Automata Theory and Languages

SITE : http://www.info.univ-tours.fr/˜mirian/

Introduction to Automata Theory

- Automata theory : **the study of abstract computing devices, or ”machines”**
- Before computers (1930), **A. Turing** studied an abstract machine (*Turing machine*) that had all the capabilities of today’ s computers (concerning what they could compute). His goal was to describe precisely the boundary between what a computing machine could do and what it could not do.
- Simpler kinds of machines (**finite automata**) were studied by a number of researchers and **useful for a variety of purposes**.
- Theoretical developments bear directly on what computer scientists do today
  - Finite automata, formal grammars: design/ construction of software
  - Turing machines: help us understand what we can expect from a software
  - Theory of intractable problems: are we likely to be able to write a program to solve a given problem? Or we should try an approximation, a heuristic...

Why Study Automata Theory?

Finite automata are a useful model for many important kinds of software and hardware:

1. Software for designing and checking the behaviour of digital circuits
2. The lexical analyser of a typical compiler, that is, the compiler component that breaks the input text into logical units
3. Software for scanning large bodies of text, such as collections of Web pages, to find occurrences of words, phrases or other patterns
4. Software for verifying systems of all types that have a finite number of distinct states, such as communications protocols of protocols for secure exchange information

The Central Concepts of Automata Theory

Alphabet

- A finite, nonempty set of symbols.
- Symbol: Σ
- Examples:
  - The binary alphabet: $\Sigma = \{0, 1\}$
  - The set of all lower-case letters: $\Sigma = \{a, b, \ldots, z\}$
  - The set of all ASCII characters

Strings

- A **string** (or sometimes a **word**) is a finite sequence of symbols chosen from some alphabet
- Example: 01101 and 111 are strings from the binary alphabet $\Sigma = \{0, 1\}$
- **Empty string**: the string with zero occurrences of symbols
  This string is denoted by ϵ and may be chosen from any alphabet whatsoever.
- **Length of a string**: the number of positions for symbols in the string
  Example: 01101 has length 5
  • There are only two symbols (0 and 1) in the string 01101, but 5 positions for symbols
- Notation of length of w: $|w|$
  Example: $|011| = 3$ and $|\epsilon| = 0$

Powers of an alphabet (1)

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

- Σ^k: the set of strings of length k, each of whose is in Σ
- Examples:
  - $\Sigma^0 : \{\epsilon\}$, regardless of what alphabet Σ is. That is ϵ is the only string of length 0
  - If $\Sigma = \{0, 1\}$, then:
    1. $\Sigma^1 = \{0, 1\}$
    2. $\Sigma^2 = \{00, 01, 10, 11\}$
    3. $\Sigma^3 = \{000, 001, 010, 011, 100, 101, 110, 111\}$

    Note: confusion between Σ and Σ^1:
    1. Σ is an alphabet; its members 0 and 1 are symbols
    2. Σ^1 is a set of strings; its members are strings (each one of length 1)

Kleen star

- Σ^*: The set of all strings over an alphabet Σ
  - $\{0,1\}^* = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, \ldots\}$
  - $\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \ldots$
- The symbol $*$ is called **Kleene star** and is named after the mathematician and logician Stephen Cole Kleene.
- $\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \ldots$
  Thus: $\Sigma^* = \Sigma^+ \cup \{\epsilon\}$

Concatenation

- Define the binary operation . called **concatenation** on Σ^* as follows:
  If $a_1 a_2 a_3 \ldots a_n$ and $b_1 b_2 \ldots b_m$ are in Σ^*, then

  $$a_1 a_2 a_3 \ldots a_n . b_1 b_2 \ldots b_m = a_1 a_2 a_3 \ldots a_n b_1 b_2 \ldots b_m$$

- Thus, strings can be concatenated yielding another string:
  If x are y be strings then $x.y$ denotes the concatenation of x and y, that is, the string formed by making a copy of x and following it by a copy of y

- Examples:

  1. $x = 01101$ and $y = 110$
     Then $xy = 01101110$ and $yx = 11001101$
  2. For any string w, the equations $\epsilon w = w\epsilon = w$ hold.
     That is, ϵ is the **identity for concatenation** (when concatenated with any string it yields the other string as a result)

- If S and T are subsets of Σ^*, then

  $$S.T = \{s.t \mid s \in S, t \in T\}$$

Languages

- If Σ is an alphabet, and $L \subseteq \Sigma^*$, then L is a (formal) **language** over Σ.
- Language: A (possibly infinite) set of strings all of which are chosen from some Σ^*
- A language over Σ need not include strings with all symbols of Σ
  Thus, a language over Σ is also a language over any alphabet that is a superset of Σ
- Examples:
  - Programming language C
    Legal programs are a subset of the possible strings that can be formed from the alphabet of the language (a subset of ASCII characters)
  - English or French

Other language examples

1. The language of all strings consisting of n 0s followed by n 1s ($n \geq 0$):

$$\{\epsilon, 01, 0011, 000111, \ldots\}$$

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

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

3. Σ^* is a language for any alphabet Σ
4. $\emptyset$, the empty language, is a language over any alphabet
5. $\{\epsilon\}$, the language consisting of only the empty string, is also a language over any alphabet
   NOTE: $\emptyset \neq \{\epsilon\}$ since $\emptyset$ has no strings and $\{\epsilon\}$ has one
6. $\{w \mid w$ consists of an equal number of 0 and $1\}$
7. $\{0^n 1^n \mid n \geq 1\}$
8. $\{0^i 1^j \mid 0 \leq i \leq j\}$

Important operators on languages: Union

The **union** of two languages L and M, denoted $L \cup M$, is the set of strings that are in either L, or M, or both.

Example

If $L = \{001, 10, 111\}$ and $M = \{\epsilon, 001\}$ then

$$L \cup M = \{\epsilon, 001, 10, 111\}$$

Important operators on languages:

Concatenation

The **concatenation** of languages L and M, denoted $L.M$ or just LM , is the set of strings that can be formed by taking any string in L and concatenating it with any string in M.

Example

If $L = \{001, 10, 111\}$ and $M = \{\epsilon, 001\}$ then

$$L.M = \{001, 10, 111, 001001, 10001, 111001\}$$

Important operators on languages: Closure

The **closure** of a language L is denoted L^* and represents the set of those strings that can be formed by taking any number of strings from L, possibly with repetitions (*i.e.*, the same string may be selected more than once) and concatenating all of them.

Examples:

- If $L = \{0, 1\}$ then L^* is all strings of 0 and 1
- If $L = \{0, 11\}$ then L^* consists of strings of 0 and 1 such that the 1 come in pairs, *e.g.*, $011, 11110$ and ϵ. But not 01011 or 101.

Formally, L^* is the infinite union $\bigcup_{i \geq 0} L^i$ where $L^0 = \{\epsilon\}$, $L^1 = L$, and for $i > 1$ we have $L^i = LL \ldots L$ (the concatenation of i copies of L).

Regular Expressions

Regular Expressions and Languages

We define the regular expressions recursively.

Basis: The basis consists of three parts:

1. The constants ϵ and $\emptyset$ are regular expressions, denoting the language $\{\epsilon\}$ and $\emptyset$, respectively. That is $L(\epsilon) = \{\epsilon\}$ and $L(\emptyset) = \emptyset$.
2. If a is a symbol, then **a** is a regular expression. This expression denotes the language $\{a\}$, *i.e.*, $L(\mathbf{a}) = \{a\}$.
   NOTE: We use boldface font to denote an expression corresponding to a symbol
3. A variable, usually capitalised and italic such as L, is a variable, representing any language.

Regular Expressions and Languages

Induction: There are four parts to the inductive step, one for each of the three operators and one for the introduction of parentheses

1. If E and F are regular expressions, then $E + F$ is a regular expression denoting the union of $L(E)$ and $L(F)$. That is, $L(E + F) = L(E) \cup L(F)$.
2. If E and F are regular expressions, then EF is a regular expression denoting the concatenation of $L(E)$ and $L(F)$. That is, $L(EF) = L(E)L(F)$.
3. If E is a regular expression, then E^* is a regular expression denoting the closure of $L(E)$. That is, $L(E^*) = (L(E))^*$.
4. If E is a regular expression, then (E) is a regular expression denoting the same as E. Formally, $L((E)) = (L(E))$.

The use of regular expressions: examples of applications

- Definition of lexical analysers (compilers).
- Used in operation systems like UNIX (a Unix-style):

  ```
  [A-Z] [a-z]* [ ] [A-Z] [A-Z]
  ```

  represents capitalised words followed by a space and two capital letters. This expressions represents patterns in text that could be a city and a state, *e.g.*, Ithaca NY.
  It misses multi-word city names such as Palo Alto CA