Automata Theory :: Words & Languages

Jörg Endrullis

Vrije Universiteit Amsterdam

Words

Word = finite sequence of **symbols** from an **alphabet** Σ.

- notation for symbols: $a, b, c, \ldots$
- notation for words: u, v, w, x, y, z
- $a \in \Sigma$ means a is a symbol from the alphabet Σ

Words

Word = finite sequence of **symbols** from an **alphabet** Σ.

- notation for symbols: $a, b, c, \ldots$
- notation for words: u, v, w, x, y, z
- $a \in \Sigma$ means a is a symbol from the alphabet Σ

We write λ for the **empty word**.

Words

Word = finite sequence of **symbols** from an **alphabet** Σ.

- notation for symbols: $a, b, c, \ldots$
- notation for words: u, v, w, x, y, z
- $a \in \Sigma$ means a is a symbol from the alphabet Σ

We write λ for the **empty word**.

Important: λ is not a letter of the alphabet !

Words

Word = finite sequence of **symbols** from an **alphabet** Σ.

- notation for symbols: $a, b, c, \ldots$
- notation for words: u, v, w, x, y, z
- $a \in \Sigma$ means a is a symbol from the alphabet Σ

We write λ for the **empty word**.

Important: λ is not a letter of the alphabet !

In programming, words are called **strings.**

Then λ is the empty string "" (has length 0).

Programs are Words

Everything stored on a computer is a word (a sequence of bits).

A bit can either be 0 or 1. So the alphabet is $\Sigma = \{ 0, 1 \}$.

Programs are Words

Everything stored on a computer is a word (a sequence of bits).

A bit can either be 0 or 1. So the alphabet is $\Sigma = \{ 0, 1 \}$.

So, in particular, a computer program is a **word**.

From an abstract point of view, a program

- takes a **words as input**
- produces a **word as output**

Programs are Words

Everything stored on a computer is a word (a sequence of bits).

A bit can either be 0 or 1. So the alphabet is $\Sigma = \{ 0, 1 \}$.

So, in particular, a computer program is a **word**.

From an abstract point of view, a program

- takes a **words as input**
- produces a **word as output**

A program can be given itself as input.

For instance, you can do

```
/bin/cat /bin/cat
```

in Linux.

Operations on Words

Concatenation

If $v = a_1 \cdots a_n$ and $w = b_1 \cdots b_m$, then

$$vw = a_1 \cdots a_n b_1 \cdots b_m$$

The concatenation of *abb* and *ba* is *abbba*.

Operations on Words

Concatenation

If $v = a_1 \cdots a_n$ and $w = b_1 \cdots b_m$, then

$$vw = a_1 \cdots a_n b_1 \cdots b_m$$

The concatenation of *abb* and *ba* is *abbba*.

Length

If $v = a_1 \cdots a_n$, then $|v| = n$.

The length can be defined inductively:

$$|\lambda| = 0 \qquad |va| = |v| + 1$$

The length of *abbba* is $|abbba| = 5$.

Operations on Words

Power

The power v^k consists of k concatenations of v's:

$$v^0 = \lambda \qquad v^{k+1} = v^k v$$

Let $w = aba$. Then

$$w^0 = \lambda \quad w^1 = aba \quad w^2 = abaaba \quad w^3 = abaabaaba$$

Operations on Words

Power

The power v^k consists of k concatenations of v's:

$$v^0 = \lambda \qquad\qquad v^{k+1} = v^k v$$

Let $w = aba$. Then

$$w^0 = \lambda \quad w^1 = aba \quad w^2 = abaaba \quad w^3 = abaabaaba$$

Reverse

The reverse of $a_1 \cdots a_n$ is

$$(a_1 \cdots a_n)^R = a_n \cdots a_1$$

The reverse can be inductively defined

$$\lambda^R = \lambda \qquad\qquad (va)^R = a(v^R)$$

The reverse of *abcb* is *bcba*.

Languages

Formal Languages

A **formal language** is a **set of words.**

Formal Languages

A **formal language** is a **set of words.**

A **(formal) language** L is a subset of Σ^*, that is, $L \subseteq \Sigma^*$.

Here Σ^* is the set of all words over Σ.

Formal Languages

A **formal language** is a **set of words**.

A **(formal) language** L is a subset of Σ^*, that is, $L \subseteq \Sigma^*$.

Here Σ^* is the set of all words over Σ.

The set of all parseable C programs form a language.

Formal Languages

A **formal language** is a **set of words**.

A **(formal) language** L is a subset of Σ^*, that is, $L \subseteq \Sigma^*$.

Here Σ^* is the set of all words over Σ.

The set of all parseable C programs form a language.

$\{\, ab,\ aab,\ bbaaabb \,\}$ is a finite language over $\Sigma = \{a, b\}$

Formal Languages

A **formal language** is a **set of words**.

A **(formal) language** L is a subset of Σ^*, that is, $L \subseteq \Sigma^*$.

Here Σ^* is the set of all words over Σ.

The set of all parseable C programs form a language.

$\{\, ab,\ aab,\ bbaaabb \,\}$ is a finite language over $\Sigma = \{a, b\}$

$\{\, ab^n a \mid n \geq 1 \,\}$ is an infinite language over $\Sigma = \{a, b\}$:

$$\{\, aba,\ abba,\ abbba,\ abbbba,\ \ldots \}$$

Formal Languages

A **formal language** is a **set of words.**

A **(formal) language** L is a subset of Σ^*, that is, $L \subseteq \Sigma^*$.

Here Σ^* is the set of all words over Σ.

The set of all parseable C programs form a language.

$\{\, ab,\ aab,\ bbaaabb \,\}$ is a finite language over $\Sigma = \{a, b\}$

$\{\, ab^n a \mid n \geq 1 \,\}$ is an infinite language over $\Sigma = \{a, b\}$:

$$\{\, aba,\ abba,\ abbba,\ abbbba,\ \ldots \}$$

$\{\, a^n b^n \mid n \geq 0 \,\}$ is an infinite language over $\Sigma = \{a, b\}$:

$$\{\, \lambda,\ ab,\ aabb,\ aaabbb,\ aaaabbbb,\ \ldots \}$$

Operations on Languages

Set operations

A language is a **set** of words. So the usual set operations have meaning for languages:

$$\in, \ \subseteq, \ \cap, \ \cup, \ \setminus, \ \ldots$$

Operations on Languages

Set operations

A language is a **set** of words. So the usual set operations have meaning for languages:

$$\in,\ \subseteq,\ \cap,\ \cup,\ \setminus,\ \ldots$$

$ba \in \{ a, aba, ba \}$ $\qquad$ $ab \notin \{ a, aba, ba \}$

Operations on Languages

Set operations

A language is a **set** of words. So the usual set operations have meaning for languages:

$$\in, \ \subseteq, \ \cap, \ \cup, \ \setminus, \ \ldots$$

$ba \in \{ a, aba, ba \}$ $\qquad$ $ab \notin \{ a, aba, ba \}$

$\{ a, ba \} \subseteq \{ a, aba, ba \}$ $\qquad$ $\{ a, b \} \not\subseteq \{ a, aba, ba \}$

Operations on Languages

Set operations

A language is a **set** of words. So the usual set operations have meaning for languages:

$$\in,\ \subseteq,\ \cap,\ \cup,\ \setminus,\ \ldots$$

$$ba \in \{\, a, aba, ba \,\} \qquad ab \notin \{\, a, aba, ba \,\}$$

$$\{\, a, ba \,\} \subseteq \{\, a, aba, ba \,\} \qquad \{\, a, b \,\} \not\subseteq \{\, a, aba, ba \,\}$$

$$\{\, a, aba, ba \,\} \cap \{\, a, ab, ba \,\} = \{\, a, ba \,\}$$

Operations on Languages

Set operations

A language is a **set** of words. So the usual set operations have meaning for languages:

$$\in, \ \subseteq, \ \cap, \ \cup, \ \setminus, \ \ldots$$

$ba \in \{ a, aba, ba \}$ $\qquad$ $ab \notin \{ a, aba, ba \}$

$\{ a, ba \} \subseteq \{ a, aba, ba \}$ $\qquad$ $\{ a, b \} \not\subseteq \{ a, aba, ba \}$

$$\{ a, aba, ba \} \cap \{ a, ab, ba \} = \{ a, ba \}$$

$$\{ a, aba, ba \} \cup \{ a, ab, ba \} = \{ a, ab, aba, ba \}$$

Operations on Languages

Set operations

A language is a **set** of words. So the usual set operations have meaning for languages:

$$\in, \ \subseteq, \ \cap, \ \cup, \ \setminus, \ \ldots$$

$$ba \in \{a, aba, ba\} \qquad ab \notin \{a, aba, ba\}$$

$$\{a, ba\} \subseteq \{a, aba, ba\} \qquad \{a, b\} \not\subseteq \{a, aba, ba\}$$

$$\{a, aba, ba\} \cap \{a, ab, ba\} = \{a, ba\}$$

$$\{a, aba, ba\} \cup \{a, ab, ba\} = \{a, ab, aba, ba\}$$

$$\{a, aba, ba\} \setminus \{a, ab, ba\} = \{aba\}$$

Operations on Languages

Complement

The complement $\overline{L}$ = all words that are not in the language L:

$$\overline{L} = \Sigma^* \setminus L$$

For $\Sigma = \{ a \}$ and $L = \{ a, aaa \}$. Then $\overline{L} = \{ \lambda, aa \} \cup \{ a^n \mid n \geq 4 \}$.

Operations on Languages

Complement

The complement $\overline{L}$ = all words that are not in the language L:

$$\overline{L} = \Sigma^* \setminus L$$

For $\Sigma = \{ a \}$ and $L = \{ a, aaa \}$. Then $\overline{L} = \{ \lambda, aa \} \cup \{ a^n \mid n \geq 4 \}$.

Reverse

The reverse of a language L is

$$L^R = \{ x^R \mid x \in L \}$$

The reverse of $L = \{ \lambda, ab, bbaba \}$ is $L^R = \{ \lambda, ba, ababb \}$.

Operations on Languages

Concatenation

The concatenation of languages L_1 and L_2 is defined as

$$L_1 L_2 = \{ xy \mid x \in L_1 \wedge y \in L_2 \}$$

Let $L_1 = \{ a,\ bb \}$ and $L_2 = \{ ab,\ ba \}$. Then

$$L_1 L_2 = \{ aab,\ aba,\ bbab,\ bbba \}$$

Operations on Languages

Concatenation

The concatenation of languages L_1 and L_2 is defined as

$$L_1 L_2 = \{ xy \mid x \in L_1 \wedge y \in L_2 \}$$

Let $L_1 = \{ a,\ bb \}$ and $L_2 = \{ ab,\ ba \}$. Then

$$L_1 L_2 = \{ aab,\ aba,\ bbab,\ bbba \}$$

Power

The n-th power of a language L is defined by induction on n:

$$L^0 = \{ \lambda \} \qquad L^{n+1} = L^n L \qquad (n \geq 0)$$

Let $L = \{ a,\ bb \}$. Then

$$L^2 = \{ aa, abb, bba, bbbb \}$$

$$L^3 = \{ aaa, aabb, abba, abbbb, bbaa, bbabb, bbbba, bbbbbb \}$$

Operations on Languages

Attention: $L^2 = \{uv \mid u, v \in L\} \neq \{uu \mid u \in L\}$!

Operations on Languages

Attention: $L^2 = \{uv \mid u, v \in L\} \neq \{uu \mid u \in L\}$!

Kleene star

$$L^* = \bigcup_{i=0}^{\infty} L^i = L^0 \cup L^1 \cup L^2 \cup L^3 \cup \cdots$$

$$L^+ = \bigcup_{i=1}^{\infty} L^i = L^1 \cup L^2 \cup L^3 \cup \cdots$$

Thus $L^* = L^+ \cup \{\lambda\}$.

Let $L = \{ a, bb \}$. Then

$$L^* = \{ \lambda, a, bb, aa, abb, bba, bbbb, aaa, aabb, abba, abbbb \ldots \}$$

L^* are all the words that you can build from '**building blocks**' L.

Exercise

Let

- $\Sigma = \{ a, b \}$
- $L = \{ ab^n \mid n \geq 0 \}$

Describe the following languages as sets:

$$L^R =$$

Exercise

Let

- $\Sigma = \{ a, b \}$
- $L = \{ ab^n \mid n \geq 0 \}$

Describe the following languages as sets:

$$L^R = \{ b^n a \mid n \geq 0 \}$$

Exercise

Let

- $\Sigma = \{ a, b \}$
- $L = \{ ab^n \mid n \geq 0 \}$

Describe the following languages as sets:

$$L^R = \{ b^n a \mid n \geq 0 \}$$
$$\overline{L} =$$

Exercise

Let

- $\Sigma = \{ a, b \}$
- $L = \{ ab^n \mid n \geq 0 \}$

Describe the following languages as sets:

$$L^R = \{ b^n a \mid n \geq 0 \}$$

$$\overline{L} = \{ \lambda \}$$

Exercise

Let

- $\Sigma = \{ a, b \}$
- $L = \{ ab^n \mid n \geq 0 \}$

Describe the following languages as sets:

$$L^R = \{ b^n a \mid n \geq 0 \}$$

$$\overline{L} = \{ \lambda \} \cup \{ bw \mid w \in \Sigma^* \}$$

Exercise

Let

- $\Sigma = \{ a, b \}$
- $L = \{ ab^n \mid n \geq 0 \}$

Describe the following languages as sets:

$$L^R = \{ b^n a \mid n \geq 0 \}$$

$$\overline{L} = \{ \lambda \} \cup \{ bw \mid w \in \Sigma^* \} \cup \{ awau \mid w, u \in \Sigma^* \}$$

Exercise

Let

- $\Sigma = \{ a, b \}$
- $L = \{ ab^n \mid n \geq 0 \}$

Describe the following languages as sets:

$$L^R = \{ b^n a \mid n \geq 0 \}$$

$$\overline{L} = \{ \lambda \} \cup \{ bw \mid w \in \Sigma^* \} \cup \{ awau \mid w, u \in \Sigma^* \}$$

Set notation is not ideal to describe languages.