Complements of mathematics for Computer Science

Mircea Crâşmăreanu

Contents

Course 1

The monoid of words of an alphabet

The aim of this Course is to introduce a main notion, useful in several theories.

Definition 1.1 An *alphabet* is a non-empty set Σ whose elements are called *symbols* or *letters*.

Examples 1.2 i) $\Sigma = B = \{0, 1\}$ is *the binary alphabet* while $\Sigma = \{0, ..., 9\}$ is *the decimal alphabet*. The alphabet $\{A, ..., Z\}$ has 26 letters.
ii) The ASCII alphabet (American Standard Code for Information Interchange) has 128 symbols for a large part of computer languages.

Usually, we will use finite alphabets but there are some important theoretical aspects involving infinite alphabets, e.g. with alef zero=the cardinal of $\mathbb{N}$.

Definition 1.3 A *non-empty word from* Σ is a map $w : \mathbb{N}_k := \{1, ..., k\} \to \Sigma$ with $k \geq 1$ called *the length of word* w and denoted $l(w)$ or $|w|$. Let Σ^+ be the set of non-empty words.

The word w above is denoted $w = w(1)w(2)...w(k)$. From some technical reasons is also useful *the empty word* defined by the empty map $\emptyset \to \Sigma$ and denoted ε (or λ) and having $l(\varepsilon) = 0$. The set of all words from Σ is denoted Σ^* and is very important in all what follows. So $\Sigma^* = \Sigma^+ \cup \{\varepsilon\}$.

Let Σ^k be the set of words with length k; hence $\Sigma^0 = \{\varepsilon\}$ and $\Sigma^1 = \Sigma$. Thus $\Sigma^* = \cup_{k \geq 0} \Sigma^k$. The elements of Σ^k are also called *k-words*.

Examples 1.4 i) For $a \in \Sigma$ we define a^k through $a^0 = \varepsilon$ respectively $a^k = a...a$ where in the right-hand-side a appears k times. So, $a^k \in \Sigma^k$.

ii) For the binary alphabet we have $\Sigma^2 = \{00, 01, 10, 11\}$ and
$\Sigma^3 = \{000, 001, 010, 100, 011, 101, 110, 111\}$.
iii) It follows that if $card\Sigma = n$ then $card\Sigma^k = n^k$.

Definition 1.5 The words $w_1, w_2 \in \Sigma^*$ are called *equals* and we write $w_1 = w_2$ if they belongs to one of the following cases:
I) $l(w_1) = l(w_2) = 0$ i.e. $w_1 = w_2 = \varepsilon$,
II) $l(w_1) = l(w_2) = k \geq 1$ and then for every $i \in \mathbb{N}_k$ we have $w_1(i) = w_2(i)$.

As we can expect it follows:

Proposition 1.6 *The equality is an equivalence relation on* Σ^*.
Proof We must verify the reflexivity, symmetry and transitivity. □

We want a structure on the sets of words and hence we consider:

Definition 1.7 We call *concatenation* on Σ the map $\cdot : \Sigma^* \times \Sigma^* \to \Sigma^*$, $(w_1, w_2) \to w_1 w_2$ given by:
I) $\varepsilon\varepsilon = \varepsilon$ and $\varepsilon w = w\varepsilon = w$,
II) $w_1 w_2 : \mathbb{N}_{|w_1|+|w_2|} \to \Sigma$ is definited by:
II1) $w_1 w_2(i) = w_1(i)$ if $i \in \mathbb{N}_{|w_1|}$,
II2) $w_1 w_2(i) = w_2(i - |w_1|)$ otherwise.

Recall that a *semigroup* is a pair $(S, \cdot)$ with S a non-empty set and "$\cdot$" an associative composition law on S and a *monoid* is a semigroup having a neutral element.

We get remarkable structures on sets of words:

Theorem 1.8 i) $(\Sigma^+, \cdot)$ *is a semigroup.*
ii) $(\Sigma^*, \cdot, \varepsilon)$ *is a monoid.*
iii) $l : (\Sigma^*, \cdot, \varepsilon) \to (\mathbb{N}, +, 0)$ *is a morphism of monoids*: $l(w_1 w_2) = l(w_1) + l(w_2)$.

Remarks 1.9 i) If $card\Sigma \geq 2$ then Σ^+ and Σ^* are non-commutative. For example, for the binary alphabet let us consider $w_1 = 10$ and $w_2 = 0$; hence $w_1 w_2 = 100 \neq w_2 w_1 = 010$.
ii) We will use powers for similar words and brackets for similar strings with length bigger than 1. Attention, the second power of 01, i.e. 0101, is $(01)^2$ and not 01^2 which is 011!

Definition 1.10 i) The semigroup S is *generated* by the finite set $X \subset S$ if any element of S is a product of elements from X: for every $a \in S$ there exist $x_1, ..., x_k \in X$ such that $a = x_1 ... x_k$. X is *a set of generators* for S. The monoid $(M, \cdot, e)$ is *generated* if the semigroupul $S_M = M \setminus \{e\}$ is generated.

ii) S is *free* if there exists a set of generators X such that the decomposition above is unique for *every* element $a \in S$. M is *free* if S_M is free.

Theorem 1.11 *Σ^+ and Σ^* are free generated by Σ.*

Examples 1.12 i) The monoid with unique element (the neutral e) is, by definition, generated by the empty set. By convention, this monoid is free.
ii) $\mathbb{N}$ is free with the generator $\{1\}$. Excepting an isomorphism, $\{e\}$ and $\mathbb{N}$ are all free commutative monoids.
iii) $\mathbb{Z}$ is generated by $\{-1, +1\}$ but is not free since, for example, $2 = (+1) + (+1) = (+1) + (-1) + (+1) + (+1)$.
iv) Is possible for a monoid to have several sets of generators. For example, $\mathbb{Z}$ have also the generators $\{-2, 3\}$. But if M is free then this set of generators is unique.

A very important result is **the universality property of free monoids**:

Theorem 1.13 *Let M be a free monooid generated by G_M, M' a monoid and $f : G_M \to M'$. Then there exists a unique $F : M \to M'$ morphism of monoids extending f.*

Proof Let $a \in M$ be arbitrary and its unique decomposition $a = x_1...x_k$. We define $F(a) = f(x_1)...f(x_k)$. $\square$

Proposition 1.14 *Given the free monoid M there exists a finite set Σ and a surjective morphism of monoids $f : \Sigma^* \to M$.*

Proof Let Σ which generated free M and the inclusion $f : \Sigma \to M$. We apply the universality property. $\square$

Other important properties of free monoids are *the simplification rules*; for $x, y, z, t \in \Sigma^*$ cu $xy = zt$:
1) if $l(x) = l(z)$ then $y = t$,
2) if $l(x) < l(z)$ then there exists $m \in M$ such that $z = xm$ and $y = mt$,
3) if $l(x) > l(z)$ then there exists $m \in M$ such that $x = zm$ and $t = my$.

Definition 1.15 If $a \in \Sigma$ and $w \in \Sigma^*$ then we denotes $|w|_a$ the number of letters a from the word w.

Examples 1.16 $|aba|_a = 2, |aba|_b = 1, |aba|_c = 0$; $l(w) = \Sigma_{a \in \Sigma} |w|_a$.

Definition 1.17 i) Let $(S, \cdot)$ be a semigroup and $T \subset S$ a non-empty subset. Then T is *a subsemigroup* of S if $T^2 \subset T$ i.e. for all $x, y \in T$ we have $xy \in T$. If S is a monoid with the neutral element 1 then T is *a submonoid* if it is a subsemigroup and $1 \in T$.

ii) A semisubgroup of S which is itself a group is called *subgroup* of the semigroup (monoid) S.

Web pages:
1) http://en.wikipedia.org/wiki/Monoid
2) http://en.wikipedia.org/wiki/Initial_and_terminal_objects
3) http://en.wikipedia.org/wiki/Universal_property

SEMINAR 1

S1.1 i) Prove that the neutral element of a monoid M is unique.
ii) Prove that an arbitrary $x \in M$ admits at most an inverse.

Solution i) Let e and e' be neutral elements. Since e is neutral we have $e' = ee'$ and since e' is neutral we have $e = ee'$.
ii) Let x' and x'' be inverses for x. Then $x' = ex' = (x''x)x' = x''(xx') = x''e = x''$.

S1.2 Prove that $G(M) = \{x \in M; x \text{ admite invers } x^{-1}\}$ is a group; it is *the group of units of* M.

Solution $e \in G(M)$ from $e = ee = ee$ i.e. $e^{-1} = e$. If $x \in G(M)$ then $x^{-1} \in G(M)$ also with $(x^{-1})^{-1} = x$.

S1.3 Let S be a no-empty set. Prove that $F(S) = \{f : S \to S\}$ is a monoid relative to the composition of functions. Determine its group of units.

Solution The composition of functions is associative and neutral element is 1_S=the identity function. We have $G(F(S)) = Bij(S) = \{f : S \to S; f \text{ bijec tion}\}$.

S1.4 Find $a, b \in \mathbb{N}$ such that *the linear composition law* $x * y = ax + by$ determine on $\mathbb{N}$ a structure of abelian monoid.

Solution The commutativity $x * y = y * x$ yields $ax + by = ay + bx$ which means $(a - b)(x - y) = 0$ for all natural numbers x, y; with $x \neq y$ it results $a = b$ and then $x * y = a(x + y)$. The associativity $a[a(x + y) + z] = a[x + a(y + z)]$ gives $a^2 = a$. The case $a = 0$ is trivial and we drop it; hence $a = 1$. In conclusion, we have $(\mathbb{N}, +, e = 0)$ as the unique linear composition law which yields a structure of commutative monoid on the set of natural numbers.

S1.5 Let $(M, *)$ and $(N, \circ)$ sets endowed with composition laws and $f : M \to N$ a surjection. Prove that if M is a (abelian) monoid then N is a (abelian) monoid.

S1.6 ! Prove the possibility of existence for a subsemigroup T of a monoid M such that T is monoid without being a submonoid of M.

Solution Let $M = \mathbb{N}_4 = \{1, 2, 3, 4\}$ with the composition law $xy = \max\{x, y\}$. Then $(M, \cdot, 1)$ is a monoid. Let $T = \{2, 3, 4\}$; T is a subsemigroup of M but is not a submonoid since 1 does not belongs to T. Also $(T, \cdot, 2)$ is a monoid.

S1.7 The element e of the semigroup S is called *idempotent* if $e^2 = e$. For an idempotent e we consider the set $H_e = \{x \in S; xe = ex = x, \exists y \in S\ xy = yx = e\}$.
i) Prove that H_e is nonempty.
ii) Prove that H_e is a subsemigroup of S.
iii) Prove that H_e is a group.
iv) Find the idempotents and the associated groups H_e for the previous monoid S1.6.
v) Prove that a finite semigroup S has idempotents.

Solution i) $e \in H_e$.
ii) Let $a, b \in H_e$ and the associated elements a', b'. Then: $(ab)e = a(be) = ae = e$ and $e(ab) = (ea)b = eb = e$ respectively $(ab)(b'a') = a(bb')a' = aea' = aa' = e$ and $(b'a')(ab) = b'(a'a)b = b'eb = b'b = e$.
iii) Let $a \in H_e$ with the correspondent a' and let $a^{-1} = ea'e$. We have: $aa^{-1} = (ae)a'e = aa'e = ee = e$ and $a^{-1}a = ea'(ea) = ea'a = ee = e$. It remains to prove that $a^{-1} \in H_e$: $a^{-1}e = ea'ee = a^{-1}$ and $ea^{-1} = eea'e = a^{-1}$.
iv) Every element $e \in \mathbb{N}_4$ is idempotent and $H_e = \{e\}$.

S1.8 Let $\sim$ an equivalence relation on the semigroup (monoid) S. We say that $\sim$ is a *congruence* on S if $a \sim b$ and $s \in S$ implies $sa \sim sb$ and $as \sim bs$. Prove that then $S/\sim$ is a semigroup (monoid).

Solution Define $[a][b] := [ab]$; we must prove the good definition. If $a \sim c$ and $b \sim d$ then $ab \sim cb$ and $cb \sim cd$ which yields $ab \sim cd$. The fact that $S/\sim$ is a semigroup (monoid with identity [1]) is immediately.

Example Equality on the monoid of words is a congruence.

S1.9 i) Prove that a morphism of semigroups $f : S \to T$ induces a congruence on S.
ii) Conversely, given a congruence $\sim$ on the S we have that $\pi : S \to S/\sim$, $\pi(a) = [a]$ is a morphism of semigroups.
So, there exists a natural correspondence between congruences and morphisms of semigroups.

Solution i) Define $\sim$ on S through: $a \sim b$ if $f(a) = f(b)$. Since the equality is an equivalence we obtain that $\sim$ is an equivalence on S. Let $s \in S$ arbitrary. f being a morphism we have: $f(as) = f(a)f(s) = f(b)f(s) = f(bs)$ and $f(sa) = f(s)f(a) = f(s)f(b) = f(sb)$; then $\sim$ is a congruence on S.
ii) $\pi(ab) = [ab] = [a][b] = \pi(a)\pi(b)$.

S1.10 Let $f : S \to T$ a surjective morphism of semigroups and $\sim$ the equivalence induced by f. Then $f^* : S/\sim \to T, f^*([a]) = f(a)$ is a bijective morphism (i.e. isomorphism) of semigroups and we denote $S/\sim \simeq T$.

Solution We must prove the good definition: $[a] = [b]$ implies $f(a) = f(b)$ i.e. $f^*([a]) = f^*([b])$. Let $t \in T$; since f is surjective there exists $s \in S$ with $f(s) = t$, hence $f^*([s]) = t$ i.e. f^* is surjective. If $f^*([a]) = f^*([b])$ then $f(a) = f(b)$ i.e. $[a] = [b]$, so f^* is also injective. We have $f^*([a][b]) = f^*([ab]) = f(ab) = f(a)f(b) = f^*([a])f^*([b])$ i.e. f^* is a morphism of semigroups.

S1.11 Let Σ a nonempty set, T a semigroup and the map $f : \Sigma \to T$. Then there exists a unique morphism of semigroups $g : \Sigma^+ \to T$ which extends f i.e. $g(s) = f(s)$ for all $s \in S$.

Solution Let $w \in \Sigma^+$ with the expression $w = w(1)w(2)...w(k)$. We define $g(w) = f(w(1))...f(w(k))$. Then g extends f and is a morphism semigroups. The uniqueness results from construction.

Course 2

Formal languages

Fix the (finite) alphabet Σ.

Definition 2.1 We call *(formal) language* over Σ a subset $L \subset \Sigma^*$ i.e. L is an element form the set of all parts of Σ^* i.e. $L \in \mathscr{P}(\Sigma^*)$. An element from L is called *proposition* or *statement*. If L is a finite set we say that L is a *finite language*; otherwise we have an *infinite language*. By convention, the empty set is accepted as the *empty language* over every alphabet as well as $\{\varepsilon\}$.

Remarks 2.2 i) $\emptyset \neq \{\varepsilon\}$; so, the first language is without words while the second language has one word.
ii) The soft languages (e.q. *Pascal*, *C*, *Java*) are over the ASCII alphabet and are infinite.
iii) We must care about the mode (rule) of definition for a language. Example, the languages over the binary alphabet $L_1 = \{0^n 1^n; n \in \mathbb{N}\}$ şi $L_2 = \{$words with the same number of 0 and 1$\}$ are different since:
$L_1 = \{\varepsilon, 01, 0011, 000111, ...\}, L_2 = \{\varepsilon, 01, 10, 0011, 1001, 1010, 1100, 0101, ...\}$.

Remarks 2.3 For the nonempty language L we consider the set $alph(L) = \{a \in \Sigma; \exists u, v \in \Sigma^*$ a. î $uav \in L\}$. Then $alph(L)$ is the minimal alphabet over with L can be defined.
I) Let us prove that $alph(L)$ is nonempty.
I1) suppose $\varepsilon \in L$; then considering $u = v = \varepsilon$ it results that $a = \varepsilon \in alph(L)$.
I2) suppose $w \in L$, $w \neq \varepsilon$. Let $u = \varepsilon$ and $v = w(2)...w(|w|)$ if $|w| > 1$ respectively $v = \varepsilon$ if $|w| = 1$; hence $a = w(1) \in alph(L)$.
II) Since $alph(L) \subset \Sigma$ and Σ is finite it results that $alph(L)$ is finite. So $alph(L)$ is an alphabet.

III) The fact that $L \subset (alph(L))^*$ is immediately using the same arguments as above.
IV) The fact that $alph(L)$ is minimal with property III is proposed as Homework.

Boolean operations with languages 2.4: from the general theory of sets we have that given the languages L_1 and L_2 then there are languages the following sets:
i) union $L_1 + L_2 := L_1 \cup L_2$, ii) intersection $L_1 \cap L_2$,
iii) complement $\bar{L_1} := \Sigma^* \setminus L_1$.
Due to the algebraic structure of Σ^* we can define the *product* $L_1L_2 = \{uv; u \in L_1, v \in L_2\}$ which is obviously u language. Is useful the exponential notation; for the language L we have the languages $L^0 = \{\varepsilon\}$, $L^n = L^{n-1}L$ with non-null n.

Definition 2.5 The language L is called *free to concatenation* if $L \cap \cup_{n\geq 2}L^n = \emptyset$.

Examples 2.6 i) For the binary alphabet and the language $L = \{01, 110\}$ we have:
$L^2 = \{0101, 01110, 11001, 110110\}$,
$L^3 = \{010101, 0111001, 1100101, 11011001, 0101110, 01110110, 11001110, 110110110\}$.
ii) The product of languages is noncommutative: with $L_1 = \{\varepsilon, 101\}$ and $L_2 = \{0, 0101\}$ we have $L_1L_2 = \{1010, 1010101, 0, 0101\} \neq \{0101, 0101101, 0\} = L_2L_1$.

Example 2.7 Let $\Sigma = \{1, 2, 3\}$. $L = \{21, 213, 1, 2222, 3\}$ is a language over Σ and $213222213 \in L^4 \cap L^5$ since $213, 2222, 1, 3 \in L$ and $21, 3, 2222, 1, 3 \in L$. Also $213222213 \in \Sigma^9$.
$M = \{2, 13, 222\}$ is a language over Σ and $213222213 \in L^4 \cap L^5 \cap M^5 \cap M^7$ although $L \cap M = \emptyset$.

Definition 2.8 Given the language L we define $L^+ = \cup_{n\geq 1}L^n$ called *positive closure* respectively $L^* = \cup_{n\geq 0}L^n$ called *star closure* or *Kleene closure* of L. (See http://en.wikipedia.org/wiki/Stephen_Cole_Kleene for a biography.)

Remark 2.9 i) The notation above is inspired by the case $L = \Sigma$ when $L^+ = \Sigma^+$ and $L^* = \Sigma^*$.
ii) For an arbitrary language L we have that L^+ is a subsemigroup in Σ^+ and L^* is submonoid in Σ^*.

Proposition 2.10 i) $\varepsilon \in L^*$, $L^+ = L^*L = LL^*$, $L^* = LL^* + \{\varepsilon\}$
ii) $\varepsilon \in L^+$ *if and only if* $\varepsilon \in L$,
iii) $L_1^* + L_2^* \subseteq (L_1 + L_2)^*$, $(L_1 \cap L_2)^* \subseteq L_1^* \cap L_2^*$,
iii) $L_1(L_2L_1)^* = (L_1L_2)^*L_1$,
iv) $L_1^*(L_2L_1^*)^* = (L_1 \cup L_2)^*$,
v) $L_3(L_1 \cup L_2L_3)^*L_2 = (L_3L_1^*L_2)^+$.

Definition 2.11 The language L is called *regular* it is empty or it can be obtained elements from Σ using only the operations: product, $+$ and $*$.

Examples 2.12 i) $L = \{$binary non-null numbers$\}$ is regular over the binary alphabet since $L = 1(0+1)^*$; a binary non-null number begins with 1 and continues with a sequence (possibly empty) of 0 and 1. Here and in all what follows, for a simple writing we give up to brakets.
ii) $\mathbb{Z}$ is $0+(\varepsilon+m)d(0+d)^*$ over $\Sigma = \{0, ..., 9, m = -\}$ with $d = 1+2+...+9$.
iii) $\mathbb{Q} = 0 + (\varepsilon + m)d(0 + d)^* + (\varepsilon + m)(0 + (d(0 + d)^*))p(0 + d)^*d$ over $\Sigma = \{0, ..., 9, m, p = .\}$ where "." is the dot =decimal comma.
iv) Binary numbers with even blocks of $1 = (0 + 11)^*$.
v) Binary numbers containing $1011 = (0 + 1)^*1011(0 + 1)^*$.

Theorem 2.13 i) *A finite language is regular.*
ii) If L_1 *and* L_2 *are regular then* $L_1 \cap L_2$ *is regular.*
iii) If L *is regular then* $\bar{L}$ *is regular.*

Definition 2.14 For the words $w, v \in \Sigma^*$ we say that v is a *sub-word* of w if there exists $x, y \in \Sigma^*$ such that $w = xvy$. If in addition, x or y are different to ε we says that v is a *proper sub-word* of w. x is called *prefix* or *initial* of w respectively *proper prefix* if is different to ε; analog y is *suffix* or *terminal* of w respectively *proper suffix* if is different to ε.

A generalization of Example 2.3 is as follows: for $w \in \Sigma^*$ let $Sub_P(w)$ be the set of proper sub-words of w. Then $Sub_P(w)$ is a language over Σ. Analog, starting with the language L we get the language $P(L) = \cup_{w \in L} Sub_P(W)$.
For example, if $L = \{0, 010, 111\}$ then $P(L) = \{0, 1, 01, 10, 11\}$.

Definition 2.15([12, p. 55]) The equivalence relation R on the monoid Σ^* is called *quasi-congruence* if uRv implies $xuyRxvy$ for all $x, y \in \Sigma$. If in addition, R has a finite number of classes we say that R is with *finite index.*

Examples 2.16 i) Every congruence is a quasi-congruence.
ii) Fix the language L and define uR_Lv if for all $x, y \in \Sigma^*$ with $l(x) = l(y)$

from $xuy \in L$ it results $xvy \in L$ and conversely. Then R_L is an equivalence on Σ^*.

Proposition 2.17 *R_L is a quasi-congruence.*

Proof Let $u, v, a, b \in \Sigma^*$ with $l(a) = l(b)$ and uR_Lv. Then $aub \in L$ if and only if $avb \in L$. Let $a = a'x$ and $b = yb'$ with $x, y \in \Sigma$. We have $a'xuyb' \in L$ if and only if $a'xvybl \in L$ with $l(a') = l(b')$. Hence $xuyR_Lxvy$ for all $x, y \in \Sigma$. $\square$

Proposition 2.18 *The language L is saturated by R_L i.e. L is the union of all classes of equivalence with respect to R_L.*

Proof From definition with $x = y = \varepsilon$ it results uR_Lv if we have the equivalence: $u \in L$ if and only if $v \in L$, which says that every class of equivalence with respect to R_L belongs to L. $\square$

Propositia 2.19 *Every quasi-congruence which saturates L is a refinement of R_L.*

Useful Web pages:
1) http://en.wikipedia.org/wiki/Formal_language
2) http://mathworld.wolfram.com/FormalLanguage.html
3) http://planetmath.org/encyclopedia/ImproperLanguage.html

SEMINAR 2

S2.1 Let $\Sigma = \{1, 2, 3, 4\}$ and the languages $L = \{12, 4\}$, $M = \{\varepsilon, 3\}$. Find:
i) LM, ML, L^2M, L^2M^2,
ii) LM^2L, M^+, LM^+, M^*.

Solution i) $LM = \{12, 3, 123, 43\}$, $ML = \{12, 4, 312, 34\}$, $L^2M = \{1212, 44, 12123, 443\}$, $L^2M^2 = \{1212, 44, 121233, 44333\}$.
ii) $LM^2L = \{1212, 44, 123312, 4334\}$, $M^+ = \{\varepsilon, 3, 33, 333, ...\}$, $LM^+ = \{12, 4, 123, 43, 1233, 433, ...\}$, $M^+ = M^*$ since $\varepsilon \in M$.

S2.2 For the previous Σ how many elements from Σ^4 begin with 22?

Solution There are 4 symbols in Σ which must be arranged in pairs to form the last two digits of required elements$=4 \cdot 4 = 16$. The asking set is $\{2211, 2222, 2233, 2244, 2212, 2221, 2213, 2231, 2214, 2241, 2223, 2232, 2224, 2242, 2234, 2243\}$.

S2.3 For $\Sigma = \{1, 2, 3, 4, 5\}$ find:
i) the number of elements in $\Sigma^0 + \Sigma^2$,
ii) the number of elements in Σ^6 which begins with 22 and ends with 1.

Solution i) $|\Sigma^0 + \Sigma^2| = 1 + 5^2 = 26$. ii) $5^3 = 125$.

S2.4 Let Σ from Ex. 2.1.
i) How many words are in Σ^+ with the length ¡ 5?
ii) How many words are in Σ^* with the length ¡ 5?
iii) How many words are in Σ^+ beginning with 12 and of length ¡ 5?

Solution i) $|\Sigma^1 + ... + \Sigma^4| = 4 + 4^2 + 4^3 + 4^4 = 4 + 16 + 64 + 256 = 4(1 + 4 + 16 + 64) = 4 \cdot 85 = 340$.
ii) $|\Sigma^0 + ... + \Sigma^4| = 1 + 340 = 341$.
iii) $4^0 + 4^1 + 4^2 = 1 + 4 + 16 = 21$ since the required set is $\{12, 12x, 12xy; x, y \in \Sigma\}$.

S2.5 Let L be a language containing ε and other 8 different elements. How many elements are in L^2?

Solution $L^2 = L + \{xy; x, y \in L \setminus \{\varepsilon\}\}$; so $|L^2| = 9 + 8^2 = 9 + 64 = 73$.

S2.6 For $\Sigma = \{0, 1, 2\}$ decide if the following statements belong to the language $L = (0 + 1)^*2(0 + 1)^*22(0 + 1)^*$:
i) 120120210, ii) 1222011, iii) 22210101.

Solution i) No, since an element of L contains 22 which is not in the given statement.
ii) Yes, writing (1)2(ε)22(011).
iii) No, since an element of L begins with 0 or 1 while the given statement begins with 2.

S2.7 The same problem for $\Sigma = \{0, 1, 2, 3\}$,
$L = (0+1)^*2(0+1)^+22(0+1+3)^*$ and: i) 120122, ii) 1222011, iii) 3202210101.

Solution i) Yes, writing (1)2(01)22(ε).
ii) No, the third digit can be 0 or 1 and it begins with 2 and the given statement does not satisfies these conditions.
iii) No, since an element of L begins with $0, 1$ or 2.

S2.8 The same problem for Σ of Ex. 2.6, $L = (0 + 1)^*(102)(1 + 2)^*$ and:
i) 012102112, ii) 1110221212, iii) 10211111, iv) 102, v) 001102102.

Solution i) No, since before 102 it appears $012 \notin (0 + 1)^*$.
ii) Yes, writing 11(102)21212.
iii) Yes, writing ε(102)11111.
iv) Yes, writing ε(102)ε.
v) No, since after 102 it appears 102 which do not belongs to $(1 + 2)^*$.

S2.9 For Σ of Ex. 2.7 find the following languages:
i) $L = \{$statements containing only one 2$\}$,

ii) $L = \{$statements containing three times 2$\}$,
iii) $L = \{$statement containing the digit 2$\}$,
iv) $L = \{$statements containing the word 2112$\}$,
v) $L = \{$statements containing digit 2 in words of length 3$\}$.

Solution i) $L = (0+1+3)^*2(0+1+3)^*$,
ii) $L = (0+1+3)^*2(0+1+3)^*2(0+1+3)^*2(0+1+3)^*$,
iii) $L = (0+1+3)^*2^+(0+1+3)^*$,
iv) $L = (0+1+2+3)^*2112(0+1+2+3)^*$,
v) $L = ((0+1+3)^*2^3)^+$.

S2.10 (Program C) Darea de la tastatură a unui caracter, afişarea acestuia, a predecesorului şi succesorului şi a codurilor ASCII aferente.

```
//predecesorul şi succesorul unui caracter şi codurile lor ASCII
#include<iostream.h>
void main ()
{
  char w, w1;
  cout<< "Dati litera:= ";
  cin>>w;
  w1=w;
  cout<< "Codul ASCII al literei " <<w<< " este: " <<hex<<int(w)<<
" " <<dec<<int(w)<<endl;
  cout<< "Codul ASCII al literei " << --w<< " este: " <<hex
<<int(w-1)<< " " <<dec<<int(w-1)<<endl;
  cout<< "Codul ASCII al literei " <<++w1<< " este: " <<hex
<<int(w1+1)<< " " <<dec<<int(w1+1)<<endl;
}
```

Exemplu:
Dati litera:= H <Enter>
Codul ASCII al literei H este: 48 72
Codul ASCII al literei G este: 47 71
Codul ASCII al literei I este: 49 73
(Pe prima coloană apare codul hexazecimal iar pe a doua coloană cel zecimal.)

Temă Scrieţi varianta Pascal a problemei.

S2.11 (Program C) Să se afişeze codurile, în hexazecimal şi zecimal, ale tuturor literelor.

```
//codurile ASCII ale tuturor literelor majuscule şi minuscule
#include<iostream.h>
void main ()
{
  char lit;
  int i;
  for (lit='A'; lit<='z'; lit++)
    {
      i=lit;
      cout<< "Codul ASCII al literei " <<lit<< " este: " <<hex<<i<<
"" <<dec<<i<< endl ;
    }
}
```

Temă Scrieţi varianta Pascal a problemei.

S2.12 (Program C) Dat un număr natural se cere caracterul corespunzător.

```
//conversia din întreg în caracter
#include<iostream.h>
void main ()
{
  int n;
  cout<< " Dati numarul natural n:= ";
  cin>>n;
  cout<< " Caracterul corespunzator lui " <<n<< " este:= " <<char(n)
<<endl;
}
```

Exemplu:
Dati numarul natural n:= 321 <Enter>
Caracterul corespunzator lui 321 este:= A
(Un caracter ocupă un singur octet spre deosebire de un întreg care ocupă 2 octeţi. De aceea se calculează numărul dat modulo 2^8=256, în exemplul nostru este 65 iar 65 este codul în zecimal al lui A.)

Temă Scrieţi varianta Pascal a problemei.

S2.13 (**Hamming distance on k-words**) For $k \in \mathbb{N}$ we define $d_H : \Sigma^k \times \Sigma^k \to \mathbb{R}_+, d_H(a,b) = |\{i; a(i) \neq b(i)\}|$; so $d_H(a,b)$ is a counter for the number of different symbols of a comparing b. Prove that d_H is a *metric* on Σ^k:

I) (positivity) $d_H(a,b) \geq 0; d_H(a,b) = 0$ if and only if $a = b$,
II) (symmetry) $d_H(a,b) = d_H(b,a)$,
III) (triangle inequality) $d_H(a,c) \leq d_H(a,b) + d_H(b,c)$.

S2.14 Let a nonempty language L satisfying $L^2 = L$. Prove:
i) if L has only one element then $L = \{\varepsilon\}$,
ii) if L has more elements then there exists $w \in L$ with $w \neq \varepsilon$ and then for a $v \neq \varepsilon$ we have $l(w) \leq l(v)$. Find from this that $\varepsilon \in L$,
iii) $L^* = L$.

Solution i) Let $L = \{w\}$ and $k = l(w)$. Then $L^2 = \{w^2\}$ with $l(w^2) = 2k$. Hence $2k = k$ i.e. $k = 0$.
iii) We have $L^3 = L^2 = L$ and analog for all $n \geq 1$ we have $L^n = L$; hence $L^+ = L$. But $\varepsilon \in L$; then $L^* = L^+ + \varepsilon = L + \varepsilon = L$.

S2.15 În **Matlab**, fie caracter (*char*) este reprezentat intern printr-o valoare numerică corespondentă. astfel, în locul accesării acestor valori, se poate lucra cu caractere, aşa cum apar acestea pe ecran. Un şir de caractere (*string*) eset, deci, un vector ale cărui elemente sunt codurile numerice corespunzătoare caracterelor de pe ecran.

Valorile de tip caracter se definesc folosind apostrofuri. De exemplu, se defineşte variabila x, cu valoare de tip caracter şi se verifică cu funcţia **class** tipul acesteia:

```
>> x='a';
>> class(x)
ans  =
char
```

Similar pentru şiruri de caractere:

```
>> y='abcd';
>> whos y
Name    Size    Bytes    Class
y       1x4     8        char array
Grand total is 4 elements using 8 bytes
```

Course 3

Regular expressions

As in the previous Course we fix from the beginning a finite alphabet Σ.

Definition 3.1 Let us consider the symbols $), (, +, *$ not belonging to Σ. We call *regular expression* a word w of the alphabet $\Sigma \cup \{\varepsilon, (,), +, *\}$ having one of the following form:
i) $w \in \Sigma$ or $w = \varepsilon$,
ii) $w = (xy)$ with x, y regular expressions,
iii) $w = x + y$ with x, y regular expressions,
iv) $w = x^*$ with x a regular expression.
Let $Rex(\Sigma)$ be the set of regular expressions.

Remarks 3.2 i) The way of defining of regular expression is inductive.
ii) As in the previous Course sometimes we give up to redundant brackets when the framework is obviously.
iii) By convention, $*$ has the maximum priority, it follows concatenation and after it the sum $+$. So, $((((xx) + y)^* + ((xy))x))$ is in fact $(xx + y)^* + xyx$.

Definition 3.3 The map $K : Rex(\Sigma) \to \mathscr{P}(\Sigma^*)$ given by:
a) $K(\varepsilon) = \emptyset$, $K(x) = \{x\}$ if $x \in \Sigma$,
b) $K(xy) = K(x)K(y)$, $K(x + y) = K(x) + K(y)$,
c) $K(x^*) = (K(x))^*$,
is called *interpretation.*

Theorem 3.4 (**Kleene**) *The language $L \in \mathscr{P}(\Sigma^*)$ is regular if and only if it belongs to the range of the interpretation map K i.e. there exists a regular expression $w \in Rex(\Sigma)$ such that $L = K(w)$.*

Examples 3.5 i) Let $x, y \in \Sigma$. Hence $K((x+y)(x+y)) = \{xx, xy, yx, yy\}$.
ii) $K(\varepsilon + (x + y)^*(yx + xyx)) = \{x, y\}^*\{yx, xyx\}$.

Definition 3.6 The regular expressions $w_1, w_2 \in Rex(\Sigma)$ are called *equivalent* if $K(w_1) = K(w_2)$. We denotes $w_1 = w_2$.

Examples 3.7 i) $w_1 + w_2 = w_2 + w_1$ and $\varepsilon + w = w + \varepsilon = w$ for all $w_1, w_2, w \in Rex(\Sigma)$.
ii) $xx^*y + y = x^*y$.

Proposition 3.8 *By denoting with E some regular expressions we have the following equivalences*:
i) $(E^*)^* = E^*$, $E_1^*E_2^* = (E_1 + E_2)^*$,
ii)(*associativity*) $E_1(E_2E_3) = (E_1E_2)E_3$, $E_1 + (E_2 + E_3) = (E_1 + E_2) + E_3$,
iii)(*distributivity*) $E_1(E_2+E_3) = E_1E_2+E_1E_3$, $(E_1+E_2)E_3 = E_1E_3+E_2E_3$.

Definition 3.9 The regular expression $E \in Rex(\Sigma)$ is called *ambiguă* if there exists a statement in $K(E)$ which can be described in two different modes (without using the void word at concatenation!).

Example 3.10 $E = (xy)^* + (x + y)^*$ is ambiguă since $xy \in K(E)$ can be written $xy + \varepsilon$ also as element in $\varepsilon + (x + y)^*$.

In some programming softwares we find expressions of type:

$$< \text{literă} >::= A|B|...|Z, \quad < \text{cifră} >::= 0|1|...|9,$$

$$< \text{identificator} >::=< \text{literă} > (< \text{literă} > | < \text{cifră} >)$$

which are exactly regular expressions having the variables $<$ literă $>$, $<$ cifră $>$ and $<$ identificator $>$ while the symbol $|$ plays the role of $+$.

For the fixed language L we define the relation: $\sigma_L = \{(w, z) \in \Sigma^* \times \Sigma^*; uwv \in L \Leftrightarrow uzv \in L, \forall u, v \in \Sigma^*\}$. Since the logic equivalence is an equivalence relation we have that σ_L is an equivalence relation. We prove that is moreover a congruence. Let $(w, z) \in \sigma_L$ and $x \in \Sigma^*$ arbitrary. Then $(xw, xz) \in \sigma_L$ and $(wx, zx) \in \sigma_L$ obviously due the rules of simplification in the monoid Σ^*. σ_L is called *the syntactic congruence* associated to the language L and the quotient monoid $Syn(L) = \Sigma^*/\sigma_L$ is called *syntactic monoid* of L.

Definition 3.11 The language L is *recognized by the monoid M* if there exists a morphism of monoids $\varphi : \Sigma^* \to M$ and $P \subset M$ such that $L = \varphi^{-1}(P)$.

Proposition 3.12 *L is recognized by the syntactic monoid $Syn(L)$.*
Proof The projection map $\pi : \Sigma^* \to Syn(L)$ is a morphism of monoids and we consider $P = \pi(L)$. Hence $L = \pi^{-1}(P)$. $\square$

The utility of the syntactic monoid is given by:

Theorem 3.13 *The following statements are equivalents*:
i) L *is regular.*
ii) $Syn(L)$ *is finite i.e.* σ_L *has a finite index.*
iii) L *is recognized by a finite monoid.*

Definition 3.14 i) Given the word $w = x_1...x_n \in \Sigma^*$ its *reverse* is the word $w^R = x_n...x_1$. We denotes also $Mi(w)$ from the English word: mirror = oglindă.
ii) The word w is called *palindrome* if $w^R = w$. Let $Pal(\Sigma)$ be the set of these symmetric words.
iii) Given the language L we define its *reverse* as the language $L^R = \{w^R; w \in L\}$.

Example 3.15 If Σ has two distinct letters a, b then $\{a^nba^n; n \in \mathbb{N}^*\} \subset Pal(\Sigma)$.

Proposition 3.16 i) $(L_1 + L_2)^R = L_1^R + L_2^R, (L_1L_2)^R = L_2^R L_1^R, (L^*)^R = (L^R)^*$.
ii) L *is regular if and only if* L^R *is regular.*

Înainte de a părăsi subiectul *limbaje* vom cita opiniile deosebit de interesante ale fizicianului Fritjof Capra din [1, p. 87-88]: "... limbajul este un sistem de comunicare simbolică. Simbolurile sale-cuvinte, gesturi şi alte semne-servesc ca indicii pentru coordonarea lingvistică a acţiunilor. Aceasta la rândul ei, creează noţiunea de obiecte, şi astfel simbolurile devin asociate cu imaginile noastre mentale ale obiectelor". În aceeaşi carte, la paginile 92-94, este prezentat ASL-American Sign Language ca exemplu de limbaj utilizat în dialogul cu persoanele surde dar şi cu cimpanzeii. La pagina 94 începe secţiunea *Originile limbajului uman* ce prezintă o foarte interesantă teorie ce pune vorbirea în conexiune cu mişcările mâinilor ca fiind "controlate de aceeaşi regiune motoare a creierului" (pag. 95). Citătm de la pagina 97: "Apariţia cuvintelor în comunicarea strămoşilor noştri a adus avantaje imediate. Cei care comunicau vocal, o puteau face şi când aveau mâinile ocupate, sau când interlocutorul era întors cu spatele. În cele din urmă, aceste avantaje evolutive vor aduce schimbările anatomice necesare vorbirii articulate desăvârşite. Timp de zeci de mii de ani, pe măsura evoluţiei traiectului nostru vocal, oamenii au comunicat printr-o combinaţie de gesturi pecise şi cuvinte vorbite până când, în final, vorbirea a surclasat semnele şi a devenit forma dominantă de comunicare umană. Chiar şi azi, folosim gesturile atunci când limbajul vorbit nu ne serveşte."

Useful Web pages:
1) http://en.wikipedia.org/wiki/Regular_language
2) http://mathworld.wolfram.com/RegularExpression.html
3) http://planetmath.org/encyclopedia/RegularLanguage.html

SEMINAR 3

S3.1 Finds examples of palindromes in Romanian language.

Solution 1) with 3 letters: *apa, ara, aba, aţa, aşa, bob, coc, ere, mim, rar, pop, sas, tot.*
2) with 4 letters: ?.
3) with 5 letters: *anina, caiac, capac, cojoc, etate, minim, soios, potop, reper, rotor, tivit.* În DEX'98, există în jur de 35 de palindroame de 5 litere.
4) 9 letters: *aerisirea.*
5) 7 letters: *rotitor, elevele, atacata, rotator, rolelor, ataşata, Elenele, etajate, ala-bala, etalate, aerarea.*

S3.2 Propozitii şi expresii palindromice:
ELE NE SEDUC CU DESENELE.
ICRE, PUI, CIUPERCI.
ELE FAC CAFELE.
ENE PURTA PATRU PENE.
O RAMĂ MARO.
ERA O TIPĂ RĂPITOARE !
NOI VOTĂM, MĂ, TOV. ION.

Dialoguri palindromice:
- EREMIA, AI MERE?
- O, N-AM, ANO!
-ACU, DUDUCA!
- IZA, CAZI ?
- DA, CAD!
- A... DA !

Scrisoare palindromica :
DRAGA ILEANA, ACI M-A ATACAT RADA CU LUCA, DAR TAC. A TA AMICA, ANA-ELIA GARD.

S3.3 (Program C): Dat un număr nr se cer numărul său de cifre, suma cifrelor sale, produsul cifrelor nenule, cifra de ordin k (cu k dat de la tastatură) şi inversatul său.

```
//operatii cu cifrele unui numar
```

```
#include<iostream.h>
#include<math.h>
void main()
{
  int n,k, nr, N=0, cifra, rasturnat=0;
  int suma=0, prod=1, cifrak;
  cout<<"dati numarul nr:= ";
  cin>>nr;
  cout<<"dati ordinul cifrei k:= ";
  cin>>k;
  n=nr;
  while (n!=0)
    {
    cifra=n%10;
    N++;
    suma+=cifra;
    if (cifra!=0) prod*=cifra;
    if (N==k) cifrak=cifra;
    rasturnat=rasturnat*10+cifra;
    n=n/10;
    }
  cout<<"numarul are:= "<<N<<" cifre"<<endl;
  cout<<"suma cifrelor este:= "<<suma<<endl;
  cout<<"produsul cifrelor nenule este:= "<<prod<<endl;
  cout<<"cifra de ordin "<<k<<" este:= "<<cifrak<<endl;
  cout<<"numarul rasturnat este:= "<<rasturnat<<endl;
}
```

Exemplu:
Dati numarul nr:= 987654321 <Enter>
Dati ordinul cifrei k:= 3 <Enter>
numarul are:= 9 cifre
suma cifrelor este:= 45
produsul cifrelor nenule este:= 362880
cifra de ordin 3 este:= 3
numarul rasturnat este:= 123456789

Temă Scrieţi varianta Pascal a problemei.

S3.4 (Program C) Se dă un număr natural N mai mare strict ca 2. Se cere afişarea unui şir de lungime N cu primele N/2 elemente în ordine crescătoare iar ultimele N/2 elemente în ordine descrecătoare.

```
/*pentru N se cere un sir de lungime N cu primele N/2 elemente crescatoare si urmatoarele elemente descrescatoare*/
#include<iostream.h>
void main ()
{
  int N, i;
  cout<<"Dati N(>2):= ";
  cin>>N;
  for (i=1;i<=N/2;i++)
    cout<<i<<" ";
  for (i=(N+N%2)/2;i;i–)
    cout<<i<<" ";
  cout<<endl;
}
```

Exemplu: Pentru N=7 apare 1 2 3 4 3 2 1, iar pentru N=8 apare 1 2 3 4 4 3 2 1.

Temă Scrieţi varianta Pascal a problemei.

S3.5 În **Matlab** funţia **strcat(s1, s2,s3, ...)** concatenează pe orizontalaă şirurile **s1, s2, s3, ...**. Spaţiile finale ale şirurilor care se concatenează sunt ignorate.

```
>> s1='elemente ';
>> s2='de baza in ';
>> s3='Matlab';
>> strcat(s1, s2, s3)
ans  =
elementede baza inMatlab
```

Tot pentru concatenare se poate utiliza operatorul [ ]. De această dată, în şirul rezultat avem spaţiile libere de la capătul şirurilor s1 şi s2:

```
>> [s1 s2 s3]
ans  =
elemente de baza in Matlab
```

Funcţia **strcvcat(t1, t2, t3, ...)** concatenează pe verticală şirurile **t1, t2, t3, ...** constituind un tablou de caractere:

```
>> t1='abc';
>> t2='def';
>> t3='ghi';
```

```
>> strvcat(t1, t2, t3)
ans  =
abc
def
ghi
```

Course 4

Varieties of monoids

LeT M be a monoid and the congruences $C_i, i \in I$ on M. Hence $C = \cap_{i \in I} C_i$ is a congruence on M. Let R be an arbitrary relation on M considered as a subset in $M \times M$ and let us remark that there exists congruences which include it in the sense of inclusion of sets e.g. *the total congruence* $M \times M$.

With this argument it results the existence of intersection of all congruences containing R which we denote $R^\sharp$ conform [8, p. 197]; then $R^\sharp$ is the small (in the sense of order relation provided by the inclusion of subsets of $M \times M$) congruence containing R: if C is a congruence and $R \subset C$ then $R^\sharp \subseteq C$.

Definition 4.1 $R^\sharp$ is called *the congruence generated by* R.

Let us recall that for regular languages we are interested in congruences of finite index; for this issue we provide:

Proposition 4.2 *Let Σ be an alphabet and C a congruence on Σ^* of finite index finit. The there exists a finite subset T of $\Sigma^* \times \Sigma^*$ such that $C = T^\sharp$. In other words, ever congruence of finite index on the monoid of words for a given alphabet is of "sharp" type.*

Proof Let $w \in \Sigma^*$ and its equivalence class $[w]$ with respect to C. Let us define $l : \Sigma^*/C \to \mathbb{N}$ through $l([w]) = \min\{|z|; z \in [w]\}$. Since C is of finite index there exists $m_C = \max\{l([w]; [w] \in \Sigma^*/C)\}$; let $k_C = m_C + 1$. It results that for every $[w] \in \Sigma^*/C$ there exists $z \in [w]$ with $|z| < k_C$; in a contrary case we have $l([w]) \geq k_C \geq 1 + l([w])$, false. Let $T = \{(u, v) \in \Sigma^* \times \Sigma^*; (u, v) \in C, |u| \leq k_C, |v| \leq k_C\}$ and define R to be the congruence $T^\sharp$. From $T \subset \cup_{i=0}^{k_C} \Sigma^i$ it results that T is a finite set.

From definition $T \subset C$ and hence $R \subset C$. It remains to prove that $C \subset R$. We prove it by reduction to absurdity. Then let $(u, v) \in C$ with

$(u, v) \notin R$. We can suppose, eventual schimbând ordinea datorită simetriei relaţiei de echivalenţă C, that $|u| \geq |v|$; also, we can consider u and v such that $|u| + |v|$ is the lower possibility. We can not have $|u| < k_C$ since we will obtain $|v| < k_C$ which says that $(u, v) \in R$. Then let $|u| \geq k_C$; then $u = zu'$ with $|u'| = k_C$. There exists $v' \in [u']$ with $|v'| < k_C$. Hence, $(u', v') \in T \subset R \subset C$. From $[v] = [u] = [zu']$ and $[zu'] = [zv']$ it results by transitivity that $[v] = [zv']$. If we suppose that $(v, zv') \in R$ it follows from this relation and $u = zu', (zu', zv') \in R$ that $(u, v) \in R$ which is contrary to the initial supposition.

In conclusion, $(v, zv') \in C$, $(v, zv') \notin R$ and $|v| + |zv'| < |v| + |zu'| = |u| + |v|$ contrary to the initial minimality hypothesis. □

Definition 4.3 A non-void subset I of the monoid M is called *ideal* if from $a \in I$ and arbitrary $s \in M$ it results $as \in I$ and $sa \in I$.

The ideals are an important source of congruences. So, given an ideal I we define the relation C_I on M by $(a, b) \in C_I$ if $a = b$ or a and b belongs both to I. Is a simple verification that C_I is a congruence on M having ad equivalence classes the singleton sets $\{a\}$ for $a \in M \setminus I$ respectively the set I. The quotient monoid M/C_I is denoted M/I and the computation rules are:
I) $\{a\}\{b\} := \{ab\}$ if $ab \notin I$ respectively I in the opposite case,
II) $\{a\}I = I\{a\} = II = I$
and then I is a "zero" of M/I. Hence, we can though M/I as being $M \setminus I$ plus a zero and the product of two non-zero elemente $\{a\}$ and $\{b\}$ is or the non-zero element $\{ab\}$ if this does not belongs to I, or the given zero. This type of congruence is called *Rees congruence Rees* and the quotient monoid M/I is called *Rees monoid* after the name of its inventor.

Definition 4.4 A class $\mathscr{V}$ of monoids is called *variety* if it is closed to the consideration of submonoids, quotient monoids and direct products.

So, $\mathscr{V}$ is a variety if:
V1) when $M \in \mathscr{V}$ and S is a submonoid of M we have $S \in \mathscr{V}$,
V2) when $M \in \mathscr{V}$ and C is a congruence on M we have $M/C \in \mathscr{V}$,
V3) when $M_i \in \mathscr{V}, i \in I$ we have $\prod_{i \in I} M_i \in \mathscr{V}$.

Examples 4.5 There are varieties the following classes:
i) the class Com of commutative monoids,
ii) the clasc Bm of *band monoids* in which every element is an idempotent.
Contrary, the class of groups is not a variety; for example if we consider the infinite cyclic group $M =< a >$ its submonoid of positive powers $\{1, a, a^2, ...\}$ is not a group.

Since for the same reason of regular languages we are interested in the finite case, we consider:

Definition 4.6 A class $\mathscr{V}$ of finite monoids is called *F-variety* is V1, V2 hold together with:
V3F) when $M_i \in \mathscr{V}, i \in \{1,2\}$ then $M_1 \times M_2 \in \mathscr{V}$.

Remark 4.7 It is obviously that instead of V3F we can work with:
V3Fn) if $M_i \in \mathscr{V}, i \in \{1,...,n\}$ then $M_1 \times ... \times M_n \in \mathscr{V}$ for every $n \geq 2$.

Examples 4.8 There are F-varieties:
i) the class of all finite monoids, particularly the class of all finite monoids from a given variety $\mathscr{V}$.
ii) the class of all finite groups.
iii) Let be given a non-void collection $\mathscr{V}_i, i \in I$ of F-varieties. Then $\cap_{i\in I}\mathscr{V}_i$ is a F-variety.
iv) Let $\mathscr{M} = \{M_j, j \in J\}$ be a non-void collection of finite monoids and $\mathscr{V}$ the intersection of all F-varieties containing $\mathscr{M}$. This intersection exists since we can consider the F-variety of all finite monoids. From iii) we have that $\mathscr{V}$ is a F-variety, the small in the sense of inclusion which contains $\mathscr{M}$. We say that $\mathscr{V}$ *is generated* by $\mathscr{M}$ we denotes sometimes as $\mathscr{B}(M_j; j \in J)$.

Definition 4.9 Let $u, v \in \Sigma^*$. We say that the monoid M *satisfies the equation* $u = v$ if for every morphism of monoids $\varphi : \Sigma^* \to M$ one have $\varphi(u) = \varphi(v)$. Also we say that $u = v$ is *the equation* of M.

Examples 4.10 i) The equation of the commutative monoid is $xy = yx$.
ii) The equation of band monoid is $x^2 = x$.

If we have a sequence (finite of infinite) (u_n, v_n) of pairs of elements from Σ^* then we can consider the class $\mathscr{C}$ of all monoids having the equations $u_n = v_n, n \geq 1$. It follows immediately that $\mathscr{C}$ is a variety and we denote it $\mathscr{V}(u_n = v_n, n \geq 1)$. For example, $Com = \mathscr{V}(xy = yx)$ and $Bm = \mathscr{V}(x^2 = x)$. In 1935, Birkhoff proves the converse result (more difficult) that every variety is defined by equations.

Fix the F-variety $\mathscr{V}$ and the alphabet Σ. Let $\mathscr{L}_{\mathscr{V}}(\Sigma)$ be the set of langauges over Σ having the syntactic monoid in $\mathscr{V}$. Any language from $\mathscr{L}_{\mathscr{V}}(\Sigma)$ is regulate having finite syntactic monoid. An useful criterion to test the membership to $\mathscr{L}_{\mathscr{V}}(\Sigma)$ is provided by:

Proposition 4.11 *Let L be a language over Σ. Then L belongs to $\mathscr{L}_{\mathscr{V}}(\Sigma)$ if and only if L is recognized by a monoid from $\mathscr{V}$.*

Also we have:

Proposition 4.12 *Let $\mathscr{V}$ be $\mathscr{W}$ two F-varieties of monoids. Then $\mathscr{V} \subseteq \mathscr{W}$ if and only if $\mathscr{L}_{\mathscr{V}}(\Sigma) \subseteq \mathscr{L}_{\mathscr{W}}(\Sigma)$ for any finite alphabet Σ.*

Hence we have:

Corollary 4.13 *Let $\mathscr{V}$ and $\mathscr{W}$ two F-varieties. We have $\mathscr{V} = \mathscr{W}$ if and only if $\mathscr{L}_{\mathscr{V}}(\Sigma) = \mathscr{L}_{\mathscr{W}}(\Sigma)$ for any finite alphabet Σ.*

Definition 4.14 i) A map $\mathscr{L}$ which associates to a finite alphabet Σ a subset $\mathscr{L}(\Sigma)$ of regular lanquages over Σ is called *RL-function.*
ii) A RL-function $\mathscr{L}$ is called *VRL-function* if:
1) for any alphabet Σ the set $\mathscr{L}(\Sigma)$ is closed to union and complement consideration,
2) for the alphabets $\Sigma_i, i = 1, 2$ and morphism of monoids $\varphi : \Sigma_1^* \to \Sigma_2^*$ we have that $L \in \mathscr{L}(\Sigma_2)$ implies $\varphi^{-1}(L) \in \mathscr{L}(\Sigma_1)$,
3) for any alphabet Σ, any $a \in \Sigma$ and any $L \in \mathscr{L}(\Sigma)$ we have that $a^{-1}L$ and La^{-1} belongs to $\mathscr{L}(\Sigma)$ where, by definition, $w \in \Sigma^*$ is element in $a^{-1}L$ if $aw \in L$.

Remarks 4.15 The condition 1) above implies that $\mathscr{L}(\Sigma)$ is closed to the boolean operations of union, intersection and complement. The condition 3) implies, by an inductive calculus after the length of word $u \in \Sigma^*$ that $u^{-1}L$ and Lu^{-1} are sets in $\mathscr{L}(\Sigma)$.

The map $\mathscr{L}_{\mathscr{V}}$ previously considered is a RL-function. Moreover:

Proposition 4.16 *If $\mathscr{V}$ is a F-variety then the map $\mathscr{L}_{\mathscr{V}}$ is a VRL-function.*

We have also the converse, proved by Eilenberg (1975):

Proposition 4.17 *Let $\mathscr{L}$ be a VRL-function. Then there exists a F-variety of monoids $\mathscr{V}$ such that $\mathscr{L} = \mathscr{L}_{\mathscr{V}}$.*

Examples 4.18 1) If Mon is the F-variety of all finite monoids then the corresponding VRL-function associates to the alphabet Σ the set of regular languages over Σ.
2) Let $Triv$ be the F-variety of trivial monoids, containing the singleton monoid $\{1\}$. Then $\mathscr{L}_{Triv}(\Sigma)$ contains the languages L having the syntactic monoid $Syn(L) = \{1\}$; so the syntactic congruence is the total congruence. In conclusion, $\mathscr{L}_{Triv}(\Sigma) = \{\emptyset, \Sigma^*\}$.

Another case when the VRL-function cab be explicitly given is when $\mathscr{V}$ is generated by a single monoid:

Proposition 4.19 *Let $\mathscr{V} = \mathscr{B} < M >$ be the F-variety generated by the finite monoid M and consider the alphabet Σ. Then $\mathscr{L}_{\mathscr{V}}(\Sigma)$ is the boolean*

algebra generated by the languages $\varphi^{-1}(z)$ for any $z \in M$ and any morphism of monoids $\varphi : \Sigma^* \to M$.

Example 4.20 Let $M = \{0, 1\}$ with the product $0 \cdot 0 = 0 \cdot 1 = 1 \cdot 0 = 0, 1 \cdot 1 = 1$. M satisfies the equations $x^2 = x, xy = yx$ and hence $\mathscr{B}(M)$ is included in the F-variety $\mathscr{V}(x^2 = x, xy = yx)$. In fact, one can prove the equality of these F-varieties and the last is called F-variety of *semilaticeal monoids* denoted SL.

Proposition 4.21 *For the alphabet Σ we have that $\mathscr{L}_{SL}(\Sigma)$ is the boolean algebra generated by the languages A^* with $A \subseteq \Sigma$.*

Useful Web pages:
1) http://en.wikipedia.org/wiki/Special_classes_of_semigroups
2) http://en.wikipedia.org/wiki/Boolean_algebra_(structure)
3) http://en.wikipedia.org/wiki/Semigroup_with_two_elements.

SEMINAR 4

S4.1 (**Bar-Hillel pumping lemma**) Let L be a regular language. Then there exists a positive integer number $n = n_L$ such that any word $w \in L$ with $|w| \geq n$ admits the decomposition $w = xyz$ having the following properties:
i) $0 < |y| \leq n$,
ii) $|xz| \leq n$,
iii) for any $k \in \mathbb{N}$ we have $xy^k z \in L$.

http://en.wikipedia.org/wiki/Pumping_lemma_for_regular_languages

S4.2 Using the pumping lemma prove that the following languages are not regular:
1) $L = \{a^n b^n; n \geq 0\}$,
2) $L = \{a^{n^2}; n \geq 0\}$,
3) $L = \{a^{10^n}; n \geq 0\}$,
4) $L = \{a^{2^n}; n \geq 1\}$.

Solution 1) We will obtain a contradiction with the pumping lemma. Let $w = a^n b^n \in L$ with $2n \geq n_L$ where n_L is provided by this lemma. We have three situations:
I) $x = a^l, y = a^m, z = a^{n-l-m} b^n$ with $m > 0$ from i). Let $i = 0$ in iii) from lemma; it results $a^{n-m} b^n \in L$, false.
II) $x = a^l, y = b^{n-l} b^m, z = b^{n-m}$ with $n > l$ and $m \geq 0$ from i). We have 3 subcases:
II1) $n = l$; it results $m > 0$. With $i = 0$ in iii) we have that $a^n b^{n-m} \in L$,

false.
II2) $n > l$ and $m = 0$. With $i = 0$ we have $a^l b^n \in L$, false.
II3) $n > l$ and $m > 0$. With $i = 2$ we have $a^l a^{n-l} b^m a^{n-l} b^m b^{n-m} \in L$, false since the symbol a must be before b.
III) $x = a^n b^m, y = b^l, z = b^{n-l-m}$ with $l > 0$. Consider then $i = 0$ and hence $a^n b^{n-l} \in L$ false.

S4.3 $X \subset \mathbb{N}$ is called *periodical* if is finite or there exists $N_0 \leq 0$ and $p \leq 1$ such that if $x \leq N_0$ then $x \in X$ if and only if $x + p \in X$.
1) If X is periodical then prove that $X = A \cup B$ where $A = X \cap \{i; 0 \leq i < N_0\}$, $B = \cup_{j=1}^{s} \{g(j)+pi; i \leq 0\}$ and $g(1), ..., g(s) = X \cap \{i; N_0 \leq i < N_0+p\}$.
2) $L = \{a\}^*$ is regular if and only if the set $X = \{i; a^i \in L\}$ is periodical.

Pentru a compara între ele şiruri de caractere, se pot folosi operatori relaţionali sau funcţii predefinite în Matlab. Operatorii relaţionali ($>, >=, <, <=, ==, \sim=$) compară valorile corespunzătoare caracterelor din şir, comparaţia realizându-se element cu element. De exemplu:

```
>> A='abcd';
>> B='xbyd';
>> A==B
ans  =
0    1    0    1
```

Valoarea 1 reprezintă *true* iar 0 reprezintă *false*. Astfel, valorile 1 arată locurile unde simbolul este acelaşi iar 0 indică locurile cu elemente diferite. Avem şi varianta:

```
>> B>A
ans  =
1    0    1    0
```

Latici

Definiţia 4.4 Fie L o mulţime nevidă înzestrată cu două legi de compoziţie $\vee$ ("sau"), $\wedge$ ("şi") : $L \times L \to L$. Tripletul $(L, \vee, \wedge)$ îl numim *latice* dacă sunt satisfăcute proprietăţile:
L1) (comutativitate) $a \vee b = b \vee a$, $a \wedge b = b \wedge a$,
L2) (asociativitate) $(a \vee b) \vee c = a \vee (b \vee c)$, $(a \wedge b) \wedge c = a \wedge (b \wedge c)$,
L3) (absorbţie) $a \vee (a \wedge b) = a$, $a \wedge (a \vee b) = a$.

Exemplul 4.5 Fie mulţimea nevidă A. Avem că $(L = \mathscr{P}(A), \cup, \cap)$ este o latice. Invităm cititorul să verifice absorbţia: pentru $X, Y \in \mathscr{P}(A)$ avem $X \cup (X \cap Y) = X$ şi $X \cap (X \cup Y) = X$.

Proprietatea 4.6 Fie laticea L şi $a \in L$. Are loc proprietatea de idempotenţă: $a \vee a = a$ and $a \wedge a = a$.

Demonstraţie $a \vee a = a \vee [a \wedge (a \vee b)] = a \vee (a \wedge x) = (L3)\ a$. Analog pentru a doua identitate. $\square$

Definiţia 4.7 Fie laticea L şi $L' \subset L$ nevidă. Spunem că L' este *sublatice* în L dacă pentru orice $x, y \in L'$ avem că $x \vee y \in L'$ şi $x \wedge y \in L'$.

Exemple 4.8 i) Fie $B \subset A$ nevidă. Atunci $\mathscr{P}(B)$ este sublatice în $\mathscr{P}(A)$.
ii) Fie $L = \mathbb{N}^*$ cu operaţiile $a \vee b = [a, b]$=cel mai mic multiplu comun al numerelor a şi b, $a \wedge b = (a, b)$=cel mai mare divizor comun al numerelor date. Avem că $(L, \vee, \wedge)$ este o latice.
Fie numerele prime distincte p şi q. Atunci $L' = \{1, p, q, pq\}$ este o sublatice în $\mathbb{N}^*$.

Definiţia 4.9 Numim *semilatice* o mulţime nevidă M înzestrată cu o operaţie internă $*$ satisfăcând:
SL1) $(a * b) * c = a * (b * c)$,
SL2) $a * b = b * a$,
SL3) $a * a = a$.
Deci $(M, *)$ este un semigrup comutativ în care orice element este idempotent.

Exemple 4.10 Dacă $(L, \vee, \wedge)$ este o latice atunci $(L, \vee)$ şi $(L, \wedge)$ sunt semilatice. Este posibil ca aceste exemple să fi sugerat denumirea.

Observaţia 4.11 Pe laticea L definim relaţia $a \leq b$ dacă $a \vee b = b$. Avem imediat că $\leq$ este o relaţie de ordine pe L. De asemeni, avem $a \leq b$ dacă şi numai dacă $a \wedge b = a$.

Definiţia 4.12 i) Laticea L se numeşte *mărginită* dacă există un cel mai mic element, notat 0, şi există un cel mai mare element, noatt 1.
ii) Dacă L este mărginită şi $x \in L$ spunem că $\overline{x} \in L$ este *complementul* lui x dacă $x \vee \overline{x} = 1$ şi $x \wedge \overline{x} = 0$.

Observaţia 4.13 Avem $\overline{0} = 1$ şi $\overline{1} = 0$ dar nu orice element dintr-o latice mărginită admite complement !

Definiţia 4.14 Laticea mărginită $(L, \vee, \wedge, 0, 1)$ se numeşte *complementată* dacă orice $a \in L$ admite complement unic.

Defniţia 4.15 Laticea L se numeşte *distributivă* dacă au loc proprietăţile de distributivitate:

D1) $(a \vee b) \wedge c = (a \wedge c) \vee (b \wedge c)$,
D2) $(a \wedge b) \vee c = (a \vee c) \wedge (b \vee c)$.

Definiţia 4.16 O latice $(L, \vee, \wedge, 0, 1, ^-)$ complementată şi distributivă se numeşte *algebră Boole* sau *latice booleană*.

Course 5

Automata

Although in literature there exists a large class of automata, the notion used in this course is the following:

Definition 5.1 We call *automata* a 5-tuple $A = (Q, \Sigma, q_0, \delta, F)$ where Q and Σ are non-void and finite sets , $q_0 \in Q$, $F \subseteq Q$is non-void and δ is a function:
$\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to \mathscr{P}(Q)$. Names:
-Q=*the set of states*,
-Σ=*the input alphabet*; an element of Σ is called *input*,
-q_0=*the initial state*,
-δ=*the transition map*,
-F=*the set of final states* or *terminal*.

Remarks 5.2 i) Due to the finite character of all above sets, sometimes it appears in name that A is a finite automata (AF).
ii) Another name of the notion above is that of *non-determinist automata with ε-transitions*. Particular variants for δ yields the notion of *determinist automata* (without ε-transitions) respectively *determinist automata*. So we have:

Definition 5.3 I) The automata A is called:
i) *non-determinist* (AFN) if $\delta(q, \varepsilon) = \emptyset$ for any $q \in Q$,
ii) *determinist* (AFD) it it satisfy the previous condition plus the condition $|\delta(q, x)| = 1$ for any $q \in Q$ and $x \in \Sigma$. We will consider mainly AFD's (see the next course) and so the transition map is a function $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to Q$ since if $\delta(q, x) = \{q'\}$ we denote simple $\delta(q, x) = q'$.
II) Given the state $q \in Q$ we call *the closure* of q as being the set:

$$Cl(q) = \{q\} + \cup_{k \in \mathbb{N}^*} \{q_{k+1} \in Q; q_2 \in \delta(q_1 = q, \varepsilon), ..., q_{k+1} \in \delta(q_k, \varepsilon)\}.$$

For $Q' \subseteq Q$ we define: $Cl(Q') = \cup_{q \in Q'} Cl(q)$ and $\delta(Q', x) = \cup_{q \in Q'} \delta(q, x)$.

An extremely important notion for this theory is:

Definition 5.4 *The extension* of the transition map to words is:
$\delta_e : Q \times \Sigma^* \to \mathscr{P}(Q)$,
i) $\delta_e(q, \varepsilon) = Cl(q)$,
ii) $\delta_e(q, wx) = Cl(\delta(\delta_e(q, w)), x)$, if $w \in \Sigma^*$ and $x \in \Sigma$.

Remarks 5.6 i) For an AFD we have $Cl(q) = \{q\}$ and then $\delta_e(q, x) = Cl(\delta(q, x)) = \delta(q, x)$ since $\delta(q, x)$ have the cardinal 1. From this reason in several references for AFD's is keeping (due to simplicity) the notation δ for the extension map!
ii) An inductive computation (after the length of the second word) yields the equality: $\delta_e(q, w_1 w_2) = \delta_e(\delta_e(q, w_1), w_2)$. It follows that for an AFD and the word $w = x_1 ... x_n$ we have:

$$\delta_e(q, w) = \delta(...(\delta(q, x_1), x_2), ...x_n).$$

We express in the following way: when the automata being in the state q receives the word w it goes in the state $\delta_e(q, w)$. If $\delta_e(q, w) = \emptyset$ we say that the automata goes into a *non-definite state* and if $|\delta_e(q, w)| \geq 2$ we say that the automata goes into an *ambiguous state*.

The most important notion in the automata theory is:

Definition 5.7 *The accepted language* or *recognized* by the automata A is: $L(A) = \{w \in \Sigma^*; \delta_e(q_0, w) \cap F \neq \emptyset\}$. So, for an AFD we have:

$$L(A) = \{w \in \Sigma^*; \delta_e(q_0, w) \in F\}.$$

An element from $L(A)$ is called *recognized word* by A.

From this reason the notion of AFD is sometimes used under the name of *automata of acceptance*. There exists an important relationship between the regular expressions and the accepted languages by automata's:

Theorea 5.8 *If E is a regular expression over Σ i.e. $E \in Rex(\Sigma)$ then there exists a finite automata with ε-transitions over the alphabet Σ such that $K(E) = L(A)$.*

Definition 5.9 The language L over Σ is called *recognizable* if there exists an automata A with alphabet Σ such that $L = L(A)$.

Examples 5.10 i) If $\Sigma = B = \{0, 1\}$ then $L = B^* \setminus B^*010B^*$ is recognizable conform Ex. 5.3.
ii) $L = \{0^n 1^n; n \geq 1\}$ is not recognizable.

Representations of finite automata's

We can represent an AF in two modes: with a table and graphic. From reason of simplicity we choose the first mode although sometimes is preferable the second due to visual advantages: $w \in \Sigma^*$ is recognized (accepted) by A id and only if in the graph of A there exists a path from the initial state q_0 to a final state $q^{prime} \in F$, this path having the arcs consecutively labeled with the letters of the word w. We say that w is *eticheta* (engleză *label*) of the path from q_0 to q'.

So if $Q = \{q_0, ..., q_n\}$ and $\Sigma = \{x_1, ..., x_m\}$ we will consider a table with $n + 2$ rows and $m + 1$ columns as follows:

$Q \setminus \Sigma$	x_1	...	x_j	...	x_m
$\rightarrow q_0$					
...					
$q_i \leftarrow$			$\delta(q_i, x_j)$		
q_n					

where the arrow $\rightarrow$ denotes the initial state and $\leftarrow$ final states.

Example 5.11 Let an AF be given with $Q = \{a, b, c\}, q_0 = a, F = \{c\}, \Sigma = \{0, 1\}$ and:
$\delta(a, 0) = \{b\}, \delta(a, 1) = \{a, b\}$, $\delta(b, 0) = \{c\}, \delta(b, 1) = \{a\}$, $\delta(c, 0) = \{c\}, \delta(c, 1) = \{b, c\}$. We have:

	0	1
$\rightarrow a$	b	$\{a, b\}$
b	c	a
$c \leftarrow$	c	$\{b, c\}$

.

Let $w_1 = 100100$ and $w_2 = 0101$. Then:
i) $\delta_e(a, w_1) = \delta_e(\delta(a, 1), 0010) = \delta_e(b, 0010) = \delta_e(\delta(b, 0), 010) = \delta_e(c, 010) = \delta_e(\delta(c, 0), 10) = \delta_e(c, 10) = \delta(\delta(c, 1), 0) = \delta(\{b, c\}, 0) = \{\delta(b, 0), \delta(c, 0)\} = \{c, c\} \cap \{c\} \neq \emptyset$
ii) $\delta_e(a, w_2) = \delta_e(\delta(a, 0), 101) = \delta_e(b, 101) = \delta_e(\delta(b, 1), 01) = \delta_e(a, 01) = \delta(\delta(a, 0), 1) = \delta(b, 1) = \{a\} \cap \{c\} = \emptyset$.
In conclusion, $w_1 \in L(A)$ and $w_2 \notin L(A)$. More generally, $(01)^+ \not\subset L(A)$.

Useful Web pages:
1) http://en.wikipedia.org/wiki/Automata_theory
2) http://mathworld.wolfram.com/AutomataTheory.html

SEMINAR 5

S5.1 Let be given the non-determinist automata:

	0	1
$\to q_0$	q_1	q_0
q_1	q_2	q_0
q_2	q_2	$\{q_2, q_3\}$
$q_3 \leftarrow$	$\emptyset$	$\emptyset$

.

Study the words $w_1 = 11001$, $w_2 = 1^n 001^m$ with $n \geq 1, m \geq 2$ and $w_3 = 0^n, n \geq 3$.

Solution $w_1 \in L(A)$ since $\delta_e(q_0, w_1) = \{q_2, q_3\}$, $w_2 \in L(A)$ since $\delta_e(q_0, w_2) = \{q_2, q_3\}$ but $w_3 \notin L(A)$ since $\delta_e(q_0, w_3) = q_2 \neq q_3$.

S5.2 Let be given the AFD:

	0	1
$\to q_0$	q_0	q_1
$q_1 \leftarrow$	q_0	q_2
$q_2 \leftarrow$	q_1	q_0

.

Study the words $w_1 = 0011$ and $w_2 = 0111$.

Solution Since $\delta_e(q_0, w_1) = q_2$ we have that $w_1 \in L(A)$ and from $\delta_e(q_0, w_2) = q_0$ we have that $w_2 \notin L(A)$.

S5.3 Let be given the AFD:

	0	1
q_0	q_0	q_0
$\to q_1 \leftarrow$	q_2	q_1
$q_2 \leftarrow$	q_2	q_3
$q_3 \leftarrow$	q_0	q_1

.

i) Study $w_1 = 1^2 0^3 1^2 0^4$ and $w_2 = 1^2 0^3 10^4$.
ii) Prove that $w = ...010... \notin L(A)$.
iii) Obtain then that $L(A) = \{0,1\}^* \setminus \{0,1\}^* 010 \{0,1\}^*$.

Solution i) $\delta_e(q_1, w_1) = q_2 \in F$ hence $w_1 \in L(A)$; $\delta_e(q_1, w_2) = q_0 \notin F$ hence $w_2 \notin L(A)$.
ii) $\delta_e(q_1, ...010...) = q_0$ since applying the word 010 we have:
1) $q_0 \to q_0 \to q_0 \to q_0$,
2) $q_1 \to q_2 \to q_3 \to q_0$,
3) $q_2 \to q_2 \to q_3 \to q_0$,
4) $q_3 \to q_0 \to q_0 \to q_0$.

Course 6

Equivalent automata

Since the most important aspect in the automata theory is given by the accepted langauge is natural to consider:

Definition 6.1 Automata A, A' over the same alphabet are called *equivalent* and we denotes $A \sim A'$ if $L(A) = L(A')$.

Since the equality of sets is an equivalence relation we have:

Proposition 6.2 *The relation $\sim$ is an equivalence relation on the set of automata with fixed alphabet Σ.*

So, we are interested to find some remarkable representatives of a given equivalence class; this search is the main reason for the following theorems of *reduction*. Hence, a first result is about the cancellation of ε-transitions:

Theorem 6.3 *given an automata A with ε-transitions there exists A' without ε-transitions and equivalent with A.*

Another important result gives an equality AFN=AFD from the point of view of accepted languages:

Theorem 6.4 *Dat AFN-ul A există un AFD A_d echivalent cu A.*

Demonstraţie Presupunem $A = (Q, \Sigma, \delta, q_0, F)$ şi fie $A_d = (Q_d, \Sigma, \delta_d, Q_0, F_d)$:
i) $Q_d = \mathscr{P}(Q), Q_0 = \{q_0\}, F_d = \{Z \in Q_d; Z \cap F \neq \emptyset\}$,
ii) $\delta_d : Q_d \times (\Sigma \cup \{\varepsilon\}) \to Q_d$ este:
ii1) $\delta_d(\emptyset, x) = \emptyset, \delta(Z, \varepsilon) = \emptyset$ pentru $Z \in Q_d$ nevidă şi $x \in \Sigma$,
ii2) $\delta_d(Z, x) = \{\delta(q, x); q \in Z\}$.
Avem că A_d este AFD deoarece $\delta_d(Z, x)$ are un singur element (=o mulţime) privit ca element în Q_d.
1) $L(A) \subseteq L(A_d)$
Fie $w \in L(A)$; deci $Z = \delta(q_0, w) \cap F \neq \emptyset$ i.e. $Z \in F_d$ şi avem $\delta_d(Q_0, w) =$

$Z \in F_d$ ceea ce spune că $w \in L(A_d)$.
2) $L(A_d) \subseteq L(A)$
Fie $w \in L(A_d)$; deci $\delta_d(Q_0, w) \in F_d$ i.e. $\{\delta(q_0, w)\} \cap F \neq \emptyset$ ceea ce spune că $w \in L(A)$. $\square$

Exemplul 6.5 Fie A cu $Q = \{q_0, q_1, q_2\}$, $\Sigma = \{0, 1\}$, $F = \{q_2\}$ şi funcţia de tranziţie dată de tabelul:

	0	1
$\rightarrow q_0$	q_1	$\{q_0, q_1\}$
q_1	q_2	q_0
$q_2 \leftarrow$	q_2	$\{q_1, q_2\}$

.

Avem $Q_d = \{Q_1, Q_2 = Q_0, Q_3, Q_4, Q_5, Q_6, Q_7, Q_8 = Q\}$ cu $Q_1 = \emptyset$, $Q_3 = \{q_1\}$, $Q_4 = \{q_2\}$, $Q_5 = \{q_0, q_1\} = Q_2 + Q_3$, $Q_6 = \{q_0, q_2\} = Q_2 + Q_4$, $Q_7 = \{q_1, q_2\} = Q_3 + Q_4$ şi:
-$\delta_d(Q_1, 0) = Q_1$, $\delta_d(Q_1, 1) = Q_1$,
-$\delta_d(Q_2, 0) = Q_3$, $\delta_d(Q_2, 1) = Q_5$,
-$\delta_d(Q_3, 0) = Q_4$, $\delta_d(Q_3, 1) = Q_2$,
-$\delta_d(Q_4, 0) = Q_4$, $\delta_d(Q_4, 1) = Q_7$,
-$\delta_d(Q_5, 0) = \delta_d(Q_2, 0) + \delta_d(Q_3, 0) = Q_3 + Q_4 = Q_7$, $\delta_d(Q_5, 1) = \delta_d(Q_2, 1) + \delta_d(Q_3, 1) = Q_5 + Q_2 = Q_8$,
-$\delta_d(Q_6, 0) = \delta_d(Q_2, 0) + \delta_d(Q_4, 0) = Q_3 + Q_4 = Q_7$, $\delta_d(Q_6, 1) = \delta_d(Q_2, 1) + \delta_d(Q_4, 1) = Q_5 + Q_7 = Q_8$,
-$\delta_d(Q_7, 0) = \delta_d(Q_3, 0) + \delta_d(Q_4, 0) = Q_4 + Q_4 = Q_4$, $\delta_d(Q_7, 1) = \delta_d(Q_3, 1) + \delta_d(Q_4, 1) = Q_2 + Q_7 = Q_8$,
-$\delta_d(Q_8, 0) = \delta_d(Q_2, 0) + \delta_d(Q_7, 0) = Q_3 + Q_4 = Q_7$, $\delta_d(Q_8, 1) = \delta_d(Q_2, 1) + \delta_d(Q_7, 1) = Q_5 + Q_8 = Q_8$.
Avem $F_d = \{Q_4, Q_6, Q_7, Q_8\}$ şi deci tabelul lui A_d este:

	0	1
Q_1	Q_1	Q_1
$\rightarrow Q_2$	Q_3	Q_5
Q_3	Q_4	Q_2
$Q_4 \leftarrow$	Q_4	Q_7
Q_5	Q_7	Q_8
$Q_6 \leftarrow$	Q_7	Q_8
$Q_7 \leftarrow$	Q_4	Q_8
$Q_8 \leftarrow$	Q_7	Q_8

Definiţia 6.6 Dat automatul A spunem că starea $q \in Q$ este *accesibilă* dacă există $w \in \Sigma^*$ aşa încât $\delta_e(q_0, w) = q$.

Observaţia 6.7 Dacă în definiţia precedentă am lua $w \in L(A)$ ar rezulta $q \in F$. Dar asta nu înseamnă că orice stare finală este accesibilă !

Teorema 6.8 *Dat A există A_{ac} cu toate stările accesibile şi echivalent cu A.*

Demonstraţie Definim $A_{ac} = (Q_{ac} = \{Q_0, ..., Q_{r+1}\}, \Sigma, \delta_{ac}, Q_0, F_{ac})$ astfel:
i) Q_{ac} va fi constituită din anumite submulţimi ale mulţimii stărilor accesibile ale lui A i.e. $\{q \in Q; \exists w \in \Sigma^*, \delta_e(q_0, w) = q\}$, $Q_0 = \{q_0\}$, $F_{ac} = \{Q_j \in Q_{ac}; Q_j \cap F \neq \emptyset\}$,
ii) δ_{ac} se determină prin recurenţă astfel:
Pasul 1. Notăm $\Sigma = \{0, ..., n-1\}$,
Pasul 2. Pentru $i \in \Sigma$ definim $\delta_{ac}(Q_0, i) = \{\delta(q_0, i)\} = Q_{i+1}$. Obţinem astfel n mulţimi ce pot coincide cu Q_0 sau între ele; cele diferite le reţinem şi le notăm $Q_1, ..., Q_j$.
Pasul 3. Pentru $i \in \Sigma$ definim $\delta_{ac}(S, i) = \{\delta(q, i); q \in S\} = Q_{j+i+1}$ cu $S \in \{Q_1, ..., Q_j\}$; din nou avem n mulţimi ce pot coincide cu $Q_0, ..., Q_j$ sau între ele; cele diferite le renumerotăm $Q_{j+1}, ..., Q_{j+k}$.

Continuăm acest procedeu până când *nu* mai obţinem mulţimi noi; deci am găsit $r \geq 1$ aşa încât $Q_{r+1} \notin \{Q_0, ..., Q_r\}$ dar $Q_{r+s} = Q_{r+1}$ pentru orice $s \geq 2$. Atunci $Q_{ac} = \{Q_0, ..., Q_{r+1}\}$. $\square$

Exemplul 6.9 Fie A cu $Q = \{q_0, q_1, q_2\}$, $\Sigma = \{0, 1\}$, $F = \{q_2\}$ şi funcţia de tranziţie dată de tabelul:

	0	1
$\rightarrow q_0$	q_1	q_2
q_1	$\{q_1, q_2\}$	q_1
$q_2 \leftarrow$	q_2	$\{q_1, q_2\}$

.

Avem deci:
Pasul 1. Observăm că mulţimea stărilor accesibile este tot Q deoarece $\delta_e(q_0, \varepsilon) = q_0$, $\delta(q_0, 0) = q_1$, $\delta(q_0, 1) = q_1$.
Pasul 2. $\delta_{ac}(Q_0, 0) = \delta(q_0, 0) = \{q_1\} = Q_1$, $\delta_{ac}(Q_0, 1) = \delta(q_0, 1) = \{q_2\} = Q_2$. Reţinem Q_1 şi Q_2.
Pasul 3. $\delta_{ac}(Q_1, 0) = \delta(q_1, 0) = \{q_1, q_2\} = Q_3$, $\delta_{ac}(Q_1, 1) = \delta(q_1, 1) = \{q_1\} = Q_1$,
$\delta_{ac}(Q_2, 0) = \delta(q_2, 0) = \{q_2\} = Q_2$, $\delta_{ac}(Q_2, 1) = \delta(Q_2, 1) = \delta(q_2, 1) = \{q_1, q_2\} = Q_3$. Deci reţinem Q_3.
Pasul 4. $\delta_{ac}(Q_3, 0) = \delta(q_1, 0) + \delta(q_2, 0) = \{q_1, q_2\} + \{q_2\} = \{q_1, q_2\} = Q_3$,
$\delta_{ac}(Q_3, 1) = \delta(q_1, 1) + \delta(q_2, 1) = \{q_1\} + \{q_1, q_2\} = \{q_1, q_2\} = Q_3$.

În concluzie, $Q_{ac} = \{Q_0, Q_1, Q_2, Q_3\}$, $F_3 = \{Q_2, Q_3\}$ şi avem tabelul pentru δ_{ac}:

	0	1
$\rightarrow Q_0$	Q_1	Q_2
Q_1	Q_3	Q_1
$Q_2 \leftarrow$	Q_2	Q_3
$Q_3 \leftarrow$	Q_3	Q_3

.

Avem $\delta_{ac,e}(Q_0, \varepsilon) = Q_0$, $\delta_{ac}(Q_0, 0) = Q_1$, $\delta_{ac}(Q_0, 1) = Q_2$, $\delta_{ac,e}(Q_0, 00) = Q_3$ ceea ce spune că toate stările sunt accesibile.

Definiţia 6.10 i) Dat AFD-ul A starea $q \in Q$ o numim *productivă* (sau *coaccesibilă*) dacă există $w \in \Sigma^*$ aşa încât $\delta_e(q, w) \in F$.
ii) Un automat cu toate stările accesibile şi productive se numeşte *trim*.

Observaţia 6.11 Din definiţie avem că q_0 este productivă dacă $L(A) \neq \emptyset$ deoarece luăm orice $w \in L(A)$.

Teorema 6.11 *Dat AFD-ul A există un AFD A_p cu toate stările productive şi echivalent cu A.*

Demonstraţie Construim $A_p = (Q_p = \{Q_1, ..., Q_{r+1}\}, \Sigma, \delta_p, Q_0, F_p)$ cu $Q_0 = \{q_0\}$ şi procedând analog cu demonstraţia anterioară luând $Q_1 = F$ şi $F_p = \{Q_j \in Q_p; Q_j \cap F \neq \emptyset\}$. $\square$

SEMINARUL 6

S6.1 Să se studieze AFD-ul cu $|Q| = |\Sigma| = 1$.

Rezolvare Avem $Q = \{q_0\} = F$, $\Sigma = \{0\}$ şi $\delta : Q \times \Sigma \rightarrow Q, \delta(q_0, 0) = q_0$. Rezultă $L(A) = \{0\}^*$. Reprezentarea tabelară:

	0
$\rightarrow q_0 \leftarrow$	q_0

.

Observaţie Un AFD cu $|\Sigma| = 1$ se numeşte *autonom*.

S6.2 Să se studieze AFD-ul cu $|Q| = 1$ şi $|\Sigma| = 2$. Generalizare.

Rezolvare Avem Q şi F ca mai sus iar $\Sigma = \{0, 1\}$. Evident $\delta(q_0, 0) = \delta(q_0, 1) = q_0$ şi deci $L(A) = \{0, 1\}^*$. Reprezentarea tabelară:

	0	1
$\rightarrow q_0 \leftarrow$	q_0	q_0

.

Generalizare: $Q = \{q_0\} = F$, $\Sigma = \{0, ..., n\}$ şi $\delta(q_0, i) = q_0$, $1 \leq i \leq n$. Avem $L(A) = \{0, ..., n\}^*$. Evident, acest automat este un trim.

S6.3 Să se studieze AFD-ul cu $|Q| = 2$ şi $|\Sigma| = 1$.

Rezolvare Fie $Q = \{q_0, q_1\}$ şi $\Sigma = \{0\}$.
I)

δ	0
$\rightarrow q_0$	q_0
q_1	q_0

.

a) $F = \{q_0\}$; avem $L(A) = \{0\}^*$.
b) $F = \{q_0, q_1\}$; avem $L(A) = \{0\}^*$.
c) $F = \{q_1\}$; avem $L(A) = \emptyset$.
II)

δ	0
$\rightarrow q_0$	q_1
q_1	q_1

.

a) $F = \{q_0\}$; avem $L(A) = \emptyset$.
b) $F = \{q_0, q_1\}$; avem $L(A) = \{0\}^*$.
c) $F = \{q_1\}$; avem $L(A) = \{0\}^*$.
III)

δ	0
$\rightarrow q_0$	q_0
q_1	q_1

.

a) $F = \{q_0\}$; avem $L(A) = \{0\}^*$.
b) $F = \{q_0, q_1\}$; avem $L(A) = \{0\}^*$.
c) $F = \{q_1\}$; avem $L(A) = \emptyset$.
IV)

δ	0
$\rightarrow q_0$	q_1
q_1	q_0

.

a) $F = \{q_0\}$; avem $L(A) = \{0^{2n}; n \in \mathbb{N}^*\}$.
b) $F = \{q_0, q_1\}$; avem $L(A) = \{0\}^*$.
c) $F = \{q_1\}$; avem $L(A) = \{0^{2n+1}; n \in \mathbb{N}^*\}$.

Automatul III se poate identifica cu permutarea identică $\pi_1 = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix}$ iar automatul IV cu permutarea "twist" $\pi_2 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$.

S6.4 Automatul ciclic $\mathscr{C}(p,r)$.

Rezolvare Fie $Q=\{q_0,...,q_{r+p-1}\}$ şi $\Sigma=\{0\}$. Considerăm funcţia de tranziţie:

δ	0
$\to q_0$	q_1
$\vdots$	$\vdots$
q_{r-1}	q_r
$\vdots$	$\vdots$
q_{r+p-2}	q_{r+p-1}
q_{r+p-1}	q_r

.

Dacă automatul părăseşte una din stările $q_0,...,q_{r-1}$ el nu se mai întoarce niciodată în acea stare; le putem numi *fără întoarcere*. Ciclul de stări $q_r,...,g_{r+p-1}$ se poate numi *ciclul fără evadare*, în engleză *no escape*.

S6.5 Să se studieze AFD-ul cu $Q=\{q_0,q_1\}$, $\Sigma=\{0,1\}$, $F=\{q_0\}$ şi funcţia:

δ	0	1
$\to q_0 \leftarrow$	q_0	q_1
q_1	q_1	q_1

.

Rezolvare $L(A)=\{0\}^*$.

S6.6 Să se studieze AFD-ul cu $Q=\{q_0,q_1,q_2\}$, $F=\{q_1\}$, $\Sigma=\{0,1\}$ şi funcţia:

δ	0	1
$\to q_0$	q_1	q_2
$q_1 \leftarrow$	q_1	q_2
q_2	q_2	q_2

.

Rezolvare $L(A)=\{0\}^+ (=\{0\}^*\cdot\{0\}=\{0\}\cdot\{0\}^*)$.

Course 7

Automat minimal

Am văzut în cursul precedent că preţul plătit pentru adăugarea anumitor proprietăţi (convenabile) unui automat se reflectă în creşterea, uneori foarte mare (rapidă), a numărului de stări. Este astfel naturală întrebarea dacă putem lucra cu anumite condiţii de minimalitate.

Definiţia 7.1 Fie A un AFD.
i) Dat numărul natural k definim relaţia $=_k$ pe Q prin: $q_1 =_k q_2$ dacă pentru orice $w \in \cup_{i=0}^{k} \Sigma^i$ avem $\delta_e(q_1, w) \in F$ dacă şi numai dacă $\delta_e(q_2, w) \in F$. Spunem că stările q_1 şi q_2 sunt *k-echivalente*.
ii) Definim relaţia $\equiv$ pe Q prin: $q_1 \equiv q_2$ dacă $q_1 =_k q_2$ pentru orice $k \in \mathbb{N}$. Vom spune că q_1 şi q_2 sunt *echivalente*.

Reamintim că apartenenţa unui element la o mulţime se studiază cu ajutorul unei funcţii:

Definiţia 7.2 Dată mulţimea nevidă X şi submulţimea sa Y definim *funcţia caracteristică* a lui Y funcţia $\chi_Y : X \to \{0, 1\}$ dată de:
i) $\chi_Y(x) = 1$ dacă $x \in Y$, ii) $\chi_Y(x) = 0$ altfel.
Există autori care fac alegerea inversă: $\chi_Y(x) = 0$ dacă $x \in Y$ ([15, p. 84]) dar în literatura clasică (românească) avem alegerea de mai sus.

Prin urmare, $q_1 =_k q_2$ dacă şi numai dacă $\chi_F(\delta_e(q_1, w)) = \chi_F(\delta_e(q_2, w))$ pentru orice $w \in \Sigma^0 + ... + \Sigma^k$. Astfel, obţinem că $q_1 =_0 q_2$ dacă şi numai dacă q_1, q_2 aparţin simultan lui F sau $Q \setminus F$. O caracterizare pentru $=_k$ cu $k \geq 1$ este:

Propoziţia 7.3 *$q_1 =_k q_2$ dacă şi numai dacă $q_1 =_{k-1} q_2$ şi $\delta(q_1, x) =_{k-1} \delta(q_2, x)$ pentru orice $x \in \Sigma \setminus \{\varepsilon\}$.*

Deoarece relaţia de egalitate este o echivalenţă avem:

Propoziţia 7.4 *Relaţiile* $=_k$ *şi* $\equiv$ *sunt echivalenţe pe* Q.

Notăm atunci Q_m şi F_m mulţimile cât determinate de $\equiv$ pe Q respectiv F. Definim $\delta_m : Q_m \times \Sigma^* \to Q_m$ prin $\delta_m([q], w) = [\delta_e(q, w)]$. Se verifică imediat buna definire a acestei aplicaţii.

Noţiunea centrală a acestui curs este:

Definiţia 7.5 AFD-ul A se numeşte *minimal* dacă pentru orice AFD A' echivalent cu A avem $|Q| \leq |Q'|$. În particular, automatul cu o singură stare este minimal.

Avem deci Problema:
Input: A AFD,
Output: A' AFD minimal şi echivalent cu A.

Trebuie subliniat că automatul minimal este unic până la un izomorfism:

Definiţia 7.6 AFD-urile A, A' cu aceeaşi Σ se numesc *izomorfe* dacă există o bijecţie $h : Q \to Q'$ aşa încât: $q_0' = h(q_0)$, $F' = h(F)$ şi $h(\delta(q, x)) = \delta'(h(q), x)$ pentru orice $q \in Q$ şi orice $x \in \Sigma$ i.e. următoarea diagramă este comutativă:

$$\begin{array}{|lll|} \hline Q \times \Sigma & \delta \to & Q \\ \downarrow h & & \downarrow h \\ Q' \times \Sigma & \delta' \to & Q' \\ \hline \end{array}.$$

Rezultă imediat că:
1) $h(\delta_e(q, w)) = \delta'_e(h(q), w)$ pentru orice $w \in \Sigma^*$,
2) $L(A) = L(A')$ şi deci A, A' sunt echivalente. Putem spune că un izomorfism realizează în fapt, o "recontorizare" a stărilor.

În determinarea automatului minimal de un mare folos este rezultatul următor:

Propoziţia 7.7 i) *Fie* $q_1, q_2 \in Q$ *pentru* $|Q| = n$. *Atunci* $q_1 \equiv q_2$ *dacă şi numai dacă* $q_1 =_{n-2} q_2$.
ii) *Dacă* Q_m *este în bijecţie cu* Q *atunci automatul* A *este minimal.*

Unul din rezultatele centrale ale acestui curs este:

Teorema 7.8 *Dat AFD-ul* A *avem că* $A_m = (Q_m, \Sigma, \delta_m, [q_0], F_m)$ *este automat minimal echivalent cu* A.

Definiţia 7.9 i) Dacă în definiţia automatului reţinem primele trei elemente $SA = (Q, \Sigma, \delta)$ atunci obţinem noţiunea de *semiautomat.*
ii) Un semiautomat (automat) se numeşte *conex* dacă orice două stări se pot uni i.e. oricare ar fi $q_1, q_2 \in Q$ există $w \in \Sigma^*$ aşa încât $\delta_e(q_1, w) = q_2$.

iii) Un semiautomat (automat) se numeşte *perfect* dacă este conex şi satisface $\delta_e(q, w_1w_2) = \delta_e(q, w_2w_1)$ pentru orice $q \in Q$ şi $w_1, w_2 \in \Sigma^*$.

Observaţii 7.10
1) Dacă automatul A este conex atunci orice stare $q \in Q$ este accesibilă.
2) Dacă automatul A este conex atunci orice stare $q \in Q$ este productivă; în adevăr, fixăm $q_f \in F$ şi din conexitate va exista $w \in \Sigma^*$ aşa încât $\delta_e(q, w) = q_f \in F$.
3) Din 1) şi 2) rezultă că un automat conex este un trim.

Definiţia 7.11 O *congruenţă* pe (semi)automatul A este o relaţie de echivalenţă $\sim$ pe Q cu proprietatea că $q_1 \sim q_2$ implică $\delta(q_1, x) \sim \delta(q_2, x)$ pentru orice $x \in \Sigma$. Rezultă că $\delta_e(q_1, w) \sim \delta_e(q_2, w)$ pentru orice $w \in \Sigma^*$.

Pagini Web utile:
1) http://planetmath.org/encyclopedia/Minimal2.html

SEMINARUL 7

S7.1 *Semiautomatul grup* şi *automatul grup.*

Rezolvare Fie grupul finit G şi semiautomatul $SA_G = (G, G, \delta)$ cu δ dată de multiplicarea în G. Avem că SA_G este conex şi dacă G este comutativ atunci SA_G este perfect. Dat $q_0 \in G$ şi $F \subset G$ avem automatul $A_G = (SA_G, q_0, F)$ care are toate stările accesibile şi productive; deci A_G este trim.

S7.2 *Sumatorul modulo n.*

Rezolvare Fie n număr natural nenul. Semiautomatul numit astfel este $SA_n = (\mathbb{N}_{<n} = \{0, ..., n-1\}, \mathbb{N}_{<n}, \delta)$ cu $\delta(q, x) = q + x(mod \quad n)$.

S7.3 Se cere un semiautomat care să recunoască constantele zecimale fără semn pornind dintr-o stare iniţială START.

Rezolvare Fie $Q = \{q_0, q_1, q_2, q_3, q_4\}$ unde: q_0=START, q_1=constantă întreagă, q_2=constantă întreagă cu punct zecimal, q_3=constantă cu parte fracţionară, q_4=eroare. Fie Σ={.,0,...,9} şi δ dată de:
1) $\delta(q_0, .) = \delta(q_1, .) = q_2, \delta(q_2, .) = \delta(q_3, .) = \delta(q_4, .) = \delta(q_4, i) = q_4$,
2) $\delta(q_0, i) = \delta(q_1, i) = q_2, \delta(q_2, i) = \delta(q_3, i) = q_3$
unde $i \in \mathbb{N}_{<9}$.

S7.4 Fie SA_n sumatorul modulo n.
i) O partiţie pe $\mathbb{N}_{<n}$ este o congruenţă pe S_n dacă şi numai dacă este de

tipul următor: există $d \geq 2$ un divizor al lui n aşa încât partiţia este de tip R_d adică are d clase cu acelaşi număr de elemente şi ordonând elementele unei clase crescător avem că diferenţa dintre două elemente consecutive este d.
ii) Fie $d_1, d_2 \geq 2$ divizori ai lui n; dacă $d_1 | d_2$ atunci $R_{d_2} \subseteq R_{d_1}$.

S7.5 Fie automatul A peste alfabetul binar cu $Q = \{q_0, q_1, q_2, q_3\}$ şi:

δ	0	1
$\rightarrow q_0$	q_1	q_2
$q_1 \leftarrow$	q_3	q_1
$q_2 \leftarrow$	q_2	q_3
q_3	q_3	q_3

Se cere $L(A)$.

Rezolvare Avem $L(A) = \{01^n; n \in \mathbb{N}\} + \{10^n; n \in \mathbb{N}\}$.

S7.6 Fie automatul având $Q = \{q_0, ..., q_n\}$, $\Sigma = \{1, ..., n\}$ şi:

δ	1	2	...	n
$\rightarrow q_0$	q_1			
q_1		q_2		
$\vdots$				
q_{n-1}				q_n
$q_n \leftarrow$				

unde în locurile necompletate apare mulţimea vidă. Se cere $L(A)$.

Rezolvare $L(A) = \{w = 12...n\}$.

S7.7 Se dă automatul cu $Q = \{q_0\} = F$, $\Sigma = \{1, ..., n\}$ şi:

δ	1	...	n
$\rightarrow q_0 \leftarrow$	$\emptyset$	...	$\emptyset$

şi se cere $L(A)$.

Rezolvare $L(A) = \{\varepsilon\}$.

S7.8 Dat alfabetul binar se cere un automat care să recunoască limbajul $L(A) = \{(01)^n; n \in \mathbb{N}\}$.

Rezolvare Fie $Q = \{q_0, q_1, q_2\}$ şi:

δ	0	1
$\rightarrow q_0 \leftarrow$	q_1	$\emptyset$
q_1	$\emptyset$	q_2
$q_2 \leftarrow$	q_1	$\emptyset$

.

S7.9 Dat alfabetul Σ mulţimea $L \subseteq \Sigma^*$ o numim *recognoscibilă* dacă există un automat A peste Σ aşa încât $L(A) = L$. Să se arate că mulţimile singleton sunt recognoscibile.

Rezolvare i) Fixăm $L = \{w = a_1...a_n \in \Sigma^*\}$. Fie A peste Σ cu $Q = \{q_0, ..., q_n\}$, $F = \{q_n\}$ şi:

δ	a_1	a_2	...	a_n
$\rightarrow q_0$	q_1			
q_1		q_2		
$\vdots$				
q_{n-1}				q_n
$q_n \leftarrow$				

unde în locurile necompletate apare mulţimea vidă. Avem $L(A) = \{w\}$.

ii) Fie $L = \{\varepsilon\}$ şi A peste Σ cu $Q = \{q_0, q_1\}$, $F = \{q_0\}$ şi:

δ	...	a_i	...
$\rightarrow q_0 \leftarrow$		q_1	
q_1		q_1	

.

Avem $L(A) = \{\varepsilon\}$.

S7.10 Să se arate că mulţimea vidă este recognoscibilă.

Rezolvare Fie A peste σ cu $Q = \{q_0, q_1\}$, $F = \{q_1\}$ şi:

δ	...	a_i	...
$\rightarrow q_0$		q_0	
$q_1 \leftarrow$		q_0	

.

Avem $L(A) = \emptyset$.

S7.11 Fie L_1 şi L_2 recognoscibile. Să se arate că $L_1 \cup L_2$ este recognoscibilă.

Rezolvare Presupunem $L_i = L(A_i), 1 \leq i \leq 2$ şi fie A cu $Q = Q_1 \times Q_2$, $q_0 = (q_0^1, q_0^2)$, $F = (F_1 \times Q_2) \cup (Q_1 \times F_2)$ şi $\delta((q_1, q_2), x) = (\delta_1(q_1, x), \delta_2(q_2, x))$. Avem imediat că $L(A) = L_1 + L_2$.

Consecinţă Orice mulţime finită din Σ^* este recognoscibilă.

S7.12 Dacă $x \in \Sigma$ să se arate că $L = x^*$ este recognoscibil.

S7.13 Pentru $L \subseteq \Sigma^*$ şi $w \in \Sigma^*$ definim $w^{-1}L = \{\alpha \in \Sigma^*; w\alpha \in L\}$ şi Σ_L^* familia acestor mulţimi cu posibilitatea includerii şi a mulţimii vide. Atunci L este recognoscibilă dacă şi numai dacă Σ_L^* este finită.

S7.14 Dacă $L \subseteq \Sigma^*$ este recognoscibilă atunci $\Sigma^* \setminus L$ este recognoscibilă.

S7.15 Dacă L_1 şi L_2 sunt recognoscibile atunci sunt recognoscibile:
i) $L_1 \cap L_2$; ii) $L_1 \cdot L_2$.

S7.16 Să se arate că, relativ la alfabetul binar, limbajul $L = \{0^n 1^n; n \in \mathbb{N}\}$ **nu** este recognoscibil.

Rezolvare Fie, prin reducere la absurd, A cu $L(A) = L$ şi fie, eventual prin renumerotare, $q_n = \delta_e(q_0, 0^n)$. Să se presupunem că $q_n = q_m$. Avem, $\delta_e(q_0, 0^m 1^n) = \delta_e(q_m, 1^n) = \delta_e(q_0, 0^n 1^n) \in F$ şi deci $0^m 1^n \in L(A) = L$. Dar atunci $m = n$ ceea ce spune că Q este infinită. Fals.

S7.17 Fie alfabetele Σ_1, Σ_2 şi $f : \Sigma_1^* \to \Sigma_2^*$ cu $f^{-1}(\varepsilon_2) = \varepsilon_1$. Să se arate că dacă $L \subseteq \Sigma_1^*$ este recognoscibilă atunci $f(L)$ este recognoscibilă.

S7.18 Dat semiautomatul SA şi $w \in \Sigma^*$ fie funcţia $F_w : Q \to Q, F_w(q) = \delta_e(q, w)$. Definim $\sim_{SA}$ pe Σ^* prin $w_1 \sim w_2$ dacă $F_{w_1} = F_{w_2}$. Să se arate că $\sim_{SA}$ este o congruenţă pe Σ^*.

Rezolvare Fie $x, y \in \Sigma^*$ oarecare. Dacă $w_1 \sim_{SA} w_2$ atunci evident $xw_1y \sim_{SA} xw_2y$.

Observaţie Fie $w_1 \sim_{SA} w_2$ şi $x, y \in \Sigma^*$. Atunci **sau** xw_1y, xw_2y aparţin lui $L(A)$ **sau** xw_1y, xw_2y aparţin lui $\Sigma^* \setminus L(A)$. Deci:

$$w_1 \sim_{SA} w_2 \Leftrightarrow [xw_1y \in L(A) \Leftrightarrow xw_2y \in L(A), \forall x, y \in \Sigma^*].$$

Fie $L \subseteq \Sigma^*$. Definim $\sim_L$ prin:

$$w_1 \sim_L w_2 \Leftrightarrow [xw_1y \in L \Leftrightarrow xw_2y \in L, \forall x, y \in \Sigma^*]$$

şi deci $\sim_{SA} = \sim_{L(A)}$.

S7.19 Fie $L \subseteq \Sigma^*$ recognoscibilă de către automatul A_L. Atunci $\sim_{A_L} = \sim_L$.

Definiţia 7.20 Congruenţa $\sim_L$ pentru L recognoscibilă se numeşte *congruenţa Myhill*.

Course 8

Acţiuni

Fixăm în acest curs o mulţime nevidă X, nu neapărat finită. Reamintim că în exerciţiul 1.3 am introdus grupul bijecţiilor lui X, $Bij(X) = \{f : X \to X; f \text{ bijecţie}\}$. Pentru simplitate, acelaşi grup îl vom nota cu $S(X)$ deoarece dacă X are n elemente atunci acest grup este bine cunoscutul grup simetric S_n.

Definiţia 8.1 Numim *grup de transformări* pe X un subgrup G al lui $S(X)$. Noţiuni analoage se definesc prin înlocuirea cuvântului "grup" cu "monoid" respectiv "semigrup".

Din condiţia de subgrup avem:
GT1) dacă $f_1, f_2 \in G$ atunci $f_2 \circ f_1 \in G$,
GT2) dacă $f \in G$ atunci f este bijecţie şi $f^{-1} \in G$.
Una din cele mai importante noţiuni matematice este:

Definiţia 8.2 Dat grupul $(G, \cdot, e)$ oarecare, numim *acţiune* (la stânga) a lui G pe X o aplicaţie $\varphi : G \times X \to X, (g, x) \to gx$ satisfăcând:
A1) $g_2(g_1x) = (g_2g_1)x$,
A2) $ex = x$,
pentru orice $g_1, g_2 \in G$ şi orice $x \in X$.

Observaţia 8.3 Există şi noţiunea de acţiune la dreapta, $\delta : X \times G \to X, (x, g) \to xg$ cu proprităţile:
A'1) $(xg_1)g_2 = x(g_1g_2)$,
A'2) $xe = x$.
Dar cele două noţiuni sunt echivalente în sensul următor:
I) dată o acţiune la stânga avem că aplicaţia $\delta(x, g) = g^{-1}x$ este o acţiune la dreapta. În adevăr:
A'1) $(xg_1)g_2 = g_2^{-1}(g_1^{-1}x) = (g_2^{-1}g_1{-}1)x = (g_1g_2)^{-1}x = x(g_1g_2)$,

A'2) $xe = e^{-1}x = ex = x$.
II) Reciproc dată o acţiune la dreapta avem că aplicaţia $\varphi(g, x) = xg^{-1}$ este o acţiune la stânga. În adevăr,
A1) $g_2(g_1x) = (xg_1^{-1})g_2^{-1} = x(g_1^{-1}g_2^{-1}) = x(g_2g_1)^{-1} = (g_2g_1)x$,
A2) $ex = xe^{-1} = xe = x$.
Prin urmare, în cele ce urmează ne vom restrânge la acţiuni la stânga, acestea fiind cel mai des întâlnite în matematică (spre exemplu în studiul proprietăţilor de simetrie ale unui obiect matematic) fără a pierde din vedere faptul că proprietăţile A'1, A'2 sunt analoage proprietăţilor extinderii funcţiei de tranziţie δ_e din teoria automatelor. Acest fapt a şi motivat alegerea subiectului acestui curs.

Fixăm deci $\varphi : G \times X \to X$ şi pentru orice $g \in G$ definim $\varphi_g : X \to X$ prin $x \to gx$.

Propoziţia 8.4 $\varphi_g \in S(X)$.

Demonstraţie Avem: $\varphi_g \circ \varphi_{g^{-1}} : x \to g^{-1}x \to g(g^{-1}x) = (gg^{-1})x = ex = x$. Analog, $\varphi_g\varphi_{g^{-1}} : x \to gx \to g^{-1}(gx) = (g^{-1}g)x = ex = x$. În concluzie, φ_g este bijecţie cu $(\varphi_g) = \varphi_{g^{-1}}$. □

Putem deci defini $\Phi : G \to S(X), g \to \varphi_g$.

Propoziţia 8.5 Φ *este morfism de grupuri.*

Demonstraţie Trebuie arătat că $\Phi(g_2g_1) = \Phi(g_2) \circ \Phi(g_1)$. Cum acestea sunt aplicaţii trebuie arătată egalitatea pentru orice $x \in X$. Dar asta înseamnă exact A1. □

Corolarul 8.6 *Imaginea prin* Φ *a lui* G *este un grup de transformări pe* X.

Demonstraţie Este o consecinţă a rezultatului clasic din teoria grupurilor: dat un morfism de grupuri $\Phi : G \to G'$ avem că $\Phi(G)$ este subgrup în G'. (Arătaţi !). □

Avem şi reciproca acestui ultim rezultat:

Propoziţia 8.7 *Dat grupul de transformări* G *al lui* X *avem că:*
i) *incluziunea* $i : G \to S(X)$ *este un morfism de grupuri,*
ii) *aplicaţia* $\varphi : G \times X \to X, (g, x) \to g(x)$ *este o acţiune a lui* G *pe* X.

Demonstraţie i) evident, $i(g_1 \circ g_2) = g_1 \circ g_2 = i(g_1) \circ i(g_2)$,
ii) A1 $g_2(g_1x) = g_2(g_1(x)) = (g_2 \circ g_1)(x) = (g_2 \circ g_1)x$; A2 $ex = 1_X(x) = x$.
□

Avem şi un al treilea rezultat analog:

Propoziţia 8.8 *Dat morfismul de grupuri* $\Phi : G \to S(X)$ *avem că aplicaţia* $\varphi : G \times X \to X, (g, x) \to \Phi(g)(x)$ *este o acţiune a lui* G *pe* X.

Demonstraţie A1) $g_2(g_1x) = \Phi(g_2)(\Phi(g_1)(x)) = \Phi(g_2) \circ \Phi(g_1)(x) = \Phi(g_2g_1)(x) = (g_2g_1)x$,
A2) $ex = \Phi(e)(x) = 1_X(x) = x$. $\square$

În concluzie, următoarele trei noţiuni sunt echivalente:
i) acţiune a lui G pe X,
ii) morfism de grupuri $\Phi : G \to S(X)$,
iii) G=grup de transformări pe X.
Astfel, în diferite surse bibliografice poate apărea una din aceste forme echivalente.

Definiţia 8.9 Spunem că elementele $x, y \in X$ sunt în relaţia "$\sim$" şi notăm $x \sim y$ dacă există $g \in G$ a.î. $y = \varphi_g(x)$ i.e. $y = gx$.

Propoziţia 8.10 "$\sim$" *este o relaţie de echivalenţă pe* X.

Demostraţie *reflexivitatea*: $x \sim x$ doarece luăm $g = e$,
simetria: presupunem $x \sim y$ cu $y = \varphi_g(x)$ şi aplicăm $\varphi_{g^{-1}}$ acestei egalităţi. Rezultă $x = g^{-1}y$ şi deci $y \sim x$,
tranzitivitatea: presupunem $x \sim y, y \sim z$ cu $y = g_1x, z = g_2y$. Rezultă $z = g_2(g_1x) \overset{A1}{=} (g_2g_1)x$ şi deci $x \sim z$. $\square$

Definiţia 8.11 Clasa de echivalenţă a lui $x \in X$ în raport cu "$\sim$" se numeşte *orbita* lui x la acţiunea φ şi o notăm Gx sau $Orb(x)$. Mulţimea orbitelor o notăm X/G şi o numim *spaţiul factor* al lui X la acţiunea lui G. Avem surjecţia $\pi : X \to X/G$ numită *proiecţie*.

Dacă discuţia anterioară ne-a plasat pe X este naturală şi o privire asupra lui G.

Definiţia 8.12 Dat $x \in X$ numim *stabilizatorul lui* x mulţimea $Stab(x) = \{g \in G; gx = x\}$.

Propoziţia 8.13 $Stab(x)$ *este subgrup în* G.

Demonstraţie Să observăm mai întâi că $Stab(x)$ este nevidă deoarece $e \in Stab(x)$.
i) Fie $g_1, g_2 \in Stab(X)$; avem imediat că $g_1g_2 \in Stab(x)$,
ii) Fie $g \in Stab(x)$ oarecare; deci $g^{-1}(gx) = g^{-1}x$ de unde rezultă că $g^{-1}x = (g^{-1}g)x = ex = x$, deci $g^{-1} \in Stab(x)$. $\square$

Observaţie Din acest motiv, uneori $Stab(x)$ este numit *grupul de izotropie a lui* $x \in X$.

O proprietate remarcabilă a stabilizatorilor este:

Propoziţia 8.14 *Dacă $y \in Orb(x)$ atunci $Stab(x)$ şi $Stab(y)$ sunt subgrupuri conjugate.*

Demonstraţie Presupunem că $y = ax, a \in G$.
i) dat $g \in Stab(x)$ avem că $aga^{-1} \in Stab(y)$ după cum se verifică imediat. Deci $aStab(x)a^{-1} \subseteq Stab(y)$,
ii) a arăta $Stab(y) \subseteq aStab(x)a^{-1}$ este echivalent cu a arăta relaţia $a^{-1}Stab(y)a \subseteq Stab(x)$ care este analoagă celei de la i) cu argumentul $x = a^{-1}y$. □

Corolarul 8.15 *Dacă un element al unei orbite O are stabilizator abelian (finit) atunci toate elementele lui O au stabilizatorul abelian (finit). Mai mult, oricare ar fi $x, y \in O$ avem $|Stab(x)| = |Stab(y)|$.*

Definiţia 8.16 Fie H subgrup al lui G şi elementele fixate $x, y \in G$. Spunem că x, y sunt *H-right echivalente* şi notăm $x_H \sim_r y$ dacă există $h \in H$ aşa încât $x = yh$.

Propoziţia 8.17 *$\sim_r$ este o relaţie de echivalenţă pe G.*

Demonstraţie 1) reflexivitatea: luăm $h = e \in H$. 2) simetria: fie $x = yh$. Rezultă că $y = xh^{-1}$ şi cum $h^{-1} \in H$ avem cerinţa. 3) tranzitivitatea: presupunem $x = yh_1$ şi $y = zh_2$. Rezultă $x = z(h_2h_1)$ şi cum $h_2h_1 \in H$ avem concluzia. □

Mulţimea cât o notăm G/H. Clasa de echivalenţă a lui $x \in G$ este $xH = \{xh; h \in H\}$ şi aplicaţia $h \in H \to xh \in xH$ este evident bijecţie; deci $|xH| = |H|$. Cum mulţimea claselor de echivalenţă constituie o partiţie a lui G rezultă că dacă G este grup finit avem $|G/H| \cdot |H| = |G|$.

Revenind la acţiuni fie $H = Stab(x)$; deci $|G/Stab(x)| \cdot |Stab(x)| = |G|$. Fie $\varphi : Orb(x) \to G/Stab(x), y = gc \to \varphi(x) = gStab(x)$. Avem că φ este corect definită: dacă $y = g_1x = g_2x$ atunci $g_2^{-1}g_1 \in Stab(x)$ şi deci $g_1 = g_2h$ cu $h \in Stab(x)$; rezultă că $g_1Stab(x) = g_2Stab(x)$, ceea ce voiam.

Propoziţia 8.18 *φ este bijecţie.*

Demonstraţie Evident φ este surjecţie căci dat $gStab(x) \in G/Stab(x)$ vom considera $y = gx \in Orb(x)$. Fie acum $\varphi(y_1) = \varphi(y_2)$ cu $y_1 = g_1x, y_2 = g_2x$; rezultă $g_1Stab(x) = g_2Stab(x)$ i.e. $g_1h_1 = g_2h_2$ cu $h_1, h_2 \in Stab(x)$. Deci $g_2 = g_1h_1h_2^{-1}$ şi avem $y_2 = (g_1h_1h_2^{-1})x = g_1(h_1h_2^{-1}x) = g_1x = y_1$ folosind că $Stab(x)$ este subgrup. □

În concluzie, dacă G şi X sunt mulţimi finite avem pentru orice $x \in X$:

$$|Orb(x)| \cdot |Stab(x)| = |G|, \tag{8.1}$$

Definiţia 8.19 Permutarea $[a_1, ..., a_k]$ din S_n o numim *permutare ciclică* sau *k-ciclu* deoarece avem $a_1 \to a_2 \to ... \to a_k \to a_1$. Un 2-ciclu îl numim *transpoziţie*.

Propoziţia 8.20 1) *Orice element din S_n este produs de permutări ciclice distincte în sensul că un element apare cel mult odată.*
2) *Transpoziţiile generează S_n.*
3) *Transpoziţiile $[12], ..., [1n]$ generează S_n deoarece $[ab] = [1a][1b][1a]$.*
4) *Transpoziţiile $[12], [23], ..., [n-1n]$ generează S_n deoarece*
$[1k] = [k-1k]...[23][12][23]...[k-1k]$.
5) *Transpoziţia $[12]$ şi n-ciclul $[12...n]$ generează S_n deoarece $[kk+1] = [12...n]^{k-1}[12][12...n]^{1-k}$.*

Definiţia 8.21 Acţiunea lui G pe X se numeşte:
i) *tranzitivă* dacă există o singură orbită i.e. X/G este o mulţime singleton,
ii) *liberă* dacă toţi stabilizatorii se reduc la $\{e\}$.

SEMINARUL 8

S8.1 Fie $N(o)$ numărul orbitelor acţiunii lui G pe X i.e. $X = Orb(x_1) \cup ... \cup Orb(x_{N(o)})$ cu $Orb(x_i) \cap Orb(x_j) = \emptyset$ pentru i, j diferite. Pentru $g \in G$ fie $Fix(g) = \{x \in X; gx = x\}$. Are loc *formula Burnside*:

$$N(o) \cdot |G| = \sum_{g \in G} |Fix(g)|. \tag{8.2}$$

Rezolvare Vom număra în două moduri distincte elementele lui X invariate de acţiunea lui G. Avem:

$$\sum_{g \in G} |Fix(g)| = \sum_{x \in X} |Stab(x)|, \tag{8.3}$$

şi deci membrul drept din (7.2) este $\sum_{i=1}^{N(o)} \sum_{y \in Orb(x_i)} |Stab(y)|$. Dar pentru orice $y, z \in Orb(x_i)$ avem: $|Stab(y)| = |Stab(z)| frac|G||Orb(x_i)$ rezultă:

$$\sum_{g \in G} |Fix(g)| = \sum_{i=1}^{N(o)} |Orb(x_i)| \frac{|G|}{|Orb(x_i)|} = N(o) \cdot |G|.$$

S8.2 Fie $X = \mathbb{N}_3 = \{1, 2, 3\}$ şi $G = \{e, f, d\}$ cu $e = [1][2][3]$ permutarea identică, $d = [1, 2, 3]$ şi $f = [1, 3, 2]$. Considerăm acţiunea naturală a lui G pe X.
i) Să se arate că G este subgrup în S_3; deci grup.
ii) Se cer numărul orbitelor acestei acţiuni.

Rezolvare i) $df = [1, 2, 3][1, 3, 2] = [1][2][3] = e$, $fd = [1, 3, 2][1, 2, 3] = [1][2][3] = 3$ şi $d^2 = f, f^2 = d$. Deci $d = f^{-1}$ sau încă $f = d^{-1}$ ceea ce arată că G este subgrup în S_3.
ii) $|Fix(e)| = 3, |Fix(d)| = |Fix(f)| = 0$ şi deci $N(o) = \frac{3+0+0}{3} = 1$. Orbita unică este întreaga mulţime X ceea ce puteam vedea şi direct: $1 \sim 1$ cu $g = e$; $1 \sim 2$ cu $g = d$; $1 \sim 3$ cu $g = f$; deci toate elementele lui X sunt echivalente între ele.

Interpretare geometrică: X este mulţimea vârfurilor unui triunghi echilateral Δ, e=aplicaţia identică, d=rotaţia în sens orar de unghi $\frac{\pi}{3}$, f=rotaţia în sens trigonometric de unghi $\frac{\pi}{3}$. Avem imediat că $f = d^{-1}$ şi $f^2 = d$.

S8.3 Se dă semiautomatul A peste alfabetul binar cu $Q = \{q_0, ..., q_5\}$ şi δ dată de: $\delta(q_0, 0) = q_1$, $\delta(q_0, 1) = q_5$, $\delta(q_i, 0) = q_i, \delta(q_i, 1) = q_{i-1}, 1 \leq i \geq 5$.
i) Se cere reprezentarea tabelară a lui δ.
ii) Dacă $F = Q \setminus \{q_0\}$ se cere $L(A)$ pentru automatul A.

Rezolvare i)

δ	0	1
q_0	q_1	q_5
q_1	q_1	q_0
q_2	q_2	q_1
q_3	q_3	q_2
q_4	q_4	q_3
q_5	q_5	q_4

.

ii) $L(A) = \{0^n; n \in \mathbb{N}^*\} + \{010^n; n \in \mathbb{N}^*\} + \{1^a0^m; a = 1, 2, 3, 4, m \in \mathbb{N}\} + \{1^50^n; n \in \mathbb{N}^*\}$.

S8.4 Fie automatul A peste alfabetul binar cu $Q = \{q_0, q_1, q_2, q_3\}$ şi:

δ	0	1
$\rightarrow q_0$	$\{q_0, q_1, q_3\}$	q_0
q_1	q_2	$\emptyset$
q_2	q_3	$\emptyset$
$q_3 \leftarrow$	$\emptyset$	$\emptyset$

.

Să se studieze cuvintele $w_1 = 0110$, $w_2 = 101101$ şi $w_3 = 1011$.

Rezolvare $w_1, w_2 \in L(A)$ iar $w_3 \notin L(A)$.

S8.5 Să se arate că $\varphi : (\mathbb{Z}, +) \times \mathbb{R} \to \mathbb{R}$, $\varphi(k, x) = k + x$ este o acţiune şi să se studieze.

Rezolvare Verificarea axiomelor de acţiune este imediată. Fie $M = \mathbb{R}/(\mathbb{Z}, +, \varphi)$ şi fie $\psi : M \to S^1$, $\psi([x]) = e^{2\pi ix} = cos(2\pi x) + isin(2\pi x)$. Să verificăm buna definire: $[x] = [y]$ implică $y = x + n$ şi deci $cos(2\pi y) + isin(2\pi y) = cos(2\pi x) + isin(2\pi x)$.
1) ψ este injectivă deoarece $\psi(x) = \psi(y)$ implică $e^{2\pi i(y-x)} = 1$ adică $y - x \in \mathbb{Z}$ ceea ce înseamnă $[x] = [y]$.
2) ψ este surjectivă în mod evident.

În concluzie, $\mathbb{R}/(\mathbb{Z}, +, \varphi) = S^1$. Avem:
i) $Orb(x) = x + \mathbb{Z}$ pentru orice $x \in \mathbb{R}$,
ii) $Stab(x) = \{0\}$.
Deci, φ este acţiune liberă şi netranzitivă.

S8.6 Să se arate că $\varphi : (\mathbb{R}, +) \times \mathbb{R} \to \mathbb{R}$, $\varphi(t, x) = e^t x$ este o acţiune şi să se studieze.

Rezolvare Verificarea axiomelor de acţiune este imediată. Avem doar 3 orbite:
i) $[0] = \{0\}$ şi $Stab(0) = \mathbb{R}$,
ii) $[1] = \mathbb{R}^*_+$,
iii) $[-1] = \mathbb{R}^*_-$.
Deci $\mathbb{R}/(\mathbb{R}, +, \varphi) = \{[0], [1], [-1]\}$; pentru orice $x \neq 0$ avem $Stab(x) = \{0\}$. Deci acţiunea este neliberă şi netranzitivă.

S8.7 Fie n număr natural nenul. Un element (vector) din $\mathbb{R}^n$ îl notăm $\bar{x} = (x^1, ..., x^n)$. Să se arate că $\varphi : (\mathbb{R}^n, +) \times \mathbb{R}^{2n} \to \mathbb{R}^{2n}$, $\varphi(\bar{a}, (\bar{x}, \bar{y})) = (\bar{a} + \bar{x}, \bar{y})$ este o acţiune şi să se studieze.

Rezolvare Verificarea axiomelor de acţiune este imediată. Avem $Stab(\bar{x}, \bar{y}) = \{\bar{0}_n\}$ şi $(\bar{x}, \bar{y}) \sim (\bar{0}, \bar{y})$ prin intermediul lui $\bar{a} = -\bar{x}$. Deci $\mathbb{R}^{2n}/(\mathbb{R}^n, +, \varphi) = (\bar{0}, \mathbb{R}^n) \simeq \mathbb{R}^n$. Acţiunea este liberă şi netranzitivă.

S8.8 Să se arate că *translaţia* $\varphi : (\mathbb{R}^n, +) \times \mathbb{R}^n \to \mathbb{R}^n$, $\varphi(\bar{a}, \bar{x}) = \bar{a} + \bar{x}$ este o acţiune şi să se studieze.

Rezolvare Verificarea axiomelor de acţiune este imediată. Orice punct $\bar{x}$ se translează în origine cu $\bar{a} = -\bar{x}$ şi deci $\mathbb{R}^n/(\mathbb{R}^n, +, \varphi) = \{[\bar{0}]\}$. Acţiunea este liberă şi tranzitivă.

S8.9 (**Grupul ciclic de ordinul** 2) Să se arate că $(\mathbb{Z}_2, +)$ este grup izomorf cu $(C_2, \cdot)$ unde $C_2 = \{1, -1\}$.

Rezolvare Cele două grupuri au aceeaşi tabelă Cayley:

$(\mathbb{Z}_2,+)$	$\widehat{0}$	$\widehat{1}$
$\widehat{0}$	$\widehat{0}$	$\widehat{1}$
$\widehat{1}$	$\widehat{1}$	$\widehat{0}$

$(C_2,+)$	1	-1
1	1	-1
-1	-1	1

.

Prin urmare corespondenţa $\widehat{0}\to 1$, $\widehat{1}\to -1$ este un izomorfism de grupuri de la $\mathbb{Z}_2$ la C_2.

Generalizare *Grupul ciclic de ordinul* n este mulţimea C_n a simetriilor de rotaţie ale unui poligon regulat cu n laturi. C_n este deci mulţimea rotaţiilor de unghi $\theta_k=\frac{2k\pi}{n}$ cu $k\in\{0,...,n-1\}$, operaţia de grup fiind compunerea rotaţiilor. Rezultă că $(C_n,\circ)$ este grup izomorf cu $(\mathbb{Z}_n,+)$.

S8.10 Să se arate că $\varphi: C_2\times\mathbb{R}\to\mathbb{R}$, $\varphi(\varepsilon,x)=\varepsilon x$ este o acţiune pe $\mathbb{R}$ şi să se studieze.

Rezolvare Verificarea axiomelor de acţiune este imediată. Avem $Orb(x)=\{x,-x\}$ şi $Stab(x)=\{1\}$; deci acţiunea dată este liberă şi netranzitivă.

S8.11 Să se arate că $\varphi:(C_2\times C_2)\times\mathbb{R}^2\to\mathbb{R}^2$, $\varphi((\varepsilon,\delta),(x,y))=(\varepsilon x,\delta y)$ este o acţiune şi să se studieze.

Rezolvare Verificarea axiomelor de acţiune este imediată. Avem $Orb(x,y)=\{(x,y),(x,-y),(-x,y),(-x,-y)\}$ şi $Stab(x,y)=\{(1,1)\}$; deci acţiunea este liberă şi netranzitivă.

S8.12 Să se arate că $\varphi:(\mathbb{R}^*,\cdot)\times\mathbb{R}^n\to\mathbb{R}^n$, $\varphi(\lambda,\bar{x})=\lambda\bar{x}$ este o acţiune şi să se studieze.

Rezolvare Verificarea axiomelor de acţiune este imediată. Avem $Orb(\bar{0})=\bar{0}$, $Stab(\bar{0})=\mathbb{R}^*$ iar pentru $\bar{x}\neq\bar{0}$ avem $Stab(\bar{x})=1$ iar $Orb(\bar{x})$ este dreapta prin originea lui $\mathbb{R}^n$ fără origine. Spaţiul cât $\mathbb{R}^n\setminus\{\bar{0}\}/(\mathbb{R}^*,+,\varphi)$ se notează $P^{n-1}\mathbb{R}$ şi se numeşte *spaţiul proiectiv real* $n-1$*-dimensional*. Acţiunea pe $\mathbb{R}^n\setminus\{\bar{0}\}$ este liberă şi netranzitivă.

Course 9

Gramatici şi limbaje generate. Ierarhia Chomsky

Definiţia 9.1 Numim *gramatică* sau *sistem generativ* un 4-uplu $G = (V_N, V_T, S, P)$ cu:
i) V_N=mulţime nevidă, finită, numită *mulţimea variabilelor* (*neterminalilor*),
ii) V_T=mulţime nevidă, finită, disjunctă de V_N, numită *mulţimea terminalilor*. $V = V_N \cup V_T$ este *vocabularul* lui G,
iii) $S \in V_N$ este *simbolul de start* sau *axioma gramaticii*,
iv) $P \subset V^* \times V^*$, finită, este *mulţimea regulilor de generare* (*producţie*). Elementele $(\alpha, \beta) \in P$ sunt supuse condiţiei ca α să conţină un simbol din V_N; putem scrie $P \subset (V^* V_N V^*) \times V^*$.

Convenţii de notaţie 9.2
I) Următoarele simboluri notează elemente din V_N:
-$A, B, C, ..., S, ...$
-numele italice scrise cu minuscule: *expresie*, *instrucţiune*, ...
II) Următoarele simboluri notează terminali:
-$a, b, c, ..., 0, ..., 9$
-operatori: $+, -, \times, /$
-simboluri de punctuaţie, paranteze,
-unităţi lexicale: id, if, while, begin, ...
III) $X, Y, Z, ...$ sunt elemente din V iar $u, v, x, y, z, w, ...$ sunt cuvinte din V^*.
IV) $\alpha, \beta, \gamma, ...$ sunt cuvinte din V. Un element $(\alpha, \beta) \in P$ îl notăm $\alpha \to \beta$.
V) Dacă $A \to \alpha_1$,..., $A \to \alpha_n$ sunt toate producţiile cu originea în A, (A-producţii), notăm: $A \to \alpha_1 | ... | \alpha_n$.
VI) O gramatică va fi precizată prin producţiile sale; de aici se deduc mulţimile V_N şi V_T iar S este partea stângă a primei producţii.

Exemplu 9.3 Gramatica $S \to SOS| - S|(S)|id, O \to +| - | \times |/$ are $V_N = \{S, O\}$ şi $V_T = \{id, +, -, \times, /, (,)\}$.

Definiţia 9.4 Date $u, v \in V^*$ scrierea $u \to^* v$ notează faptul că $u = v$ sau există un şir finit $u_1 = u, ..., u_n = v$ de elemente din V^* astfel că $u_i \to u_{i+1}, 1 \leq i \leq n-1$. Un astfel de şir îl numim *derivare*.
Vom mai folosi notaţiile:
-"$\to^{(i)}$": derivarea directă a folosit regula i din P,
-"$\to^{i}$": derivarea s-a făcut aplicând i paşi (reguli),
-"$\to^{(i)j}$": se aplică regula i de j ori.
ii) *Limbajul* generat de G este $L(G) = \{w \in V_T^*; S \to^* w\}$ iar $w \in L(G)$ îl numim *frază* în G.
iii) $FP(G) = \{\alpha \in V^*; S \to \alpha\}$ este *mulţimea formelor propoziţionale* ale lui G. Deci $L(G) = FP(G) \cap V_T^*$ i.e. limbajul generat este mulţimea formelor propoziţionale ce conţin doar simboli terminali.
v) Gramaticele G_1, G_2 se numesc *echivalente* dacă $L(G_1) = L(G_2)$.

Ierarhia Chomsky 9.5 Fixăm gramatica G:
0) G generală se numeşte *de tip* 0,
1) dacă toate producţiile $\alpha \to \beta$ satisfac condiţia $|\alpha| \leq |\beta|$ atunci spunem că G este *de tip* 1 sau *monotone*,
1') dacă toate producţiile sunt de forma $\alpha A\beta \to \alpha\rho\beta$ cu $A \in V_N, \alpha, \beta \in V^*, \rho \in V^+$ atunci spunem că G este *dependentă de context*,
2) dacă toate producţiile sunt *context-free* i.e. de forma $A \to \alpha$ cu $A \in V_N, \alpha \in V^+$ atunci spunem că G este *de tip* 2 sau *context-free*,
lin) dacă toate producţiile sunt de forma $A \to \alpha$ sau $A \to \alpha B\beta$ cu $A, B \in V_n, \alpha, \beta \in V_T^*$ atunci spunem că G este *gramatică liniară*,
3) dacă toate producţiile sunt de forma $A \to aB$ sau $A \to a$ cu $A, B \in V_N$ şi $a \in V_T$ atunci spunem că G este *de tip* 3 sau *regulată*,
3') dacă toate producţiile sunt de forma $A \to \alpha$ sau $A \to \alpha B$ (respectiv $A \to B\alpha$) cu $A, B \in V_N, \alpha \in V_T$ atunci spunem că G este *drept liniară* (respectiv *stâng liniară*).
Limbajul $L \subset V_T^*$ se numeşte *de tip* $r, 0 \leq r \leq 3$ sau *liniar* dacă există o gramatică G de tip r sau liniară a. î. $L = L(G)$.

Gramaticile 1 şi 1' sunt echivalente şi la fel 3 şi 3'. Notând cu Lin respectiv L_r mulţimea limbajelor liniare respectiv de tip r avem:

$$L_3 \subset Lin \subset L_2 \subset L_1 \subset L_0$$

şi acest şir de incluziuni stricte se numeşte *ierarhia lui Chomsky* (1956).

Lingvistul american Noam Chomsky a introdus gramaticile libere de context în scopul descrierii limbilor naturale. Deşi acest scop s-a dovedit

mult prea ambiţios, limbajele (gramaticile) context-free s-au arătat utile în descrierea limbajelor de programare. Astfel, au fost utilizate de Bachus pentry FORTRAN şi Naur pentru ALGOL (din acest motiv, gramaticile libere de context se numesc uneori *gramatici în forma Backus-Naur*) în timp ce recent HTML-ul a fost descris cu un limbaj independent de context.

"An HTML document (with arbitrary text content) has this sort of structure:
<HTML>
<HEAD> <TITLE> Jane Doe's Home Page </TITLE> </HEAD>
<BODY> <H1> Jane Doe </H1> <H2> Home Page </H2>
<P>
<CENTER> <IMG src="jane.jpg"> </CENTER>
</P>
</BODY>
</HTML>
The expression <HTML> must be followed by </HTML>, <HEAD> must be followed by </HEAD>, and so on, in the same pattern as matched parentheses. Thus recognizing that a string belongs to a certain CFL (context-free language) is one of the tasks performed by a web browser."

Aceleaşi limbaje context-free sunt deosebit de importante în proiectarea compilatoarelor.

Pentru generalităţi asupra operei lui Chomsky a se vedea
http://en.wikipedia.org/wiki/Noam_Chomsky ca şi site-ul personal
http://www.chomsky.info/. Alte site-uri recomandate:
-http://en.wikipedia.org/wiki/Chomsky_hierarchy
-http://mathworld.wolfram.com/Grammar.html.

În aplicaţii, să notăm că dat $L \in L_0$ se caută r maximal pentru care $L \in L_0$. Astfel, este posibil ca iniţial să avem $L = L(G)$ cu G de tip r dar să existe $r' > r$ şi G' de tip r' astfel încât $L = L(G')$.

SEMINARUL 9

S9.1([13, p. 18]) Să se arate că $L_1 = \{a^n b^n; n \geq 0\} \in Lin - L_3$.

Rezolvare([10, p. 11]) Fie gramatica G cu $V_N = \{S\}, V_T = \{a, b\}$ şi $P : S \to ab|aSb$. Avem că G este liniară dar nu este regulată. Arătăm prin inducţie că $L_1 \subseteq L(G)$.
1) pasul de pornire e evident din prima regulă.
2) presupunem că $a^k b^k \in L(G), k \geq 1$. Deci avem o derivare $S \to^* a^k b^k$ cu

$u_n = a^k b^k$ şi $u_{n-1} = a^{k-1} S b^{k-1}$. Avem atunci $u_{n-i} \to^{(2)} a^{k-1}(aSb)b^{k-1} = a^k S b^k \to^{(1)} a^{k+1} b^{k+1}$ ceea ce voiam.

A mai rămas de arătat că $L(G) \subseteq L_1$. Analizând derivările posibile din G avem $PF(G) = \{a^k S b^k, a^k b^k\}$ dar ultima derivare nu mai poate continua datorită regulilor de producţie din G. Avem deci concluzia.

S9.2([13, p. 18]) Să se arate că $L_2 = L_1 L_1 \in L_2 - Lin$.

S9.3([13, p. 18]) Să se arate că $L_3 = \{a^n b^n c^n; n \geq 1\} \in L_1 - L_2$.

Rezolvare([10, p. 20]) Fie G cu $V_N = \{S, A\}, V_T = \{a, b, c\}$ şi $P : S \to abc|aSA, bA \to bbc, cA \to AC$. Este o gramatică monotonă dar nu de tipul 2. Arătăm prin inducţie că $L_3 \subseteq L(G)$.

S9.4([13, p. 18]) Să se arate că $L_4 = \{a^{2^n}; n \geq 0\} \in L_1 - L_2$.

Rezolvare([10, p. 24]) Fie G cu $V_N = \{S, A, B, C\}, V_T = \{a\}$ şi $P : S \to BAB, BA \to BC, CA \to AAC, CB \to AAB, A \to a, B \to \varepsilon$.

S9.5([13, p. 18]) Să se arate că $L_5 = \{a^{n^2}; n \geq 0\} \in L_1 - L_2$.

S9.6([13, p. 18]) Să se arate că $L_6 = \{a^n; n = prim\} \in L_1 - L_2$.

S9.7([13, p. 18]) Să se arate că $L_7 = \{a^n b^m a^n b^m; n, m \geq 1\} \in L_1 - L_2$.

S9.8([13, p. 18]) Să se arate că $L_8 = \{a^n b^m; n \geq 1, 1 \leq m \leq 2^n\} \in L_1 - L_2$.

S9.9([13, p. 18]) Să se arate că $L_9 = \{a^n b^m c^p; 1 \leq n \leq m \leq p\} \in L_1 - L_2$.

S9.10([13, p. 18]) Să se arate că $L_{10} \in L_2 - Lin$ unde L_{10} este *limbajul lui Dyck* pentru vocabularul $\{a, b\}$ i.e. limbajul generat de gramatica independentă de context $G = (\{S\}, \{a, b\}, S, P)$ cu $P : S \to SS|aSb|\varepsilon$.

S9.11 .

Rezolvare .

S9.12 .

Rezolvare .

Course 10

Problema cuvintelor

Deoarece vom lucra în acest curs pe grupuri, extindem mai întâi definiţia cuvintelor pentru a formaliza noţiunea de invers. Fixăm $X = \{x_1, ..., x_k\}$ o mulţime finită. Numim *cuvânt de lungime* n pe mulţimea X o funcţie $w : \mathbb{N}_n = \{1, ..., n\} \to X \times \{1\}$. Dacă $w(i) = (w_i, \varepsilon_i)$ atunci cuvântul w se mai notează $w = x_{w_1}^{\varepsilon_1} ... x_{w_n}^{\varepsilon_n}$. Operaţia de concatenare se defineşte uzual: $ww' := x_{w_1}^{\varepsilon_1} ... x_{w_n}^{\varepsilon_n} x_{w'_1}^{\varepsilon'_1} ... x_{w'_n}^{\varepsilon'_n}$ şi rămâne asociativă. Fie $W(X)$ acest monoid relativ la cuvântul vid.

Definiţia 10.1 i) O *echivalenţă elementară* pe $W(X)$ este o pereche de cuvinte de forma $(d_1 x_w^{\varepsilon} x_w^{-\varepsilon} d_2, d_1 d_2)$ cu $\varepsilon \in \{1\}$ şi d_1, d_2 cuvinte arbitrare.
ii) Pe $W(X)$ definim o relaţie în modul următor: două cuvinte le numim *echivalente* dacă pot fi legate printr-un lanţ finit de echivalenţe elementare. Astfel, cele două cuvinte se transformă unul în celălalt prin ştergerea sau introducerea unor perechi $x_w x_w^{-1}$ sau $x_w^{-1} x_w, w = 1, ..., n$.

Propoziţia 10.2 *Relaţia astfel introdusă este o congruenţă.*

Monoidul cât devine grup definind inversul astfel: $[w = x_{w_1}^{\varepsilon_1} ... x_{w_n}^{\varepsilon_n}]^{-1} = [w = x_{w_n}^{-\varepsilon_n} ... x_{w_1}^{-\varepsilon_1}]$ şi notând cu 1 clasa cuvântului vid.. Acest grup numit *grupul liber generat de* X şi notat $F(X)$; are o proprietate de universalitate în categoria grupurilor: pentru orice grup G şi elemente fixate $g_1, ..., g_k \in G$ există un unic morfism de grupuri $\varphi : F(X) \to G$ satisfăcând $\varphi(x_i) = g_i, i = 1, ..., k$.

Mai general, fixăm $R = \{c_1, ..., c_m\}$ o mulţime de cuvinte peste Σ numite *relaţii*. Intersecţia tuturor subgrupurilor normale ce conţine R îl notăm $N(R)$ şi este subgrup normal. Putem vorbi atunci de grupul factor $F(\Sigma)/N(R)$ notat $< X|R >$ şi pentru care elementele lui X le numim *generatori*. Mai spunem că grupul $G =< X|R >$ este *prezentat prin generatori şi relaţii*.

Observaţia 10.3 Trebuie avut grijă atât în precizarea generatorilor cât şi a relaţiilor:
1) Prezentarea $< x, y|xy = yx, xyx^{-1} = yxy^{-1} >$ este greşită deoarece $x = y$. Înmulţim a doua relaţie la dreapta cu y: $xyx^{-1}y = yx = xy$. Simplificăm prin xy la stânga şi avem $x^{-1}y = 1$ de unde concluzia.
2) Relaţiile $x^2 = y^2 = (xy)^2 = 1$ implică comutativitatea $xy = yx$. În adevăr, din ultima relaţie: $xyxy = 1$ pe care o înmulţim la stânga succesiv cu x şi apoi y.

Exemple 10.4 1) $C_n :< x|x^n = 1 >$ este prezentarea *grupului ciclic de ordin* $n \geq 2$ din cursul 8.
2) $< x, y|x^2 = y^3 = 1, yxy = x >$ este o prezentare a grupului simetric S_3 luând $x = [12]$ şi $[123]$. De aici rezultă că S_3 este primul grup simetric neabelian deoarece din a treia relaţie avem $yx = xy^{-1}$ iar $y^{-1} = y$ are da $y^2 = 1$ care împreună cu a doua relaţie conduce la contradicţia $y = 1$.

Noţiunea centrală a acestui curs a fost formulată de Dehn sub numele de *problema cuvintelor* astfel: dat $w \in X^*$ să se găsească o procedură conţinând un număr finit de instrucţiuni pentru a decide dacă $w = 1$ sau nu. O formulare modernă este următoarea:

Definiţia 10.5 Grupul $G =< X|R >$ *are soluţie la problema cuvintelor* dacă limbajul $W(G) = \{w \in X^*; w = 1\}$ este recunoscut de un automat determinist.

Exemplul 10.6 Dacă X este finită atunci grupul liber $F(X) =< X|\emptyset >$ are soluţie la problema cuvintelor.

Definiţia 10.7 Dat grupul $G =< X|R >$ şi limbajul $L \subset X^*$ spunem că perechea (X, L) este o *structură raţională* pentru G dacă L este regulat şi L generează pe G.

Considerăm un simbol $\$ \notin X$ ("the padding simbol"=simbolul auxiliar) şi definim $X' = X \cup \{\$\}$ şi $X(2, \$) = X' \times X' \setminus \{\$, \$\}$. Definim $\mu : X^* \times X^* \to X(2, \$)^*$ prin:
1) $\mu(u = x_1...x_m, v = y_1...y_n) = (x_1, y_1)...(x_n, y_n)(x_{n+1}, \$)...(x_m, \$)$ dacă $n < m$,
2) $\mu(u, v) = (x_1, y_1)...(x_n, y_n)$ dacă $n = m$,
3) $\mu(u, v) = (x_1, y_1)...(x_m, y_m)(\$, y_{n+1})...(\$, y_n)$ dacă $n > m$.
Fie (X, L) structură raţională pentru G şi $w \in X^*$; definim
$L_w = \{\mu(w_1, w_2); w_1, w_2 \in L, w_1 = ww_2\}$.

Definiţia 10.8 i) Structura raţională (X, L) a lui G se numeşte *structură automată* dacă L_ε şi L_x, pentru orice $x \in L$, sunt limbaje regulate.

ii) Grupul G se numeşte *automat* dacă admite o structură automată.

Exemple 10.9 Sunt grupuri automate următoarele clase de grupuri: grupurile finite, grupurile libere finit generate, grupurile abeliene finit generate, grupuri *braid*.

Dat $w \in X^*$ cu $w = 1$ reamintim că $w =_{F(X)} \prod_i = 1^k (u_i r_i^{\pm 1} u_i^{-1})$ cu $u_i \in F(X), r_i \in R, k \in \mathbb{N}$. Fie $a(w)$ cea mai mică valoare a lui k.

Definiţia 10.10 *Funcţia izoperimetrică* a lui $G =< X|R >$ este $f_G : \mathbb{N} \to \mathbb{N}$:

$$f(n) = max\{a(w); |w| \leq n, w = 1\}.$$

Problema cuvintelor pe grupuri automate este rezolvată de:

Teorema 10.11 *Dacă G este grup automat atunci*:
1) *G admite o prezentare finită cu funcţia izoparametrică marginită superior de o funcţie pătratică.*
2) *G are soluţie la problema cuvintelor.*

Demonstaţia acestui rezultat fundamental se bazează pe:

Propoziţia 10.12 *Fie $< X|R >$ o prezentare finită a grupului G. Următoarele sunt echivalente*:
i) *Funcţia izoparamerică este mărginită superior de o funcţie recursivă.*
ii) *G are soluţie la problema cuvintelor.*
iii) *Funcţia izoparametrică este mărginită.*

SEMINARUL 10

S10.1 Să se arate că următoarele sunt prezentări ale grupului trivial:
a) $< x, y|x^2 = y^3, xyx = yxy >$.
b) $< x, y|xy = y^2x, yx = x^2y >$.
c) $< x, y, z|xy = y^2x, yz = z^2y, zx = x^2z >$.

Rezolvare a) Din a doua relaţie prin înmulţirea cu x la stânga şi la dreapta avem: $x^2yx^2 = xyxyx$ adică $y^7 = y^3yy^3 = xyxyx$. Tot din a doua relaţie prin înmulţire la dreapta cu yx avem $xyxyx = yxy^2x$; deci $y^7 = yxy^2x$ de unde avem $y^6(= x^4) = xy^2x$. Obţinem $x^2 = y^2$ şi din $y^2 = y^3$ avem $y = 1$. Revenind la a doua relaţie rezultă şi $x = 1$.

S10.2 Grupul diedral (*dihedral group* în engleză) D_n este grupul simetriilor de rotaţie al unei plăci în formă de poligon regulat cu n laturi. (Alţi

autori îl notează D_{2n}.) Fie r rotaţia de unghi $\frac{2\pi}{n}$ în jurul unei axe de simetrie perpendiculară pe poligon şi s rotaţia de unghi π în jurul unie axe de simetrie din planul poligonului. Atunci o prezentare a lui D_n este:

$$D_n =< r, s | r^n = 1, s^2 = 1, sr = r^{n-1}s >$$

şi orice element din D_n este de forma r^k sau $r^k s$ cu $0 \leq k \leq n-1$. Identităţile de calcul în D_n:
i) $r^a r^b = r^k$ cu $k = a + b(mod\ n)$,
ii) $(r^a s)r^b = r^l s$ cu $l = a - b(mod\ n)$,
iii) $(r^a s)(r^b s) = r^l s$.
Rezultă şi alte perechi de generatori:
I) (rs, s) deoarece $r = (rs)s$,
II) $(rs, r^2 s)$.

Rezolvare .

S10.3 Există c si *grupul diedral infinit* D_∞ generat de $t, s : \mathbb{Z} \to \mathbb{Z}, t(z) = z + 1, s(z) = -z$. Avem $s^2 = 1$ în timp ce t are ordin infinit. Identitatea $tst = s$ este evidentă din schema membrului stâng: $z \to z + 1 \to -z - 1 \to -z = s(z)$. Deci: $D_\infty =< t, s | s^2 = 1, tst = s >$.

Rezolvare .

S10.4

Rezolvare

Course 11

Funcţii recursive

În acest curs considerăm funcţii f de tipul următor:
-*funcţie parţială* dacă $f : X \to \mathbb{N}$ cu X submulţime (nevidă) a lui $\mathbb{N}^n$,
-*funcţie totală* dacă f este definită pe tot $\mathbb{N}^n$.
Pentru simplitatea scrierii, ambele tipuri de funcţii le notăm la fel $f : \mathbb{N}^n \to \mathbb{N}$, înţelegându-se din context tipul. Fie $\mathscr{P}$ mulţimea tuturor funcţiilor parţiale (deci orice n) şi $\mathscr{T}$ mulţimea tuturor funcţiilor totale. De asemenea, n-uplul $(x_1, ..., x_n)$ îl notăm $\bar{x}$.

Definiţia 11.1 1) Fie numerele naturale $n, k \geq 1$ şi funcţiile $g : \mathbb{N}^k \to \mathbb{N}, h_1, ..., h_k : \mathbb{N}^n \to \mathbb{N}$ din $\mathscr{P}$. Numim *compunerea* lor funcţia $f = g \circ (h_1, ..., h_k) : \mathbb{N}^n \to \mathbb{N}$ dată de $f(\bar{x}) = g(h_1(\bar{x}), ..., h_k(\bar{x}))$. Evident $f \in \mathscr{P}$ membrul stâng fiind definit acolo unde se poate defini membrul drept. Mai spunem căm definit *operatorul de superpoziţie* SUP.
2) Fie $g : \mathbb{N}^n \to \mathbb{N}$ şi $h : \mathbb{N}^{n+2} \to \mathbb{N}$ din $\mathscr{P}$. Funcţia parţială $f : \mathbb{N}^{n+1} \to \mathbb{N}$ dată de $f(\bar{x}, 0) = g(\bar{x}), f(\bar{x}, y + 1) = h(\bar{x}, f(\bar{x}), y)$ se numeşte *obţinută prin recursie primitivă*. Cazul $n = 0$ este admis şi atunci g se consideră număr natural fixat. Mai spunem căm definit *operatorul de recursie primitivă* REC.

Observaţia 11.2 Dat $\bar{x}$ avem că $f(\bar{x}, y)$ este definit: sau pentru niciun y sau pentru orice $y \in \mathbb{N}$ sau pentru $y \in \mathbb{N}_k = \{1, ..., k\}$ cu k determinat de g şi h.

Definiţia 11.3 1) Următoarele funcţii le numim *iniţiale*:
i) *funcţia zero* $z : \mathbb{N} \to \mathbb{N}, z(x) = 0$,
ii) *fucţia succesor* $\sigma : \mathbb{N} \to \mathbb{N}, \sigma(x) = x + 1$,
iii) *funcţii proiecţie* $\pi_{kr} : \mathbb{N}^k \to \mathbb{N}, \pi_{kr}(\bar{x}) = x_r, k \geq 1, 1 \leq r \leq k$.
Evident, toate funcţiile iniţiale aparţin lui $\mathscr{T}$.

2) Numim *clasă de funcţii* o submulţime a lui $\mathscr{P}$ şi *clasă de funcţii totale* o submulţime a lui $\mathscr{T}$.
3) O clasă $\mathscr{C}$ de funcţii totale o numim *închisă primitiv recursiv* dacă:
i) toate funcţiile iniţiale aparţin lui $\mathscr{C}$.
ii) $\mathscr{C}$ este închisă la compunere i.e. dacă $g, h_1, ..., h_k \in \mathscr{C}$ atunci $g \circ (h_1, ..., h_k) \in \mathscr{C}$.
iii) $\mathscr{C}$ este închisă la recursia primitivă i.e. dacă $g, h \in \mathscr{C}$ atunci f oţinută din g şi h prin recursie primitivă este element din $\mathscr{C}$.

Există o cea mai mică mulţime închisă primitiv recursiv $\mathscr{F}(pr)$ anume intersecţia tuturor claselor închise primitiv recursiv.

Definiţia 11.4 Un element $f \in \mathscr{F}(pr)$ îl numim *funcţie primitiv recursivă.*

Teorema 11.4 (de caracterizare) *Fie $f \in \mathscr{T}$. Atunci $f \in \mathscr{F}(pr)$ dacă şi numai dacă există un şir $f_0, ..., f_k = f$ unde f_i este sau funcţie iniţială sau se obţine prin compunere din unele f_j cu $j < i$ sau se obţine prin recursie primitivă din două funcţii $f_j, j < i$.*

Definiţia 11.6 Un şir de tipul celui precedent îl numim *definiţie primitiv recursivă* a lui f.

Exemplu 11.7 Fie $f : \mathbb{N}^n \to \mathbb{N}$ element dintr-o clasă închisă primitv recursiv $\mathscr{C}$ şi definim $g : \mathbb{N}^m \to \mathbb{N}$ prin $g(x_1, ..., x_m) = f(y_1, ..., y_n)$ unde y_i este sau o constantă sau x_j pentru un j fixat. Atunci $g \in \mathscr{C}$ deoarece $g = f \circ (h_1, ..., h_m)$ cu h_j sau funcţie constantă (care este primitiv recursivă; vezi Ex. 10.?) sau o funcţie π_{kj}.

Propoziţia 11.8 *Fie $\mathscr{C}$ o clasă închisă primitiv recursiv şi $g \in \mathscr{C}$ de forma $g : \mathbb{N}^{n+1} \to \mathbb{N}$. Atunci următoarele funcţii aparţin lui $\mathscr{C}$:*
1)(*adunarea repetată*) $f_1 : \mathbb{N}^{n+1} \to \mathbb{N}, f_1(\bar{X}, y) = \sum_{t=0}^{y} g(\bar{X}, t)$.
2)(*înmulţirea repetată*) $f_2 : \mathbb{N}^{n+1} \to \mathbb{N}, f_1(\bar{X}, y) = \prod_{t=0}^{y} g(\bar{X}, t)$.
Demonstraţie Definiţia primitiv recursivă a acestor funcţii este:
1) $f_1(\bar{x}, 0) = g(\bar{x}, 0), f_1(\bar{x}, y+1) = f_1(\bar{x}, y) + g(\bar{x}, y+1)$.
2) $f_2(\bar{x}, 0) = g(\bar{x}, 0), f_2(\bar{x}, y+1) = f_2(\bar{x}, y) g(\bar{x}, y+1)$. □

Definiţia 11.9 1) Dat $n \geq 1$ numim *predicat n-ar* o afirmaţie $P(x_1, ..., x_n)$ în n variabile ce este adevărată sau falsă în funcţie de valorile variabilelor considerate ca elemente din $\mathbb{N}$. Predicatul dat se identifică cu mulţimea $T(P) = \{\bar{x} \in \mathbb{N}; P(\bar{x}) = \text{adevărată}\}$.
2) Dată $\mathscr{C}$ o clasă închisă primitiv recursiv. O submulţime A a lui $\mathbb{N}^n$ o numim *în $\mathscr{C}$* dacă funcţia caracteristică $\chi_A \in \mathscr{C}$. În particular, un predicat n-ar este *în $\mathscr{C}$* dacă $\chi_{T(P)} \in \mathscr{C}$.

Pentru rezultatul următor reamintim că date predicatele P şi Q avem: $P \vee Q$ înseamnă "P sau Q", $P \wedge Q$ înseamnă "P şi Q" iar $\neg P$ este negaţia lui P.

Propoziţia 11.10 *Fie $\mathscr{C}$ o clasă închisă primitiv recursivă şi $A, B \subset \mathbb{N}^n$. Dacă A şi B sunt în $\mathscr{C}$ atunci $A \cup B, A \cap B$ şi $\mathbb{N}^n \setminus A$ sunt în $\mathscr{C}$. În consecinţă, date predicatele n-are P şi Q din $\mathscr{C}$ avem că $P \vee Q, P \wedge Q$ şi $\neg P$ sunt în $\mathscr{C}$.*

Demonstraţie $\chi_{A \cup B} = \chi_A \chi_B$, $\chi_{A \cap B} = sg(\chi_A + \chi_B)$ şi $\chi_{cA} = 1 \dot{-} \chi_A$. □

În cele ce urmează $x = y$ înseamnă predicatul $P(x, y)$ adevărat doar când x şi y sunt egale, etc.:

Propoziţia 11.11 *Predicatele $x = y, x \neq y, x < y, x \leq y, x > y, x \geq y$ sunt primitiv recursive.*

Demonstraţie $\chi_{\neq}(x, y) = sg(|x - y|)$, $\chi_{<}(x, y) = sg(x \dot{-} y)$. Analog, $=$ este $\neg(\neq)$, $\leq$ este $< \vee =$, $\geq$ este $\neg(<)$. □.

Definiţia 11.12 1)Fie mulţimea X şi funcţia parţială $f : X \to X$. Se numeşte *iteraţia* sau *iterata* lui f funcţia parţială $F : X \times \mathbb{N} \to X$ dată de $F(x, 0) = f(x)$ şi $F(x, n + 1) = f(F(x, n))$. Dacă f este totală notăm $F(x, n) = f^n(x)$.
2) Spunem că functţa $f : \mathbb{N}^n \to \mathbb{N}^k$ *aparţine* clasei $\mathscr{C}$ dacă toate funcţiile $\pi_{kj} \circ f$ sunt din $\mathscr{C}$ pentru $1 \leq j \leq k$.
3) Clasa $\mathscr{C}$ de funcţii se numeşte *închisă la iteraţii* dacă odată cu functia $f : \mathbb{N}^n \to \mathbb{N}^n$ din $\mathscr{C}$ avem că şi iterata $F : \mathbb{N}^{n+1} \to \mathbb{N}^n$ este din $\mathscr{C}$.

Se poate arăta că dacă $\mathscr{C}$ este închisă primitiv recursiv atunci $\mathscr{C}$ este închisă la iteraţii. Suntem interesaţi în reciproca acestui fapt.

Propoziţia 11.13 *Fie $\mathscr{C}$ o clasă de funcţii ce conţine funcţiile iniţiale şi este închisă la iteraţii. Atunci $\mathscr{C}$ este închisă primitiv recursiv.*

Vom introduce acum cea mai genrală clasă de funcţii recursive pentru care trebuie considerat un nou mod de generare de funcţii. Fie deci $f : \mathbb{N}^{n+1} \to \mathbb{N}$ o funcţie parţială. Vom defini o altă funcţie $g : \mathbb{N}^n \to \mathbb{N}$ prin $g(\bar{x})$=cea mai mică valoare a lui $y \in \mathbb{N}$ pentru care $f(\bar{x}, y) = 0$. Datorită caracterului parţial al lui f sunt necesare câteva precizări şi de aceea introducem:

Definiţia 11.14 *Funcţia de minimizare* a lui f este funcţia parţială $g : \mathbb{N}^n \to \mathbb{N}$ dată de:
1) $g(\bar{x}) = r$ dacă $f(\bar{x}, r) = 0$ şi pentru $0 \leq s < r$, $f(\bar{x}, s)$ este definită şi nenulă,

2) $g(\bar{x})$ este nedefinită în caz contrar. Folosim notaţia $g(\bar{x}) = \mu_y(f(\bar{x}, y = 0))$. Atenţie, g poate fi parţială chiar şi când f este totală. Spunem că am definit *operatorul de minimizare* MIN

Definiţia 11.15 Date funcţiile f şi g ca mai sus, spunem că g se obţine din f prin *minimizare regulată* dacă f este totală şi pentru orice $\bar{x} \in \mathbb{N}^n$ există $y \in \mathbb{N}$ aşa încât $f(\bar{x}, y) = 0$. Rezultă că g este atunci funcţie totală.

Definiţia 11.16 *Clasa funcţiilor recursive* ete cea mai mică clasă $\mathscr{C}$ de funcţii totale care este închisă primitiv recursiv şi închisă la minimizare reguluată i.e. dacă $f \in \mathscr{C}$ şi g se obţine din f prin minimizare regulată atunci $g \in \mathscr{C}$.

O astfel de clasă există, fiind de fapt intersecţia tuturor claselor ce verifică proprietăţile menţionate.

Definiţia 11.17 1) O submulţime A a lui $\mathbb{N}^n$ se numeşte *recursivă* dacă χ_A este funcţie recursivă.
2) Predicatul n-ar P se numeşte *recursiv* dacă mulţimea
$A_P = \{\bar{x} \in \mathbb{N}^n; P(\bar{x}) = \text{adevărat}\}$ este recursivă.

Definiţia 11.18 *Clasa funcţiilor parţial recursive* este cea mai mică clasă de funcţii parţiale ce conţine funcţiile iniţiale şi este închisă la compunere, primtiv recursivitate şi minimizare.

Clasa funcţiilor parţial recursive ce sunt totale este închisă primitiv recursiv şi închisă la minimizare regulată; prin urmare conţine clasa funcţiilor recursive. Deci, o funcţie recursivă este parţial recursivă şi totală dar reciproca nu este valabilă.

Exemple 11.19 Funcţia $f : \mathbb{N} \to \mathbb{N}, f(x) = \mu_y(x(y+1) = 0)$ este parţial recursivă dar nefiind totală nu este recursivă. În adevăr, $f(0) = 0$ şi în rest f este nedefinită.

Definiţia 11.20 Se dau funcţiile $f, g : \mathbb{N}^2 \to \mathbb{N}$. Spunem că f este *func ţia de minimizare limitată* a lui g dacă $g(x, z) = \mu_y \leq z(f(x, y) = 0)$ i.e. valoarea $g(x, z)$ se obţine astfel: dacă există $0 \leq y_0 \leq z$ astfel ca $f(x, 0) > 0, ..., f(x, y_0 - 1) > 0$ şi $f(x, y_0) = 0$ atunci $g(x, z) = y_0$; în caz contrar $g(x, z) = z + 1$. Spunem că am definit *operatorul de minimizare limitată*.

Concluzii: Criteriul de recunoaştere a funcţiilor primitiv recursive sau recursive Se dau şirul finit de funcţii $f_0, ..., f_k$ şi funcţia f:
1) şirul dat îl numim *pr-şir* dacă orice element al său este funcţie iniţială sau se obţine din precedentele elemente cu operatorii SUP sau REC. Dacă

în plus folosim şi operatorul MIN spunem că avem un *r-şir*.
2) f este *primitiv recursivă* dacă există un pr-şir cu f ca element final.
3) f este *recursivă* dacă există un r-şir cu f ca element final.

SEMINARUL 11

S11.1 Să se arate că următoarele funcţii sunt primitiv recursive:
1) *funcţia sumă* $s : \mathbb{N}^2 \to \mathbb{N}, s(x,t) = x + y$.
2) *funcţia produs* sau *multiplicare* $m : \mathbb{N}^2 \to \mathbb{N}, m(x,y) = xy$.
3) *funcţia exponenţială* $exp : \mathbb{N}^2 \to \mathbb{N}, exp(x,y) = x^y$.
4) *funcţia factorial* $Fac : \mathbb{N} \to \mathbb{N}, Fac(x) = x!$.
5) *orice funcţie constantă* $c : \mathbb{N}^n \to \mathbb{N}, c(\bar{x}) = c$ cu $c \in \mathbb{N}$ fixat.
6) *funcţia predecesor* $Pred : \mathbb{N} \to \mathbb{N}, Pred(x) = x-1$ dacă $x > 1$ şi $Pred(0) = 0$.
7) *scăderea proprie* $\dot{-} : \mathbb{N}^2 \to \mathbb{N}, x \dot{-} y = max\{x - y, 0\}$.
8) *funcţia modul* $| \ | : \mathbb{N} \to \mathbb{N}$.
9) *funcţia semn* $sg : \mathbb{N} \to \mathbb{N}, sg(x) = 1$ dacă $x > 0$ şi $sg(0) = 0$.
10) $\overline{sg} : \mathbb{N} \to \mathbb{N}, \overline{sg}(x) = 0$ dacă $x > 0$ şi $\overline{sg}(0) = 1$.

Rezolvare 1) $s(x,0) = \pi_{11}(x), s(x,y+1) = s(x,y)+1 = \sigma \circ \pi 33(x,y,s(x,y))$. Putem spune că $\pi_{11}, \pi_{33}, \sigma, \sigma \circ \pi_{33}$ este o definiţie primitiv recursivă pentru s. În cele ce urmează vom restrânge demonstraţia la indicarea recursivităţii.
2) $m(x,0) = z(x), m(x,y+1) = m(x,y) + x$.
3) $exp(x,0) = 1, exp(x,y+1) = m(exp(x,y),x)$.
4) $Fac(0) = 1, Fac(x+1) = m(Fac(x), x+1)$.
5) Să considerăm $n = 1$; atunci funcţia constantă 0 este z, funcţia constantă 1 este $\sigma \circ z$, funcţia constantă 2 este $\sigma^2 \circ z$, etc. Pentru n general, funcţia constantă c este $c' \circ \pi_{n1}$ cu $c' : \mathbb{N} \to \mathbb{N}$ funcţia constantă c.
6) $Pred(0) = 0, Pred(x+1) = x = s(x,0)$.
7) $x \dot{-} 0 = x = s(x,0), x \dot{-} (y+1) = Pred(x \dot{-} y)$.
8) $|x - y| = (x \dot{-} y) + y \dot{-} x$.
9) $sg(0) = 0, sg(x+1) = 1$.

S11.2 Să se arate că funcţiile următoare sunt primitiv recursive:
1) $\lfloor ./. \rfloor : \mathbb{N}^2 \to \mathbb{N}, \lfloor x/y \rfloor =$ cel mai mic număr natural mai mic sau egal cu x/y dacă $y > 0$ respectiv $\lfloor x/0 \rfloor = 0$.
2) $\lceil ./. \rceil : \mathbb{N}^2 \to \mathbb{N}, \lceil x/y \rceil =$ cel ami mic număr natural mai mare sau egal cu x/y dacă $y > 0$ respectiv $\lceil x/0 \rceil = 0$.
3) $rest : \mathbb{N}^2 \to \mathbb{N}, rest(x,y) =$ restul împărţirii lui x la y dacă $y > 0$ respectiv $rest(x,0) = 0$.
4) $prim : \mathbb{N} \to \mathbb{N}, prim(n) =$ al n-lea număr prim cu $prim(0) = 2$.

Rezolvare .

S11.3 .

Rezolvare

Course 12

Mulţimi şi limbaje recursiv enumerabile

Definiţia 12.1 Submulţime a A a lui $\mathbb{N}$ se numeşte *recursiv enumerabilă* (r. e. pe scurt) dacă $A = \emptyset$ sau există $f : \mathbb{N} \to \mathbb{N}$ recursivă aşa încât $A = f(\mathbb{N}$. Alţi autori folosesc denumirea de mulţime *semirecursivă*, [15, p. 93]

Observaţia 12.2 Noţiunea astfel introdusă formalizează conceptul de mulţime *listabilă* deoarece elementele lui A se pot lista: $f(0), f(1), ...$, printr-o procedură cu un număr finit de instrucţiuni.

Pentru a caracteriza acest tip de mulţime considerăm *funcţia caracteristică parţială* a lui A, $\chi_{pA} : \mathbb{N} \to \mathbb{N}$: $\chi_{pA}(x) = 1$ dacă $x \in A$ şi $\chi_{pA}(x)$ este nedefinită dacă x nu aparţine lui A.

Propoziţia 12.3 *Pentru $A \subset \mathbb{N}$ următoarele afirmaţii sunt echivalente*:
1) *A este r.e.*.
2) *A este domeniul de definiţie al unei funcţii parţial recursive $g : \mathbb{N} \to \mathbb{N}$.*
3) *funcţia caracteristică parţială χ_{pA} este parţial recursive.*
4) *A este imaginea unie funcţii parţial recursive.*
5) *sau $A = \emptyset$ sau există o funcţie primitiv recursivă $f : \mathbb{N} \to \mathbb{N}$ aşa încât $A = f(\mathbb{N})$.*

Demonstraţie 1) $\Rightarrow$ 2). Dacă $A = \emptyset$ atunci putem considera A ca domeniu al unei funcţii g parţial recursive având domeniul vid: spre exemplu $g(x) =$ cel mai mic y natural pentru care $x + y + 1 = 0$. Fie acum $A = f(\mathbb{N})$ cu f recursivă şi definim $g(x) =$ cel mai mic y natural pentru care $f(y) = x$. Avem că g este parţial recursivă şi $A = dom(g)$.
2 $\Rightarrow$ 3). Avem $\chi_{pA} = 1 \dot{-} z \cdot g$ cu z funcţia zero. Rezultă că χ_{pA} este parţial recursivă fiind obţinută din g şi funcţii primitiv recursive prin compunere.

3) $\Rightarrow$ 4). Fie $f = \pi_{11} + (1 \dot{-} \chi_{pA})$, reamintind că π_{11} este funcţia identică pe $\mathbb{N}$. Avem concluzia.
4) $\Rightarrow$ 5). Avem existenţa funcţiilor primitiv recursive $u : \mathbb{N} \to \mathbb{N}$ şi $v : \mathbb{N}^2 \to \mathbb{N}$ aşa încât $f(x) = u(h(t))$ unde $h(t) =$ cel mai mic t pentru care $v(x,t) = 0$. Să presupunem acum A nevidă şi fie $a_0 \in A$ pentru care definim $F : \mathbb{N}^2 \to \mathbb{N}$ prin:
i) $F(x,n) = u(r(t))$ unde $r(t) =$ cel mai mic t pentru care $t \leq n(v(x,t) = 0)$ dacă există astfel de t,
ii) $F(x,n) = a_0$ în caz contrar.
Avem că F este primitiv recursivă şi $F(\mathbb{N}^2) = A$. Fie $J : \mathbb{N}^2 \to \mathbb{N}$ bijecţia primitiv recursivă de la Exerciţiul ??. Atunci $F \circ J^{-1} = F \circ (K, L) : \mathbb{N} \to \mathbb{N}$ este primitiv recursivă cu imaginea A.
5) $\Rightarrow$ 1). Evident. $\square$

Propoziţia 12.4(**Kleene**) *Mulţimea $A \subset \mathbb{N}$ este recursivă dacă şi numai dacă A şi $\mathbb{N} \setminus A$ sunt ambele r.e..*

Definiţia 12.5 1)Fie X numţime numărabilă şi $\varphi : X \to \mathbb{N}$ o bijecţie fixată. Atunci submulţimea A a lui X se numeşte *recursivă*(respectiv *r.e.*) relativ la φ dacă $\varphi(A)$ este recursivă (respectiv r.e.).
2) Numim *enumerare Gödel* pentru X o aplicaţie injectivă $\varphi : X \to \mathbb{N}$ pentru care $\varphi(X)$ este recursivă.

Fixăm în cele ce urmează mulţimea finită A de cardinal n şi de asemeni bijecţia $\{1, ..., n\} \to A, i \to a_i$. Avem următoarele enumerări Gödel ale lui A^*:
I) $\varphi_1(a_{i_1}...a_{i_k}) = \sum_{j=1}^{k} i_j (n+1)^{j-1}$,
II) $\varphi_1'(a_{i_1}...a_{i_k}) = \sum_{j=1}^{k} i_j n^j$,
III) $\varphi_2(a_{i_1}...a_{i_k}) = 2^k \prod_{j=1}^{k} p_j^{i_j}$ unde p_j este al j-lea număr prim impar.

Definiţia 12.6 Limbajul L peste A se numeşte *recursiv* (respectiv *r.e.*) dacă $\varphi(L)$ este recursiv (respectiv r.e.) unde φ este φ_1, φ_1' sau φ_2.

Fie gramatica $G = (V_N, V_T, S, P)$ şi $A = V_N \cup V_T$; presupunem $A = \{a_1, ..., a_n\}$. Să notăm producţiile $P = \{\alpha_1 \to \beta_1, ..., \alpha_l \to \beta_l\}$ şi fie $\lambda_i = |\alpha_i|$, $\mu_i = |\beta_i|$.

Propoziţia 12.7 *Pentru $i \in \{1, ..., l\}$ există o funcţie primitiv recursivă $f_i : \mathbb{N}^2 \to \mathbb{N}$ astfel ca, dacă $m = \varphi_2(x_1...x_k)$ şi $x_1...x_k = x_1...x_{r-1}\alpha_i x_{r+\lambda_i}...x_k$ atunci $f_i(r,m) = \varphi_2(x_1...x_{r-1}\beta_{r+\lambda_i}...x_k)$ respectiv $f_i(r,m) = m$ alftel $f_i(r,m) = m$.*

Teorema 12.8 *Un limbaj de tip 0 este r.e. şi reciproc.*

Teorema 12.9 *Un limbaj de tip* 1 *este recursiv.*

Observaţia 12.10 i) Reciproca teoremei precedente nu este adevărată: există limbaje recursive care nu sunt dependente de context.
ii) Fie $S \subset \mathbb{N}$ mulţime r.e. ce nu este recursivă şi $\varphi : A^* \to \mathbb{N}$ una din enumerările Gödel precedente. Există o funcţie recursivă strict crescătoare $f : \mathbb{N} \to \varphi(A^*)$; atunci $f(S)$ este r.e. şi nerecursivă. Mai mult, $\varphi(\varphi^{-1}(f(S))) = f(S)$ şi deci $\varphi^{-1}(f(S))$ este limbaj r.e. nerecursiv.

În concluzie avem schema următoare:

$$L_1 \subsetneq L_{\mathrm{r}} \subsetneq L_{\mathrm{r.e.}} = L_0.$$

cu L_{r} mulţimea limbajelor recursive şi $L_{\mathrm{r.e.}}$ mulţimea limbajelor r.e.

Propoziţia 12.11 *Dacă L este un limbaj r.e. atunci la fel este închierea sa Kleene L^*.*

SEMINARUL 12

S12.1 .

Rezolvare .

S12.2 .

Rezolvare .

S12.3 .

Rezolvare .

S12.4 .

Rezolvare .

S12.5 .

S12.6 .

Rezolvare .

S12.7 .

Rezolvare .

S12.8 .

Rezolvare .

Course 13

Entropia, energia şi corelaţia surselor de informaţie

Definiţia 13.1 Numim *sursă de informaţie* o pereche $S = (\Sigma, \pi)$ cu Σ alfabet şi π o *distribuţie de probabilitate* pe Σ adică o aplicaţie $\pi : \Sigma \to \mathbb{R}_+$ sastisfăcând $\sum_{s\in\Sigma} \pi(s) = 1$. Distribuţia π o numim *pozitivă* dacă $\pi(s) > 0$ pentru orice $s \in \Sigma$.

Observaţii 13.2 i) Avem că $\pi(s) \in [0, 1]$ pentru orice $s \in \Sigma$.
ii) O sursă de informaţie poate fi gândită ca un dispozitiv "black-box" care emite simboluri din Σ fiecare astfel de simbol s fiind emis cu probabilitatea $\pi(s)$.
iii) Fixăm $|\Sigma| = n$ şi notăm $S = (\Sigma = (s_i), \pi = (\pi_i)), 1 \leq i \leq n$ cu convenţia $p_1 \geq ... \geq p_n$. Vom nota tabelar:

S	s_1	...	s_n
π	p_1	...	p_n

Exemplul 13.3 $\pi_u(s) = \frac{1}{n}$ pentru orice $s \in \Sigma$ este o distribuţie pozitivă de probabilitate numită *distribuţia uniformă*.

Putem extinde π la monoidul cuvintelor obţinându-se un morfism de la Σ^* la monoidul multiplicativ al lui $[0, 1]$, $\pi : \Sigma^* \to ([0, 1], \cdot, 1)$ considerând:
i) $\pi(\varepsilon) = 0$,
ii) $\pi(w) = \pi(w(1))...\pi(w(k))$ pentru orice $w \in \Sigma^k, k \geq 2$.
Proprietatea de morfism o interpretăm astfel: probabilitatea emiterii unui simbol este independentă de simbolurile emise anterior; din acest motiv, sursele de informaţii astfel definite mai sunt numite *fără memorie*, cele cu memorie mai fiind numite *surse Markov*. Obţinem astfel o extindere a lui π

la limbaje peste Σ:
iii) $\pi(\emptyset) = 0$,
iv) $\pi(L) = \sum_{w \in L} \pi(w)$ dacă L este submulţime nevidă a lui Σ^*.
Numărul real nenegativ $\pi(L)$ îl numim *π-măsura lui L*. Astfel, π_u-măsura lui L o numim *indicatorul de cod* al lui L.

Definiţia 13.4 Limbajul produs L_1L_2 se numeşte *neambiguu* dacă pentru orice $w \in L_1L_2$ există cuvintele unice $u \in L_1$ şi $v \in L_2$ aşa încât $w = uv$.

Propoziţia 13.5 1) $\pi(\Sigma^k) = 1$ *pentru orice* $k \geq 1$.
2) $\pi(\cup_{i \in I} L_i) \leq \sum_{i \in I} \pi(L_i)$ *pentru orice familie* $L_i, i \in I$ *cel mult numărabilă de submulţimi ale lui* Σ^* *cu următorea convenţie: dacă există* $i \in I$ *aşa încât* $\pi(L_i) = \infty$ *atunci* $\pi(\cup_{i \in I}) = \infty$. *Dacă familia* L_i *are mulţimile disjuncte atunci avem egalitate.*
3) $\pi(L_1L_2) \leq \pi(L_1)\pi(L_2)$. *Dacă produsul* L_1L_2 *este neambiguu atunci avem egalitate.*
4) $\pi(L^*) \leq \sum_{i \geq 0} \pi(L^n) \leq \sum_{i \geq 0} (\pi(L))^n$ *cu convenţia:* $\pi(L) = \infty$ *implică* $\pi(L^*) = \infty$.

Conceptul de entropie ca măsură a informaţiei şi a gradului de incertitudine, a fost introdus în 1948 de către Claude Shannon. Această noţiune este profund analoagă conceputului similar din termodinamică unde a fost introdus de către Clausius în 1864 ca măsură a gradului de dezordine al unui sistem fizic.

Deoarece din punct de vedere matematic, informaţia furnizată de simbolul $s_k \in \Sigma$ este $I_k = -\log p_k$ rezultă că media ponderată a informaţiilor furnizate de sursa dată este:

Definiţia 13.6 Se numeşte *entropie* a sursei S numărul real nenegativ:

$$H(\pi) = -\sum_{i=1}^{n} p_i \log p_i$$

unde logaritmul este în baza 2 şi avem convenţia $0 \cdot \log 0 = 0$. Unitatea de măsură a entropiei este "biţi/simbol".

Alegerea bazei 2 se poate considera ca fiind neimportantă matematic datorită proprietăţii de schimbare a bazei logaritmilor şi este impusă din punct de vedere tehnic de utilizarea calculului binar în procesarea datelor de către calculatoare.

Lema 13.7 *Dacă* $b > 0$ *şi* $x > 0$ *atunci* $\log_b x \leq x - 1$ *cu egalitate doar pentru* $x = 1$.

Propoziţia 13.8 Inegalitatea Gibbs *Fie numerele reale* $(p_i, q_i), 1 \leq i \leq n$ *satisfăcând*:
i) $0 \leq p_i, q_i \leq 1$,
ii) $\sum_{i=1}^n q_i \leq 1 = \sum_{i=1}^n p_i$.
Atunci: $-\sum_{i=1}^n p_i \log p_i \leq -\sum_{i=1}^n p_i \log q_i$. *Avem egalitate dacă şi numai dacaă* $p_i = q_i$ *pentru totţi* $i \in \mathbb{N}_n$.

Corolar 13.9 *Pentru orice sursă de* n *informaţii avem*: $0 \leq H(\pi) \leq \log n = H(\pi_u)$.

Demonstraţie Avem $H(\pi_u) = -\sum_{i=1}^n \frac{1}{n} \log \frac{1}{n} = -\log \frac{1}{n} = \log n$. Deoarece $p_i \in [0,1]$ avem $-p_i \log p_i \geq 0$ şi rezultă membrul stâng. Pentru membrul drept aplicăm inegalitatea Gibbs cu $q_i = \frac{1}{n}, 1 \leq i \leq n$. □

Cazurile de egalitate pentru inegalitatea precedentă sunt precizate de:

Propoziţia 13.10 i) $H(\pi) = 0$ *dacă şi numai dacă* $p_1 = 1$.
ii) $H(\pi) = \log n$ *dacă şi numai dacă* $\pi = \pi_u$.

Demonstraţie i) $H(\pi) = 0$ dacă şi numai pentru orice $i \in \mathbb{N}_n$ avem $p_i \log p_i = 0$. Cum nu putem avea că toţi p_i sunt nuli deoarece suma lor este 1 rezultă că trebuie să existe măcar un indice i aşa încât $p_i = 1$. Din ordonarea probabilităţilor p_i rezultă $p_1 = 1$.
ii) rezultă din cazul de egalitate al Inegalităţii Gibbs. □

Observaţia 13.11 În termodinamică unui sistem fizic izolat, o stare de echilibru este caracterizată de entropie maximă. Prin analogie, am putea numi distribuţia uniformă ca fiind starea de echilibru "informaţional" al sursei date, toate cele n simboluri (stări) fiind la fel de probabile.

Definiţia 13.12 Se dau sursele de informaţie $S_1 = (\Sigma_1, \{p_i\}, i \in I), S_2 = (\Sigma_2, \{q_j\}, j \in J)$. Numim *produsul* lor sursa $S_1 S_2 = (\Sigma_1 \times \Sigma_2, \{p_i q_j\})$. (Avem imediat $\sum_{i,j} p_i q_j = 1$.)

Putem spune că sursa produs S^2 generează grupe de câte două mesaje ale sursei S. Analog pentru o putere $k \geq 2$ oarecare.

Propoziţia 13.13 $H(S_1 S_2) = H(S_1) + H(S_2)$. *În consecinţă* $H(S^k) = kH(S)$.

Demosntraţie $-\sum_{i,j} p_i q_j \log(p_i q_j) = -\sum_{i,j} p_i q_j \log p_i - \sum_{i,j} p_i q_j \log q_j = \sum_j q_j H(S_1) + \sum_i p_i H(S_2)$. □

Definiţia 13.14 Pentru sursa dată iniţial definim:
1) *redundanţa* $R = \log n - H(\pi)$,
2) *eficienţa* $\eta = \frac{H(\pi)}{\log n}$.

Exemplul 13.15 (n=2) Notând $p_1 = p$ avem:

S	s_1	s_2
π	p	$1-p$

i) $H(p, 1-p) = \eta = -p\log p - (1-p)\log(1-p)$,
ii) $R = 1 + p\log p + (1-p)\log(1-p)$.

Inspirat de expresia energiei cinetice care este suma pătratelor vitezelor, matematicianul român Octav Onicescu a introdus în 1964 conceptul următor:

Definiţia 13.16 Numim *energia* sursei date numărul real strict pozitiv:

$$E(\pi) = \sum_{i=1}^{n} p_i^2.$$

Propoziţia 13.17 i) $E(\pi_u) = \frac{1}{n} \leq E(\pi) \leq 1$.
ii) $E(\pi) = \frac{1}{n}$ *dacă şi numai dacă* $\pi = \pi_u$.
iii) $E(\pi) = 1$ *dacă şi numai dacă* $p_1 = 1$.
iv) $E(S_1S_2) = E(S_1)E(S_2)$.

Demonstraţie i) Faptul că $E(\pi_u) = \frac{1}{n}$ este imediat ca şi inegalitatea din dreapta deoarece p_i fiind subunitare avem $E(\pi) \leq \sum_{i=1}^{n} p_1$. Pentru a arăta inegalitatea din stânga fie $x_i = p_i - \frac{1}{n}$; rezultă $\sum_{i=1}^{n} x_i = 0$. Avem $E(\pi) = \frac{1}{n} + \sum_{i=1}^{n} x_i^2$.
ii) Avem egalitate în stânga dacă şi numai dacă toţi x_i sunt nuli.
iii) Avem egalitate în dreapta dacă şi numai dacă $p_i^2 = p_i$ ceea ce revine la $p_1 = 1$ şi $p_2 = ... = p_n = 0$.
iv) $\sum_{i,j}(p_iq_j)^2 = (\sum p_i^2)(\sum q_j^2)$ din independenţa celor două surse. □

Definiţia 13.18 Date sursele S_1 şi S_2 de aceeaşi dimensiune n numim:
i) *corelaţia* lor numărul real nenegativ $C(\pi_1, \pi_2) = \sum_{i=1}^{n} p_iq_i$.
ii) *coeficientul de corelaţie* numărul real nenegativ $CC(\pi_1, \pi_2) = \frac{C(\pi_1,\pi_2)}{\sqrt{E\pi_1)E(\pi_2)}}$.

Propoziţia 13.19 $CC(\pi_1, \pi_2) \leq 1 = CC(\pi, \pi)$ *cu egalitate dacă şi numai dacă* $\pi_1 = \pi_2$.

Demonstraţie Faptul că $CC(\pi, \pi) = 1$ este imediat iar inegalitatea este exact inegalitatea Cauchy-Buniakowski-Schartz (CBS) din teoria produselor scalar. Avem egalitate în inegalitatea CBS dacă şi numai dacă vectorii n-dimensionali π_1, π_2 sunt coliniari adică există numărul real λ aşa încât $\pi_2 = \lambda\pi_1$. Dar din $\sum p_i^1 = \sum p_i^2 = 1$ rezultă $\lambda = 1$. □

SEMINAR 13

S13.1 Se dă o sursă cu $n = 5$ şi $p_1 = \frac{1}{2}, p_2 = \frac{1}{4}, p_3 = \frac{1}{8}, p_4 = p_5 = \frac{1}{16}$. Se cere entropia, redundanţa, eficienţa şi energia acestei surse.

Rezolvare $H = \frac{1}{2} \cdot 1 + \frac{1}{4} \cdot 2 + \frac{1}{8} \cdot 3 + \frac{1}{16} \cdot 4 \cdot 2 = \frac{15}{8} = 1.875$ biţi/simbol.
$R = \log 5 - H = 2.3219 - 1.8750 = 0.4469, \eta = \frac{1.875}{2.3219} = 0.8075,$
$E = \frac{1}{4} + \frac{1}{16} + \frac{1}{64} + \frac{2}{256} = 0.25 + 0.0625 + 0.0156 + 0.0078 = 0.3359$

S13.2 .

Rezolvare .

S13.3 .

Rezolvare .

S13.4 .

Rezolvare .

S13.5 .

Rezolvare .

S13.6 .

Rezolvare .

S13.7 .

Rezolvare .

Course 14

Compilatoare 1: Analiză lexicală

Definiţia 14.1 i) Un *translator* este un program (cutie neagră) care primeşte la intrare un text scris într-un limbaj de programare, *limbajul sursă*, şi produce la ieşire un text echivalent scris în alt limbaj de programare, *limbajul obiect*.
ii) Dacă limbajul sursă este un limbaj de nivel înalt iar limbajul obiect este un limbaj de nivel inferior (limbaj de asamblare sau cod maşină), atunci translatorul respectiv se numeşte *compilator*.

Procesul de compilare a unui program are loc în mai multe faze. O fază este o operaţie unitară în cadrul căreia are loc transformarea programului sursă dintr-o reprezentare în alta.

Principalele faze ale unei compilări sunt:
1) *Analiza lexicală*: textul sursa este preluat sub forma unei secvenţe de caractere care sunt grupate apoi în entitati numite *atomi*; atomilor li se atribuie coduri lexicale, astfel că, la ieşirea acestei faze, programul sursă apare ca o secvenţa de asemenea coduri. Exemple de atomi: cuvinte cheie, identificatori, constante numerice, semne de punctuaţie etc.
2) *Analiza sintactică* are ca scop gruparea atomilor rezultaţi în urma analizei lexicale în structuri sintactice. O structură sintactică poate fi văzută ca un arbore ale cărui noduri terminale reprezintă atomi, în timp ce nodurile interioare reprezintă şiruri de atomi care formează o entitate logică . Exemple de structuri sintactice: expresii, instrucţiuni, declaraţii etc.
3) Pe durata analizei sintactice, de obicei are loc şi o *analiză semantică*, ceea ce înseamnă efectuarea unor verificări legate de:
-compatibilitatea tipurilor datelor cu operaţiile în care ele sunt implicate,

-respectarea regulilor de vizibilitate impuse de limbajul sursă.

4) *Generarea de cod intermediar*: în această fază are loc transformarea arborelui sintactic într-o secvenţa de instrucţiuni simple, similare macroinstrucţiunilor unui limbaj de asamblare. Diferenţa dintre codul intermediar şi un limbaj de asamblare este în principal aceea că în codul intermediar nu se specifică registrele utilizate în operaţii. Exemple de reprezentări pentru codul intermediar: instrucţiunile cu trei adrese, notaţia postfix, etc. Codul intermediar are avantajul de a fi mai uşor de optimizat decât codul maşină .

5) *Optimizarea de cod* este o fază opţională al cărei rol este modificarea unor porţiuni din codul intermediar generat astfel încât programul rezultat să satisfacă anumite criterii de performanţă vizând timpul de execuţie sau/şi spaţiul de memorie ocupat.

6) *Generarea codului final* presupune transformarea instrucţiunilor codului intermediar (eventual optimizat) în instrucţiuni maşină (sau de asamblare) pentru calculatorul ţinta (cel pe care se va executa programul compilat).

În afară de aceste acţiunile, procesul de compilare mai include următoarele:
7) *Gestionarea tabelei de simboluri*: tabela de simboluri este o structură de date destinată păstrării de informaţii despre simbolurile (numele) care apar în programul sursă; compilatorul face referire la această tabelă aproape în toate fazele compilării. 8) *Tratarea erorilor*: un compilator trebuie să fie capabil să recunoască anumite categorii de erori ce pot apare în programul sursă; tratarea unei erori presupune detectarea ei, emiterea unui mesaj corespunzător şi revenirea din eroare, adică, pe cât posibil, continuarea procesului de compilare până la epuizarea textului sursă, astfel încât numărul de compilări necesare eliminării tuturor erorilor dintr-un program să fie cât mai mic. Practic, există erori specifice fiecărei faze de compilare.

Derularea procesului de compilare

Fazele unui proces de compilare se pot înlănţui, în principiu, în două moduri:

-la ieşirea/finalul fiecărei faze se va genera un fişier intermediar conţinând forma de reprezentare a programului sursă rezultată în faza respectivă, fişier ce va constitui intrare pentru faza următoare. În acest caz, în fiecare fază va avea loc cel puţin o parcurgere a programului sursă, de la început la sfârşit. O asemenea parcurgere se numeşte *trecere*.

-două sau mai multe faze de compilare se întrepătrund astfel încăt ele să se execute printr-o singură trecere.

Aplicarea uneia sau alteia dintre aceste două modalităţi depinde de natura limbajului compilat precum şi de mediul în care urmează să ruleze compilatorul. Astfel, în sprijinul proiectanţilor de compilatoare au fost create

instrumente software precum generatoarele de analizoare lexicale şi sintactice, generatoarele de compilatoare sau sistemele de scriere a translatoarelor. Aceste instrumente sunt programe care produc compilatoare sau părţi din compilatoare, primind la intrare o specificare a limbajului sursă precum şi a calculatorului ţintă.

Un *analizor lexical* citeşte textul sursă caracter cu caracter şi-l transformă într-o secvenţă de unităţi primitive (elementare) numite *unităţi lexicale*, în engleză *tokens*.

O unitate lexicală descrie o secvenţă de caractere cu o anumită semnificaţie şi este tratată ca o entitate logică. Astfel, pentru fiecare limbaj de programare se stabilesc, atunci când se proiectează acel limbaj, unităţile lexicale corespunzătoare.

Majoritatea limbajelor utilizează următoarele unităţi lexicale:
1) CONSTANTE; exemplu: $737, -68.94, 3e+2$,
2) IDENTIFICATORI; exemplu: *alpha*, *un_ident*,
3) OPERATORI; exemplu: $+, *, /, <, >$,
4) CUVINTE REZERVATE; exemplu: *begin, while*,
5) SEMNE SPECIALE; de exemplu: ; . : .

Problema analizei lexicale comportă cel puţin două aspecte:
I) găsirea unei modalităţi de descriere a unităţilor lexicale; astfel, se constată că expresiile regulate sunt instrumentele ce pot descrie orice unitate lexicală.
II) recunoaşterea acestor unităţi lexicale ceea ce constituie analiza lexicală propriu zisă; dacă descrierea se face prin intermediul expresiilor regulate atunci mecanismul de recunoaştere este automatul finit determinist conform Teoremei 5.8.

Unităţile lexicale sunt de două categorii:
a) unităţi ce descriu un şir anume de caractere; exemplu *if, while*, ++ := ;
b) unităţi ce descriu o clasă de şiruri: identificatori, constante, etc.
În al doilea caz, vom considera o unitate lexicală ca fiind o pereche (*tipul*, *valoarea*).

Pentru unităţi lexicale ce descriu un şir anume, prin convenţie tipul este acel şir iar valoarea coincide cu tipul. Astfel, caracterul (este de tip paranteză stângă iar *alpha* este unitate lexicală de tip *identificator* care are valoarea *alpha*. Mai spunem că *alpha* este o instanţă a tokenului identificator sau *lexem*.
LEXEM ~e 1) Cuvânt sau parte de cuvânt care serveşte ca suport minimal al semnificaţiei; morfem lexical. 2) Unitate de bază a vocabularului care reprezintă asocierea unuia sau a mai multor sensuri; cuvânt; unitate lexicală. din fr. lexeme. Sursa : NODEX (341523)

limbaje cu compilator C, FORTRAN, Pascal, ALGOL, BASIC

Pagini Web utile:
1) http://en.wikipedia.org/wiki/Compiler
2) http://ro.wikipedia.org/wiki/GNU_Compiler_Collection

Bibliography

[1] Capra, F., *Conexiuni ascunse*, Ed. Tehnică, Bucureşti, 2004.

[2] Cazacu, C., *Teoria calculabilităţii efective*, Ed. Univ. "Al. I. Cuza", Iaşi, 1996.

[3] Chiswell, I., *A Couse in Formal Languages, Automata and Groups*, Universitext, Springer-Verlag, London, 2009.

[4] Grigoraş Gh., *Limbaje formale şi tehnici de compilare*, Ed. Univ. "Al. I. Cuza", Iaşi, 1985.

[5] Grigoraş Gh., *Construcţia compilatoarelor. Algoritmi fundamentali*, Ed. Univ. "Al. I. Cuza", Iaşi, 2005.

[6] Gontineac M., *Limbaje formale şi automate*, Note de curs, http://www.math.uaic.ro/∼mgonti/.

[7] Holcombe W. M. L., *Algebraic automata theory*, Cambridge studies in advanced mathematics 1, Cambridge Univ. Press, 1982.

[8] Howie J. M., *Automata and Languages*, Claredon Press, Oxford, 1991.

[9] Ivan Gh.; Ivan M., *Concepte algebrice fundamentale în studiul limbajelor formale. Teorie şi exerciţii*, Editura de Vest, Timişoara, 2006.

[10] Jucan, T.; Andrei, Ş., *Limbaje formale şi teria automatelor: Teorie şi practică*, Ed. Univ. "Al. I. Cuza", Iaşi, 2002.

[11] Onicescu, O.; Ştefănescu V., *Elemente de statistică informaţionalču aplicaţii*, Ed. Tehnică, Bucureşti, 1979.

[12] Orman G., *Limbaje formale*, Ed. Tehnică, Bucureşti, 1982.

[13] Păun, Gh., *Gramatici contextuale*, Ed. Academiei, Bucureşti, 1982.

[14] Simovici D., *Limbaje formale şi tehnici de compilare*, EDP, Bucureşti, 1978.

[15] Srivastava, S. M., *A course in mathematical logic*, Universitext, Springer-Verlag, 2008.

[16] Şerbănaţi L. D., *Limbaje de programare şi compilatoare*, Ed. Academiei, Bucureşti, 1987.

[17] Tomescu I., *Introducere în combinatorică*, Ed. Tehnică, Bucureşti, 1972. (Cap. 8, p. 93.)

[18] Ţiplea F. L., *Fundamentele algebrice ale informaticii*, Ed. Polirom, Iaşi, 2006.

Index