Chapter 2: Formal Languages

In this chapter, we

- say what symbols, strings, alphabets and (formal) languages are,
- show how to use various induction principles to prove language equalities, and
- give an introduction to the Forlan toolset.

In subsequent chapters, we will study four more restricted kinds of languages: the regular (Chapter 3), context-free (Chapter 4), recursive and recursively enumerable (Chapter 5) languages.

2.1: Symbols, Strings, Alphabets and (Formal) Languages

In this section, we define the basic notions of the subject: symbols, strings, alphabets and (formal) languages.

Symbols

A *symbol* is one of the following finite sequences of ASCII characters:

- One of the digits 0–9;
- One of the upper case letters A–Z;
- One of the lower case letters a–z

Symbols

A *symbol* is one of the following finite sequences of ASCII characters:

- One of the digits 0–9;
- One of the upper case letters A–Z;
- One of the lower case letters a–z; and
- A $\langle$, followed by any finite sequence of digits, letters, commas, $\langle$ and $\rangle$, in which $\langle$ and $\rangle$ are properly nested, followed by a $\rangle$.

Symbols

A *symbol* is one of the following finite sequences of ASCII characters:

- One of the digits 0–9;
- One of the upper case letters A–Z;
- One of the lower case letters a–z; and
- A $\langle$, followed by any finite sequence of digits, letters, commas, $\langle$ and $\rangle$, in which $\langle$ and $\rangle$ are properly nested, followed by a $\rangle$.

For example, $\langle$id$\rangle$ and $\langle\langle$a,$\rangle$b$\rangle$ are symbols. On the other hand, $\langle$a$\rangle\rangle$ is not a symbol since $\langle$ and $\rangle$ are not properly nested in a$\rangle$.

Symbols

A *symbol* is one of the following finite sequences of ASCII characters:

- One of the digits 0–9;
- One of the upper case letters A–Z;
- One of the lower case letters a–z; and
- A $\langle$, followed by any finite sequence of digits, letters, commas, $\langle$ and $\rangle$, in which $\langle$ and $\rangle$ are properly nested, followed by a $\rangle$.

For example, $\langle$id$\rangle$ and $\langle\langle$a,$\rangle$b$\rangle$ are symbols. On the other hand, $\langle$a$\rangle\rangle$ is not a symbol since $\langle$ and $\rangle$ are not properly nested in a$\rangle$.

We write **Sym** for the set of all symbols. It is countably infinite.

Strings

A *string* is a list of symbols.

Strings

A *string* is a list of symbols.

We typically abbreviate the empty string $[\,]$ to $\%$, and abbreviate $[a_1, \ldots, a_n]$ to $a_1 \cdots a_n$, when $n \geq 1$.

Strings

A *string* is a list of symbols.

We typically abbreviate the empty string $[\,]$ to $\%$, and abbreviate $[a_1, \ldots, a_n]$ to $a_1 \cdots a_n$, when $n \geq 1$.

We write **Str** for **List Sym**, the set of all strings. It is countably infinite.

Strings

A *string* is a list of symbols.

We typically abbreviate the empty string $[\,]$ to $\%$, and abbreviate $[a_1, \ldots, a_n]$ to $a_1 \cdots a_n$, when $n \geq 1$.

We write **Str** for **List Sym**, the set of all strings. It is countably infinite.

Because strings are lists, we have that $|x|$ is the *length* of a string x, and that $x \mathbin{@} y$ is the *concatenation* of strings x and y.

Strings

A *string* is a list of symbols.

We typically abbreviate the empty string $[\,]$ to $\%$, and abbreviate $[a_1, \ldots, a_n]$ to $a_1 \cdots a_n$, when $n \geq 1$.

We write **Str** for **List Sym**, the set of all strings. It is countably infinite.

Because strings are lists, we have that $|x|$ is the *length* of a string x, and that $x \mathbin{@} y$ is the *concatenation* of strings x and y.

We typically abbreviate $x \mathbin{@} y$ to xy.

Strings

A *string* is a list of symbols.

We typically abbreviate the empty string $[\,]$ to $\%$, and abbreviate $[a_1, \ldots, a_n]$ to $a_1 \cdots a_n$, when $n \geq 1$.

We write **Str** for **List Sym**, the set of all strings. It is countably infinite.

Because strings are lists, we have that $|x|$ is the *length* of a string x, and that $x \mathbin{@} y$ is the *concatenation* of strings x and y.

We typically abbreviate $x \mathbin{@} y$ to xy.

Concatenation is associative: for all $x, y, z \in \mathbf{Str}$,

$$(xy)z = x(yz).$$

Strings

A *string* is a list of symbols.

We typically abbreviate the empty string $[\,]$ to $\%$, and abbreviate $[a_1, \ldots, a_n]$ to $a_1 \cdots a_n$, when $n \geq 1$.

We write **Str** for **List Sym**, the set of all strings. It is countably infinite.

Because strings are lists, we have that $|x|$ is the *length* of a string x, and that $x \mathbin{@} y$ is the *concatenation* of strings x and y.

We typically abbreviate $x \mathbin{@} y$ to xy.

Concatenation is associative: for all $x, y, z \in \mathbf{Str}$,

$$(xy)z = x(yz).$$

$\%$ is the identify for concatenation: for all $x \in \mathbf{Str}$,

$$\%x = x = x\%.$$

Raising a String to a Power

We define the string x^n resulting from *raising* a string x *to a power* $n \in \mathbb{N}$ by recursion on n:

$$x^0 = \%, \text{ for all } x \in \mathbf{Str};$$
$$x^{n+1} = xx^n, \text{ for all } x \in \mathbf{Str} \text{ and } n \in \mathbb{N}.$$

We assign this operation higher precedence than concatenation, so that xx^n means $x(x^n)$ in the above definition.

Raising a String to a Power

We define the string x^n resulting from *raising* a string x *to a power* $n \in \mathbb{N}$ by recursion on n:

$$x^0 = \%, \text{ for all } x \in \mathbf{Str};$$
$$x^{n+1} = xx^n, \text{ for all } x \in \mathbf{Str} \text{ and } n \in \mathbb{N}.$$

We assign this operation higher precedence than concatenation, so that xx^n means $x(x^n)$ in the above definition.

Proposition 2.1.1
For all $x \in \mathbf{Str}$ *and* $n, m \in \mathbb{N}$, $x^{n+m} = x^n x^m$.

Raising a String to a Power

We define the string x^n resulting from *raising* a string x *to a power* $n \in \mathbb{N}$ by recursion on n:

$$x^0 = \%, \text{ for all } x \in \mathbf{Str};$$
$$x^{n+1} = xx^n, \text{ for all } x \in \mathbf{Str} \text{ and } n \in \mathbb{N}.$$

We assign this operation higher precedence than concatenation, so that xx^n means $x(x^n)$ in the above definition.

Proposition 2.1.1
For all $x \in \mathbf{Str}$ *and* $n, m \in \mathbb{N}$, $x^{n+m} = x^n x^m$.

Proof. An easy mathematical induction on n. The string x and the natural number m can be fixed at the beginning of the proof. □

Prefixes, Suffixes and Substrings

Suppose x and y are strings. We say that:

- x is a *prefix* of y iff $y = xv$ for some $v \in \mathbf{Str}$;
- x is a *suffix* of y iff $y = ux$ for some $u \in \mathbf{Str}$;
- x is a *substring* of y iff $y = uxv$ for some $u, v \in \mathbf{Str}$.

A prefix, suffix or substring of a string other than the string itself is called *proper*.

For example:

- 12 is a of 1234;
- 234 is a of 1234;
- 23 is a of 1234.

Prefixes, Suffixes and Substrings

Suppose x and y are strings. We say that:

- x is a *prefix* of y iff $y = xv$ for some $v \in \mathbf{Str}$;
- x is a *suffix* of y iff $y = ux$ for some $u \in \mathbf{Str}$;
- x is a *substring* of y iff $y = uxv$ for some $u, v \in \mathbf{Str}$.

A prefix, suffix or substring of a string other than the string itself is called *proper*.

For example:

- 12 is a proper prefix of 1234;
- 234 is a of 1234;
- 23 is a of 1234.

Prefixes, Suffixes and Substrings

Suppose x and y are strings. We say that:

- x is a *prefix* of y iff $y = xv$ for some $v \in \mathbf{Str}$;
- x is a *suffix* of y iff $y = ux$ for some $u \in \mathbf{Str}$;
- x is a *substring* of y iff $y = uxv$ for some $u, v \in \mathbf{Str}$.

A prefix, suffix or substring of a string other than the string itself is called *proper*.

For example:

- 12 is a proper prefix of 1234;
- 234 is a proper suffix of 1234;
- 23 is a of 1234.

Prefixes, Suffixes and Substrings

Suppose x and y are strings. We say that:

- x is a *prefix* of y iff $y = xv$ for some $v \in \mathbf{Str}$;
- x is a *suffix* of y iff $y = ux$ for some $u \in \mathbf{Str}$;
- x is a *substring* of y iff $y = uxv$ for some $u, v \in \mathbf{Str}$.

A prefix, suffix or substring of a string other than the string itself is called *proper*.

For example:

- 12 is a proper prefix of 1234;
- 234 is a proper suffix of 1234;
- 23 is a proper substring of 1234.

Alphabets

An *alphabet* is a finite subset of **Sym**. We use Σ to name alphabets.

Alphabets

An *alphabet* is a finite subset of **Sym**. We use Σ to name alphabets.

We write **Alp** for the set of all alphabets. **Alp** is countably infinite.

Alphabets

An *alphabet* is a finite subset of **Sym**. We use Σ to name alphabets.

We write **Alp** for the set of all alphabets. **Alp** is countably infinite.

We define $\mathbf{alphabet} \in \mathbf{Str} \to \mathbf{Alp}$ by right recursion:

$$\begin{aligned} \mathbf{alphabet}\,\% &= \emptyset; \\ \mathbf{alphabet}(ax) &= \{a\} \cup \mathbf{alphabet}\,x, \text{ for all } a \in \mathbf{Sym} \text{ and } x \in \mathbf{Str}. \end{aligned}$$

Alphabets

An *alphabet* is a finite subset of **Sym**. We use Σ to name alphabets.

We write **Alp** for the set of all alphabets. **Alp** is countably infinite.

We define $\mathbf{alphabet} \in \mathbf{Str} \to \mathbf{Alp}$ by right recursion:

$$\begin{aligned} \mathbf{alphabet}\,\% &= \emptyset; \\ \mathbf{alphabet}(ax) &= \{a\} \cup \mathbf{alphabet}\,x, \text{ for all } a \in \mathbf{Sym} \text{ and } x \in \mathbf{Str}. \end{aligned}$$

I.e., $\mathbf{alphabet}\,w$ consists of all of the symbols occurring in the string w. E.g., $\mathbf{alphabet}(01101) = \{0, 1\}$.

Alphabets

An *alphabet* is a finite subset of **Sym**. We use Σ to name alphabets.

We write **Alp** for the set of all alphabets. **Alp** is countably infinite.

We define $\mathbf{alphabet} \in \mathbf{Str} \to \mathbf{Alp}$ by right recursion:

$$\begin{aligned} \mathbf{alphabet}\,\% &= \emptyset; \\ \mathbf{alphabet}(ax) &= \{a\} \cup \mathbf{alphabet}\,x, \text{ for all } a \in \mathbf{Sym} \text{ and } x \in \mathbf{Str}. \end{aligned}$$

I.e., $\mathbf{alphabet}\,w$ consists of all of the symbols occurring in the string w. E.g., $\mathbf{alphabet}(01101) = \{0, 1\}$.

If Σ is an alphabet, then we write Σ^* for $\mathbf{List}\,\Sigma$.

Languages

We say that L is a *language* iff $L \subseteq \Sigma^*$, for some $\Sigma \in \mathbf{Alp}$.

Languages

We say that L is a *language* iff $L \subseteq \Sigma^*$, for some $\Sigma \in \mathbf{Alp}$.

If $\Sigma \in \mathbf{Alp}$, then we say that L is a Σ-*language* iff $L \subseteq \Sigma^*$.

Languages

We say that L is a *language* iff $L \subseteq \Sigma^*$, for some $\Sigma \in \mathbf{Alp}$.

If $\Sigma \in \mathbf{Alp}$, then we say that L is a Σ-*language* iff $L \subseteq \Sigma^*$.

Here are some example languages (all are $\{0,1\}$-languages):

- $\emptyset$;
- $\{0,1\}^*$;
- $\{010, 1001, 1101\}$;
- $\{\, 0^n 1^n \mid n \in \mathbb{N} \,\}$;
- $\{\, w \in \{0,1\}^* \mid w \text{ is a palindrome} \,\}$.

Languages

We say that L is a *language* iff $L \subseteq \Sigma^*$, for some $\Sigma \in \mathbf{Alp}$.

If $\Sigma \in \mathbf{Alp}$, then we say that L is a Σ-*language* iff $L \subseteq \Sigma^*$.

Here are some example languages (all are $\{0,1\}$-languages):

- $\emptyset$;
- $\{0,1\}^*$;
- $\{010, 1001, 1101\}$;
- $\{\, 0^n 1^n \mid n \in \mathbb{N} \,\}$;
- $\{\, w \in \{0,1\}^* \mid w \text{ is a palindrome} \,\}$.

Every language is countable.

Languages

We say that L is a *language* iff $L \subseteq \Sigma^*$, for some $\Sigma \in \mathbf{Alp}$.

If $\Sigma \in \mathbf{Alp}$, then we say that L is a Σ-*language* iff $L \subseteq \Sigma^*$.

Here are some example languages (all are $\{0,1\}$-languages):

- $\emptyset$;
- $\{0,1\}^*$;
- $\{010, 1001, 1101\}$;
- $\{\, 0^n 1^n \mid n \in \mathbb{N} \,\}$;
- $\{\, w \in \{0,1\}^* \mid w \text{ is a palindrome} \,\}$.

Every language is countable.

Furthermore, Σ^* is countably infinite, as long as the alphabet Σ is

Languages

We say that L is a *language* iff $L \subseteq \Sigma^*$, for some $\Sigma \in \mathbf{Alp}$.

If $\Sigma \in \mathbf{Alp}$, then we say that L is a Σ-*language* iff $L \subseteq \Sigma^*$.

Here are some example languages (all are $\{0,1\}$-languages):

- $\emptyset$;
- $\{0,1\}^*$;
- $\{010, 1001, 1101\}$;
- $\{\, 0^n 1^n \mid n \in \mathbb{N} \,\}$;
- $\{\, w \in \{0,1\}^* \mid w \text{ is a palindrome} \,\}$.

Every language is countable.

Furthermore, Σ^* is countably infinite, as long as the alphabet Σ is nonempty. ($\emptyset^* = \{\%\}$.)

Languages (Cont.)

We write **Lan** for the set of all languages. It is uncountable: even $\mathcal{P}\,\{0\}^*$, the set of all $\{0\}$-languages, has the same size as $\mathcal{P}\,\mathbb{N}$.

Languages (Cont.)

We write **Lan** for the set of all languages. It is uncountable: even $\mathcal{P}\,\{0\}^*$, the set of all $\{0\}$-languages, has the same size as $\mathcal{P}\,\mathbb{N}$.

Given a language L, we write **alphabet** L for the alphabet

$$\bigcup \{\, \textbf{alphabet}\, w \mid w \in L \,\}.$$

Languages (Cont.)

We write **Lan** for the set of all languages. It is uncountable: even $\mathcal{P}\,\{0\}^*$, the set of all $\{0\}$-languages, has the same size as $\mathcal{P}\,\mathbb{N}$.

Given a language L, we write **alphabet** L for the alphabet

$$\bigcup \{\, \textbf{alphabet}\, w \mid w \in L \,\}.$$

For all languages L, $L \subseteq (\textbf{alphabet}\, L)^*$.

Languages (Cont.)

We write **Lan** for the set of all languages. It is uncountable: even $\mathcal{P}\,\{0\}^*$, the set of all $\{0\}$-languages, has the same size as $\mathcal{P}\,\mathbb{N}$.

Given a language L, we write **alphabet** L for the alphabet

$$\bigcup \{\, \textbf{alphabet}\, w \mid w \in L \,\}.$$

For all languages L, $L \subseteq (\textbf{alphabet}\, L)^*$.

If A is an infinite subset of **Sym** (and so is not an alphabet), we allow ourselves to write A^* for **List** A.

Languages (Cont.)

We write **Lan** for the set of all languages. It is uncountable: even $\mathcal{P}\,\{0\}^*$, the set of all $\{0\}$-languages, has the same size as $\mathcal{P}\,\mathbb{N}$.

Given a language L, we write **alphabet** L for the alphabet

$$\bigcup \{\, \textbf{alphabet}\, w \mid w \in L \,\}.$$

For all languages L, $L \subseteq (\textbf{alphabet}\, L)^*$.

If A is an infinite subset of **Sym** (and so is not an alphabet), we allow ourselves to write A^* for **List** A.

For example, $\textbf{Sym}^* =$.

Languages (Cont.)

We write **Lan** for the set of all languages. It is uncountable: even $\mathcal{P}\,\{0\}^*$, the set of all $\{0\}$-languages, has the same size as $\mathcal{P}\,\mathbb{N}$.

Given a language L, we write **alphabet** L for the alphabet

$$\bigcup \{\, \textbf{alphabet}\, w \mid w \in L \,\}.$$

For all languages L, $L \subseteq (\textbf{alphabet}\, L)^*$.

If A is an infinite subset of **Sym** (and so is not an alphabet), we allow ourselves to write A^* for **List** A.

For example, $\textbf{Sym}^* = \textbf{Str}$.