Limbaje Formale si Automate

Instructor *Andrei PĂUN*

Semestrul II

Departmentul de Infomrmatica
Universitatea Bucuresti
Bucuresti, Romania

Structura cursului:

1. Cunostinte de baza
   multimi, multiseturi, secvente, functii, relatii, metode si tehnici de demonstratie, alfabet, cuvinte, limbaje

2. Automate finite
   automate finite nedeterministe (NFA), automate finite deterministe (DFA), transformarea de la NFA la DFA, proprietati pentru limbajele acceptate de automate,
   lema de pompare

3. Expresii regulate
   echivalenta dintre expresiile regulate si automatele finite
   (in definirea limbajelor)

4. Gramatici independente de context (CFG)
   simplificare si forme normale, algoritmi de parsare, lema de pompare si alte proprietati

5. Automate pushdown (PDA)
nondeterministic pushdown automata (NPDA),
deterministic pushdown automata (DPDA),
PDA vs. CFGs

6. Masini Turing (TM)
diverse tipuri de TM, TM universale,
decidabilitate si nedecidabilitate,
complexitate de timp si spatiu

Motivare pentru curs

Theory of Computing

(1) Putere de calcul

(2) Complexitate

(3) Software design

(4) Circuit design

. . . . . .

I. Cunostinte de baza

1. Multimi

1) O multime este o colectie de elemente luate dintr-un univers.

2) O multime este finita daca contine un numar finit de elemente, si este infinita in caz contrar.

3) Cardinalitatea unei multimi este numarul sau de elemente si se noteaza cu $\#A$ sau $|A|$ (pentru multimea A).

– **Cum specificam o multime:**

(1) Listam toate elementele sale:

{} (or $\emptyset$),

$A = \{2,\ 3,\ 5,\ 7\}$**,**

$B = \{3,\ 2,\ 2,\ 7,\ 5,\ 7\}$

(2) Precizam o proprietate: $\{x \mid P(x)\}$

$C = \{x \mid x$ **este numar prim**$\}$**,**

$D = \{x \mid x$ **este un cuvant in romana si** x **este si palindrom**$\}$**,**

$E = \{x \mid x$ **este un intreg divizibil cu 3 si mai mic decat 10** $\}$

(3) Definite recursiv:

Forma generala:

i) Anumite elemente initiale sau de baza sunt in A

ii) Daca $a,\ b \in A$**, atunci** $a \circ b \in A$

iii) Nici un alt element nu apartine lui A

Exemplu Definirea lui A:

i) $2 \in A$

ii) Daca $a,\ b \in A$**, atunci** $a + b \in A$

iii) Nici un alt element nu apartine lui A

(4) Inchiderea la operatii

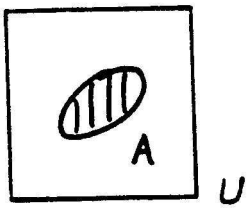
- **Spunem ca o multime** A **este inchisa la o operatie binara** $\circ$ **daca pentru toate** $a,\ b \in A$**,** $a \circ b \in A$
- **Fie** $B \supseteq A$ **si** B **este inchisa la** $\circ$**. Spunem ca** B **este o inchidere a lui** A **la operatia** $\circ$ **daca nu exista** B' **astfel incat** $A \subseteq B' \subset B$ **si** B' **este inchisa la** $\circ$**.**
- **Forma generala (pentru multimi definite prin inchidere)**
  A **este cea mai mica multime care contine anumite elemente initiale (de baza) si care este inchisa la operatia** $\circ$**.**

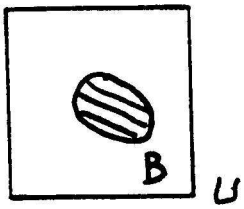
- **Exemplu**

  B este cea mai mica multime care il contine pe τ si este inchisa la "+".

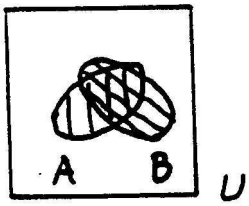
— **Operatii cu multimi**

Daca se dau A si B doua multimi:

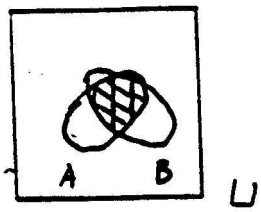




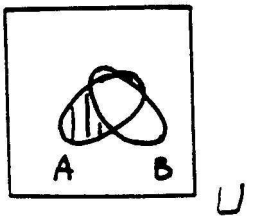
$A \cup B = \{x \mid x$ **este in** A **sau** x **este in** $B\}$



$A \cap B = \{x \mid x$ **este in** A **si** x **este si in** $B\}$



$A - B = \{x \mid x$ **este in** A **si** x **nu este in** $B\}$



$\cup_{i=1}^{n} A_i = \{x \mid x \textbf{ este in } A_i, \textbf{ pentru un } i,\ 1 \leq i \leq n\}$
$\cap_{i=1}^{n} A_i = \{x \mid x \textbf{ este in } A_i,\ 1 \leq i \leq n\}$

— Proprietati ale operatiilor cu multimi

Distributivitate

$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$
$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$

Indempotenta

$A \cup A = A;\ A \cap A = A$

Involutie

$\neg(\neg A) = A$

Comutativitate

$A \cap B = B \cap A;\ A \cup B = B \cup A$

Asociativitate

$A \cup (B \cup C) = (A \cup B) \cup C$
$A \cap (B \cap C) = (A \cap B) \cap C$

Regulile lui De Morgan

$\neg(A \cup B) = \neg A \cap \neg B$

$\neg(A \cap B) = \neg A \cup \neg B$

— **Multimea submultimilor**

Multimea tuturor submultimilor unei multimi A ("power set" of A) se noteaza cu 2^A.

Exemplu:

$A = \{2, 3, 5\}$

$2^A = \{\emptyset, \{2\}, \{3\}, \{5\}, \{2, 3\}, \{2, 5\}, \{3, 5\}, \{2, 3, 5\}\}$

— **Cateva multimi speciale:**

$\emptyset$: multimea vida;

$\mathcal{Z}$: multimea intregilor;

$\mathcal{N}$: multimea intregilor pozitivi;

$\mathcal{N}_0$: multimea intregilor care nu sunt negativi;

$\mathcal{R}$: multimea numerelor reale.

. . . . . .

2. **Multiseturi**

   Un multiset este o colectie de elemente dintr-un univers oarecare, colectie in care repetitiile nu sunt ignorate.

3. **Secvente**

   O secventa de elemente dintr-un univers oarecare este o lista de elemente care sunt ordonate (fiecare element are o pozitie).

4. Functii

Definitie Daca se dau doua multimi A si B, o functie (partiala) de la A la B asociaza cu fiecare a din A (cel mult) un element b din B.

Notatii:

functie $f : A \to B$
asocierea $f(a) = b$
nedefinita $f(a) = undef$

Functii speciale:

functia identitate:
$f : A \to A$ **si**
$f(a) = a$ **pentru toti** $a \in A$

Functia caracteristica a lui A **pentru** B**:**
$B \subseteq A$
$f : A \to \{true,\ false\}$
$f(a) = true,\ if\ a \in B$
$f(a) = false,\ if\ a \in A - B$

Cateva proprietati ale functiilor $f : A \to B$

totala:
$f(a)$ **este definita pentru toate elementele** a **din** A

surjectiva:
pentru toate elementele b **din** B **exista un** a **din** A **astfel incat** $f(a) = b$

injectiva:
$a,\ a' \in A,\ a \neq a'$ **implica faptul ca** $f(a) \neq f(a')$

bijectiva:
in acelasi timp si surjectiva si injectiva

Notatia big-O

$f(n) = O(g(n))$ **daca exista** $c > 0$ **si intregul pozitiv** d **astfel incat pentru toate** $n \geq d$

$$f(n) \leq cg(n)$$

$f(n) = \Omega(g(n))$ **daca**

$$cf(n) \geq g(n)$$

$f(n) = \Theta(g(n))$ **daca**

$$f(n) = O(g(n)) \text{ si } f(n) = \Omega(g(n))$$

functie (partiala) m,n

$$f : A_1 \times A_2 \times \ldots \times A_m \to B_1 \times \ldots \times B_n$$

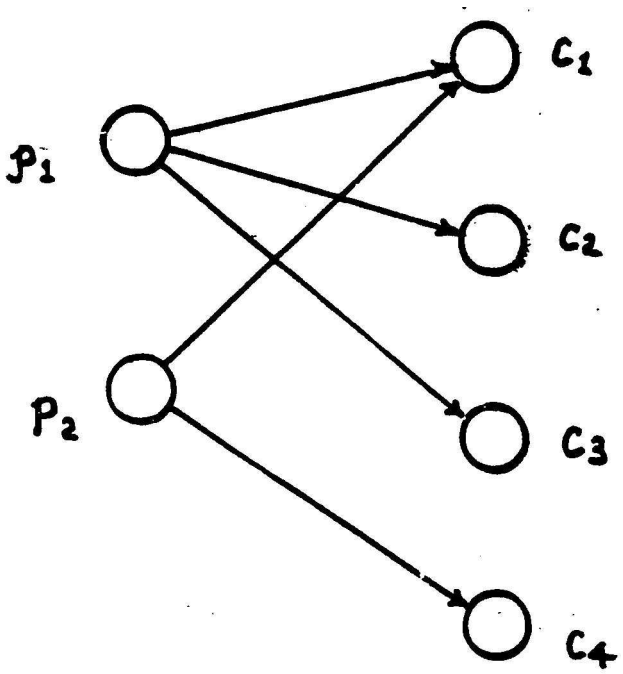
$$f(a_1, a_2, \ldots, a_m) = (b_1, \ldots, b_n)$$

sau $f(a_1, a_2, \ldots, a_m) = undef$

5. Relatii

O relatie n-ara R, $n \geq 1$, pentru multimile $A_1, \ldots, A_n$ este orice submultime R a produsului cartezian $A_1 \times \ldots \times A_n$.

Exemplu



$P = \{p_1, p_2\}, \ C = \{c_1, c_2, c_3, c_4\}$
$T \subseteq P \times C$
$T = \{(p_1, c_1), (p_1, c_2), (p_1, c_3), (p_2, c_1), (p_2, c_4)\}$

Compunerea relatiilor

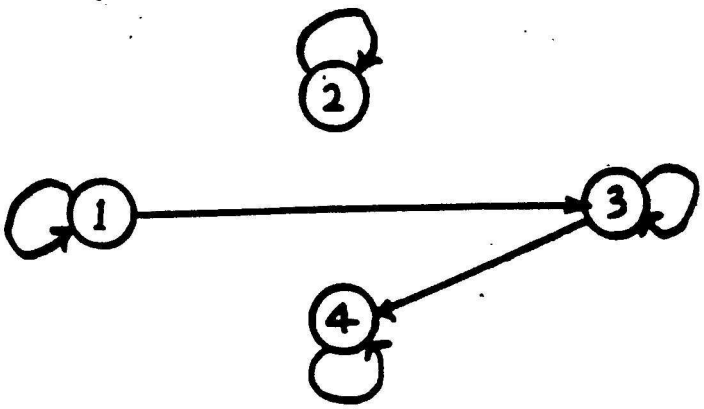
Fie $R \subseteq A \times B$ si $S \subseteq B \times C$ doua relatii. Atunci compunerea lui R cu S, notata prin $R \circ S$, este relatia $R \circ S \subseteq A \times C$, definita prin:
$R \circ S = \{(a, c) \mid (a, b)$ **este in** R **si** (b, c) **este in** S
pentru un b **din** $B\}$

Compunerea unei relatii $R : A \times A$ cu ea insasi

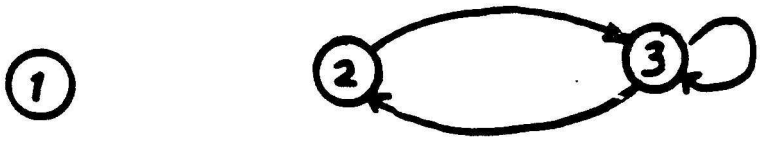
$R^0 \ : \ \{(a, a) \mid a$ **este in** $A\}$
$R^1 \ : \ R$
$R^i, \ _{i \geq 1} \ : \ R \circ R^{i-1}$
$R^+ = \cup_{i=1}^{\infty} R^i$ **inchiderea tranzitiva a lui** R
$R^* = \cup_{i=0}^{\infty} R^i$ **inchiderea reflexiva si tranzitiva a lui** R

Proprietatile relatiei $R \subseteq A \times A$

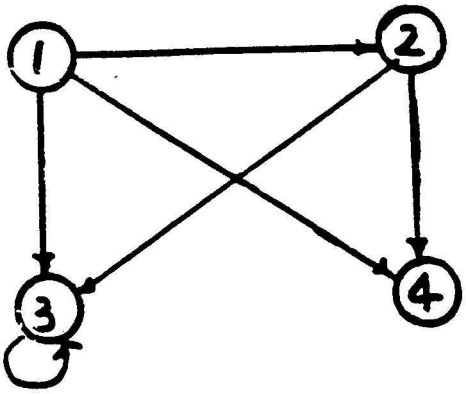
- **reflexiva:** aRa **pentru toti** a **din** A



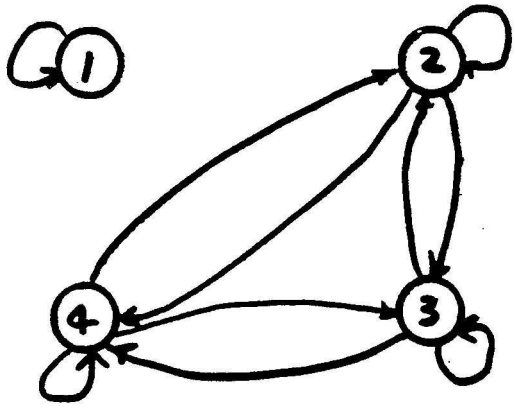
- **simetrica:** aRb **implica** bRa



- **antisimetrica:** $aRb,\ bRa$ **implica** $a = b$



- **tranzitiva:** $aRb,\ bRc$ **implica** aRc



△ **O relatie binara** R **peste** A **este o relatie de echivalenta daca este**

- **reflexiva,**
- **simetrica, si**
- **tranzitiva.**

△ **O relatie de echivalenta** R **peste** A **defineste submultimi disjuncte ale lui** A**, numite clase de echivalenta.**

$$[a]_R = \{b \mid b \text{ este in } A \text{ si } aRb\}$$

• **Fie R o relatie binara de echivalenta peste A si $B \subseteq A$.**
Spunem ca B este R-inchisa daca pentru toti a din B, aRb implica faptul ca b este in B.

Exemplu A : **multimea intregilor;**
$B = \{9, 4, 3, 2\}$**;**
R :**"este patratul lui" ("is the square of")**
Atunci B este R-inchisa.

• **R este o relatie n-ara peste A. $B \subseteq A$ este R-inchisa daca pentru toate $b_1, \ldots, b_{n-1}$ din B, avem ca daca $(b_1, \ldots, b_{n-1}, b_n)$ este in R atunci b_n este in B.**

Exemplu A : **multimea tuturor intregilor;**
$R = \{(a, b, c) \mid a \times b = c\}$**;**
$B = \{ia \mid i \geq 1\}$ **pentru un intreg a**
Atunci B este R-inchisa.

<u>Lema</u> Fie R o relatie de echivalenta peste A. Atunci pentru toti a, b din A avem sau $[a]_R = [b]_R$ sau $[a]_R \cap [b]_R = \emptyset$

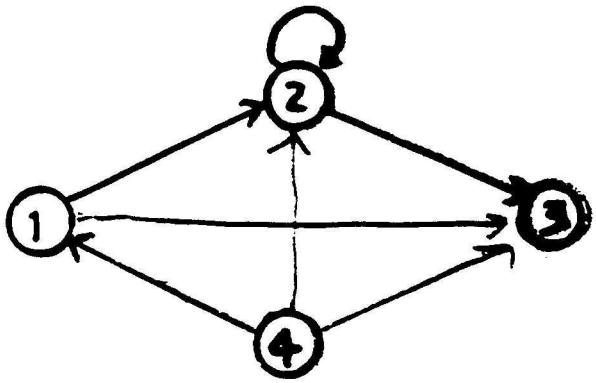
△ **O relatie binara R peste A este o ordine partiala daca este**

- **reflexiva,**
- **tranzitiva, si**
- **antisimetrica.**

△ **<u>Inchidere</u>**

– R^+ este inchiderea tranzitiva a lui R

– O reprezentare in digraf pentru o inchidere tranzitiva:



6. Cardinalitate

△ **Daca se dau doua multimi A si B, spunem ca ele au cardinalitate egala daca exista o bijectie $f : A \to B$.**

Exemplu $\#\mathcal{Z} = \#\mathcal{N}_0$

$$f(i) = -2i - 1, \ i < 0$$
$$f(i) = 2i, \ i \geq 0$$

△ **O multime este numarabila sau enumerabila daca este ori finita ori infinita, caz in care are aceeasi cardinalitate cu $\mathcal{N}$.**

7. Metode de demonstratie

— demonstratie directa

— demonstratie prin contradictie

— demonstratie prin inductie

△ **Demonstratie directa**

Enumeram toate cazurile pentru a arata ca proprietatea este adevarate.

Exemplu Orice numar par dintre 4 si 2^{32} este suma a doua numere prime.

△ **Demonstratie prin contradictie**

Stabilim validitatea propozitiei presupunand ca nu este adevarata si inferam o contradictie.

Exemplu Exista un numar infinit de numere impare.

△ Demonstratia prin inductie

– baza

– ipoteza inductiva

– pasul inductiv

– concluzie

Two forms:
(1) — Prove it holds for a basis value b,
— Hypothesize it holds for value n,
— Prove it holds for value $n+1$.

(2) — Prove it holds for basis values,
— Hypothesize it holds for all values less than n and greater than basis value,
— Prove it holds for value n.

Example (proof by induction)

(i) no two lines are parallel;

(ii) no three lines have a common intersection point.

Prove that the number of regions in an arrangement of n lines is

$$1 + n(n+1)/2$$

8. Proof Techniques

— pigeonhole principle

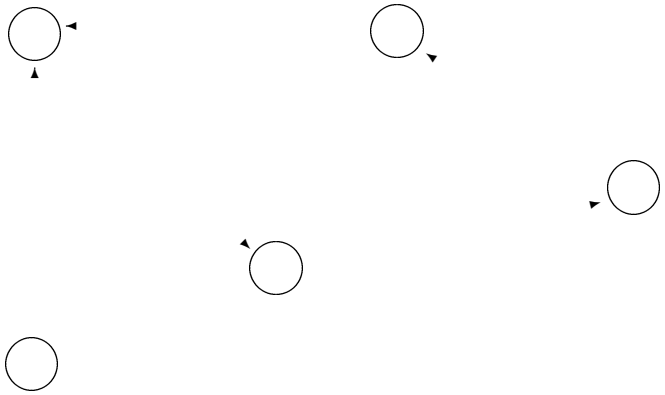
— counting

— diagonalisation

The Pigeonhole Principle
Let $n \geq 1$ be an integer, let there be n pigeonholes, and there be $n+1$ items of mail to be placed in the pigeonholes. Then, however the the items are placed, at least one pigeonhole will contain at least two items.

The Pigeonhole Principle(Extended version)
Let $q_1, \ldots, q_t \geq 1$. There exists a positive integer $N(q_1, \ldots, q_t)$. If $n \geq N(q_1, \ldots, q_t)$, then however n items are placed into t pigeonholes, there exists i, $1 \leq i \leq t$ such that the i-th pigeonhole contains at least $q_1 + 1$ items.

Example **Let $G = (V, E)$ be a digraph with $\#V = n \geq 1$. Then G has a cycle if and only if G has a path of length at least n.**



9. Alphabets, Words, and Languages

△ **An alphabet is a finite, nonempty set of elements.**
The elements of the alphabet are called symbols or letters.

△ **A word over an alphabet Σ is a finite sequence of symbols from Σ.**

△ **A language is a set of words.**

Examples:
(1) English alphabet, words and language.

(2) Alphabet: $\{a,\ b\}$
words: $\lambda,\ a,\ b,\ ab,\ ba,\ \ldots\ldots$
language: $\{\lambda,\ ab,\ aabb,\ aaabbb,\ \ldots\ldots\}$
i.e. $\{a^i b^i \mid i \geq 0\}$

(3) Alphabet : Fortran reserved words and identifiers
words : a Fortran program
Language : the set of all Fortran programs.

△ **Empty word is a word over every alphabet.**

$$\lambda$$

△ **The length of a word is the number of symbols in the given word.**

Example: $|\lambda| = 0$

$|abbc| = 4$

△ **The a-length of a word x is the number of times a occurs in x.**

Example: $|01101|_0 = 2$

$|cbcbc|_a = 0$

△ **The <u>catenation</u> of two words x and y, denoted by xy, is the word obtained by appending the word y to x.**

<u>Example</u> $x = abc$
$y = defg$
$xy = abcdefg$
$yx = defgabc$

$$x\lambda = \lambda x = x$$
$$\lambda\lambda = \lambda$$

△ **Catenation is <u>associative</u>:**

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

△ $|xy| = |x| + |y|$

△ $x^0 = \lambda$ **(Power of a word)**
$x^i = xx^{i-1},\ i \geq 1$

<u>Example</u> $(abc)^0 = \lambda$
$(abc)^3 = abcabcabc$

△ **Prefix: x is a prefix of y if $\exists z$ such that**

$$xz = y$$

Example
- ***not* is a prefix of *notice***
- **λ is a prefix of any word**
- **Any word is a prefix of itself**

△ **Suffix: x is a suffix of y if $\exists z$ such that**

$$zx = y$$

Example
- ***allow* is a suffix of *swallow***
- **λ is a suffix of any word**
- **01011 is a suffix of 01011**

△ **Subword: x is a subword of y if $\exists w, z$ s.t.**

$$wxz = y$$

Example The subwords of *beat* are:

$$\lambda,\ b,\ e,\ a,\ t,\ be,\ ea,\ at,\ bea,\ eat,\ beat$$

△ **Proper prefix (suffix, subword):**
x is a proper prefix (suffix, subword) of y if x is a prefix (suffix, subword) of y and $x \neq \lambda$ and $x \neq y$.

△ **Reversal:**

(i) $x^R = x$, **if** $x = \lambda$;

(ii) $x^R = y^R a$, **if** $x = ay$ **for** $a \in \Sigma$, y **is a word**

Lemma If $x = a_1 \dots a_n$ **for some** $n \geq 0$, **then**

$$x^R = a_n \dots a_1.$$

Proof: By induction on $|x|$.

Basis: $|x| = 0$. $\lambda^R = \lambda$ **by (i).**

I.H.: Assume Lemma holds for all x **with** $|x| = n$.

I.S.: Consider a word x **with** $|x| = n + 1$.

Then $x = a_1 \dots a_{n+1} = a_1 u$, **where**

$$u = a_2 \dots a_{n+1}.$$

Then $x^R = u^R a_1$.

But $|u| = n$, **by I.H.,** $u^R = a_{n+1} \dots a_2$, **so we get**

$$x^R = a_{n+1} \dots a_1. \; qed$$

△ **Universal Language over** Σ: Σ^*

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

△ Mappings (Morphism and Substitution)

Definition Let Σ and Δ be two alphabets. Then a mapping $h : \Sigma^* \to \Delta^*$ is said to be a morphism if

(i) $h(\lambda) = \lambda$**;**

(ii) $h(xy) = h(x)h(y)$**, for all** $x,\ y$ **in** Σ^***.**

Note: (ii) implies that we need only define the images of letters to define the images of words.

Example:

$\Sigma = \{0,\ 1\}, \quad \Delta = \{a,\ b,\ c,\ d\},$

$h(0) = baac, \quad h(1) = \lambda.$

Then $h(1001) = h(1)h(0)h(0)h(1) = \lambda baacbaac\lambda = baacbaac$

A morphism h is said to be λ-free if, for all $x \neq \lambda, \quad h(x) \neq \lambda.$

<u>Definition</u> Let Σ and Δ be two alphabets. A mapping $\sigma : \Sigma^* \to 2^{\Delta^*}$ is said to be a <u>substitution</u> if

i) $\sigma(\lambda) = \{\lambda\}$**;**

ii) $\sigma(xy) = \sigma(x)\sigma(y)$**, for all** $x,\ y \in \Sigma^*$**.**

<u>Example</u>:

$\Sigma_1 = \{a,\ b\}, \quad \Delta_1 = \{a,\ b,\ c\},$

a substitution σ_1**:**

$\sigma_1(a) = \{ab,\ ac,\ b\};$

$\sigma_1(b) = \{b,\ ba\}.$

Then $\sigma_1(ba) =$

<u>Example</u>:

$\Sigma_2 = \{0,\ 1,\ 2\}, \quad \Delta_2 = \{a,\ b\},$

$\sigma_2(0) = \{a^i \mid i \geq 1\}$

$\sigma_2(1) = \{b^i \mid i \geq 1\}$

$\sigma_2(2) = \{\lambda\}$

Then $\sigma_2(1020) =$

△ **Given an alphabet** Σ, Σ^* **is defined as follows:**

(i) $\lambda \in \Sigma^*$**;**

(ii) if $x \in \Sigma^*$**, then** $ax \in \Sigma^*$ **for all** $a \in \Sigma$**.**

△ **A language** L **over an alphabet** Σ **is a subset of** Σ^***, i.e.,** $L \subseteq \Sigma^*$**.**

Example:

ϕ **is a language over every alphabet .**

$\{\lambda\}$ **is a language over every alphabet.**

$\{a^i b^i \mid i \geq 0\}$ **is a language over** $\{a,\ b\}$

$\{a^i b^i c^i \mid i \geq 0\}$ **is a language over** $\{a,\ b,\ c\}$

△ Catenation of Languages

$L_1 \circ L_2 = \{x_1 \circ x_2 \mid x_1 \textbf{ in } L_1,\ x_2 \textbf{ in } L_2\}$,
usually written as L_1L_2.

Examples

$L_1 = \{a,\ ab\}; \quad L_2 = \{bc,\ c\}$

$L_1L_2 = \{abc,\ ac,\ abbc\}$

(Note that $L_1 \times L_2 = \{(a, bc), (a, c), (ab, bc), (ab, c)\}$**)**

$\phi L_1 =$

$\{\lambda\}L_1 =$

$\{\lambda\}\phi =$

$L_A = \{a,\ \lambda\}; \quad L_B = \{ab,\ b\}$

$L_AL_B =$

△ **Powers of Languages:**

$L^0 = \{\lambda\}$

$L^{n+1} = L^n \circ L, \ n \geq 1$

△ **Plus**

$L^+ = \cup_{i=1}^{\infty} L^i$

△ **Star**

$L^* = \cup_{i=0}^{\infty} L^i \qquad\qquad L^0 \cup L^1 \cup L^2 \cup \ldots\ldots$

Examples

$L = \{ab, \ b\}$**. Then**

$L^0 = \{\lambda\}$

$L^1 = L = \{ab, \ b\}$

$L^2 = \{abab, \ bab, \ abb, \ bb\}$

$\phi^0 = \{\lambda\} \neq \phi$

$\{\lambda\}^0 = \{\lambda\}$

$L^* = \{\lambda, \ ab, \ b, \ abab, \ bab, \ abb, \ bb, \ \ldots\ldots\}$

(Note $ababbabab \in L^4$**)**

△ Reversal of Languages

$$L^R = \{x^R \mid x \textbf{ is in } L\}$$

Examples

$$L_F = \{aab,\ abb,\ abbb\};$$
$$L_F^R = \{baa,\ bba,\ bbba\}$$
$$L_I = \{a^m b^n \mid m > n > 0\}$$
$$L_I^R = \{b^n a^m \mid m > n > 0\}$$

— **properties of reversal**

$$\begin{aligned}(A \cup B)^R &= A^R \cup B^R \\ (A \cap B)^R &= A^R \cap B^R \\ (AB)^R &= B^R A^R \\ (A^+)^R &= (A^R)^+ \\ (A^*)^R &= (A^R)^*\end{aligned}$$

△ Complementation of Languages

Given a language L over Σ,

$$\overline{L} = \Sigma^* - L$$

$\overline{\Sigma^*} = \phi$ $\qquad$ $\overline{\Sigma^*}\ ?\ \{\lambda\}$ $\qquad$ $(\overline{A})^R = \overline{A^R}$

$L = \{a^n b^n \mid n \geq 0\}$ **over** $\Sigma = \{a,\ b\}$

$\overline{L} =$

△ **Alternative definition of L^+ and L^***

$$L^+ = \{w \text{ in } \Sigma^* \mid w = w_1 w_2 \dots w_n, \text{ for } w_1, w_2, \dots, w_n \in L \text{ and } n \geq 1\}$$

$$L^* = \{w \text{ in } \Sigma^* \mid w = w_1 w_2 \dots w_n, \text{ for } w_1, w_2, \dots, w_n \in L \text{ and } n \geq 0\}$$

△ **Given a word x in Σ^*, and a language $L \subseteq \Sigma^*$ to test if x is in L^* we use $L^* = \cup_{i=0}^{\infty} L^i$ and test if x is in $L^0,\ L^1,\ \dots$.**

△ Property Summaries

— properties of catenation (of languages)

$$\begin{aligned}
\textbf{Associativity} &\text{ — } L_1(L_2L_3) = (L_1L_2)L_3 \\
\textbf{Identity} &\text{ — } L\{\lambda\} = \{\lambda\}L = L \\
\textbf{Zero} &\text{ — } L\phi = \phi L = \phi \\
\textbf{Distributivity} &\text{ — } L_1(L_2 \cup L_3) = L_1L_2 \cup L_1L_3 \\
\textbf{w.r. } \cup &\text{ — } (L_1 \cup L_2)L_3 = L_1L_3 \cup L_2L_3
\end{aligned}$$

Distributivity does not hold with respect to $\cap$

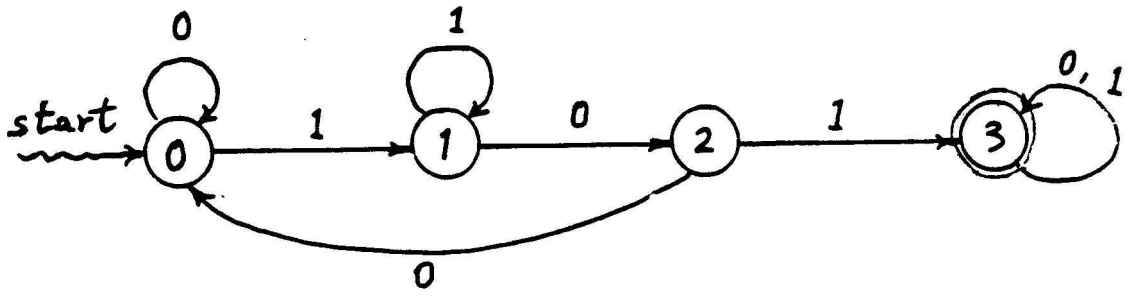
— properties of Plus and Star

$$\begin{aligned}
L^* &= L^+ \cup \{\lambda\} \quad (\textbf{ But } L^+ = L^* - \{\lambda\}?) \\
L^+ &= LL^* \\
(L^+)^+ &= L^+, \quad (L^*)^* = L^* \\
\phi^+ &= \phi, \qquad \phi^* = \{\lambda\}, \quad \phi^0 = \{\lambda\} \\
\{\lambda\}^+ &= \{\lambda\}, \quad \{\lambda\}^* = \{\lambda\}, \quad \phi^2 = \phi \\
L^+L &= LL^+, \quad L^*L = LL^* \\
A \subseteq B &\Rightarrow A^+ \subseteq B^+ \textbf{ and } A^* \subseteq B^*
\end{aligned}$$

Chapter 2. FINITE AUTOMATA

Example Design a "sequential lock". The lock has 1-bit sequential input. Initially the lock is closed. If the lock is closed it will open when the last three input signals are "1", "0", "1", and then remains open.

— state (transition) diagram



— state (or transition) table

Present state	Present symbol 0	1
0	0	1
1	2	1
2	0	3
3	3	3

state set : $\{0,\ 1,\ 2,\ 3\}$
input alphabet : $\{0,\ 1\}$
Transition function : $\delta(0,0) = 0,\ \delta(0,1) = 1,\ \ldots$
Start state : 0
Final state set : $\{3\}$

Deterministic Finite Automata (DFA)

$M = (Q,\ \Sigma,\ \delta,\ s,\ F)$ **where**

Q **is a finite nonempty set of states**

Σ **is the input alphabet**

$\delta : Q \times \Sigma \to Q$ **transition function**

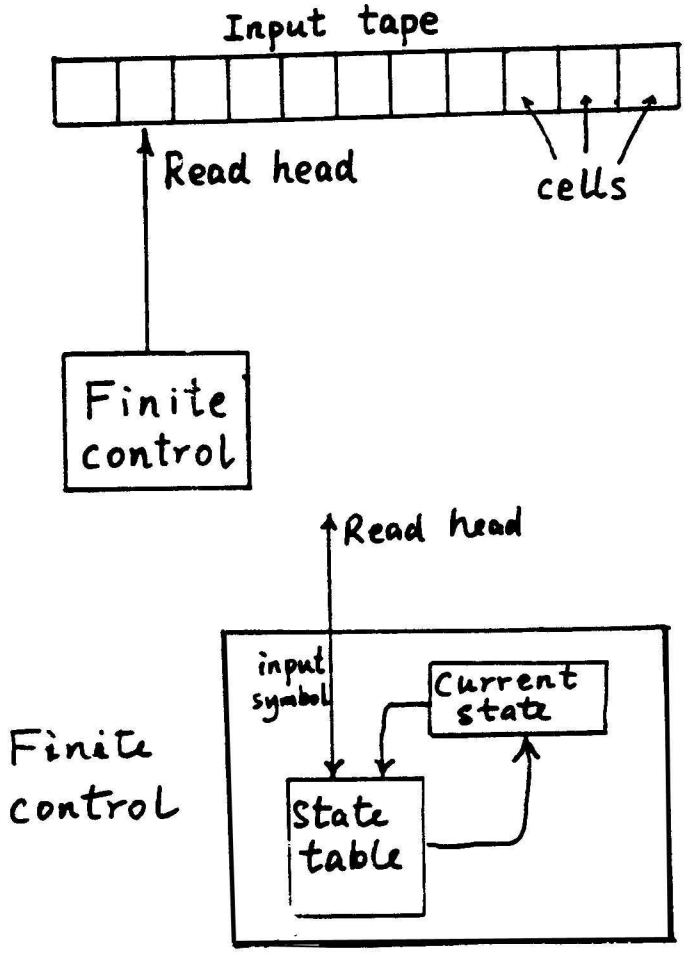
s **start state**

$F \subseteq Q$ **final state set**

A computer is a finite state system (i.e. FA) which has millions of states.

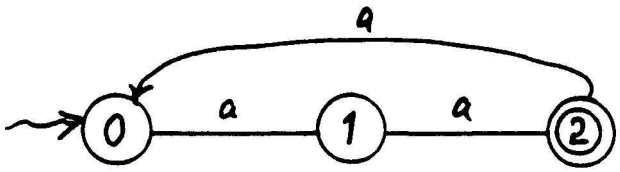
There are many examples of <u>FINITE STATE SYSTEMS</u>. A finite automaton is an <u>ABSTRACTION</u> of them.

<u>View a DFA as a machine</u>



Specifying δ

1)



State diagram
(Transition diagram)

Start state

Final state

2)

Present state	Present symbol a
0	1
1	2
2	0

Configurations

a word in $Q\Sigma^*$

$$px$$

where p **is the present state, and**
x **is the remaining input**

Example:

$0aa \quad \ldots \quad 1a \quad \ldots \quad 2$

(start configuration) (final configuration)

Moves of a DFA

$0aa \vdash 1a$ $\qquad$ $px \vdash qy$

$1a \vdash 2$ $\qquad$ **if** $x = ay$ **and** $\delta(p, a) = q$

Configuration sequence

$0aa \vdash 1a \vdash 2$

$\vdash^+$ **and** $\vdash^*$

$\vdash$ **is a binary relation over** $Q\Sigma^*$**.**

$\vdash^+$: **transitive closure of** $\vdash$**.**

$\vdash^*$: **reflexive transitive closure of** $\vdash$**.**

$$0aa \vdash^+ 2$$
$$0aa \vdash^* 2$$
$$0aa \vdash^* 0aa$$
$$0aa \vdash^2 2$$

$$px \vdash^k qy$$

if $px \underbrace{\vdash p_{i_1}x_{i_1} \vdash p_{i_2}x_{i_2} \vdash \ldots}_{k \textbf{ steps}} \vdash qy$

Accepting Configuration Sequence

$0aa \vdash 1a \vdash 2$

$\vdash$ can also be viewed as a function

$$\vdash : Q\Sigma^* \to Q\Sigma^*,$$

since the next configuration is determined uniquely for a given configuration.

The DFA stops when:

(i) we have no more input,

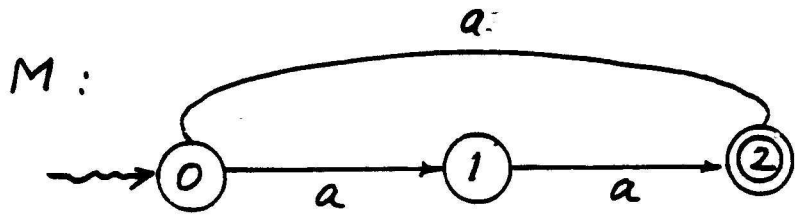
or (ii) the next configuration is undefined.

A word x is said to be accepted by a DFA M if $sx \vdash^* f,\ f \in F$.

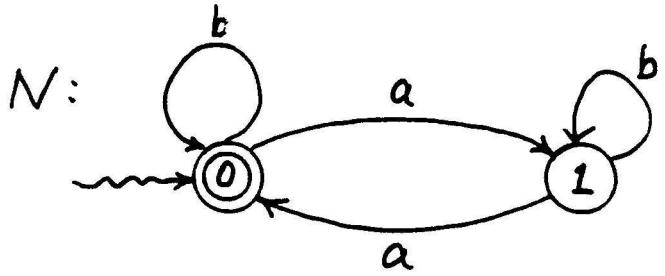
The language of a DFA M, $L(M)$, is defined as:

$$L(M) = \{x \mid sx \vdash^* f, \text{ for some } f \in F\}$$

Examples



$L(M) =$



$L(N) =$

DFA membership problem

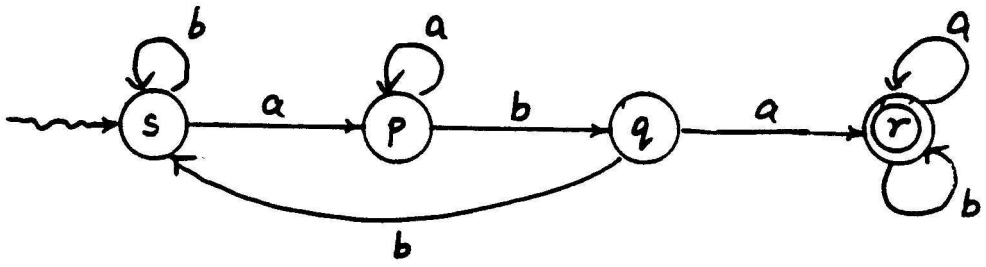
DFA MEMBERSHIP

INSTANCE: A DFA, $M = (Q, \Sigma, \delta, s, F)$
and a word $x \in \Sigma^*$.

QUESTION: Is x **in** $L(M)$ **?**

Run the DFA M **with input** x**.**
In at most $|x|$ **steps it accepts, rejects or aborts.**

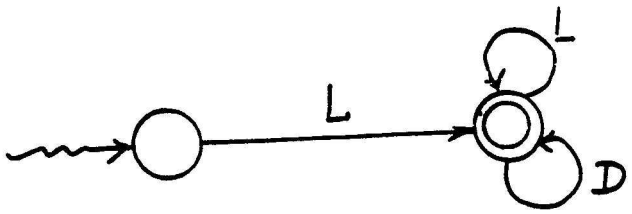
Examples



Checking for words that contain *aba* **as subword.**

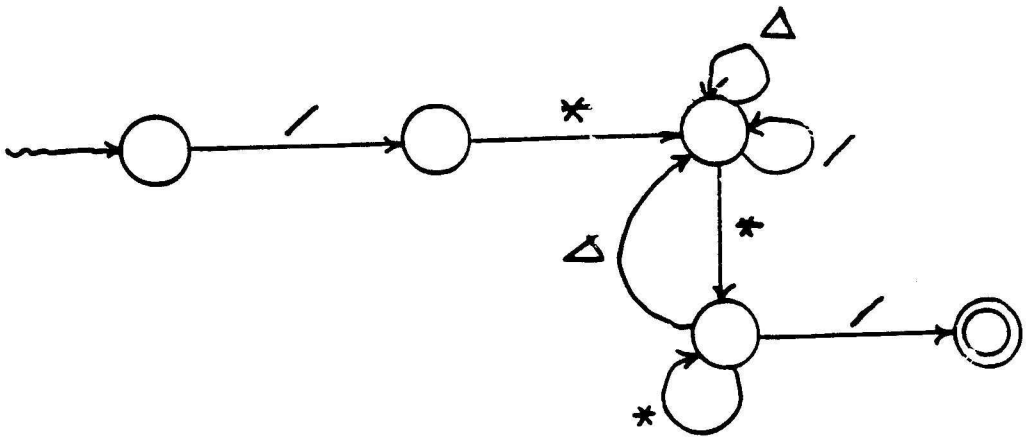
Check: *ababba*
abbaabbbab

Let L denote any letter of English alphabet and D any decimal digit; the form of PASCAL IDENTIFIERS can be specified by



Recognizing comments that may go over several lines.

/*.....*/

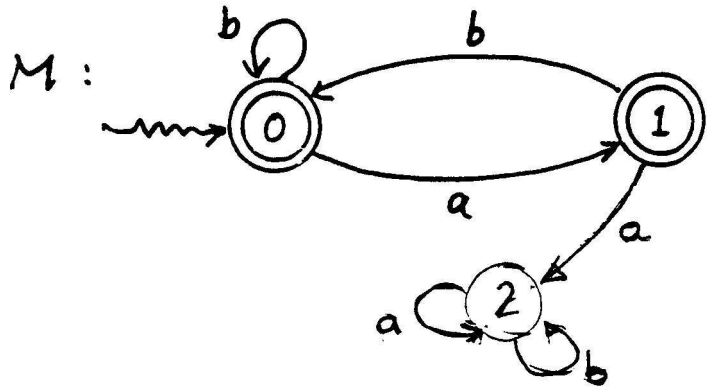


Δ: symbols other than " $*$ " and "/"

A DFA which has a total δ is said to be complete; if δ is nontotal it is incomplete.

Theorem. Every incomplete DFA M can be "completed" by adding one new state ("sink") to give DFA M' such that $L(M') = L(M)$.

Example:



$L(M)$ is the set of all words that do not contain two consecutive a's.

$\triangle$ **Two DFA M_1 and M_2 are equivalent if $L(M_1) = L(M_2)$.**

$\triangle$ **The collection of languages accepted by DFA's is denoted by**

$$\mathcal{L}_{DFA}.$$

It is called the family of DFA languages and it is defined as:

$$\mathcal{L}_{DFA} = \{L \mid L = L(M) \text{ for some DFA } M\}$$

$\triangle$
$K = \{a^i b^i \mid i \geq 1\}$ **is not accepted by any DFA.**

Proof: Use contradiction and Pigeonhole principle.

Assume $K = L(M)$, for some DFA

$$M = (Q, \{a, b\}, \delta, s, F).$$

Let $n = \#Q$. Consider the accepting configuration sequence for $a^n b^n$,

$$s_0 a^n b^n \vdash s_1 a^{n-1} b^n \vdash \ldots \vdash s_n b^n \vdash \ldots \vdash s_{2n}$$

where $s_0 = s$ and $s_{2n} \in F$. Now $n + 1$ states appear during the reading of a^n, but there are only n distinct states in Q. By Pigeonhole principle at least one state must appear at least twice during the reading of a's.

Assume $s_i = s_j, 0 \leq i < j \leq n$.
Then

$$s_0 a^{n-(j-i)} b^n \vdash \ldots \vdash s_i a^{n-j} b^n$$
$$s_j a^{n-j} b^n \vdash \ldots \vdash s_n b^n \vdash \ldots \vdash$$
$$\vdash s_{2n}$$

Therefore $a^{n-(j-i)} b^n \in K$.

This is a contradiction.