

Reading Books

- ◆ Introduction to Automata Theory, Languages, and Computation: John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman
- ◆ Introduction to the Theory of Computation-Michael Sipser
- ◆ Elements of the Theory of Computation-Harry R. Lewis, Christos H. Papadimitriov

Alphabets

- ◆ An *alphabet* is a finite, nonempty set of symbols.
- ◆ Σ : alphabet
- ◆ Common alphabets include:
 1. $\Sigma = \{0, 1\}$, the binary alphabet
 2. $\Sigma = \{a, b, \dots, z\}$, the set of all lower-case letters
 3. The set of all ASCII characters, or the set of all printable ASCII characters

Strings

- ◆ A *string* (*word*) is a finite sequence of symbols chosen from some alphabet.
- ◆ 01101 is a string from the binary alphabet $\Sigma = \{0, 1\}$
- ◆ The string 111 is another string chosen from this alphabet

The Empty String

- ◆ The *empty string* is the string with zero occurrences of symbols.
- ◆ This string, denoted ϵ , is a string that may be chosen from any alphabet whatsoever.

Length of a String

- ◆ It is often useful to classify strings by their *length*: the number of positions for symbols in the string.
- ◆ Number of positions
- ◆ $|011| = 3$ and $|\varepsilon| = 0$

Powers of an Alphabet

◆ If Σ is an alphabet, express the set of all strings of a certain length from that alphabet by using an exponential notation

◆ Σ^k : set of strings of length k , each of whose symbols is in Σ

◆ **Example:** $\Sigma^0 = \{\epsilon\}$ length = 0.

$\Sigma = \{0, 1\} \Rightarrow \Sigma^1 = \{0, 1\}, \Sigma^2 = \{00, 01, 10, 11\}$

$\Sigma^3 = \{000, 001, 010, 011, 100, 101, 110, 111\}$

Σ : alphabet, members 0, 1 are symbols

Σ^1 : a set of strings, members 0, 1 are strings⁷

Powers of an Alphabet

◆ The set of all strings over an alphabet Σ is conventionally denoted Σ^*

◆ $\{0, 1\}^* = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, \dots\}$

◆ Another way:

$$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \dots$$

Sometimes, exclude empty string. Set of non-empty strings from alphabet Σ is denoted Σ^+ .

$$\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \Sigma^4 \cup \dots$$

$$\Sigma^* = \Sigma^+ \cup \{\epsilon\}$$

Concatenation of Strings

- ◆ Let x & y be strings
- ◆ Then xy denotes the concatenation of x & y
-that is the string formed by making a *copy of x and following it by a copy of y*
- ◆ If x is the string composed of i symbols $x = a_1a_2\dots a_i$ and y is the string composed of j symbols $y = b_1b_2\dots b_j$, then xy is the string of length $i+j$: $xy = a_1a_2\dots a_ib_1b_2\dots b_j$

Concatenation of Strings

◆ Example:

$x = 01101$, $y = 110$, $xy = ?$

$xy = 01101110$, $yx = ?$

$yx = 11001101$

For any string w , the equations $\epsilon w = w = w \epsilon$ hold

ϵ : the *identity for concatenations* since when concatenated with any string it yields the other strings as a result

Languages

- ◆ A set of strings all of which are chosen from some Σ^* , [where Σ is a particular alphabet] is called a language
- ◆ If Σ is alphabet, $L \subseteq \Sigma^*$, then L is a language over Σ .
- ◆ Common languages can be viewed as sets of strings.
- English: collection of legal English words is a set of strings over the alphabet that consists of all the letters

Languages

◆ C: the legal programs are a subset of the possible strings that can be formed from the alphabet of the language. This subset of the ASCII characters

◆ Many other languages:

1. The language of all strings consisting of n 0's followed by n 1's for some

$n \geq 0: \{\epsilon, 01, 0011, 000111, \dots\}$

2. The set of strings of 0's & 1's with an equal number of each:

$\{\epsilon, 01, 10, 0011, 0101, 1001, \dots\}$

Languages

3. The set of binary numbers whose value is a prime:

$\{10, 11, 101, 111, 1011, \dots\}$

4. Σ^* is a language for any alphabet Σ

5. \emptyset the empty language, is a language over any alphabet

6. $\{\epsilon\}$ the language consisting of only the empty string, is also a language over any alphabet

-important constraint on what can be a language is that all alphabets are finite

Problems

- ◆ A problem is the question of deciding whether a given string is a member of some particular language
- ◆ If Σ is an alphabet, & L is a language over Σ , problem L is:
 - Given a string w is Σ^* , decide whether or not w is in L

Set-Formers as a way to define languages

- ◆ It is common to describe a language using a “set-former”:

$\{w \mid \text{something about } w\}$

- read “the set of words w such that (whatever is said about w to the right of the vertical bar)”
- $\{w \mid w \text{ consists of an equal number of } 0\text{'s \& } 1\text{'s}\}$
- $\{w \mid w \text{ is a binary integer that is prime}\}$
- $\{w \mid w \text{ is a syntactically correct C program}\}$

Set-Formers as a way to define languages

- ◆ It is common to replace w by some expression with parameters & describe the strings in the language by stating conditions on the parameters.
- ◆ Ex: first with parameter n , the second with i & j
 1. $\{0^n 1^n \mid n \geq 1\}$. Read "the set of 0 to the n 1 to the n such that n is greater than or equal to 1"
-this language consists of strings $\{01, 0011, 000111, \dots\}$. As with alphabets, we can raise a single symbol to a power n in order to represent n copies of that symbol

Set-Formers as a way to define languages

2. $\{0^i1^j \mid 0 \leq i \leq j\}$. This language consists of strings with some 0's (possibly none) followed by at least as many 1's

Example: Protocol for Sending Data

