CMSC 141 Automata and Language Theory

Regular Languages

Mark Froilan B. Tandoc
Instructor 1
ICS UPLB

August 15, 2014

Automata and Language Theory

Automaton

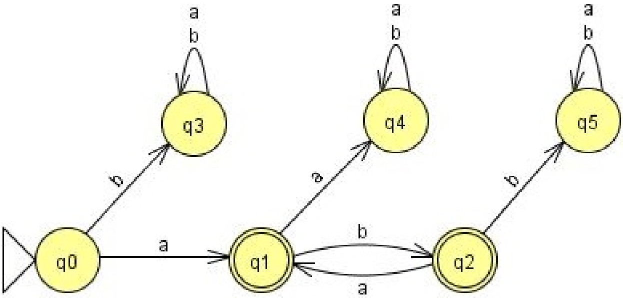
a theoretical machine capable of computation (string processing)

Formal Language

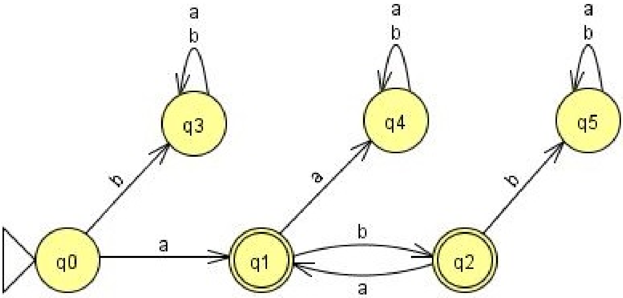
a set of strings (set can be finite, but most interesting languages are infinite)

Example

$L = \{a, ab, aba, abab, ababa, ...\}$
$L =$ set of strings x over the alphabet $\Sigma = \{a, b\}$, such that x starts with a and alternates with b.



Finite Automaton



- This finite automaton (finite state machine) accepts all strings in L and rejects all others
- The machine has a finite number of nodes, but accepts an infinite number of strings

Why study automata?

- Working with abstract machines and languages train the mind (will train the mind in creating algorithms)
- Automata are essential and a good place to start in studying the limitations of computation
- Applications in various computer science topics like compiler design, natural language processing, fractal graphics, etc.

Alphabets and Strings

Alphabet

A finite, non-empty set of symbols. Conventionally, we use the symbol Σ for an alphabet.

- $\Sigma = \{0, 1\}$
- $\Sigma = \{0, 1, 2, ..., 8, 9\}$
- $\Sigma = \{a, b, c, ..., x, y, z\}$
- $\Sigma = \mathit{ASCII\ character\ set}$

Alphabets and Strings

Strings

A finite sequence of symbols over some alphabet
Example strings from the binary alphabet
$\Sigma = \{0, 1\}$

- 111
- 01
- 1
- 1010
- 111001

Alphabets and Strings

The empty string

We represent an empty string by ϵ. Other references use λ.

Length of a string

The number of symbols in a string.
Standard notation for the length of string w is $|w|$

- $|101| = 3$
- $|1101| = 4$
- $|\epsilon| = 0$

Alphabets and Strings

Powers of an alphabet

The set of all strings of a certain length from an alphabet Σ can be expressed using an exponent notation.

If $\Sigma = \{0, 1\}$ then

- $\Sigma^0 = \{\epsilon\}$
  - This applies to any alphabet
- $\Sigma^1 = \{0, 1\}$
- $\Sigma^2 = \{00, 01, 10, 11\}$

The set of all strings over an alphabet Σ is denoted by Σ^*.

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

Alphabets and Strings

String concatenation

If $x = abb$ and $y = ab$ then $xy = abbab$

For any string w, $w = w\epsilon = \epsilon w$ holds since ϵ is the *identity* for string concatenation.

Self-concatenation

Self-concat uses exponent notation

If $x = abb$ then

- $x^0 = \epsilon$
  - This applies to any string
- $x^1 = abb$
- $x^2 = abbabb$
- $x^3 = abbabbabb$

Alphabets and Strings

Given $w = xyz$

- x is a prefix of w
- y is a substring of w
- z is a suffix of w

- Is concatenation commutative?
  - Is $xy = yx$ for all strings x and y?
- Is concatenation associative?
  - Is $(xy)z = x(yz)$ for all strings x, y and z?

Languages

Formal language

A formal language is a set of strings over some alphabet

- $\Sigma^* = \{\epsilon, 0, 1, 00, 01, 10, 11, ...\}$ over $\Sigma = \{0, 1\}$
- $ID = \{x \in \Sigma^* : x$ starts with a letter followed by 0 or more letters or digits$\}$ over $\Sigma = \{a..z, 0..9\}$
- $INT = \{x \in \Sigma^* : x$ is a sequence of one or more digits, prefixed by an optional $+$ or - sign$\}$ over $\Sigma = \{0..9, +, -\}$

Languages

Set-Formers

A language can be described using a set former:
$\{w|$ something about $w\}$

- $\{w|w$ consists of an equal number of 0's and 1's $\}$
- $\{w|w$ is a binary integer that is a prime $\}$
- $\{0^n1^n|n \geq 1\}$

Operations on Languages

(Most) set operations are applicable to languages since it is a *set* of strings, along with other string-specific operations.

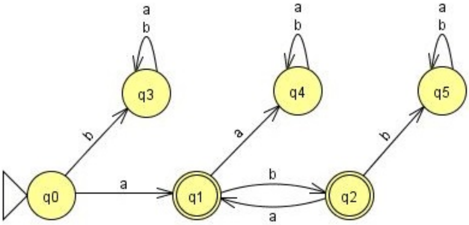
- Union $\Rightarrow L_1 \cup L_2$ (also denoted as $L_1 + L_2$)
- Intersection $\Rightarrow L_1 \cap L_2$
- Concatenation $\Rightarrow L_1 L_2 = \{xy : x \in L_1, y \in L_2\}$
- Kleene Closure $\Rightarrow L^* = L^0 \cup L^1 \cup L^2 \cup \ldots$
- Complement $\Rightarrow \Sigma^* - L$

Operations on Languages

Examples

- Concatenation:
  - $\{0, 1\}\{0, 1\} = \{00, 01, 10, 11\}$
- Kleene Closure
  - $\{0, 1\}^* = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, ...\}$
  - $\{a, b, c\}^* =$
    $\{\epsilon, a, b, c, aa, ab, ac, ba, bb, bc, ca, cb, cc, ...\}$
  - $\{a, ab, abb\}^* =$
    $\{\epsilon, a, ab, abb, aa, aab, aabb, aba, abab, ...\}$

Finite Automata



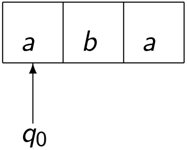
- Finite automata can describe Regular Languages
- A finite automaton has a finite set of states
- Its "control" transfers from state to state in response to external "inputs"
- Can be deterministic or non-deterministic

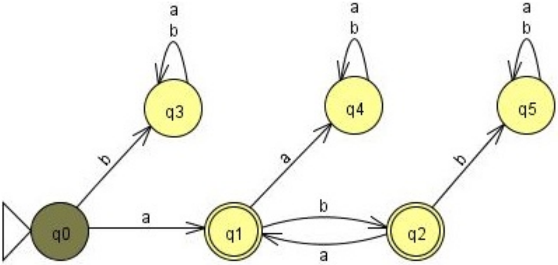
How FA process strings

- A FA decides whether to "accept" or "reject" a string
- The set of all strings a FA accepts is the language it represents
- Starting from the start state, we transition among the states using the transition function δ using each symbol in the input string until the whole string is read
- If we eventually ended in a final state, the input string is accepted

Sample Run

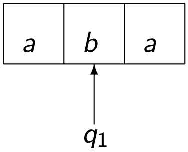
$L = \{a, ab, aba, abab, ababa, ...\}$ over $\Sigma = \{a, b\}$
Test if "aba" belong to the language.

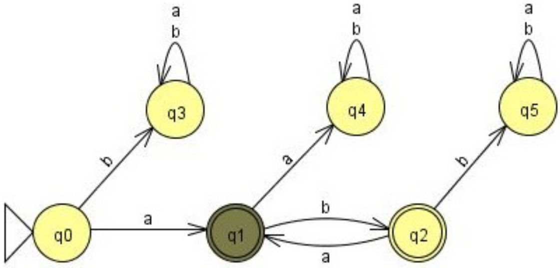




Sample Run

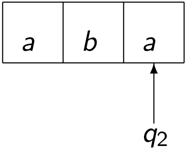
$L = \{a, ab, aba, abab, ababa, ...\}$ over $\Sigma = \{a, b\}$
Test if "aba" belong to the language.

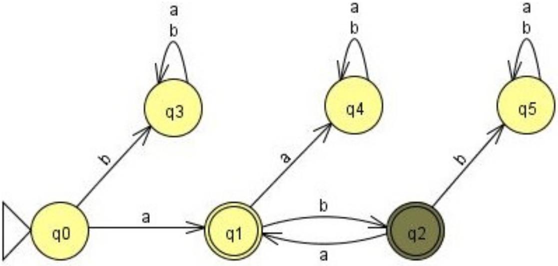




Sample Run

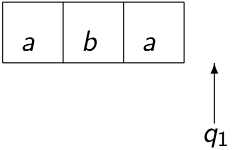
$L = \{a, ab, aba, abab, ababa, ...\}$ over $\Sigma = \{a, b\}$
Test if "aba" belong to the language.

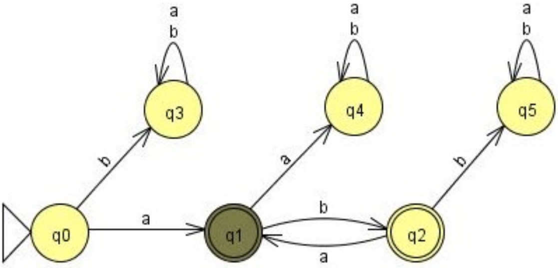




Sample Run

$L = \{a, ab, aba, abab, ababa, ...\}$ over $\Sigma = \{a, b\}$
Test if "aba" belong to the language.





References

- Previous slides on CMSC 141
- M. Sipser. Introduction to the Theory of Computation. Thomson, 2007.
- J.E. Hopcroft, R. Motwani and J.D. Ullman. Introduction to Automata Theory, Languages and Computation. 2nd ed, Addison-Wesley, 2001.
- E.A. Albacea. Automata, Formal Languages and Computations, UPLB Foundation, Inc. 2005
- JFLAP, www.jflap.org