2]\}$
- $L_2 = \{[3], [4]\}$

Solve the following exercises

1. $L_1 \cup \{\epsilon\} = \{[0], [1], [2], \epsilon\}$
2. $L_1 \cup L_2 = \{[0], [1], [2], [3], [4]\}$
3. $L_1 \cdot L_2 = \{[0, 3], [0, 4], [1, 3], [1, 4], [2, 4]\}$
4. $L_2 \cdot \{\epsilon\} = L_2$
5. $L_1 \cup \Sigma^* = \Sigma^*$
6. $L_2 \cup \emptyset =$
### Exercises

<table>
<thead>
<tr>
<th>Coq</th>
<th>Notation</th>
<th>Math</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nil</td>
<td>{ε}</td>
<td>{ε}</td>
</tr>
<tr>
<td>Char c</td>
<td>c</td>
<td>{c}</td>
</tr>
<tr>
<td>Union L1 L2</td>
<td>L1 U L2</td>
<td>L₁ U L₂</td>
</tr>
<tr>
<td>App L1 L2</td>
<td>L1 &gt;&gt; L2</td>
<td>L₁ L₂</td>
</tr>
<tr>
<td>Void</td>
<td>\∅</td>
<td>\∅</td>
</tr>
<tr>
<td>All</td>
<td>\Σ*</td>
<td>\Σ*</td>
</tr>
</tbody>
</table>

- \(L₁ = \{[0], [1], [2]\}\)
- \(L₂ = \{[3], [4]\}\)

Solve the following exercises

1. \(L₁ \cup \{ε\} = \{[0], [1], [2], ε\}\)
2. \(L₁ \cup L₂ = \{[0], [1], [2], [3], [4]\}\)
3. \(L₁ \cdot L₂ = \{[0, 3], [0, 4], [1, 3], [1, 4], [2, 4]\}\)
4. \(L₂ \cdot \{ε\} = L₂\)
5. \(L₁ \cup \Sigma* = \Sigma*\)
6. \(L₂ \cup \emptyset = L₂\)
7. \(L₂ \cdot \emptyset = \)
### Exercises

<table>
<thead>
<tr>
<th>Coq</th>
<th>Notation</th>
<th>Math</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nil</td>
<td>{ε}</td>
<td></td>
</tr>
<tr>
<td>Char c</td>
<td>c</td>
<td>{c}</td>
</tr>
<tr>
<td>Union L1 L2</td>
<td>L1 U L2</td>
<td>L₁ U L₂</td>
</tr>
<tr>
<td>App L1 L2</td>
<td>L1 &gt;&gt; L2</td>
<td>L₁ ⋅ L₂</td>
</tr>
<tr>
<td>Void</td>
<td>Ø</td>
<td></td>
</tr>
<tr>
<td>All</td>
<td>Σ*</td>
<td></td>
</tr>
</tbody>
</table>

- \( L₁ = \{[0], [1], [2]\} \)
- \( L₂ = \{[3], [4]\} \)

Solve the following exercises

1. \( L₁ \cup \{ε\} = \{[0], [1], [2], ε\} \)
2. \( L₁ \cup L₂ = \{[0], [1], [2], [3], [4]\} \)
3. \( L₁ \cdot L₂ = \{[0, 3], [0, 4], [1, 3], [1, 4], [2, 4]\} \)
4. \( L₂ \cdot \{ε\} = L₂ \)
5. \( L₁ \cup Σ* = Σ* \)
6. \( L₂ \cup Ø = L₂ \)
7. \( L₂ \cdot Ø = Ø \)
The power operator for languages
The power operator for languages

- $L^{n+1} = L \cdot L^n$
- $L^0 = \{\epsilon\}$

Example

- $L = \{[0], [1], [2]\}$
- $L^0 = \{\epsilon\}$
- $L^1 = L \cdot \{\epsilon\} = L$
- $L^2 = L \cdot L = \{[0, 0], [0, 1], [0, 2], [1, 0], [1, 1], [1, 2], [2, 0], [2, 1], [2, 2]\}$
Implementing power

Inductive Pow (L:language) : nat → word → Prop :=
| pow_nil:
  Pow L 0 nil
| pow_cons:
  forall n w1 w2 w3,
  In w2 (Pow L n) →
  In w1 L →
  w3 = w1 ++ w2 →
  Pow L (S n) w3.

Rules in the form of:

\[
\begin{array}{c}
P_1 \quad P_2 \\
Q \quad P_3
\end{array}
\]

Are read as: If \( P_1 \) \textbf{and} \( P_2 \) \textbf{and} \( P_3 \) all hold, \textbf{then} we have \( Q \).
Exercise

Require Import Coq.Lists.List.
From Turing Require Import Lang.
From Turing Require Import Util.
Import Lang.Examples.
Import LangNotations.
Import ListNotations.
Open Scope lang_scope.
Open Scope char_scope.

Lemma in_aaa:
  In ["a"; "a"; "a"] (Pow "a" 3).
Proof.
Qed.

Lemma pow_char_in_inv:
  forall c n w,
  In w (Pow (Char c) n) →
  w = Util.pow1 c n.
Proof.
Qed.

CS420 ☾ Power, Kleene star, equivalence ☾ Lecture 9 ☾ Tiago Cogumbleiro
Kleene operator
Kleene operator

\[ L^* = L^0 \cup L^1 \cup L^2 \cup L^3 \cup \ldots \]

Inductive definition

\[ \frac{w \in L^n}{w \in L^*} \]

Wait, what is \( n \)?

Any \( n \) will do. If you can build a proof object such that \( w \in L^n \), then \( w \in L^* \).

Does this mean that there is only one \( n \)? Say, \( L^* = L^{1000} \)?

**NO** it does not. Each word membership will have its possibly distinct \( n \).

Example: \( L = [a] \), we have that \( \epsilon \in L^0 \) and that \([a, a] \in L^2 \), thus \( \epsilon \in L^* \) and \([a, a] \in L^* \).
Lemma in_aaa_2:
In ['a'; 'a'; 'a'] (Star 'a').
Proof.
Language Equivalence
Language equivalence (equality)

- Mathematically, we write $L_1 = L_2$ to mean that two languages are equal.
- How do you prove language equality?
Language equivalence (equality)

- Mathematically, we write $L_1 = L_2$ to mean that two languages are equal.
- How do you prove language equality?
- You have to show that all words in $L_1$ are also in $L_2$ and vice-versa.
Language equivalence in Coq

**Definition** \( \text{Equiv} \ (L1 \ L2: \text{language}) \ := \ \forall \ w, \ L1 \ w \leftrightarrow L2 \ w. \)

Show that Vowel is equivalent to previous example

**Lemma** \( \text{vowel\_eq}: \)

\[
\text{Vowel} \equiv (\text{Char} \ "a" \ U \ \text{Char} \ "e" \ U \ \text{Char} \ "i" \ U \ \text{Char} \ "o" \ U \ \text{Char} \ "u").
\]

**Proof.**
Language equivalence in Coq

**Definition**  
\[ \text{Equiv} \ (L1 \ L2: \text{language}) := \forall w, \ L1 \ w \leftrightarrow L2 \ w. \]

Show that \text{Vowel} is equivalent to previous example

**Lemma**  
\text{vowel\_eq}:  
\text{Vowel} == (\text{Char} \ "a" \ U \ \text{Char} \ "e" \ U \ \text{Char} \ "i" \ U \ \text{Char} \ "o" \ U \ \text{Char} \ "u").

**Proof.**

- \text{apply} \ \text{vowel\_iff}.
- \text{Qed}.
Exercise

Show that Void is a neutral element in union.

Lemma union_l_void:
    \forall L, L \cup \text{Void} = L.
Exercise

Show that Void is a neutral element in union.

Lemma union_l_void:
    forall L,
    L U Void == L.

Proof.
    split; intros.
    - destruct H. {
        assumption.
    }
    apply not_in_void in H.
    contradiction.
    - left.
    assumption.

Qed.
Exercise

Show that Void is an absorbing element in concatenation.

Lemma app_l_void:
  forall L, L >> Void == Void.
Exercise

Show that Void is an absorbing element in concatenation.

```
Lemma app_l_void:
   forall L,
   L ++ Void == Void.

Proof.
   unfold App; split; intros.
   destruct H as (w1, (w2, (Ha, (Hb, Hc)))).
   subst.
   apply not_in_void in Hc.
   contradiction.
   apply not_in_void in H.
   contradiction.
Qed.
```
Exercise

**A language that accepts any words that consists of two vowels**
Exercise

A language that accepts any words that consists of two vowels

Definition TwoVowels := Vowel >> Vowel.

Show that ["a"; "e"] is in TwoVowels
Exercise

A language that accepts any words that consists of two vowels

**Definition** TwoVowels := Vowel >> Vowel.

Show that ['"a"'; '"e"'] is in TwoVowels

**Goal** In ['"a"'; '"e"'] (Vowel >> Vowel).

**Proof.**
Exercise

A language that accepts any words that consists of two vowels

Definition TwoVowels := Vowel >> Vowel.

Show that ["a"; "e"] is in TwoVowels

Goal In ["a"; "e"] (Vowel >> Vowel).

Proof.

unfold App.
exists ["a"], ["e"]. (* Existential in the goal *)

split. { reflexivity. }

split. { left. reflexivity. }

right. left. reflexivity.

Qed.
Exercise

What words are accepted by $L_2$?

Definition $L_2 := \text{All} \Rightarrow \text{Char} \ "a"$. 
Exercise

Rewrite Vowels to use only language operators.
Exercise

Rewrite Vowels to use only language operators.

Definition \( \text{Vowels2} := \text{Char } "a" \ U \text{Char } "e" \ U \text{Char } "i" \ U \text{Char } "o" \ U \text{Char } "u". \)
Exercise

**Lemma vowel_length:**

\[
\text{forall } w, \\
\text{Vowel } w \rightarrow \text{length } w = 1.
\]
Exercise

Lemma vowel_length:
  \(\forall w,\) Vowel \(w\) \(\rightarrow\) length \(w = 1\).

Proof.
  intros.
  destruct \(H\) as \([H|[H|[H|[H]|H]]]]\); subst; reflexivity.
Qed.
Exercise

Goal for all $w$, $(\text{Vowel} \Rightarrow \text{Vowel}) w \Rightarrow \text{length } w = 2$. 
Exercise

Goal for all \( w \), \((\text{Vowel} \gg \text{Vowel})\) \( w \) \( \rightarrow \) length \( w = 2 \).

Proof.

intros.
unfold App in *.
destruct H as (w1, (w2, (Ha, (Hb, Hc)))). (* Existential in hypothesis *)
subst. apply vowel_length in Hb. apply vowel_length in Hc.
SearchAbout (length(_ ++ _)). (* Search for lemmas *)
rewrite app_length. rewrite Hb. rewrite Hc. reflexivity.
Qed.
Exercise

Show that all strings are rejected by Void.
Exercise

Show that all strings are rejected by Void.

Lemma not_in_void:
  \forall w,
  \neg \text{In } w \text{ Void.}

Proof.
  intros.
  intros N.
  inversion N.
Qed.