Meaning function $\mathscr{L}$

Meaning Function

Once you defined a way to describe the strings in a language it is important to define a meaning function $\mathscr{L}$ that maps syntax to semantics

- e.g. the case for numbers

- Why using a meaning function?
  - Makes clear what is syntax, what is semantics
  - Allows us to consider notation as a separate issue
  - Expressions and meanings are not 1 to 1

Warning

It should never happen that the same syntactical structure has more meanings

Meaning function $\mathscr{L}$

Meaning Function

Once you defined a way to describe the strings in a language it is important to define a meaning function $\mathscr{L}$ that maps syntax to semantics

- e.g. the case for numbers

- Why using a meaning function?
  - Makes clear what is syntax, what is semantics
  - Allows us to consider notation as a separate issue
  - Expressions and meanings are not 1 to 1

Warning

It should never happen that the same syntactical structure has more meanings

Meaning function $\mathscr{L}$

Meaning Function

Once you defined a way to describe the strings in a language it is important to define a meaning function $\mathscr{L}$ that maps syntax to semantics

- e.g. the case for numbers

- Why using a meaning function?
  - Makes clear what is syntax, what is semantics
  - Allows us to consider notation as a separate issue
  - Expressions and meanings are not 1 to 1

Warning

It should never happen that the same syntactical structure has more meanings

ToC

Languages

We need to define which is the set of strings in any token class. Therefore we need to choose the right mechanisms to describe such sets:

- Reducing at minimum the complexity needed to recognise lexemes
- Identifying effective and simple ways to describe the patterns

- Regular languages seem to be enough powerful to define all the lexemes in any token class
- Regular expressions are a suitable way to syntactically identify strings belonging to a regular language

Strings

Parts of a string

Terms related to stings:

- a prefix of a string *s* is the string obtained removing zero or more characters from the end of *s*
- a suffix of a string *s* is the string obtained removing zero or more characters from the beginning of *s*
- a substring of a string *s* is obtained deleting any prefix and any suffix from *s*
- proper prefixes, suffixes and substrings of a string *s* are those prefixes, suffixes and substrings of *s*, respectively, that are not empty (ϵ) or not equal to *s* itself
- a subsequence of a string *s* is any string formed by deleting zero or more not necessarily consecutive positions of *s*

Regular expressions (regexp): Syntax

To form a syntactically correct regexp we have the following rules:

- Single character: $'c'$ is a regexp for each $c \in \Sigma$;
- Epsilon: ϵ is a regexp;
- Union: $a + b$ is a regexp if a and b are regexps (also written $a|b$);
- Concatenation: $a \cdot b$ is a regexps if a and b are regexps (also written ab);
- Iteration (Kleene star): a^* is a regexp if a is a regexp;
- Brackets: (a) is a regexp if a is a regexp

Regular expressions (regexp): Syntax

To avoid too much brackets we fix the following precedence and associativity rules:

- $*$ has the highest precedence and is left associative
- $\cdot$ has the second highest precedence and is left associative
- $+$ has the lowest precedence and is left associative
- e.g., $a + bc^*$ means $a + (b(c^*))$; $abc + d + e$ means $(((ab)c) + d) + e$; ...

Moreover we will use the following shorthands:

- At least one: $a^+ \equiv aa^*$
- Option: $a? \equiv a + \epsilon$
- Range: $[a - z] \equiv {'a'} + {'b'} + \cdots + {'z'}$
- Excluded range: $[^\wedge a - z] \equiv$ complement of $[a - z]$

Meaning function $\mathscr{L}$

- The meaning function $\mathscr{L}$ maps syntax to semantics: $\mathscr{L}(e) = \mathscr{M}$ where e is a regexp and $\mathscr{M}$ is a set of strings

Given an alphabet Σ and regular expressions a and b over Σ:

- $\mathscr{L}(\epsilon) = \{\epsilon\}$
- $\mathscr{L}('c') = \{c\}$, where $c \in \Sigma$
- $\mathscr{L}(a + b) = \mathscr{L}(a) \cup \mathscr{L}(b)$
- $\mathscr{L}(ab) = \mathscr{L}(a) \odot \mathscr{L}(b)$
- $\mathscr{L}(a^*) = \bigcup_{i \geq 0} \mathscr{L}(a)^i$ where $\begin{cases} \mathscr{L}(a)^0 = \{\epsilon\} \\ \mathscr{L}(a)^i = \mathscr{L}(a) \odot \mathscr{L}(a)^{i-1} \end{cases}$

$\odot$ is the concatenation of languages:

$$L_1 \odot L_2 = \{s_1 s_2 \mid s_1 \in L_1 \wedge s_2 \in L_2\}$$

Some equivalence laws for regexps

Given regexps e_1 and e_2, they are equivalent, written $e_1 \equiv e_2$, if and only if $\mathscr{L}(e_1) = \mathscr{L}(e_2)$

Let a, b, c be regexps, then:

$a + b \equiv b + a$	$+$ is commutative
$a + (b + c) \equiv (a + b) + c$	$+$ is associative
$a + a \equiv a$	$+$ is idempotent
$a(bc) \equiv (ab)c$	$\cdot$ is associative
$a(b + c) \equiv ab + bc$	$\cdot$ distributes over $+$ on the left
$(a + b)c \equiv ac + bc$	$\cdot$ distributes over $+$ on the right
$a\epsilon \equiv \epsilon a \equiv a$	ϵ is the identity for $\cdot$
$(\epsilon + a)^* \equiv a^*$	ϵ is guaranteed in a closure
$a^{**} \equiv a^*$	the Kleene star is idempotent

Regular Languages

Semantics of Regular Expressions

Regular expressions (syntax)
specify regular languages (semantics)

A language L is regular if and only if there exists a regular expression e such that $\mathscr{L}(e) = L$

Closure Properties of Regular Languages

Regular languages are closed with respect to union, intersection, complement

If L_1 and L_2 are regular languages then $L_1 \cup L_2$, $L_1 \cap L_2$ and L_1^c are regular languages

Exercise

Consider $\Sigma = \{0, 1\}$. What are the sets defined by the following REs?

- 1^*
- $(1 + 0)1$
- $0^* + 1^*$
- $(0 + 1)^*$

Exercise

Given the regular language identified by $(0 + 1)^*1(0 + 1)^*$ which are the regular expressions identifying the same language among the following one:

- $(01 + 11)^*(0 + 1)^*$
- $(0 + 1)^*(10 + 11 + 1)(0 + 1)^*$
- $(1 + 0)^*1(1 + 0)^*$
- $(0 + 1)^*(0 + 1)(0 + 1)^*$

Exercise

Consider $\Sigma = \{0, 1\}$. What are the sets defined by the following REs?

- 1^*
- $(1 + 0)1$
- $0^* + 1^*$
- $(0 + 1)^*$

Exercise

Given the regular language identified by $(0 + 1)^*1(0 + 1)^*$ which are the regular expressions identifying the same language among the following one:

- $(01 + 11)^*(0 + 1)^*$
- $(0 + 1)^*(10 + 11 + 1)(0 + 1)^*$
- $(1 + 0)^*1(1 + 0)^*$
- $(0 + 1)^*(0 + 1)(0 + 1)^*$

Exercise

Choose the regular languages that are correct specifications of the following English-language description:

Twelve-hour times of the form “04:13PM”. Minutes should always be a two digit number, but hours may be a single digit

- $(0+1)?[0-9]:[0-5][0-9](AM+PM)$
- $((0+\epsilon)[0-9]+1[0-2]):[0-5][0-9](AM+PM)$
- $(0^*[0-9]+1[0-2]):[0-5][0-9](AM+PM)$
- $(0?[0-9]+1(0+1+2):[0-5][0-9](A+P)M$

Exercise

Describe the languages denoted by the following RegExp:

- $a(a|b)^*a$
- $a^*ba^*ba^*ba^*$
- $((\epsilon|a)b^*)^*$