Criptografie avansată

Notițe de curs

CONF. DR. MIHAI PRUNESCU

17 octombrie 2018

Cuprins

CURS 1

RECAPITULARE ALGEBRĂ

Amintim noțiunile esențiale de algebră comutativă ce vor fi necesare pe parcursul acestui curs. Începem cu structurile algebrice și cîteva dintre proprietățile lor fundamentale.

1.1 Grupuri. Grupuri de permutări

Definiţie 1.1: Fie G o mulțime nevidă și $\cdot : G \times G \longrightarrow G$ o operație binară.

Perechea $(G, \cdot)$ se numește *grup* dacă au loc:

(G1) $\forall x, y \in G, x \cdot y \in G$ (operația este *internă*);

(G2) $\forall x, y, z \in G, x \cdot (y \cdot z) = (x \cdot y) \cdot z$ (operația este *asociativă*);

(G3) $\exists 1 \in G$ un element distins astfel încît $1 \cdot x = x \cdot 1 = x$, pentru orice $x \in G$ (operația admite *element neutru (unitate)*);

(G4) $\forall x \in G, \exists y \in G$ cu $x \cdot y = y \cdot x = 1$ (orice element din G este *inversabil* — în acest caz, x este inversul lui y și reciproc).

Dacă, în plus, $x \cdot y = y \cdot x, \forall x, y \in G$, grupul se numește *comutativ* sau *abelian*.

Ne interesează în mod particular *grupuri de permutări*. Le putem defini într-un sens foarte general, astfel:

Definiţie 1.2: Fie A o mulțime oarecare (nevidă). Definim:

$$S(A) = \{f : A \longrightarrow A \mid f \text{ bijecție } \}.$$

Se poate vedea imediat că $S(A)$ are, de fapt, o structură de grup, cu compunerea funcțiilor. Numim $S(A)$ *grupul simetric* al mulțimii A. În particular, dacă $A = \{1, 2, \dots, n\}$, obținem *grupul permutărilor de n elemente*, notat, de obicei, cu S_n.

Amintim, din noțiunile elementare de combinatorică, faptul că $|S_n| = n!$. De asemenea, următoarea teoremă este fundamentală.

Teoremă 1.1 (Lagrange)**:** *Fie $H \subseteq G$ un subgrup. Atunci $\#H \mid \#G$, unde $\#$ notează* ordinul grupului, *i.e. cardinalul mulțimii subiacente.*

În particular, H partiționează *G, adică:*

$$G = \bigcup_{b \in G} bH,$$

reuniunea fiind disjunctă ($b_1H \cap b_2H \neq \emptyset \iff b_1 = b_2$).

O altă teoremă importantă, legată de grupuri simetrice de data aceasta, este:

Teoremă 1.2 (Cayley)**:** *Pentru orice grup G, avem $G \leq S(G)$.*

Demonstrație. Fie $g \in G$ un element arbitrar. Atunci putem defini:

$$f_g : G \longrightarrow G, \quad f_g(x) = gx, \forall x \in G.$$

Se pot verifica imediat proprietățile:

- f bijectivă, pentru orice $g \in G$;
- $f_1 = \mathrm{Id}_G$, unde $1 \in G$ este elementul neutru, iar Id_G este aplicația identitate a lui G, $x \mapsto x$;
- $f_g \circ f_h = f_{gh}, \forall g, h \in G$;
- $f_g = f_h \iff g = h$.

Definind acum $\varphi : G \longrightarrow S(G), \varphi(g) = f_g$ obținem un morfism injectiv, adică $G \simeq \varphi(G) \leq S(G)$. □

Amintim acum construcția grupului factor. Fie H un subgrup al lui G. Pentru $a \in G$ fixat, folosim notația $aH = \{ah \mid h \in H\}$, mulțimi pe care le numim *clase de echivalență la stînga modulo H* (eng. *left cosets*). Motivul denumirii este că putem defini relația pe G:

$$a \sim b \iff a \in bH,$$

care este o echivalență (reflexivă, simetrică, tranzitivă), iar clasa de echivalență a elementului a este $\hat{a} = aH$.

Avem nevoie de:

Definiție 1.3: Fie $H \leq G$ un subgrup.

H se numește *subgrup normal* al lui G, notat $H \trianglelefteq G$ dacă $\forall g \in G, gHg^{-1} = G$. Echivalent, $gH = Hg$, adică clasele de echivalență la stînga generate de g coincid cu cele la dreapta.

Observaţie 1.1: Evident, dacă grupul G este comutativ, orice subgrup al său este normal.

Cu acestea, ajungem la:

Definiţie 1.4: Fie $H \trianglelefteq G$. Pe mulțimea factor G/H avem o structură de grup, numit *grupul factor* al lui G modulo H.

Un exemplu la îndemînă este următorul: fie $\mathbb{Z}$ grupul întregilor. Fie $n \in \mathbb{Z}$. Atunci $n\mathbb{Z}$ este subgrup normal, deoarece $\mathbb{Z}$ este comutativ și se poate observa că $\mathbb{Z}/n\mathbb{Z} \simeq \mathbb{Z}_n$, grupul claselor de resturi modulo n.

Pentru următorul exemplu fundamental, avem nevoie de:

Definiţie 1.5: Fie $f : G \longrightarrow H$ o funcție. Ea se numește *morfism* (unitar) (eng. *homomorphism*) dacă:

- $f(1_G) = 1_H$;
- $f(xy) = f(x)f(y), \forall x, y \in G$.

Dat un astfel de morfism, definim:

$$\mathrm{Ker} f = \{x \in G \mid f(x) = 1_H\} = f^{-1}(1_G),$$

numit *nucleul* (eng. *kernel*) morfismului.

De asemenea, amintim:

$$\mathrm{Im} f = \{y \in H \mid \exists x \in G \text{ a.î. } f(x) = y\} = f(G),$$

pe care o numim *imaginea* morfismului.

Cu acestea, avem:

Teoremă 1.3 (Teorema fundamentală de izomorfism (TFI))**:** *Fie $f : G \longrightarrow H$ un morfism de grupuri.*

Atunci $\mathrm{Ker} f \trianglelefteq G$ *și* $G/\mathrm{Ker} f \simeq \mathrm{Im} f \leq G$.

Demonstraţie. (Schiță:) Demonstrația se bazează pe asocierea:

$$\psi : G/\mathrm{Ker} f \longrightarrow \mathrm{Im} f, \quad \psi(\hat{g}) = f(g).$$

□

Un exemplu simplu de aplicare a teoremei este izomorfismul $\mathbb{Z}/n\mathbb{Z} \simeq \mathbb{Z}_n$ menționat mai sus. Definim:

$$f : \mathbb{Z} \longrightarrow \mathbb{Z}, \quad f(x) = x \bmod n.$$

Atunci se poate verifica imediat faptul că $\mathrm{Ker} f = n\mathbb{Z}$, iar $\mathrm{Im} f = \mathbb{Z}_n$. TFI ne oferă izomorfismul căutat.

Următoarea noțiune introduce un nou tip de grupuri.

Definiţie 1.6: Fie G un grup și $A \subseteq G$ o submulțime. Definim:

$$\langle A \rangle = \bigcap_{A \subseteq H \leq G} H,$$

numit *subgrupul generat de A*. Altfel spus, este cel mai mic subgrup care conține A ca submulțime.

În particular, pentru $A = \{a\}$, obținem $\langle a \rangle$, (sub)grup numit *ciclic*, generat de a și care conține toți multiplii elementului a.

Un exemplu tipic este $\mathbb{Z} = \langle 1 \rangle$, deoarece orice număr întreg se poate scrie pornind cu 1 și adunări succesive.

Acum putem caracteriza ordinul elementului cu ajutorul ordinului grupului:

Propoziţie 1.1: *Elementul a al grupului G are ordin finit dacă și numai dacă subgrupul ciclic generat de a, $\langle a \rangle$ este finit.*

O proprietate la fel de importantă ne arată că grupurile ciclice sînt, în esență, clasificate de grupurile de clase de resturi.

Teoremă 1.4: *În contextul și cu notațiile de mai sus, avem*

$$\langle a \rangle \simeq \mathbb{Z}/n\mathbb{Z},$$

unde $n = \operatorname{ord}(a)$*, adică cel mai mic n cu proprietatea că* $a^n = 1$.

Cu aceasta, se poate demonstra ușor că subgrupurile lui $\mathbb{Z}/n\mathbb{Z}$ au următoarea formă particulară.

Teoremă 1.5: *Fie H un subgrup al lui $\mathbb{Z}/n\mathbb{Z}$. Atunci există $m \in \mathbb{Z}$, cu proprietatea că $m \mid n$, astfel încît $H \simeq \mathbb{Z}/m\mathbb{Z}$.*

Amintim acum cîteva proprietăți specifice grupurilor de permutări. Lucrînd în S_n, cu n arbitrar, avem *morfismul semn*:

$$\varepsilon : S_n \longrightarrow (\{\pm 1\}, \cdot).$$

Pentru acesta, $\operatorname{Ker} \varepsilon = A_n$, care se numește *grupul altern* de ordin n și conține toate permutările pare.

Deoarece orice permutare este fie pară, fie impară, avem $\#(S_n/A_n) = 2$ și spunem că A_n este un *subgrup de indice 2* în S_n, notat uneori $[S_n : A_n] = 2$.

Descompunerea permutărilor se face conform rezultatelor cunoscute:

- Orice permutare se scrie ca un produs de transpoziții;
- Orice permutare se scrie ca un produs de cicluri disjuncte.

În general, avem:

$$(k\ k+1) = (1\ 2 \dots n)^{k-1}(1\ 2)(1\ 2 \dots n)^{1-k}$$
$$(i\ j) = (j-1\ j)(j-2\ j-1)\dots(i+1\ i+2)(i\ i+1)(i+1\ i+2)\cdots(j-2\ j-1)(j-1\ j),\ i<j$$
$$(a_1\ a_2 \dots a_k) = (a_1\ a_2)(a_2\ a_3)\cdots(a_{k-1}\ a_k)$$

Rezultă din aceste descompuneri că:

- Transpozițiile de forma $(k\ k+1)$ generează S_n $(k<n)$;
- Cele două elemente $(1\ 2)$ și $(1\ 2\ \dots\ n)$ generează S_n.

1.2 Inele și proprietăți aritmetice

Celelalte structuri algebrice care vor fi foarte importante în acest curs sînt *inelele*. Amintim definiția:

Definiție 1.7: Tripletul $(R, +, \cdot)$ se numește *inel* dacă:

- $(R, +)$ este un grup comutativ;
- $(R, \cdot)$ este un *monoid*, adică nu orice element este inversabil;
- Are loc proprietatea de *distributivitate* a celei de-a doua operații („înmulțirea“) față de prima („adunarea“), adică:

$$x\cdot(y+z) = x\cdot y + x\cdot z$$
$$(y+z)\cdot x = y\cdot x + z\cdot x,$$

  pentru orice $x, y, z \in R$.

Dacă monoidul $(R, \cdot)$ este comutativ, inelul se numește *comutativ*.

În plus, amintim faptul că un inel are două elemente neutre, notate, în general, cu 0 (elementul neutru al grupului aditiv) și 1 (pentru monoidul multiplicativ).

Principalul exemplu de inel este inelul numerelor întregi, $\mathbb{Z}$. În plus, pentru aplicațiile aritmetice, vom folosi și inele de clase de resturi, adică de forma $(\mathbb{Z}_n, +, \cdot)$.

O substructură a inelelor care va fi mai utilă în aplicații decît cea de subinel este structura de *ideal*, pe care o definim mai jos:

Definiție 1.8: Fie R un inel și $I \subseteq R$ o submulțime.

I se numește *ideal* dacă:

- $(I, +)$ este subgrup al $(R, +)$;

- $\forall r \in R, x \in I, rx \in I$.

Notația pentru ideal este $I \trianglelefteq R$, care nu este întîmplătoare. $(I, +)$ este chiar *subgrup normal* al lui R, deoarece $(R, +)$ este comutativ, deci putem considera grupul factor $(R/I, +)$. Acest grup devine chiar inel, cu operația:

$$\hat{x} \cdot \hat{y} = \widehat{xy}, \quad \forall x, y \in R.$$

Rezultă că obținem chiar *inelul factor* R/I.

Observație 1.2: Atenție: deoarece I este subgrup normal în grupul *aditiv* al inelului, rezultă că $\hat{x} = x + I$ și atunci operația pe inelul factor se scrie detaliat:

$$(x + I) \cdot (y + I) = xy + I.$$

Morfismele de inele sînt definite mai jos:

Definiție 1.9: Fie R, S două inele și o funcție $f : R \longrightarrow S$.
Funcția se numește *morfism* dacă „respectă ambele operații“, adică:

- f este *aditivă*, $f(x + y) = f(x) + f(y), \forall x, y \in R$;
- f este *multiplicativă*, $f(xy) = f(x)f(y), \forall x, y \in R$.

În plus, cerem ca morfismul să fie *unitar*, adică $f(1_R) = 1_S$ și $f(0_R) = 0_S$.

Ca în cazul grupurilor, dat un morfism ca mai sus, definim:

$$\mathrm{Ker}f = \{x \in R \mid f(x) = 0_S\}.$$

Avem imediat:

Teoremă 1.6 (Teorema fundamentală de izomorfism pentru inele)**:** *În contextul și cu notațiile de mai sus,* $\mathrm{Ker}f \trianglelefteq R$ *și* $R/\mathrm{Ker}f \simeq \mathrm{Im}f \leq R_2$ *(subinel).*

Exemplul esențial care ne va fi de folos este detaliat mai jos.
Pentru inelul $\mathbb{Z}$, orice ideal este de forma $n\mathbb{Z}$, pentru un anumit $n \in \mathbb{Z}$. Într-adevăr, faptul că $n\mathbb{Z}$ este ideal este clar, iar pentru reciprocă, fie I un ideal al lui $\mathbb{Z}$. Se poate arăta că, dacă n este cel mai mic element pozitiv din I, atunci $I = n\mathbb{Z}$.
Dacă I și J sînt două ideale arbitrare, definim *suma* lor prin:

$$I + J = \{a + b \mid a \in I, b \in J\}.$$

Pentru cazul idealelor inelului de întregi, un exercițiu simplu arată că:

- $n\mathbb{Z} + m\mathbb{Z} = \gcd(m, n)\mathbb{Z}$;
- $n\mathbb{Z} \cap m\mathbb{Z} = \mathrm{lcm}(m, n)\mathbb{Z}$.

Pentru orice inel R, definim :

$$R^{\times} = U(R) = \{x \in R \mid \exists y \in R, xy = 1\}$$

și-l numim *grupul unităților* lui R (deoarece formează un grup față de operația de înmulțire din inel).

În general, se poate arăta folosind algoritmul lui Euclid că:

Propoziție 1.2: $x \in \left(\mathbb{Z}/n\mathbb{Z}\right)^{\times} \iff \gcd(x, n) = 1$.

Demonstrația se bazează pe faptul că, din algoritmul lui Euclid, cel mai mare divizor comun al două numere se poate scrie ca o combinație liniară a celor două. Adică:

$$\gcd(a, b) = d \Longrightarrow \exists \alpha, \beta \in \mathbb{Z}, \alpha a + \beta b = d.$$

Pentru inelele de clase de resturi, următorul rezultat este fundamental:

Teoremă 1.7 (Lema chineză a resturilor)**:** *Fie descompunerea lui n în factori primi ca $n = p_1^{\alpha_1} \cdots p_k^{\alpha_k}$. Atunci are loc izomorfismul de inele:*

$$\mathbb{Z}/n\mathbb{Z} \simeq \mathbb{Z}/p_1^{\alpha_1}\mathbb{Z} \times \cdots \mathbb{Z}/p_k^{\alpha_k}\mathbb{Z}.$$

În particular, avem că, dacă m și n sînt relativ prime, atunci:

$$\mathbb{Z}_{mn} \simeq \mathbb{Z}_m \times \mathbb{Z}_n,$$

izomorfism de inele. Justificarea este simplă, folosind teorema de izomorfism. Definim:

$$\mathbb{Z} \xrightarrow{p} \mathbb{Z}_n \times \mathbb{Z}_m, \quad p(a) = (a \bmod n, a \bmod m).$$

Atunci $\mathrm{Ker} p = mn\mathbb{Z}$ și avem rezultatul.

O funcție aritmetică importantă este *indicatorul lui Euler*.

Definiție 1.10: Se definește funcția $\varphi : \mathbb{N} \longrightarrow \mathbb{N}$, numită *indicatorul lui Euler* (eng. *Euler's totient function*) prin:

- $\varphi(0) = 0, \varphi(1) = 1$;
- $\varphi(n) = \#\{k \mid k < n, \gcd(k, n) = 1\}$,

adică numărul de numere mai mici decît n, coprime cu n.

Din propoziția 1.2, observăm că $\varphi(n) = \#\mathbb{Z}_n^{\times}$.

De asemenea, folosind cazul particular al lemei chineze, avem:

Lemă 1.1: *Dacă m și n sînt coprime, atunci $\varphi(mn) = \varphi(m)\varphi(n)$.*

Cazul general este următorul:

Teoremă 1.8: *Dacă avem descompunerea în factori primi* $n = p_1^{\alpha_1} \cdots p_k^{\alpha_k}$, *atunci:*

$$\varphi(n) = n\left(1 - \frac{1}{p_1}\right) \cdots \left(1 - \frac{1}{p_k}\right).$$

Demonstrație. Chiar din definiția funcției lui Euler, avem:

$$\begin{aligned}
\varphi(n) &= \varphi(p_1^{\alpha_1}) \cdots \varphi(p_k^{\alpha_k}) \\
&= \left(p_1^{\alpha_1} - p_1^{\alpha_1 - 1}\right) \cdots \left(p_k^{\alpha_k} - p_k^{\alpha_k - 1}\right) \\
&= p_1^{\alpha_1} \cdots p_k^{\alpha_k} \cdot \left(1 - \frac{1}{p_1}\right) \cdots \left(1 - \frac{1}{p_k}\right) \\
&= n \cdot \left(1 - \frac{1}{p_1}\right) \cdots \left(1 - \frac{1}{p_k}\right).
\end{aligned}$$

□

Această funcție apare și în următorul rezultat:

Teoremă 1.9 (Euler)**:** *Fie* $a, n > 0$ *astfel încît* $\gcd(a, n) = 1$.
Atunci $a^{\varphi(n)} \equiv 1$ *mod* n.

Demonstrație. Putem da o demonstrație elegantă folosind inele. Cum $\#\mathbb{Z}_n^\times = \varphi(n)$, fie $a \in \mathbb{Z}_n^*$. Atunci, conform teoremei lui Lagrange pentru grupuri (1.1), aplicată subgrupului generat de a, avem că $\operatorname{ord}(a) \mid \#\mathbb{Z}_n^\times$. Astfel, dacă $\varphi(n) = f$, iar $\operatorname{ord}(a) = o \mid f$, rezultă că $f = ox$, pentru un $x \in \mathbb{Z}$. Atunci:

$$a^f = a^{ox} = (a^o)^x = 1^x = 1,$$

din definiția ordinului. Cu alte cuvinte, $a^f = 1$ în $\mathbb{Z}_n^\times$, adică $a^{\varphi(n)} \equiv 1 \bmod n$. □

Un caz particular este:

Teoremă 1.10 (Fermat)**:** *Dacă* p *este un număr prim și* a *e astfel încît* $p \nmid a$, *atunci:*

$$a^{p-1} \equiv 1 \ mod\ p.$$

Inelul $\mathbb{Z}$ mai are o proprietate care îl face potrivit pentru aritmetică. El este *inel euclidian*, adică putem aplica teorema împărțirii cu rest și algoritmul lui Euclid. În simboluri,

$$\forall a, b \in \mathbb{Z}, \exists! r, q \in \mathbb{Z}, q > 0, 0 \leq r < |b| a = bq + r.$$

Amintim *algoritmul lui Euclid* pentru cel mai mare divizor comun, pe un exemplu.

Fie $a = 100$, $b = 17$. Aplicăm succesiv teorema împărțirii cu rest:

$$\begin{aligned} a = 100 &= 5 \cdot 17 + 15 \\ 17 &= 1 \cdot 15 + 2 \\ 15 &= 7 \cdot 2 + 1. \end{aligned}$$

Recuperăm acum expresia:

$$1 = 15 - 7 \cdot 2 = 15 - 7(17 - 15) = 8 \cdot 15 - 7 \cdot 17 = 8 \cdot (100 - 5 \cdot 17) - 7 \cdot 17 = 8 \cdot 100 - 47 \cdot 17.$$

Deci $1 = \gcd(100, 17) = 8 \cdot 100 - 47 \cdot 17$.

În particular, rezultă

$$1 \equiv -47 \cdot 17 \bmod 100 \equiv 53 \cdot 17 \bmod 100,$$

adică $\widehat{17}^{-1} = \widehat{53}$ în $\mathbb{Z}/100\mathbb{Z}$.

De asemenea, mai putem calcula:

$$\#\mathbb{Z}_{100}^{\times} = \varphi(100) = 100 \cdot \left(1 - \frac{1}{2}\right) \cdot \left(1 - \frac{1}{5}\right) = 40.$$

În plus, $\mathbb{Z}_{100}^{\times} = \mathbb{Z}_4^{\times} \times \mathbb{Z}_{25}^{\times}$, fiecare dintre inele conținînd doar elementele coprime cu, respectiv, 100, 4 și 25.

1.3 Corpuri. Corpuri finite

Pentru definiția corpurilor, putem, cel mai ușor, să pornim de la definiția inelelor. Astfel, inelul $(K, +, \cdot)$ se numește *corp* dacă, în plus, $(K - \{0\}, \cdot)$ este grup. Dacă, mai mult, grupul multiplicativ este comutativ, corpul se numește *comutativ*.

Exemplele tipice și utile pentru aritmetică sînt $\mathbb{Q}$ și $\mathbb{Z}_p$, cu p număr prim.

Vom folosi o notație specială pentru corpuri finite: $\mathbb{F}_n$ va nota corpul finit cu n elemente. De asemenea, amintim:

Definiție 1.11: Fie K un corp. Se numește *caracteristica lui K* ordinul elementului 1 față de adunare. Adică, cel mai mic t astfel încît $1 + 1 + \cdots + 1 = 0$, adunarea avînd exact t termeni.

Cîteva proprietăți fundamentale pentru corpuri urmează.

Teoremă 1.11 (Wedderburn)**:** *Orice corp finit este comutativ.*

De asemenea, avem și:

Teoremă 1.12: *Orice corp finit are caracteristică p, un număr prim.*

Mai mult, corpurile finite au p^k elemente, cu p un număr prim și au o structură de spațiu vectorial peste $\mathbb{Z}_p$.

În plus:

Teoremă 1.13: *Două corpuri finite cu același număr de elemente sînt izomorfe.*
De asemenea, avem și că $\mathbb{F}_{p^2} \subseteq \mathbb{F}_{p^r}$ dacă și numai dacă $s \mid r$.

De exemplu, să studiem corpul $\mathbb{F}_4$. Pornim de la polinomul $X^2 + X + 1$, care este ireductibil peste $\mathbb{F}_2 = \mathbb{Z}_2$ (putem testa elementele). Atunci, fie $\omega \in \mathbb{C}$ cu $\omega^2 + \omega + 1 = 0$, adică rădăcinile strict complexe de ordinul 3 ale unității (obs. $\omega^3 = 1$). Definim:

$$\mathbb{F}_4 = \mathbb{F}_2[\omega] = \{0, 1, \omega, \omega + 1\}$$

și observăm că am obținut o structură de corp, unde $\mathbb{F}_4^\times \simeq \mathbb{Z}_3$, care este ciclic, $\mathbb{F}_4$ fiind generat de ω, adică:

$$\mathbb{F}_4^\times = \{1 = \omega^3, \omega, \omega^2\}.$$

Avem următorul rezultat:

Teoremă 1.14: *Fie K un corp comutativ și G un grup finit, care este subgrup al grupului multiplicativ $(K - \{0\}, \cdot)$.*
Atunci G este grup ciclic.

Demonstrație. Presupunem $\#G = h$. Vrem să arătăm că G conține un element de ordin h. Descompunem în factori primi $h = p_1^{r_1} \cdots p_k^{r_k}$.

Facem observația că, pentru orice $i = 1, \dots, k$, există elemente x_i din G cu $x_i^{\frac{h}{p_i}} \neq 1$, altfel, polinomul $X^{\frac{h}{p_i}} - 1$ ar avea un număr de rădăcini mai mare decît gradul.

Demonstrăm acum că, dacă $y_i = x_i^{\frac{h}{p_i^{r_i}}}$, atunci y_i este un element de ordin $p_i^{r_i}$.
Evident, $y_i^{p_i^{r_i}} = x_i^h = 1$. Dacă y_i ar avea ordinul p_i^s, cu $s < r_i$, atunci:

$$y_i^{p_i^{r_i - 1}} = x^{\frac{h}{p_i}} = 1,$$

ceea ce contrazice faptul că ordinul y_i ar fi $p_i^{r_i}$.
Fie acum $y = y_1 \cdots y_k$; ordinul fiecărui factor fiind coprim cu celelalte, rezultă că $\operatorname{ord} y = h$ și atunci $G = \langle y \rangle$, adică este grup ciclic. □

De exemplu: $\mathbb{Z}_7^\times = \{1, 2, 3, 4, 5, 6\} = \langle 3 \rangle$.
Un alt caz particular de corp finit pe care îl putem folosi și pentru logică este $\mathbb{Z}_2 = \mathbb{F}_2 = \{0, 1\}$. Din tablele pentru adunare și înmulțire, observăm o similitudine cu tabelele de adevăr pentru conjuncții și disjuncții:

$+/\vee$	0	1
0	0	1
1	1	0

$\cdot/\wedge$	0	1
0	0	0
1	0	1

Astfel, putem identifica $x \cdot y$ cu $x \wedge y$ sau $\min(x, y)$, iar $x + y = x \vee y = \max x, y$.

Mai mult, avem următoarea teoremă care caracterizează *funcțiile booleene*:

Teoremă 1.15: *Orice funcție* $f : \{0,1\}^k \longrightarrow \{0,1\}$ *se poate exprima folosind* $\wedge, \vee, \neg$.

Demonstrație. Definim direct:

$$f = \bigvee_{\substack{(\varepsilon_1,\dots,\varepsilon_k) \\ f(\varepsilon_1,\dots,\varepsilon_k)=1}} x_1^{\varepsilon_1} \wedge \dots \wedge x_n^{\varepsilon_k},$$

unde $x_i^0 = \neg x_i$ și $x_i^1 = x_i$. □

De exemplu, luăm expresia $x + y$, gîndită funcția sumă

$$f = + : \{0,1\}^2 \longrightarrow \{0,1\}.$$

Deoarece există două posibilități ca f să dea 1, adică $f(0,1) = f(1,0) = 1$, avem descompunerea:

$$f(x, y) = x + y = x^0 y^1 \vee x^1 y^0.$$

Folosind teorema, obținem mai departe $(x \wedge \neg y) \vee (\neg x \wedge y)$.

De fapt, folosind o observație logică simplă, avem:

Observaţie 1.3: Cum $x \wedge y = \neg(\neg x \vee \neg y)$, rezultă că $\neg$ și $\vee$ vor fi suficiente pentru expresii logice, conjuncția rezultînd din ele.

Putem introduce un nou operator, numit *Sheffer's stroke*[1]sau, în teoria porților logice, NAND, notat $x \mid y$, cu tabelul:

$\mid$	0	1
0	1	1
1	1	0

Acest operator poate fi definit ca *negația conjuncției* (exprimat în limbaj natural prin „nu amîndouă“) și avem:

$$\neg x = x \mid x$$
$$x \vee y = (x \mid x) \mid (y \mid y).$$

Rezultă, folosind și observația anterioară, că Sheffer's stroke generează termeni care reprezintă toate funcțiile booleene.

Mai general, pentru corpuri finite cu n elemente, avem:

[1] wiki

Teoremă 1.16: *Fie $\mathbb{F}$ un corp finit cu n elemente.*
Atunci orice funcție $f : \mathbb{F}^k \longrightarrow \mathbb{F}$ este dată de un polinom.

Demonstrație. Fără a intra în detalii, demonstrația este imediată, folosind *polinomul de interpolare Lagrange*.[2]

Alternativ, putem da o demonstrație folosind observația:

$$\forall x \in \mathbb{F}, x^{n-1} = \begin{cases} 1, & x \neq 0 \\ 0, & x = 0 \end{cases}$$

în cazul în care $\#\mathbb{F} = n$. (Aceasta implică faptul că $\#\mathbb{F}^\times = n - 1$ și putem folosi un argument de tip Teorema lui Lagrange, 1.1).

Rezultă că putem defini polinomul:

$$P(X_1, \dots, X_k) = \sum_{(\varepsilon_1,\dots,\varepsilon_k)\in\mathbb{F}^k} (1 - (X_1 - \varepsilon_1)^{n-1}) \cdots (1 - (X_k - \varepsilon_k)^{n-1}) \cdot f(\varepsilon_1, \dots, \varepsilon_k),$$

pentru funcția $f : \mathbb{F}^k \longrightarrow \mathbb{F}$ ca în enunț. □

1.4 SEMINAR

Definim $\mathbb{F}_q = \mathbb{F}_p[X]/f(X)$ pentru f un polinom ireductibil, astfel încît $p^{\deg f} = q$. Atunci putem privi $\mathbb{F}_q$ ca mulțimea polinoamelor de grad mai mic decît gradul lui f într-o rădăcină a lui f. Adică, dacă α este o rădăcină a lui f, definim:

$$\mathbb{F}_q = \left\{ \sum_{i=0}^{\deg f - 1} a_i \alpha^i \mid a_i \in \mathbb{F}_p \right\},$$

adunarea și înmulțirea fiind făcute modulo p, împreună cu relația $f(\alpha) = 0$.

Atunci putem descrie corpul $\mathbb{F}_8$ ca:

$$\mathbb{F}_8 = \mathbb{F}_2[X]/(X^3 + X + 1) = \mathbb{F}_2[\omega],$$

unde ω este o rădăcină a polinomului $X^3 + X + 1$, adică $\omega^3 = \omega + 1$ (lucrînd peste $\mathbb{F}_2$ unde $-1 = 1$). Așadar:

$$\mathbb{F}_2[\omega] = \{a_0 + a_1\omega + a_2\omega^2 \mid a_0, a_1, a_2 \in \mathbb{F}_2\}.$$

Conform Teoremei 1.14, $\mathbb{F}_2[\omega] - \{0\}$ este grup ciclic. Căutăm un generator și observăm că $\mathbb{F}_2[\omega] \simeq \langle\omega\rangle$, grup ciclic de ordin 7 (verificați).

Avem nevoie de următoarea:

[2] Polinomul de interpolare Lagrange poate fi folosit pentru a aproxima cu un polinom o funcție dată. Mai precis, pentru a genera cea mai potrivită curbă de tip polinomial care mediază între valori date. Detalii, împreună cu legătura cu corpurile finite și criptografie, se pot găsi aici sau în discuția de aici.

Teoremă 1.17: *Pentru orice $\alpha > 1$, există un unic corp finit cu p^α elemente.*

Atenție, însă! $\mathbb{F}_{p^\alpha} \neq \mathbb{Z}_{p^\alpha}$, deoarece $\mathbb{Z}_{p^\alpha}$ are divizori ai lui zero (e.g. $p \cdot p^{\alpha-1} = 0$). Singurul caz particular cînd izomorfismul are loc este cu $\alpha = 1$, unde $\mathbb{F}_p \simeq \mathbb{Z}_p$.

De fapt, cazurile cînd $\mathbb{Z}_n^\times$ produc grupuri ciclice sînt date în:

Teoremă 1.18: *Mulțimea* $\{n \in \mathbb{N} \mid n > 1, \mathbb{Z}_n^\times \text{ ciclic }\}$ *coincide cu mulțimea* $\{2, 4, p^\alpha, 2p^\alpha \mid p \text{ prim impar }, \alpha \geq 1\}$.

De exemplu, știm că $\#\mathbb{Z}_9^\times = \varphi(9) = 6$ și trebuie să fie grup ciclic. Deci căutăm un generator și vom avea izomorfismul:

$$(\mathbb{Z}_9^\times, \cdot, 1) \simeq (\mathbb{Z}_6, +, 0).$$

Avem $\mathbb{Z}_9^\times = \langle 2 \rangle$, iar $\mathbb{Z}_6 = \langle 1 \rangle = \langle 5 \rangle$ (generatorii sînt coprimi cu 6). Pentru a stabili izomorfismul, este suficient să găsim o corespondență între generatori. Definim:

$$\gamma : \mathbb{Z}_9^\times \longrightarrow \mathbb{Z}_6, \quad \gamma(2) = 5,$$

care se poate extinde folosind operațiile grupurilor.

Deoarece avem proprietatea de morfism, $\gamma(a \cdot b) = \gamma(a) + \gamma(b)$, pentru orice $a, b \in \mathbb{Z}_9^\times$, rezultă că γ se comportă ca un *logaritm discret*.

★★★

Fie $\mathcal{A}_{26} \simeq \mathbb{Z}_{26}$ alfabetul cu 26 de litere. Introducem o codificare pe digrame:

$$y_1 = 6x_1 + x_2 \bmod 26$$
$$y_2 = 5x_1 + x_2 \bmod 26.$$

Astfel, din cuvîntul $y_1 y_2$ trecem în cuvîntul $x_1 x_2$. Care este regula de decriptare?

Scriind sistemul în formă matriceală, obținem *regula de criptare*:

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} 6 & 1 \\ 5 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

Decriptarea se face prin înmulțire cu A^{-1} la stînga, unde:

$$A^{-1} = \begin{pmatrix} 1 & -1 \\ -5 & 6 \end{pmatrix} = \begin{pmatrix} 1 & 25 \\ 21 & 6 \end{pmatrix} \bmod 26.$$

Acest exemplu conduce la întrebarea generală: Fie $A \in M_n(\mathbb{F}_2)$. Care este probabilitatea de a alege A cu determinant nenul?

Cum elementele lui A sînt din $\mathbb{F}_2$, rezultă că determinant nenul înseamnă determinant 1. Atunci putem calcula direct:

- Cazurile posibile (totalitatea matricelor din $M_n(\mathbb{F}_2)$) înseamnă 2^{n^2} elemente;
- Pentru cazurile favorabile, alegem liniile una cîte una:
  - prima linie poate fi aleasă în 2^n moduri, din care scădem 1 (linia nulă);
  - a doua linie nu poate fi prima sau un multiplu al ei (cu cît 0 sau 1), deci poate fi aleasă în $2^n - 2$ moduri;
  - a treia linie nu poate fi de forma $\alpha L_1 \pm \beta L_2$, unde L_1 este prima linie, L_2 este a doua, iar $\alpha, \beta \in \mathbb{F}_2$. Deci poate fi aleasă în $2^n - 4$ moduri.

În final, se obține:

$$\begin{aligned} P(\det A = 1) &= \frac{(2^n - 1)(2^n - 2)(2^n - 4) \cdots (2^n - 2^{n-1})}{2^{n^2}} \\ &= \left(1 - \frac{1}{2^n}\right)\left(1 - \frac{1}{2^{n-1}}\right) \cdots \left(1 - \frac{1}{2}\right). \end{aligned}$$

Se poate arăta, folosind analiză, că $P(\det A = 1) > 0,4$, pentru orice n.

★★★

Fie C regula de criptare, D regula de decriptare, peste alfabetul $\mathcal{A}_{26}$. Presupunem că pentru orice mesaj $m \in \mathcal{A}_{26}^n$ și orice $n \geq 1$, există o unică permutare $\sigma \in S_n$, astfel încît $C(m) = \sigma(m)$ (mesajul este criptat cu permutarea σ).

Pentru decriptare, dacă c este un mesaj de lungime n, $D(c) = \sigma^{-1}(c) = m$, mesajul inițial.

Oscar are un exemplar din mașina C, dar nu are D și interceptează un mesaj de lungime n. Cum îl poate decripta?

Presupunem n fixat. Atunci, putem obține valorile permutării folosind mesaje prestabilite, astfel:

- Din $C(ABC \dots YZZZ \dots Z)$, obținem $\sigma(1), \dots, \sigma(25)$, observînd cum sînt permutate primele 25 de litere;
- Din $C(ZZ \dots ZABC \dots YZZ \dots)$, unde la început punem 25 de litere Z, putem obține $\sigma(26)$ pînă la $\sigma(50)$.
- Continuînd, putem obține σ în $\left[\frac{n}{25}\right]$ pași, apoi putem calcula σ^{-1} și decriptăm.

Alternativ, putem recripta ceea ce rezultă, deoarece $C(C(m))$ va folosi σ^2 și, într-un număr de maxim $n!$ pași, putem obține mesajul inițial, de unde deducem ordinul lui σ.

★★★

Shamir's No Key Protocol: Presupunem că Alice și Bob au fiecare cîte un lacăt și o cheie unică, pe care o au doar ei, pentru propriul lacăt. Folosind o cutie căreia i se poate atașa un lacăt sau mai multe, cum poate trimite Alice un mesaj lui Bob, care să nu poată fi deschis de nimeni altcineva?

Soluție:

(1) Alice pune mesajul în cutie și pune lacătul ei, apoi trimite cutia lui Bob;

(2) Bob adaugă propriul lacăt cutiei și trimite înapoi lui Alice;

(3) Alice scoate lacătul ei și trimite lui Bob cutia (pe care a mai rămas doar lacătul lui);

(4) Bob deschide lacătul și citește mesajul.

Această problemă poate fi modelată astfel.

Folosind **One Time Pads**, cu chei de lungimea mesajului. Astfel, dacă notăm cu m mesajul, cu K_A cheia lui Alice și cu K_B cheia lui Bob, iar $\oplus$ este suma binară, schimbul de mesaje poate fi modelat prin:

$$\begin{aligned}\text{Alice} &\xrightarrow{m\oplus K_A \Rightarrow M_1} \text{Bob}\\ &\xleftarrow{m\oplus K_A\oplus K_B \Rightarrow M_2}\\ &\xrightarrow{m\oplus K_B \Rightarrow M_3} \oplus K_B \implies m,\end{aligned}$$

deoarece fiecare cheie acționează ca un element de ordin 2.

Dar atunci, Oscar poate calcula, dacă interceptează toate mesajele:

$$K_B = M_1 \oplus M_2 \implies m = M_3 \oplus K_B,$$

deci procedura este total nesigură.

Alternativ, fie p un număr prim foarte mare și $m \in \{0,1\}^*$ un mesaj binar, deci putem privi $m \in \mathbb{F}_p^\times$. Luăm $0 < K_A < p-1$ și $0 < K_B < p-1$, cu $K_A, K_B \in \mathbb{Z}_{p-1}^\times$. Schimbul de mesaje devine:

$$\begin{aligned}\text{Alice} &\xrightarrow{m^{K_A}} \text{Bob}\\ &\xleftarrow{(m^{K_A})^{K_B}}\\ &\xrightarrow{(m^{K_A K_B})^{K_A^{-1}}}\end{aligned}$$

La final, Bob aplică $(m^{K_A K_B K_A^{-1}})^{K_B^{-1}} = m$.

★★★

Conceptul de *unicity distance* (notată n_0) înseamnă cea mai mică lungime de text cifrat pentru care numărul de chei false este aproximativ 0.

Pentru One Time Pads, avem că $n_0 = \infty$, deoarece din orice două mesaje putem obține o cheie aplicînd XOR, așadar avem o regulă de criptare ce se poate obține din orice mesaj.

Faptul că $n_0 = \infty$ este un alt mod de a exprima că One Time Pads oferă securitate totală.

★★★

Următoarea problemă se va rezolva folosind lema chineză.

Presupunem că vrem să aflăm vîrsta unei persoane și știm că:

- Acum un an, vîrsta era multiplu de 3;
- Peste 2 ani, vîrsta va fi multiplu de 5;
- Peste 4 ani, vîrsta va fi multiplu de 7.

Fie x vîrsta actuală. Avem sistemul de congruențe:

$$\begin{aligned} x &\equiv 1 \bmod 3 \\ x &\equiv 3 \bmod 5 \\ x &\equiv 3 \bmod 7. \end{aligned}$$

Folosind *varianta efectivă* a lemei chineze (v. §2) calculăm:

$$\begin{aligned} 35^{-1} &= (5 \cdot 7)^{-1} \bmod 3 = 2 \\ 21^{-1} &= (3 \cdot 7)^{-1} \bmod 5 - 1 \\ 15^{-1} &= (3 \cdot 5)^{-1} \bmod 7 = 1 \end{aligned}$$

Rezultă că:

$$x = (1 \cdot 35 \cdot 2 + 3 \cdot 21 \cdot 1 + 3 \cdot 15 \cdot 1) \bmod 3 \cdot 5 \cdot 7,$$

adică $x = 73$.

★★★

Teoremă 1.19: *Fie S o mulțime finită și $f : S \longrightarrow S$ o funcție.*

Atunci există o submulțime $S_0 \subseteq S$ astfel încît $f(S_0) = S_0$ (adică f este o permutare a lui S_0), S_0 este maximală cu această proprietate, $S_0 \neq \emptyset$, iar $f|_{S-S_0}$ este o reuniune finită de arbori.

Demonstrație. Fie $x_0 \in S$. Formăm șirul $x_0, x_1 = f(x_0), x_2 = f(f(x_0)) \dots$. Acest șir va fi periodic, deoarece S este finită, adică există $n, m > n$ astfel încît $x_m = x_n$.

Definim S_0 ca fiind reuniunea ciclurilor obținute. □

De exemplu, considerăm funcția definită pe $S = \{1, 2, \dots, 10\}$:

$$\begin{aligned}
1 &\longmapsto 7 \\
2 &\longmapsto 5 \\
3 &\longmapsto 10 \\
4 &\longmapsto 1 \\
5 &\longmapsto 8 \\
6 &\longmapsto 3 \\
7 &\longmapsto 7 \\
8 &\longmapsto 2 \\
9 &\longmapsto 5 \\
10 &\longmapsto 6.
\end{aligned}$$

Diagramatic, obținem:

$$\begin{tikzcd}
4 \arrow[r] & 1 \arrow[r] & 7 \arrow[loop right] & & 9 \arrow[d] & & 3 \arrow[r] & 10 \arrow[r] & 6 \arrow[ll, bend left] \\
 & & & 2 \arrow[r] & 5 \arrow[r] & 8 \arrow[ll, bend left] & & &
\end{tikzcd}$$

Atunci, avem:

$$S_0 = \{7, 2, 5, 8, 3, 10, 6\},$$

iar $S - S_0 = \{1, 4, 9\}$, formată din 3 arbori.

CURS 2

APLICAȚII ALE ARITMETICII MODULARE

Cel mai rapid algoritm al lui Euclid lucrează modulo 2.

Algoritm 1 Euclid modulo 2

```
Procedură EUCLID MOD 2(a,b)
    g ← 1
    while (a%2 = 0&&b%2 = 0) do
        a ← a/2
        b ← b/2
        g ← 2g
    while (a ≠ 0) do
        while (a%2 = 0) do
            a ← a/2
        while (b%2 = 0) do
            b ← b/2
        // acum ambii sînt impari
        if (a ≥ b) then a ← (a − b)/2
        else b ← (b − a)/2
    return g · b
```

O variantă optimizată putem formula și pentru lema chineză a resturilor și obținem *varianta efectivă*, adică: presupunem că avem $m_1, \dots, m_r$ astfel încît $\forall i \neq j, \gcd(m_i, m_j) = 1$. Vrem să găsim x astfel încît pentru orice i, $x = a_i \bmod m_i$.

Fie $M = m_1 \cdots m_r$ și $M_i = \dfrac{M}{m_i}$. Definim $y_i = M_i^{-1} \bmod m_i$. Atunci soluția este:

$$x = \sum_{i=1}^{r} a_i M_i y_i \bmod M.$$

De exemplu: căutăm x astfel încît:

$$\begin{aligned} x &= 5 \bmod 7 \\ x &= 3 \bmod 11 \\ x &= 10 \bmod 13. \end{aligned}$$

Fie $M = 7 \cdot 11 \cdot 13 = 1001$. Atunci:

$$\begin{aligned} M_1 &= 11 \cdot 13 = 143, \quad y_1 = 143^{-1} \bmod 7 = 3^{-1} \bmod 7 = 5 \\ M_2 &= 7 \cdot 13 = 91, \quad y_2 = 91^{-1} \bmod 11 = 3^{-1} \bmod 11 = 4 \\ M_3 &= 7 \cdot 11 = 77, \quad y_3 = 77^{-1} \bmod 13 = 12^{-1} \bmod 13 = 12. \end{aligned}$$

Rezultă:

$$x = \sum a_i M_i y_i \bmod M = 894 \bmod 1001.$$

Obținem de aici că în izomorfismul $\mathbb{Z}_{1001} \simeq \mathbb{Z}_7 \times \mathbb{Z}_{11} \times \mathbb{Z}_{13}$, 894 corespunde tripletului $(5, 3, 10)$.

2.1 Coduri liniare

Fie $\mathcal{A}$ un alfabet, cu $\#A = n$. Putem identifica $\mathcal{A}$ cu $\mathbb{Z}_n$ și atunci o aplicație $C : \mathcal{A}^k \longrightarrow \mathcal{A}^k$ este dată de:

$$C(a_1, \dots, a_k) = M \cdot (a_1 \; \dots \; a_k)^t,$$

unde $M \in M_k(\mathbb{Z}_n)$ este o matrice inversabilă.

Amintim că, în general, dacă $M \in M_k(R)$, cu R un inel oarecare, avem că M este inversabilă dacă și numai dacă $\det(M) \in R^\times$.

De exemplu, să luăm $\mathcal{A}$ alfabetul latin cu 26 de litere și aplicăm codarea $x_1x_2 \rightsquigarrow y_1y_2$, conform ecuațiilor:

$$\begin{cases} y_1 &= 6x_1 + 2x_2 \bmod 26 \\ y_2 &= 5x_1 + 2x_2 \bmod 26. \end{cases}$$

Căutăm cuvinte x_1x_2 și $x_1'x_2'$ care au aceeași codificare y_1y_2. Dacă acestea există, atunci această codificare nu poate fi folosită în criptografie, din cauza ambiguităților introduse!

Rezolvarea problemei revine la rezolvarea sistemului de congruențe, folosind matricea sistemului:

$$A = \begin{pmatrix} 6 & 1 \\ 5 & 1 \end{pmatrix} \bmod 26.$$

Cum $\det(A) = 1$, matricea este inversabilă, deci sistemul are soluție unică.

Regula de decriptare este dată de înmulțirea la dreapta a formei matriceale a sistemului cu A^{-1}.

2.2 Securitate bazată pe teoria informației

Fie P mulțimea textelor clare (*plain text*), K, mulțimea cheilor și C, mulțimea textelor codificate. Atunci avem funcțiile care ne dau criptarea și decriptarea:

- $\mathrm{Enc} : P \times K \longrightarrow C$;
- $\mathrm{Dec} : C \times K \longrightarrow P, \mathrm{Dec}_k(\mathrm{Enc}_k(m)) = m$, adică are loc o *criptare simetrică.*

Definiţie 2.1: Spunem că sistemul $(\mathrm{Enc}, \mathrm{Dec})$ *are securitate perfectă* dacă și numai dacă, $\forall m \in P, \forall c \in C, p(m \mid c) = p(m)$, unde p este probabilitatea.

Cu alte cuvinte, chiar dacă îl știm pe c, nu avem nicio informație despre m.

Din proprietăți elementare ale funcțiilor, avem:

Lemă 2.1: *Dacă securitatea este perfectă, atunci* $\#K \geq \#C \geq \#P$.

Demonstraţie. Se poate vedea că Enc_k este injectivă, pentru orice cheie fixată $k \in K$, deci $\#C \geq \#P$.

Acum, pentru orice $c \in C, p(c) > 0$ (altfel, c este inutilă). Rezultă că pentru orice $m \in P, c \in C$, $p(c|m) = p(c) > 0$[1]. Atunci, pentru orice mesaj $m \in P$ și orice codificare $c \in C$, există o cheie $k \in K$ cu $\mathrm{Enc}_k(m) = c$. De aici, $\#K \geq \#C$. □

Teoremă 2.1 (Shannon)**:** *Dacă* $\#P = \#C = \#K$, *atunci securitatea perfectă este echivalentă cu afirmațiile:*

- *orice cheie este folosită cu probabilitate egală,* $\dfrac{1}{\#K}$*;*
- *pentru orice mesaj* $m \in P$ *și codare* $c \in C$*, există o cheie unică* $k \in K$*, cu* $\mathrm{Enc}_k(m) = c$.

Demonstraţie. „$\Longrightarrow$“: Trebuie să arătăm că pentru orice mesaj m din P, există o cheie unică pentru descifrare. Cum $\#C = \#K$, rezultă:

$$\#\{\mathrm{Enc}_k(m) \mid k \in K\} = \#K$$

și avem concluzia.

Vrem să arătăm că fiecare cheie este folosită cu probabilitate egală, anume $p(k) = \dfrac{1}{\#K}$, pentru orice $k \in K$.

Fie $\#K = n, P = \{m_i \mid 1 \leq i \leq n\}$ și fie $c \in C$ fixat. Indexăm cheile $k_1, \ldots, k_n$ astfel încît $\mathrm{Enc}_{k_i}(m_i) = c, \forall i$.

[1] Am folosit *formula probabilităților condiționate*:

$$P(A|B) = \frac{P(A \cap B)}{P(B)}.$$

Securitatea perfectă ne asigură că $p(m_i \mid c) = p(m_i)$, deci:

$$p(m_i) = p(m_i \mid c) = \frac{p(c \mid m_i)p(m_i)}{p(c)} = \frac{p(k_i)p(m_i)}{p(c)}.$$

Rezultă că pentru orice i, avem $p(k_i) = p(c)$, adică toate cheile au probabilitate egală și ea este $\frac{1}{n}$.

„$\Longleftarrow$" Reciproc, știind că $\#K = \#P = \#C$ și $p(k) = \frac{1}{\#K}, \forall k \in K$ și că pentru orice mesaj există o cheie unică de descifrare, arătăm că $p(m \mid c) = p(m)$.

Calculăm:

$$p(c) = \sum_{k \in K} p(k)p(m = \mathrm{Dec}_k(c)) = \frac{1}{\#K} \sum_k p(m = \mathrm{Dec}_k(c))$$

egalitatea avînd loc deoarece cheile au probabilitate egală.

Deoarece pentru orice mesaj m și criptare c, există o cheie unică k, cu $\mathrm{Enc}_k(m) = c$, rezultă:

$$\sum_{k \in K} p(m = \mathrm{Dec}_k(c)) = \sum_m p(m) = 1,$$

adică, încercînd toate cheile, sigur descifrăm mesajul. Rezultă $p(c) = \frac{1}{\#K}$. Iar dacă $c = \mathrm{Enc}_k(m)$, atunci $p(c \mid m) = p(k) = \frac{1}{\#K}$.

Folosim teorema lui Bayes:

Teoremă 2.2 (Bayes)**:** *Fie X și Y două variabile aleatoare, cu $p(Y = y) > 0$. Atunci:*

$$p(X = x \mid Y = y) = \frac{p(X = x) \cdot p(Y = y \mid X = x)}{p(Y = y)}.$$

Se obține:

$$p(m|c) = \frac{p(m) \cdot p(c \mid m)}{p(c)} = \frac{p(m) \cdot \frac{1}{\#K}}{\frac{1}{\#K}} = p(m),$$

ceea ce finalizează demonstrația. □

Exemplu 2.1: Fie $\mathcal{A} = \{A, B, \dots, Z\}$ alfabetul cu 26 de litere.

Atunci $\#K = \#P = \#C = 26^n$, iar $p(k) = \frac{1}{26^n}$.

Criptarea se face prin $c = m + k \bmod 26$, pe componente.

Exemplu 2.2: Vernam's Code (OTP): Fie $\#K = \#P = \#C = 2^n$, iar $p(k) = 2^{-n}$. Atunci criptarea se face prin $c = m \oplus k$, unde $\oplus$ este adunarea binară (i.e. suma pe $\mathbb{F}_2 = \mathbb{Z}_2$).

Atenție, însă: dacă se folosește aceeași cheie, întrucît:

$$c_1 \oplus c_2 = m_1 \oplus k \oplus m_2 \oplus k = m_1 \oplus m_2,$$

se poate face analiză de frecvența, iar mesajul poate fi descifrat.

2.3 Entropia

Definiţie 2.2: Fie X o variabilă aleatoare, care ia valorile $x_1, \dots, x_n$, cu probabilitățile $p_i = p(X = x_i)$.

Se definește *entropia* variabilei aleatoare prin:

$$H(X) = -\sum_{i=1}^{n} p_i \log_2 p_i,$$

cu convenția că dacă $p_i = 0$, atunci $p_i \log_2 p_i = 0$.

De exemplu, dacă la o întrebare răspund mereu „da", atunci $p_1 = 1, p_2 = 0$.
Entropia este $H(X) = -1 \log_2 1 - 0 \log_2 0 = 0$, adică nu ofer nicio informație.
Dacă răspund la întîmplare cu „da" sau „nu", atunci $p_1 = p_2 = \frac{1}{2}$ și:

$$H(X) = \frac{1}{2} \cdot \left(-\log_2 \frac{1}{2} - \log_2 \frac{1}{2} \right) = 1,$$

adică ofer 1 bit de informație.

Se poate vedea că funcția entropie are proprietățile:

- $H(X) \geq 0$, pentru orice variabilă aleatoare;
- $H(X) = 0$ dacă și numai dacă există un singur eveniment sigur, iar restul sînt imposibile (i.e. $\exists! i, p_i = 0 \wedge \forall j \neq i, p_j = 0$);
- Dacă variabila este uniform distribuită, adică $p_i = \frac{1}{n}, \forall i$, atunci $H(X) = \log_2 n$.

În calcule, ne va fi de folos **inegalitatea lui Jensen**:

$$\sum_{i=1}^{n} a_i = 1 \Longrightarrow \sum_{i=1}^{n} a_i \log_2 x_i \leq \log_2 \sum_{i=1}^{n} a_i x_i,$$

cu egalitate dacă și numai dacă $x_1 = x_2 = \cdots = x_n$.

De asemenea, avem:

Teoremă 2.3: *Dacă X are n valori posibile, atunci:*

$$0 \leq H(X) \leq \log_2 n.$$

Pentru justificare, este suficient să observăm:

$$H(X) = -\sum p_i \log_2 p_i = \sum p_i \log_2 \frac{1}{p_i} \leq \log_2 \sum p_i \frac{1}{p_i} = \log_2 n.$$

Definiţie 2.3: Se definește *entropia condiționată* pentru variabila aleatoare X și evenimentul y prin:

$$H(X \mid y) = -\sum_X p(X = x \mid Y = y) \log_2 p(X = x \mid Y = y)$$

În general, pentru o întreagă variabilă aleatoare Y, avem:

$$H(X \mid Y) = \sum p(Y = y) \cdot H(X \mid y).$$

Fie acum X, Y două variabile aleatoare. Definim:

$$r_{ij} = p(X = x_i \wedge Y = y_j).$$

Definim și *entropia comună* prin:

$$H(X, Y) = -\sum_{i=1}^{n} \sum_{j=1}^{m} r_{ij} \log_2 r_{ij}.$$

Aceasta are proprietățile:

- $H(X, Y) \le H(X) + H(Y)$, cu egalitate dacă și numai dacă variabilele sînt independente $(p(X = x \mid Y = y) = p(X = x))$;
- $H(X, Y) = H(Y) + H(X \mid Y)$;
- $H(X \mid Y) \le H(X)$, cu egalitate dacă și numai dacă variabilele sînt independente.

2.3.1 Aplicații în criptografie

Interpretările pe care le putem da în criptografie sînt următoarele:

- Dacă $H(P \mid K, C) = 0$, atunci, dacă știm mesajul criptat și cheia, vom ști mesajul inițial;
- Dacă $H(C \mid P, K) = 0$, atunci, dacă știm mesajul și cheia, vom ști mesajul criptat.

Folosind faptul că $H(X, Y) = H(Y) + H(X \mid Y)$, avem:

$$H(K, P, C) = H(P, K) + H(C \mid P, K) = H(P, K) = H(K) + H(P),$$

deoarece K și P sînt independente.

Rezultă $H(K, C) = H(K) + H(P)$.

Vom numi entropia $H(K \mid C)$ *key equivocation* (cantitatea de echivoc din cheie), reprezentînd incertitudinea care rămîne privitoare la cheie chiar după ce s-a aflat codul cifrat. Aceasta se calculează simplu;

$$H(K \mid C) = H(K, C) - H(C) = H(K) + H(P) - H(C).$$

Să luăm un exemplu. Fie mulțimile:

$$P = \{a, b, c, d\}, \quad K = \{k_1, k_2, k_3\}, \quad C = \{1, 2, 3, 4\}.$$

Considerăm probailitățile:

$$p(a) = 0,25; \quad p(b) = p(d) = 0,3; \quad p(c) = 0,15$$
$$p(k_1) = p(k_3) = 0,25; \quad p(k_2) = 0,5;$$
$$p(1) = p(2) = p(3) = 0,2625; \quad p(4) = 0,2125.$$

Obținem entropiile:

$$H(P) = 1,9257; \quad H(K) = 1,5; \quad H(C) = 1,9944.$$

Cum $H(C) = 1,9527 + 1,5 - 1,9944 = 1,4583$, rezultă că, dacă știm un mesaj cifrat, mai trebuie să găsim aproximativ 1,5 biți de informație despre cheie. Această cantitate este foarte mică, deci cifrarea este foarte nesigură!

Cifrarea poate fi dată de:

	a	b	c	d
k_1	3	4	2	1
k_2	3	1	4	2
k_3	4	3	1	2

Fie acum L limbajul natural, iar H_L entropia corespunzătoare unei litere (adică informația purtată de o literă). Un șir aleatoriu de litere are $H = \log_2 25 \simeq 4,70$, deci $H_L \leq 4,70$.

Probabilitățile de apariție a unor litere poate fi calculată ținînd seama de statistici. De asemenea, putem ține cont și de reguli sintactice, precum faptul că Q este mereu urmat de U, TH apare foarte frecvent etc.

Așadar, putem studia P^2, variabila aleatoare a bigramelor, adică a perechilor de litere. Avem:

$$H(P^2) = -\sum_{i,j} p(P = i, P' = j) \log_2(P = i, P' = j) \simeq 7,12.$$

Definiţie 2.4: Putem defini entropia limbajului natural L prin:

$$H_L = \lim_{n \to \infty} \frac{H(P^n)}{n}.$$

Se poate arăta că $1 \leq H_l \leq 1,5$, deci rezultă că o literă din engleză folosește aproximativ 5 biți de informație, dar conține aproximativ 1,5 biți de informație.

Pornind de la entropie, putem defini:

Definiţie 2.5: Se definește *redundanța limbajului natural* prin:

$$R_L = 1 - \frac{H_L}{\log_2 \#P}.$$

Dacă $H_L = 1,25$ pentru engleză, avem:

$$R_L = 1 - \frac{1,25}{\log_2 26} \simeq 0,75,$$

deci putem comprima texte în această limbă de la 10 MB la 2,5 MB!

2.4 Chei false

Fie $c \in C, |c| = n$ și definim $K(c) = \{k \in K \mid \text{Enc}_k(c) \text{ are sens }\}$.

Numărul $\#K(c) - 1$ se numește *numărul de chei false.*

Numărul mediu de chei false se calculează prin:

$$s_n = \sum_{c \in C} p(c)(\#K(c) - 1) = \Big(\sum_{c \in C} p(c) \cdot \#K(c)\Big) - 1.$$

Pentru cazul practic cînd n este mare, atunci $\#P = \#K$ implică:

$$\begin{aligned}
\log_2(s_n + 1) &= \log_2 \sum_{c \in C} p(c) \cdot \#K(c) \\
&\geq \sum p(c) \log_2(\#K(c)) && \text{(Jensen)} \\
&\geq \sum p(c) H(K \mid c) \\
&= H(K \mid C) \\
&= H(K) + H(P) - H(C) \\
&\simeq H(K) + nH_L - H(C) && \text{(pentru n foarte mare)} \\
&= H(K) - H(C) + n(1 - R_L) \log_2 \#P && \text{(def. redundanță)} \\
&\geq H(K) - n \log_2 \#C + n(1 - R_L) \log_2 \#P && H(C) \leq n \log_2 \#C \\
&= H(K) - nR_L \log_2 \#P && (\#P = \#C).
\end{aligned}$$

Concluzia:

$$s_n \geq \frac{\#K}{(\#P)^{nR_L}} - 1.$$

Legat de acest concept, avem:

Definiţie 2.6: Se definește *distanța de unicitate* (eng. *unicity distance*) n_0 valoarea lui n astfel încît numărul de chei false devine 0.

Conform calculului de mai sus, avem:

$$n_0 \simeq \frac{\log_2 \#K}{R_L \log_2 \#P}.$$

De exemplu, pentru cazul *substituțiilor în limbajul natural*, avem $\#P = 26, \#K = 26! \simeq 4 \cdot 10^{25}$ și $R_L = 0,75$, deci:

$$n_0 \simeq \frac{88,4}{0,75 \cdot 4,7} \simeq 25.$$

Rezultă că, pentru $|c| \geq 25$ se presupune că există o unică descifrare cu sens.

Pentru cazul *șirurilor de biți și chei de lungime l*, avem $\#P = 2$, $\#K = 2^l$ și $R_L = 0,75$. Avem:

$$n_0 \simeq \frac{l}{0,75} = \frac{4l}{3}.$$

Dacă comprimăm datele înainte de a le transmite, deci facem redundanța (aproape) nulă, avem $n_0 \longrightarrow \frac{l}{0} \longrightarrow \infty$!

2.5 Linear feedback stream registers (LFSR)

Fie un registru de lungime L și biții $c_1, \ldots c_L$ stocați în el. Considerăm starea inițială dată de vectorul $[s_{L-1}, \ldots, s_1, s_0]$, iar șirul de ieșire (starea finală) $s_0, s_1, \ldots, s_{L-1}, s_L, s_{L+1}, \ldots$, unde am calculat prin:

$$s_j = c_1 s_{j-1} \oplus c_2 s_{j-2} \oplus \cdots \oplus c_L s_{j-L}, \forall j \geq L.$$

Dacă $s_{i+N} = s_i$, șirul este periodic, de perioadă N, iar $N \leq 2^L - 1$.

Putem scrie tranziția matriceal, folosind:

$$\begin{aligned} M &= \begin{pmatrix} 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ c_L & c_{L-1} & c_{L-2} & \ldots & c_1 \end{pmatrix} \\ v &= (1, 0, 0, \ldots, 0) \\ s &= (s_1, s_2, \ldots, s_L) \quad \text{starea internă.} \end{aligned}$$

Atunci tranziția la starea următoare este dată de $s = M \cdot s$, iar bitul de ieșire este $v \cdot s$.

Definim *polinomul de conexiune* prin:

$$C(X) = 1 + c_1 X + \cdots + c_L X^L = \det(XM - I_L) \in \mathbb{F}_2[X]$$

Exemplu 2.3: Pentru $C(X) = X^3 + X + 1$, avem figura 2.1.

Vom fi interesați de cazul particular din definiția următoare.

Definiție 2.7: Polinomul $C(X)$ este *primitiv* dacă și numai dacă este ireductibil. În acest caz, vom avea $\mathbb{F}_2[X]/(C(X)) = \mathbb{F}_{2^L}$, iar $\mathbb{F}_{2^L}^\times = \langle \theta \rangle$, grup ciclic, generat de o rădăcină a lui C.

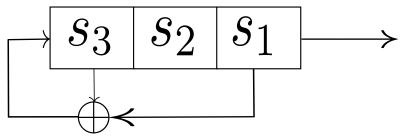


Figura 2.1: LFSR cu polinomul de conexiune $C(X) = X^3 + X + 1$

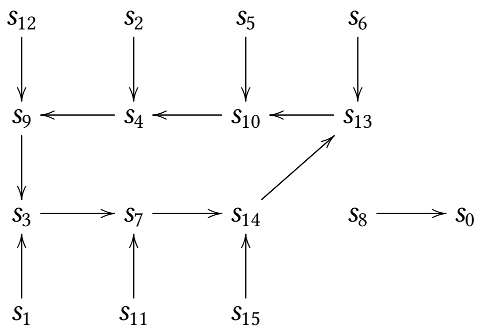
De asemenea:

- dacă $c_L = 0$ (spunem că polinomul este *singular*), atunci șirul devine periodic mai tîrziu;
- dacă $c_L = 1$ și C este ireductibil, atunci obținem un șir periodic, cu perioada N, care divide $2^L - 1$ (va fi cea mai mică valoare astfel încît $C(X) \mid 1 + X^N$;
- dacă $c_L = 1$ și C este primitiv, atunci chiar $N = 2^L - 1$.

Exemplu 2.4: Dacă $L = 4$ și $C = X^3 + X + 1$ (singular) și:

$$M = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{pmatrix}$$

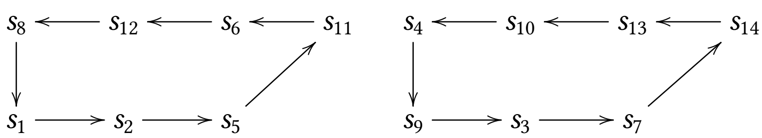
iar figura arată astfel:



Exemplu 2.5: Dacă $C(X) = X^4 + X^3 + X^2 + X + 1 = (X + 1)(X^3 + X + 1)$, avem matricea:

$$M = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix},$$

iar figura:



Exemplu 2.6: Pentru cazul $C(X) = X^4 + X + 1$, care este ireductibil și primitiv, avem:

$$M = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

și se obține un ciclu de lungime 15.

CURS 3

DES ȘI AES

3.1 LFSR (completări)

În cazul LFSR, dacă se cunosc $2L$ valori, se pot afla coeficienții $c_1, \ldots, c_L$, din sistemul:

$$\begin{pmatrix} s_{L-1} & s_{L-2} & \ldots & s_1 & s_0 \\ s_L & s_{L-1} & \ldots & s_2 & s_1 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ s_{2L-2} & s_{2L-3} & \ldots & s_L & s_{L-1} \end{pmatrix} \cdot \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_L \end{pmatrix} = \begin{pmatrix} s_L \\ s_{L-1} \\ \vdots \\ s_{2L-1} \end{pmatrix}$$

Rezultă că, în general, un LFSR este nesigur pentru un atac pe text clar.

Definiție 3.1: Fie $s = s_1, s_2, \ldots$. Definim *complexitatea liniară* a șirului s, notată $L(s)$, prin:

- $L(s) = 0$ dacă $s = 0, 0, 0, \ldots$;
- $L(s) = \infty$ dacă niciun LFSR nu produce $L(s)$;
- $L(s) =$ lungimea celui mai scurt LFSR care produce s.

În general, se pot constata următoarele proprietăți, pentru un șir finit $s_0, \ldots, s_{n-1}$:

- $0 \leq L(s) \leq n$;
- Dacă s este periodic, de perioadă N, atunci $L(s) \leq N$;
- $L(s \oplus t) \leq L(s) + L(t)$.

În cazuri concrete, putem combina mai multe LFSR, de exemplu:

$$f(x_1, x_2, x_3, x_4, x_5) = 1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_1 \odot x_2 \odot x_3 \odot x_5.$$

Dacă vrem să calculăm complexitatea combinației, $f(L_1, \dots, L_n)$, înlocuim $\oplus$ cu $+_{\mathbb{Z}}$ și $\odot$ cu $\cdot_{\mathbb{Z}}$.

Exemplu 3.1: Geffe generator: fie funcția:

$$f(x_1, x_2, x_3) = x_1 x_2 \oplus x_2 x_3 \oplus x_3.$$

Complexitatea liniară este $L_1 L_2 + L_2 L_3 + L_3$, iar perioada, $(2^{L_1} - 1)(2^{L_2} - 1)(2^{L_3} - 1)$.

Modul în care codifică un mesaj $x_1 x_2 x_3$ este dat în tabelul de mai jos.

x_1	x_2	x_3	z
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Se poate vedea de aici că $p(z = x_1) = p(z = x_3) = \frac{3}{4}$, care sînt valori mult prea mari pentru a da o securitate bună.

Exemplu 3.2: Atac bazat pe corelație: Presupunem că știm L_1, L_2, L_3. Atunci putem proceda astfel:

```
pentru toate polinoamele de conexiune primitive de grad L1
   pentru toate stările inițiale ale LFSR1
         calculează 2L1 biți din LFSR1
         calculează cîți sînt egali folosind Geffe
repetă pentru LFSR3
recuperează LFSR2 din x1x2 ⊕ x2x3 ⊕ x3.
```

Exemplu 3.3: Ron's Cypher RC4: Fie S un vector cu valorile $0, 1, \dots, 255$ permutate cumva. Procedura este:

```
i = 0; j = 0
   i := (i + 1) mod 256
   j := (j + S_i) mod 256
   swap(S_i, S_j);
   t := (S_i + S_j) mod 256
   k := S_t
```

Starea inițială este generată folosind cheia:

```
for i = 0 to 255 S_i = i
j = 0
for i = 0 to 255 do
    j := j + S_i + k_i mod 256
    swap(S_i, S_j)
```

Problema principală este că procedura nu generează valori atît de aleatorii precum s-ar dori.

INDEX

BIBLIOGRAFIE

[Smart, 2013] Smart, N. P. (2013). *Cryptography, an Introduction.* online.

[Smart, 2016] Smart, N. P. (2016). *Cryptography Made Simple.* Springer.