Eötvös Loránd Tudományegyetem

Természettudományi Kar

Nógrádi Ábel

Lineáris hibajavító kódok

BSc Szakdolgozat

Alkalmazott Matematika

Témavezető:

Hermann Péter

Algebra és Számelmélet Tanszék

Budapest, 2015

Köszönetnyilvánítás

Köszönöm Hermann Péter tanáromnak, témavezetőmnek a témaválasztástól a szakmai magyarázatokon át a nyelvhelyességi hibák javításáig nyújtott segítségét.
Köszönöm örök első olvasóimnak, anyámnak és apámnak a figyelmét.

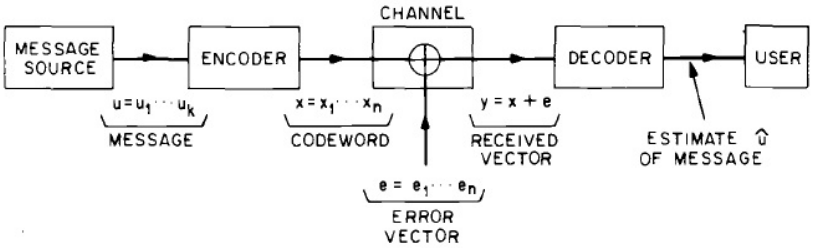
Tartalomjegyzék

"A királynét megölni nem kell félnetek jó lesz ha mindenki egyetért én nem ellenzem." - János esztergomi érsek híres kétértelmű mondata Gertrudis megölése kapcsán

1. fejezet

Bevezetés

Az információelmélet-kódelmélet a hírközlés matematikai elmélete. Foglalkozik az adatátvitellel, az adattömörítéssel és a titkosítással. A mai nevén kódelmélet az 1940-es években indult világhódító útjára a gépészmérnökök körében. Azóta a matematika önálló része lett. Leginkább a számítástudományban használják.
Dolgozatomban a kódolás adatvédelemmel foglalkozó részéről írok. Egyik helyről a másikra küldött üzenet a csatornán áthaladva meghibásodhat, elveszhetnek elemei: az üzenet tartalma megváltozhat. A meghibásodás megállapítására és esetleges kijavítása érdekében kódoljuk az üzeneteket. A kódolás hatékonysága függ a tömörségétől, és a hibák valószínűségétől. Minthogy különböző csatornák másképp hibásodhatnak meg, a különböző csatornákra különböző kódokat használunk.
A dolgozat a lineáris kódokkal foglalkozik. A lineáris kódok nagyon egyszerűek, viszont annál hatékonyabbak a gyakorlatban. A CD-ket, a szatellitről érkező üzeneteket, az e-maileket, a QR-kódokat is lineárisan kódolják. Miután megnézzük, hogy a lineáris kódok milyen tulajdonságokkal bírnak, felső becslést adunk a jó kódokra. Megnézzük Shannon tételét, amely a kódelmélet kiindulópontja.
Ezután kicsit részletesebben foglalkozom a ciklikus kódokkal.



2. fejezet

Lineáris kódok

A lineáris kódok a gyakorlatban nagyon jól használhatók.

Egy $u = u_1u_2 \dots u_{k-1}u_k$ k darabból álló üzenetblokkot akarunk átküldni egy zajos csatornán. Kódolni fogjuk ezeket az üzeneteket annak érdekében, hogy védettebbé tegyük őket a hibáktól. Ezt úgy tesszük, hogy ezt a k hosszú blokkot átalakítjuk egy n hosszú $x = x_1x_2 \dots x_n$ kódszóvá, ahol $n \geqslant k$. Majd ezeket a kódszavakat küldjük el a csatornán, ahol meghibásodhatnak, így a dekóder esetleg az eredetitől eltérő üzenetté fejti vissza a kódot. A kódszavak összessége a kód.

A kód egy k dimenziós altere az n dimenziós térnek, tehát a kódokra vektorként tekintünk. Miután q^k üzenet van, amit kódszóvá alakítunk, q^{n-k}-val kevesebb mint q^n, azaz mint az egész tér. Azt szeretnénk elérni, hogy a nem kódszóként használt vektorok a különböző kódszók körül diszjunkt t sugarú gömböket alkossanak. Ekkor tud ugyanis t hibát kijavítani a kód. Ezért egy kódnak nagyon fontos tulajdonsága, hogy a benne levő kódszavak milyen távolságra vannak egymástól. Ha például ezek a gömbök kiteszik az egész teret, akkor perfekt kódokról beszélünk. Ebből következik, hogy ha t-nél több, de legfeljebb $2t$ hiba esik a zajos csatornán való küldéskor, akkor az átküldött üzenet hibás lesz. Ezt észleli a dekóder. Megeshet, hogy 2t+1 hiba esik, akkor egy kódszóból egy másikat kapunk, a dekóder nem észlel hibát. Ennek nagyon kicsi az esélye, ha viszonylag kicsi a p valószínűsége, azaz a csatornán küldött szimbólumok meghibásodási lehetőségének valószínűsége. A következőkben definiálom a későbbiekben használt fogalmakat.

2.0.1. Definíció. *Legyen $f : \mathbb{F}_q^k \to \mathbb{F}_q^n$ injektív függvény. Ekkor $\mathcal{C} := Imf = \{f(x)|x \in \mathbb{F}_q^k\}$* ***kód****.*
Lineáris *a kód, ha teljesíti minden $a, b \in \mathbb{F}_q^k, \lambda \in \mathbb{R}$-ra, hogy:*

$i) f(a+b) = f(a) + f(b)$
$ii) f(\lambda * a) = \lambda * f(a)$

Vagyis a kód egy (lineáris) altere $\mathbb{F}_q^n$-nek.

2.0.2. Megjegyzés. A dolgozatban végig felteszem a $q = 2$-t. Bináris, azaz $\mathbb{F}_2$ felett vagyunk, azaz modulo 2 összeadást és modulo 2 szorzást használunk.(Magasabb elemszámú test felett bonyolultabb, de lényegében hasonló a dolog.)

Egy kódnak a dimenziója k és hossza n után a harmadik legfontosabb paramétere a távolsága.

2.0.3. Definíció. *$x = x_1x_2...x_n$ és $y = y_1y_2 \dots y_n$ vektorok **Hamming-távolsága** az a szám, ahány helyen a két vektor eltér. Azaz $d(x, y) = |\{i \mid x_i \neq y_i\}|$*

2.0.4. Definíció. *$x = x_1x_2 \dots x_n$ vektor **Hamming-súlya** az a szám, ahány helyen a vektor nem nulla. Azaz $w(x) = |\{i \mid x_i \neq 0\}|$*

Nyilván $d(x, y) = w(x - y)$.

2.0.5. Megjegyzés. Egy $\mathcal{C}$ kód **távolsága** a benne levő kódszavak minimális távolsága. Ezeket a lineáris kódokat $[n, k, d]$-val jelöljük.

2.0.6. Tétel. *Egy $\mathcal{C}$ kód, aminek a minimális távolsága d, akkor $t = \lfloor \frac{1}{2}(d-1) \rfloor$ hibajavító. Ha d páros, akkor $\frac{1}{2}(d-2)$ hibát kijavít és $d/2$-t jelez.*

Bizonyítás. Tegyük fel, hogy a minimális távolság két kódszó között $d = 2t+1$. Ekkor x kódszó körülötti t sugarú gömbben van minden y kódszó, amelyre igaz, hogy $d(x, y) \leqslant t$. Ezek a kódszavak körülötti gömbök diszjunktak. Azaz ha a küldött u kódszóban t hiba szerepel, akkor is még u körülötti gömbben lesz benne, tehát közelebb van u-hoz, mint bármelyik másik kódszóhoz. Így a legközelebbi szomszéd dekodólás kijavítja a hibát. (lásd: 2.3) Ha d páros, akkor $\frac{1}{2}(d-2)$ sugarú gömbök diszjunktak, és a dekódoló ki is javít ennyi hibát. Ha viszont $d/2$ hiba esik a kapott kódszóban, a vektor pont két kódszó közé esik. Ebben az esetben ezt csak jelzi a dekódoló. □

2.1. Lineáris kódok tulajdonságai

A $\mathcal{C}$ akkor lineáris kód, ha van hozzá egy $\mathbf{H}$ mátrix, amire teljesül az, hogy:

$$\mathbf{H}x^T = 0 \tag{2.1}$$

minden x kódszóra a kódból. $\mathbf{H}$ mátrixot **paritásellenőrző mátrix**nak nevezzük.
A lineáris kódok a következő tulajdonságokkal bírnak:

- A paritásellenőrző mátrix egy $(n-k)*n$ méretű mátrix. Standard alakja $\mathbf{H} = [A \mid I_{n-k}]$.
- Ezeknek az $(n-k)$ rangú mátrixoknak az oszlopai lineárisan függetlenek, nincs null-soruk, vagy két megegyező soruk.
- Tartozik hozzá egy $\mathbf{G}$ **generátormátrix**, amely egy $k*n$ méretű mátrix, (amely ha **szisztematikus** $\mathbf{G} = [I_k \mid -A^T]$ alakú) és igaz rá, hogy

  $$x = uG, \tag{2.2}$$

  tehát $\mathbf{G}$ sorai generálják a kódszavakat. Vagyis $\mathbf{G}$ sortere a kód. (2.1) és (2.2) egyenletekből következik, hogy $\mathbf{GH}^T = 0$ és $\mathbf{HG}^T = 0$.
- Egy $x = x_1x_2 \dots x_n$ kódszó hossza n. Ha $\mathbf{H}$-nak $n-k$ lineárisan független sora van, akkor a $\mathcal{C}$ kódnak 2^k kódszava van. k-t a kód **dimenzió**jának nevezzük. Ekkor $[n,k]$-val jelöljük a kódot. Ha a kód d minimális távolságát is tudjuk, akkor $[n,k,d]$-val jelöljük.
- Azt mondjuk, hogy a kód **hatékonysága** $R = k/n$. Magyarul **jelsebesség**nek is hívják. Mértékegysége bit/csatornahasználat, ami a csatorna kihasználtságát méri. Általánosan: $R = \frac{1}{n} log_q|\mathcal{C}|$.
- Ha x, y kódszó, akkor $x+y$ is az a linearitás miatt.

2.1.1. Tétel. *Lineáris esetben a kód minimális távolsága egyenlő a nemnulla kódszavak minimális súlyával. Azaz $d = min\ w(z)$, ahol $0 \neq z \in \mathcal{C}$.*

Bizonyítás. $d(x,y) = d(x-y,0) = w(x-y)$ és ha $x, y \in \mathcal{C}$ akkor $x-y \in \mathcal{C}$ $\square$

2.2. Kódolás

Mikor egy üzenetet egy szisztematikus lineáris kóddal kódolunk, akkor a kódszó két részre bomlik. Az első része tartalmazza az üzenetet magát:

$$x_1 = u_1, x_2 = u_2, \ldots, x_k = u_k,$$

a maradék $n - k$ szimbólum a paritásellenőrző karakter. Ezeket a paritásellenőrző mátrix határozza meg. (2.1)

2.2.1. Példa. Adva van egy paritásellenőrző mátrix:

$$\mathbf{H} = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$k = 3$ hosszú az üzenet, amit $n = 6$ hosszú kódszóvá alakítunk át. Ebből kiszámoljuk a generátormátrixot:

$$\mathbf{G} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 \end{pmatrix}$$

Tehát a lehetséges $2^k = 8$ kódszót úgy kapjuk meg, hogy az üzenetekkel balról megszorozzuk a $\mathbf{G}$ generátormátrixot. Például $u = 101$ üzenetet szeretnénk kódolni, akkor összeadjuk a $\mathbf{G}$-nek a megfelelő sorait, azaz az első és harmadik sorát, s megkapjuk a keresett $x = 101010$ kódszót.

2.3. Dekódolás

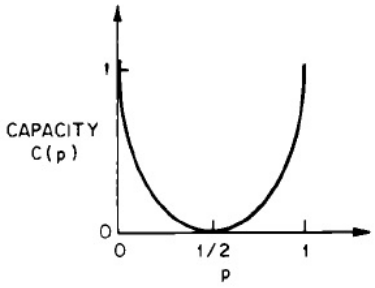
Most, hogy az u üzenetet kódoltuk x-é, és elküldtük a zajos kommunikációs csatornán, jön a visszafejtés. Sajnos nem mindig x-et kapja meg a fogadó, hanem az esetleges hibák miatt egy y-t.

2.3.1. Definíció. *Az $e = y - x = e_1 e_2 \ldots e_n$ vektort hibavektornak nevezzük.*

Hogy visszafejtsük az eredeti üzenetet, elég a hibavektort megfejteni. Csakhogy a dekódoló sosem lehet teljesen biztos a hibavektorban. Azt kell hibavektornak választania, aminek a legnagyobb a valószínűsége. Feltéve, hogy minden x kódszó egyenlő valószínűséggel

szerepel, és hogy $0 \leqslant p < 1/2$, egy karakter meghibásodásának valószínűsége, azaz a csatorna zajossága miatt egy karakter 0-ról 1-re (vagy fordítva) változhat meg, akkor ezt **maximum valószínűségi dekódolás**nak hívják. A dekódolás bemutatásához kell még pár fontos definíció.

2.3.2. Definíció. *A **bináris szimmetrikus csatornán** egy csatornát értünk, ami bináris inputot kap, binárisat ad vissza, p valószínűségű hibával. A csatorna **kapacitása** a p valószínűségtől és a test q elemszámától függ.*
Esetünkben $q = 2$ a kapacitás $1 + p\,log_2 p + (1-p)\,log_2(1-p)$. Ez mindig 0 és 1 közé esik.



Capacity of binary symmetric channel.

Egy bináris szimmetrikus csatornán adott egy $e = e_1 e_2 \dots e_n$ hibavektor. Ekkor $e_i = 1$ (azaz i-edik karakter rossz) p valószínűséggel, $e_i = 0$ (azaz ha az i-edik karakter jó) $1 - p$ valószínűséggel fordul elő. Feltesszük, hogy $0 \leqslant p < 1/2$. Ebben az esetben egy a súlyú v rögzített vektorra teljesül, hogy:

$$P(e = v) = p^a (1-p)^{n-a} \tag{2.3}$$

és mivel $(1-p) > p$, igaz, hogy:

$$(1-p)^n > p(1-p)^{n-1} > p^2(1-p)^{n-2} > \dots \tag{2.4}$$

Ezért egy 1 súlyú hibavektor nagyobb valószínűséggel szerepel, mint egy 2 súlyú, és így tovább. A dekóder így azt az x kódszót fogja választani, ami a legközelebb van az y-hoz, azaz a legkisebb súlyú e hibavektort. Ezt hívják **legközelebbi szomszéd dekódolás**nak. Ehhez azonban össze kell hasonlítani a 2^k kódszót, ami már egy kicsivel nagyobb k-ra szinte lehetetlen. Ezért egy másik módszert használunk, aminek a neve **standard array módszer**.

2.3.3. Definíció. *Legyen $\mathcal{C}$ egy $[n,k]$ lineáris kód $\mathbb{F}_q^n$-ban. Ekkor $\mathcal{C}$-nek az a vektor szerinti* ***mellékosztálya****: $a+\mathcal{C}=\{a+x : x\in\mathcal{C}\}$*

A mellékosztályok néhány tulajdonsága:

- minden $b\in\mathbb{F}_q^n$ vektor benne van egy mellékosztályban
- minden mellékosztály q^k vektort tartalmaz
- két mellékosztály vagy diszjunkt, vagy egyenlők
- $$\mathbb{F}^n=\mathcal{C}\cup(a_1+\mathcal{C})\cup(a_2+\mathcal{C})\cup\ldots\cup(a_{q^{n-k}-1}+\mathcal{C}) \tag{2.5}$$

Tehát a dekóder kap egy y kódszót, amelyik beletartozik a (2.5) egyik mellékosztályába. Legyen ez az $y=a_i+x$. Ha az eredeti üzenet x' volt, akkor a hibavektor $e=y-x'=a_i+x-x'=a_i+x''$, amit $a_i+\mathcal{C}$ tartalmaz. Ezért az összes lehetséges hibavektor egy mellékosztályban van y-nal. A dekódoló a legkisebb súlyú $\tilde{e}$ vektort fogja választani a mellékosztályból. Ezt a vektort hívjuk **osztályelső**nek. Ekkor y visszafejthető: $\tilde{x}=y-\tilde{e}$. Feltesszük, hogy az a_i-k a (2.5) állításból osztályelsők.

A standard array egy táblázat, aminek az első sorában a lehetséges eredeti üzenetek, a másodikban a kódszavak, első oszlopban 0 vektorral, a többi sorban pedig az $a_i+\mathcal{C}$ mellékosztályok az osztályelsőkkel (hibavektorokkal) az elején. A visszafejtés menete a következő: a dekóder megkapja az y vektort, megkeresi melyik mellékosztályban szerepel, s veszi annak az osztályelsőjét. Ebből kiszámolható az eredeti kódszót, $\tilde{x}=y-\tilde{e}$. Ebből pedig az üzenet.

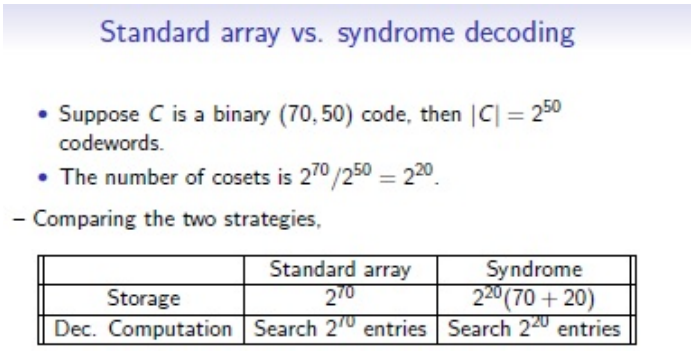
2.3.4. Példa. 2.2.1 példa folytatása:

üzenet	000	001	010	011	100	101	110	111
kódszó	000000	001100	010101	011001	100110	101010	110011	111111
1.mellékosztály	100000	101100	110101	111001	000110	001010	010011	011111
2.mellékosztály	010000	011100	000101	001001	110110	111010	100011	101111
3.mellékosztály	001000	000100	011101	010001	101110	100010	111011	110111
4.mellékosztály	000010	001110	010111	011011	100100	101000	110001	111101
5.mellékosztály	000001	001101	010100	011000	100111	101011	110010	111110
6.mellékosztály	110000	111100	100101	101001	010110	011010	000011	001111
7.mellékosztály	100001	101101	110100	111000	000111	001011	010010	011110

Mint látjuk, mind a $2^6 = 64$ darab lehetséges vektor szerepel a táblázatban. Az y vektort szeretnénk visszafejteni. Legyen ez most $y = 010001$. Ez a harmadik mellékosztályban van. Megkeressük az osztályelsőt, ami a táblázat bal szélén szerepel, vagyis a $e = 001000$ vektor. Ebből megkapjuk az eredeti kódszavunkat: $x = y - e = 011001$, ami a 011 üzenetet jelenti.

A **szindróma** arra szolgál, hogy a dekóder könnyedén találja meg, hogy az y vektor melyik mellékosztályban van. A szindróma $S = \mathbf{H}y^T$ egy $n - k$ hosszú vektor, amely akkor és csakis akkor nulla, ha az y vektor egy kódszó (definícióból, ld. (2.1)). Tehát ha nem esett hiba a küldött kódszóban, akkor a szindróma nulla. Fordítva nem igaz az állítás: a szindróma lehet nulla, ha hiba van a kódszóban. Amennyiben előfordul hiba, a szindróma $S = \mathbf{H}y^T = \mathbf{H}x^T + \mathbf{H}e^T = \mathbf{H}e^T$ egyenlő, azaz a $S = \sum_{i=1}^{n} \mathbf{H}_i e_i$, ami egyenlő a paritásellenőrző mátrix azon oszlopainak összegével, ahol a hibavektorban egyes szerepel. Amiért ez fontos nekünk az az, hogy a szindrómák és a mellékosztályok között kölcsönös megfeleltetés van. Ezáltal a dekóder csak kiszámolja a kapott y vektor szindrómáját, és rögtön megkapjuk, hogy y melyik mellékosztályban van.

A standard array módszer és a szindróma használata között mind a tárhely nagyságában, mind a számítások mennyiségében hatalmas a különbség.



2.3.5. Példa. 2.3.4 példa folytatása: A kapott $y = 010001$ vektort meg kell keresnünk, hogy melyik mellékosztályban van. Ezt a szindróma segítségével tesszük meg. Kiszámoljuk előre a különböző mellékosztályok különböző szindrómáját: $S = \mathbf{H}e^T$

üzenet	szindróma
kódszó	000
1.mellékosztály	110
2.mellékosztály	101
3.mellékosztály	100
4.mellékosztály	010
5.mellékosztály	001
6.mellékosztály	011
7.mellékosztály	111

$S = \mathbf{H}y^T = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$ így az y a 3.mellékosztályban van. Vegyük észre, hogy ha más hibavektort választunk osztályelsőnek, például a második elemet a 3. mellékosztályból, ugyanezek a vektorok szerepelnének a mellékosztályban. Akkor viszont a rossz hibavektor miatt rossz kódszót is kapnánk.

2.4. Hibalehetőség

Amikor egy dekódoló standard array-t használ, mindig egy osztályelsőt választ hibavektornak. Akkor és csak akkor nem hibázik, ha a valódi hibavektor tényleg egy osztályelső. Ha nem az, akkor rossz kódszót fog visszaadni.

2.4.1. Definíció. *Legyen $P_{\mathcal{C}}$ annak a valószínűsége, hogy a dekóder hibás kódszót ad vissza.*

Ha P_i a valószínűsége annak, hogy a dekódoló rosszul dönt, amikor x_i -t küldtük az $M := 2^k$ kódszó közül, akkor $P_{\mathcal{C}} = \frac{1}{M}\sum_{i=1}^{M} P_i$. Ha a dekódoló standard array-t használ, csak akkor hibázhat, ha nem az osztályelsőt választja hibavektornak, azaz $P_{\mathcal{C}} = P(e \neq \text{osztályelső})$. Tegyük fel, hogy α_i darab osztályelső van i súllyal. Használva a (2.3)-t, azt kapjuk hogy:

$$P_{\mathcal{C}} = 1 - \sum_{i=0}^{n} \alpha_i p^i (1-p)^{n-i} \tag{2.6}$$

Ha a $\mathcal{C}$ kód minimális távolsága $d = 2t+1$ vagy $d = 2t+2$, akkor t hibajavító a kód. (ld. (2.0.6) tétel) Azaz minden hibavektor, amelynek kevesebb a súlya mint t, osztályelső. Ha

ugyanis x nem volna osztályelső, akkor lenne még egy y vektor, amelynek a súlya nem nagyobb nála. Ekkor viszont $w(x-y) \leqslant w(x)+w(y) \leqslant \lfloor \frac{d-1}{2} \rfloor + \lfloor \frac{d-1}{2} \rfloor \leqslant d-1$, ami ellentmond a d minimális távolságnak. Speciálisan $\alpha_i = \binom{n}{i}$, ha $0 \leq i \leq t$. Csakhogy $i > t$ esetén nagyon nehéz kiszámolni α_i-t. Ha a csatorna p meghibásodási valószínűsége kicsi, akkor $1-p \approx 1$ és $p^i(1-p)^{n-i} \gg p^{i+1}(1-p)^{n-i-1}$, azaz eggyel nehezebb súlyú hibavektor sokkal kisebb valószínűséggel fordul elő. Ebben az esetben nagy i mellett a (2.6) egyenlet erre módosul:

$$P_{\mathcal{C}} \approx 1 - \sum_{j=0}^{t} \binom{n}{j} p^j (1-p)^{n-j}. \tag{2.7}$$

(2.7) jobb oldala minden esetben egy felső becslést ad a $P_{\mathcal{C}}$-re.

2.4.2. Definíció. *Ha $\alpha_i = 0$ minden $i > t = \lfloor (d-1)/2 \rfloor$-re, akkor (2.7) pontosan teljesül. Ezeket a kódokat **perfekt kódok**nak nevezzük.*

3. fejezet

Felső korlátok

A kódolásban sokféle felső korlát létezik. Van olyan, amelyik a kód minimális távolságára ad felső becslést, van amelyik a hibavalószínűségre, de mind közül talán a legfontosabb a csatorna kihasználtságára vonatkozó korlát.

3.1. Gömbpakolási-korlát

Egy t-hibajavító kód kijavít minden t-nél kisebb súlyú hibavektort, viszont t-nél nagyobb súlyút nem. Ekvivalensen, a $\sum_{i=1}^{M} B_t(x_i) = \mathbb{F}^n$, ahol $x_i \in \mathcal{C}$, a $B_t(x_i)$ gömbök diszjunktak, és együtt kiadják a teret. Mindegyik $B_t(x_i)$ gömbben $1 + \binom{n}{1} + ... + \binom{n}{t}$ vektor van, az n dimenziós térben viszont 2^n vektor található.

3.1.1. Tétel. (Gömbpakolási-korlát) *Egy t-hibajavító (n, k, d) bináris kód kielégíti ezt az egyenlőtlenséget:*

$$2^k \left(1 + \binom{n}{1} + \ldots + \binom{n}{t}\right) \leq 2^n \tag{3.1}$$

3.1.2. Megjegyzés.

- Ehhez nem kellett a linearitás.
- Definíció alapján ha perfekt a kód, egyenlőség áll fenn.

3.2. Shannon-tétel

3.2.1. Általánosan

1948-ban Claude Shannon kidolgozott egy tételt, amely a kódelmélet egyik alapkövének bizonyult. A tétel azt mondja ki, hogy kódolt üzenet "hiba nélkül" küldhető a csatorna kapacitásán belül.

Tehát minden $\varepsilon > 0$ és minden R kisebb, mint a csatorna kapacitása (ld. (2.3.2)), létezik egy olyan $\mathcal{C}$ kód $k/n \geqslant R$ jelsebességgel (a csatorna kapacitásán belül), amelynek a hibavalószínűsége $P_{\mathcal{C}} \leqslant \varepsilon$. Sajnos a tétel csak a létezést mondja ki, nem ad konkrét kódot.

3.2.1. Tétel. (Shannon-tétel) *Ha* $0 < R < 1 + p\ log_2 p + (1-p)\ log_2(1-p)$, *akkor* $P_{[k,n,p]} \longrightarrow 0$ *ha* $n \longrightarrow \infty$, *ahol* $P_{[k,n,p]}$ *a minimális értéke* $P_{\mathcal{C}}$*-nek, ahol* $P_{\mathcal{C}}$ *minden szóba jöhető* $\mathcal{C}$ *kódnak a hibavalószínűsége.*

A bizonyításhoz szükségünk van néhány lemmára és megjegyzésre:

3.2.2. Megjegyzés. A hibák száma a fogadott kódszóban egy valószínűségi változó. Csak a hibák a darabszámától függ. Binomiális eloszlású, tehát a várható értéke np, a szórása $np(1-p)$. Ha $b := \left(\frac{np(1-p)}{\varepsilon/2}\right)^{1/2}$, akkor a Csebisev egyenlőtlenségből adódik, hogy

$$P(a > np + b) \leqslant \frac{1}{2}\varepsilon \tag{3.2}$$

3.2.3. Lemma. *Mivel* $p < \frac{1}{2}$, *ezért* $\varrho := \lfloor np + b \rfloor$ *kisebb mint* $\frac{1}{2}n$, *ha* n *elég nagy. Jelölje* $B_\varrho(x)$ *azokat a kódszavakat, amik az* x *középpontú,* ϱ *sugarú gömbön belül vannak, azaz amelyekre* $d(x,y) \leqslant \varrho$. *Ekkor*

$$|B_\varrho(x)| = \sum_{i \leqslant \varrho} \binom{n}{i} < \frac{1}{2} n \binom{n}{i} \leqslant \frac{1}{2} n \frac{n^n}{\varrho^\varrho (n-\varrho)^{n-\varrho}} \tag{3.3}$$

3.2.4. Lemma.

$$\begin{gathered} \frac{\varrho}{n}\ log_2\ \frac{\varrho}{n} = \frac{1}{n}\lfloor np+b \rfloor\ log_2\ \frac{\lfloor np+b \rfloor}{n} = p\ log_2\ p + o(n^{-1/2}), \\ \left(1 - \frac{\varrho}{n}\right)\ log_2\ \left(1 - \frac{\varrho}{n}\right) = (1-p)\ log_2\ (1-p) + o(n^{-1/2}), (n \longrightarrow \infty) \end{gathered} \tag{3.4}$$

3.2.5. Megjegyzés. Legyen $u \in \{0,1\}^n, v \in \{0,1\}^n$. Ekkor legyen

$$f(u,v) := \begin{cases} 0 & \text{,ha } d(u,v) > \varrho \\ 1 & \text{,ha } d(u,v) \leqslant \varrho \end{cases} \tag{3.5}$$

3.2.6. Megjegyzés. Legyen $x_i \in C, y \in \{0,1\}^n$. Ekkor legyen

$$g_i(y) := 1 - f(y, x_i) + \sum_{j \neq i} f(y, x_j) \tag{3.6}$$

Érdemes észrevenni, hogy ha csak x_i van az y középpontú, ϱ sugarú gömbben, akkor $g_i(y) = 0$, egyébként $g_i(y) \geqslant 1$.

Bizonyítás. [Shannon-tétel] Feltesszük, hogy a kódszavak egyenlő eséllyel fordulnak elő. A dekódoló megkapja y-t. Ha pontosan egy darab olyan x_i kódszó van, amelyre igaz, hogy $d(x_i, y) \leqslant \varrho$, akkor y-t x_i-nek vesszük. Legyen továbbra is P_i az x_i kódszót küldve a hiba valószínűsége.

$$P_i = \sum_{y \in \{0,1\}^n} P(y|x_i) g_i(y) = \sum_{y \in \{0,1\}^n} P(y|x_i)(1 - f(y, x_i)) + \sum_{y \in \{0,1\}^n} \sum_{j \neq i} P(y|x_i) f(y, x_j) \tag{3.7}$$

Az egyenlet jobb oldalán levő első tag annak a valószínűségét adja meg, hogy y nincs benne az x_i középpontú ϱ sugarú gömbben. Ezt viszont maximalizáltuk, az (3.2) szerint ez maximum $\frac{1}{2}\varepsilon$. Így az összes kódszóra azt kapjuk, hogy

$$P_{\mathcal{C}} \leqslant \frac{1}{2}\varepsilon + \frac{1}{M} \sum_{i=1}^{M} \sum_{y \in \{0,1\}^n} \sum_{j \neq i} P(y|x_i) f(y, x_j). \tag{3.8}$$

$P_{[k,n,p]}$ a minimuma a $P_{\mathcal{C}}$ értékeknek, tehát minden $P_{\mathcal{C}}$ várható értéke nagyobb nála. Ezek szerint

$$\begin{aligned} P_{[k,n,p]} &\leqslant \frac{1}{2}\varepsilon + \frac{1}{M} \sum_{i=1}^{M} \sum_{y \in \{0,1\}^n} \sum_{j \neq i} E(P(y|x_i)) E(f(y, x_j)) \\ &= \frac{1}{2}\varepsilon + \frac{1}{M} \sum_{i=1}^{M} \sum_{y \in \{0,1\}^n} \sum_{j \neq i} E(P(y|x_i)) \frac{|B_\varrho(y)|}{2^n} \\ &= \frac{1}{2}\varepsilon + (M-1)\frac{|B_\varrho(y)|}{2^n} \end{aligned} \tag{3.9}$$

Rendezzük, aztán vegyük a logaritmusát és használjuk a (3.3) és (3.4) összefüggéseket.

$$log(P_{[k,n,p]} - \frac{1}{2}\varepsilon) \leqslant log\left((M-1)\frac{|B_\varrho(y)|}{2^n}\right) =$$

$$= log\left((M-1)\frac{\frac{1}{2}n\frac{n^n}{\varrho^\varrho(n-\varrho)^{n-\varrho}}}{2^n}\right)$$

$$\frac{log(P_{[k,n,p]} - \frac{1}{2}\varepsilon)}{n} \leqslant \frac{log(M-1)}{n} + \left(\frac{n+1}{n}\right) log\ n - \frac{\varrho}{n} log\ \varrho -$$

$$- \left(\frac{n-\varrho}{n}\right) log(n-\varrho) - \left(\frac{n+1}{n}\right) log\ 2 =$$

$$= \frac{log\ M}{n} - \frac{log\left(\frac{M}{M-1}\right)}{n} + \left(\frac{n+1}{n}\right) log\ n - \frac{\varrho}{n} log\frac{\varrho}{n} - \frac{\varrho}{n} log\ n -$$

$$- \left(1-\frac{\varrho}{n}\right) log\left(1-\frac{\varrho}{n}\right) - \left(1-\frac{\varrho}{n}\right) log\ n - \left(\frac{n+1}{n}\right) =$$

$$= \frac{log\ M}{n} + \left(\frac{n+1}{n}\right) log\ n - p\ log\ p - o(n^{-1/2}) - (1-p) log\ (1-p) -$$

$$- log\ n - o(n^{-1/2}) - \left(\frac{n+1}{n}\right) =$$

$$= \frac{log\ M}{n} - (1 + p\ log\ p + (1-p) log\ (1-p)) + \mathcal{O}(n^{-1/2})$$

Helyettesítsük be $M = 2^{Rn}$-t, és használva a tétel kikötését, azt kapjuk, hogy

$$\frac{log(P_{[Rn,n,p]} - \frac{1}{2}\varepsilon)}{n} < -\vartheta < 0 \tag{3.10}$$

egy $n > n_0$ küszöbindextől. Ezt rendezve, és 2 hatványra emelve kijön, hogy $P_{[k,n,p]} < \frac{1}{2}\varepsilon + 2^{-\vartheta n}$. Ezt akartuk belátni. □

3.2.2. Lineáris esetben

A Shannon-tétel megfelelője igaz lineáris kódok esetében is.

3.3. Singleton-korlát

A Singleton-korlát egy felső becslést ad $\mathcal{C}$ kód minimális távolságára.

3.3.1. Tétel. (Singleton-korlát általánosan) *Ha $\mathcal{C}$ egy $[n, M, d]$ kód (nem feltétlenül lineáris), akkor $M \leqslant q^{n-d+1}$.*

3.3.2. Tétel. (Singleton-korlát lineáris esetben) *Ha $\mathcal{C}$ egy lineáris $[n,k,d]$ kód, akkor $d \leqslant n-k+1$.*

Bizonyítás. $r = n-k$ a rangja **H**-nak, azaz ennyi a független oszlopainak maximális száma. Tehát minden $(n-k+1)$ darab oszlop összefügg. Ebből következik, hogy létezik egy $n-k+1$ súlyú kódszó $\mathcal{C}$. Ennél nem lehet nagyobb a d minimális távolság.

□

Ha az egyenlőtlenség pontosan teljesül, akkor **MDS** (maximum distance separable) kódokról beszélünk.

3.4. Plotkin-korlát

3.4.1. Tétel. (Plotkin-korlát általánosan) *Legyen $\mathcal{C}$ egy $[n,M,d]$ kód, ahol $n < 2d$. Ekkor $M \leqslant 2\left[\frac{d}{2d-n}\right]$.*

3.4.2. Tétel. (Plotkin korlát lineáris esetben) *Legyen $\mathcal{C}$ egy $[n,k,d]$ lineáris kód, ahol $d = 2t+1$. Ekkor $\frac{k}{n} \leqslant 1 - 4 \cdot \frac{t}{n}$.*

3.5. Gilbert-Varshamov korlát

3.5.1. Definíció. *Legyen $A(n,d) := max\{M|$ létezik $[n,M,d]$ kód$\}$. Ezeket a kódokat **optimális**nak nevezzük.*

3.5.2. Lemma. *$A(n,d) \geqslant \frac{q^n}{V_q(n,d-1)}$, ahol $V_q(n,r) = |B_r(x)| = \sum_{i=0}^{r} \binom{n}{i}(q-1)^i$.*

Bizonyítás. Legyen $\mathcal{C} = [n,M,d]$ kód **maximális**. Ez azt jelenti, hogy nem bővíthető több szóval úgy a kód, hogy ne sérüljön a $\mathcal{C}$ kód d minimális távolsága. Más szóval: a $B_{d-1}(c)$ gömbök, ahol $c \in \mathcal{C}$ lefedik az egész $\mathbb{F}_q^n$ teret. $q^n = |\cup_{c\in\mathcal{C}} B_{d-1}(c)| \leqslant \sum_{c\in\mathcal{C}} |B_{d-1}(c)| = |\mathcal{C}|\ V_q(n,d-1)$, és mivel $A(n,d) \geqslant |\mathcal{C}|$, átrendezve adódik a lemma. □

3.5.3. Tétel. (Gilbert-Varshamov korlát általánosan) *Ha teljesül az, hogy $V_q(n,d-1) < q^{n-k+1}$, akkor létezik egy $[n,k,d]$ kód.*

Bizonyítás. k szerinti teljes indukcióval bizonyítjuk. A $k = 0$-ra nyilvánvaló. Tegyük fel, hogy igaz $k-1$-re, tehát létezik egy $\mathcal{C}_{k-1} = [n,k-1,d]$ kód. Mivel $|\mathcal{C}_{k-1}| \cdot V(n,d-1) < q^n$ a feltevésünk szerint, ezért ez a kód nem maximális. Így létezik egy $x \in \mathbb{F}_g^n$-beli kódszó,

amely nincs közelebb semelyik kódszóhoz, mint d. Legyen $\mathcal{C}_k$ a $\mathcal{C}_{k-1}$ kiterjesztése $\{x\}$-szel. Ekkor $w = ax + y$ kódszó $\mathcal{C}_k$-beli, ahol az a nemnulla $\mathbb{F}_q$-beli, az y pedig $\mathcal{C}_{k-1}$-beli. Így $w(z) = w(a^{-1}z) = w(x + a^{-1}y) = d(x, -a^{-1}y) \geqslant d$. $\square$

3.5.4. Tétel. (Gilbert-Varshamov korlát lineáris esetben) *Tegyük fel, hogy teljesül a következő egyenlőtlenség:*

$$1 + \binom{n-1}{1} + \ldots + \binom{n-1}{d-2} < 2^{n-k}. \tag{3.11}$$

Ekkor létezik bináris $\mathcal{C} = [n, k, d]$ lineáris kód.

4. fejezet

Néhány lineáris kód

4.0.5. Definíció. *Legyen $\mathcal{C}$ egy $[n,k,d]$ kód.* ***Bővített kód****nak nevezzük a $\tilde{\mathcal{C}}$ $(n+1)$ hosszú kódot az $\mathbb{F}_q^{n+1}$ felett, ha minden $x \in \mathcal{C}$-re teljesül, hogy $\tilde{x} = (x_1, x_2, \ldots, x_n, -\sum_{i=1}^{n} x_i)$.*

Legyen $\mathcal{C}$ kódnak a paritásellenőrző mátrixa $\mathbf{H}$, és generátormátrixa $\mathbf{G}$. Ekkor az új bővített kódunknak a $\tilde{\mathbf{G}}$ generátormátrixát megkapjuk a $\mathbf{G}$-ből, ha hozzáadunk egy $(n+1)$-edik oszlopot úgy, hogy az oszlopösszeg a nullvektor legyen. A paritásellenőrző mátrixot is megkaphatjuk az eredeti $\mathbf{H}$-ból, ha hozzáadva egy csupa egyesből álló $(n-k+1)$-edik sort, és egy $(n+1)$-edik oszlopot, amely $(0,0,\ldots,0,1)^T$ alakú. Ha $\mathcal{C} = [n,k,d]$ kód, páros d minimális távolsággal, akkor a bővített $\tilde{\mathcal{C}}$ kód $[n+1,k,d+1]$ lesz.

4.0.6. Példa. Tegyük fel, hogy létezik $[7,4,3]$ bináris kód. Erre alkalmazva a bővítést $[8,4,4]$ kódot kapunk. Azt, hogy létezik ilyen kód, a következő fejezetben mutatom meg.

4.1. Hamming kódok

A Hamming kódokat Hamming és Golay írta le először, bár már Shannon jegyzeteiben is szerepelt a bináris $[7,4]$ kód. Ők alkották meg a perfekt kódok osztályát.

A Hamming kódok nagyon fontos egyhibajavító kódok, mert könnyű őket kódolni és dekódolni. Mint láttuk a 2.3 bekezdés végén taglalt szindrómáknál, hogy a kapott vektor szindrómája megegyezik a paritásellenőrző mátrix azon oszlopainak összegével, ahol meghibásodott a küldött kódszó. Tehát a $\mathbf{H}$ mátrix oszlopainak nemnulla vektoroknak kell lenniük, különben nem észrevehető a meghibásodás. És nem lehet két oszlopa megegyező, mert megkülönböztethetetlen lenne, hogy hol történt a hiba. Tehát ha $r = n-k$ darab

sora van a $\mathbf{H}$ mátrixnak, akkor maximum $2^r - 1$ különböző oszlopa lehet. A Hamming kódok $\mathbf{H}$ mátrixában az összes oszlopot használjuk, így $n = 2^r - 1$.

4.1.1. Definíció. *A bináris* ***Hamming kód*** $\mathcal{H}_r$ *egy* $[n = 2^r - 1, k = 2^r - 1 - r, d = 3]$ *kód minden* $r \geqslant 2$*, amihez tartozó* $\mathbf{H} \in \mathcal{M}^{r \times 2^r - 1}$ *paritásellenőrző mátrixnak az oszlopai minden nemnulla* r *hosszú különböző vektor.*

Ezek a $\mathcal{H}_r$ kódok családot alkotnak , mert a $\mathbf{H}$ mátrix oszlopainak felcserélésétől nem változik a kód hibajavító képessége, vagy a hibavalószínűsége (ekvivalencia szintjéig egyértelműek). Ezek a kódok mind egyhibajavítóak, tehát az 2.0.6 állítás szerint a minimális távolság az egyes kódszavak között legalább $d \geqslant 3$. De valójában egyenlőség áll fent, mert a kódszavak körülötti diszjunkt gömbökben $1 + n = 2^r$ darab vektor van. Összesen $2^k = 2^{2^r - 1 - r}$ gömb van, s ezek kiadják az egész teret: $2^{2^r - 1 - r} 2^r = 2^n$. Ez azt jelenti, hogy a Hamming kódok perfektek.

4.1.2. Megjegyzés. Két $\mathbb{F}_q^n$-beli lineáris kódra azt mondjuk hogy ekvivalensek, ha egyikből előállíthatjuk a másikat permutálva annak koordinátáit, és/vagy nemnulla testelemmel megszorozva őket.

Ezeken kívül nem sok perfekt kód van.

4.1.1. Alkalmazás

A Hamming kódokat sok helyen használjuk, de a mai digitális világban talán a chipek kódolásánál játsszák a legfontosabb szerepet. A számítógép memória chipje szilíciumból épül, s nagyon megbízható. Ellenben mikor sok ezer chipet kombinálunk a memóriában, már számottevő a meghibásodás esélye. Ezért szeretnénk úgy kódolni, hogy a hibákat észrevegyük, és javítsuk is ki őket. A memória chip egy adattárolási cellákból álló négyzetes tömb. Például a $64Kbit$ chip olyan chip (ahol $K = 2^{10}$), amelyik a $2^{16} = 65536$ bitet, azaz bináris adatot tárol. A $256Kbit$ 2^{18}, a $1Gbit$ 2^{20} cellából álló chip. A $64Kbit$ chip egy $2^8 \times 2^8$ tömb, ahol a különböző cellákban egyesek és nullák lehetnek. Minden cellát külön kezelünk, mindegyikhez külön cím tartozik, azaz különböző sor és oszlop index. $64Kbit$ chip esetén az index 0-tól 255-ig terjed. A legnagyobb cím a $(255, 255) = (2^8 - 1, 2^8 - 1)$, ami bináris számrendszerben 11111111, 8 darab 1-es bit. Tehát a $64Kbit$ chip cellánkénti címzéséhez $8 + 8 = 16$ bitre van szükség. A $256Kbit$ chipnek $2^9 = 512$ sora és oszlopa van, így 18 bitre van szüksége a címzéshez, az $1Gbit$ chipnek pedig 20-ra. A memória chipben

tárolt 0-ásokat és 1-eseket negatív elektromos töltésekkel, és azok hiányával reprezentáljuk. A cella tartalmát 0-nak tekintjük, ha elektronokat tartalmaz (negatív töltés), ellenkező esetben a cella nem tartalmaz elektronokat, tehát a cella tartalmát 1-nek tekintjük. A cella értékének megállapítása a cella töltésének mérésén alapul. Nyilvánvaló, ha a cella elveszti a töltését, hibás lesz az értéke. Két fajta hiba fordulhat elő, hard (nem javítható) és soft (javítható). A nem javítható hiba például fizikai sérülés. Javítható a hiba, amikor maga a chip nem sérült, de alfa részecske sugárzás hatására a cella töltése megváltozik. Ez a fajta hiba gyakori, és nem elkerülhető. Ennek kezelésére alkalmazunk kódolást. Tegyük fel, hogy van $8Mbit$ memóriánk, amely 128 darab $64Kbit$ chipből áll. Ennél a memóriánál négy sorba szervezik a chipeket, soronként tehát 32 chip található. Mindegyik chip tartalmaz 2^{16} memória cellát, ez tehát összesen 2^{23} cellát jelent. A tartalmazott adat 32 bites szavakra van osztva. Minden szó tartalmaz egy-egy bitet egy sor mind a 32 chipjéből. Hibajavítás céljából hozzáadunk további 7 chipet minden sorhoz, így 156 chipet kapunk. Minden sorban 39 chip van, a 7 extra chip tartalmazza a paritásellenőrző biteket. A hibajavításhoz a bővített Hamming $[64, 57, 4]$ kódot használjuk. Ez a kód valójában 57 bit adatot tud védeni, de mi csak 32 bitet használunk ebből.

4.2. Ciklikus kódok

A ciklikus kódokat sok helyen, előszeretettel használják, mert sokféleképp generálhatjuk őket, könnyű dekódolni, és ide tartoznak a nagyon fontos BCH kódok családja. (ld. 4.3.1)

4.2.1. Ciklikus kódok tulajdonságai

4.2.1. Definíció. *Egy $\mathcal{C}$ kód **ciklikus**, ha lineáris és ciklikus, azaz ha $c = (c_0, c_1, \ldots, c_{n-1}) \in \mathcal{C}$, akkor $(c_{n-1}, c_0, \ldots, c_{n-2})$ is eleme a $\mathcal{C}$ kódnak.*

Ahhoz, hogy algebrailag le tudjunk írni a ciklikus kódokat, $c = (c_0, c_1, \ldots, c_{n-1})$-t megfeleltetjük $c(x) = c_0 + c_1x + \ldots + c_{n-1}x^{n-1}$-nek, ahol $c \in \mathbb{F}^n$, $\mathbb{F}$ bármely $\mathbb{GF}(q)$. Ha $\mathbb{F}$ egy test, akkor $\mathbb{F}[x]$ jelenti x-nek azokat a polinomjait, amelyeknek az együtthatói az $\mathbb{F}$-ből vannak. Ez egy gyűrű.

Tekintsük a $R_n = \mathbb{F}[x]/(x^n - 1)$ faktorgyűrűt. Minden maradékosztály egyértelműen reprezentálható egy n-nél kisebb fokú polinommal. Így azt mondhatjuk, hogy $c(x)$ beletartozik R_n-be. R_n vektortér $\mathbb{F}$ felett, n dimenzióval.

Az x-szel való szorzás eltolást eredményez: $xc(x) = c_0x+c_1x^2+\ldots+c_{n-2}x^{n-1}+c_{n-1}x^n = c_{n-1}+c_0x+\ldots+c_{n-2}x^{n-1}$, miután $x^n = 1$ teljesül R_n-ben. R_n-nek lineáris altere $\mathcal{C}$ ideál, ha $c(x) \in \mathcal{C}$ akkor $r(x)c(x)$ is eleme, ahol $r(x) \in R_n$. Ez másképpen azt jelenti, hogy ha $c(x) \in \mathcal{C}$ akkor a $xc(x)$ is az. Ezek után azt mondhatjuk, hogy egy n hosszú ciklikus kód megfelel az R_n az ideáljának. R_n-ben minden ideál főideál. Egy főideálban egy fix ú.n. generátorelem a különböző többszörösei szerepelnek. Tehát a ciklikus kódokat $g(x)$ polinom generálja. Legyen $\mathcal{C}$ egy n hosszú ciklikus kód, ekkor a következő tulajdonságokkal rendelkezik:

- Egyértelműen létezik egy $g(x)$ főpolinom, aminek minimális a fokszáma $\mathcal{C}$-ben. Ugyanis ha két ilyen lenne, a különbségük is $\mathcal{C}$-beli lenne, aminek a fokszáma kisebb. Ez csak úgy nem ellentmondás, ha a két polinom megegyezik.

- $g(x)$ osztja x^n-1-et. Ugyanis $x^n-1 = h(x)g(x)+r(x)$, ahol $deg\ r(x) < r = deg\ g(x)$, akkor $r(x) = -h(x)g(x) \in \mathcal{C}$, ami ellentmondás, kivéve ha $r(x) = 0$.

- Minden $c(x) \in \mathcal{C}$ felírható $c(x) = f(x)g(x)$ $\mathbb{F}[x]$-beliként, ahol $f(x) \in \mathbb{F}[x]$ és a fokszáma $< n-r$, $r = deg\ g(x)$. $\mathcal{C}$ dimenziója $n-r$. Így az $f(x)$ polinomnak megfelel egy $k = n-r$ hosszú üzenet, és ebből lesz az $f(x)g(x)$ kódszó. Mivel q^k darab kódszó van, $r = n-k$.

- Ha $g(x) = g_0 + g_1x + \ldots + g_rx^r$, akkor $\mathcal{C}$-nek generátormátrixa az alábbi mátrix:

$$\mathbf{G} = \begin{pmatrix} g_0 & g_1 & g_2 & \ldots & g_r & & 0 \\ 0 & g_0 & g_1 & \ldots & g_{r-1} & g_r & 0 \\ & & & \ldots & \ldots & & 0 \\ 0 & & & g_0 & \ldots & \ldots & g_r \end{pmatrix}$$

$$= \begin{pmatrix} g(x) & & & & \\ & xg(x) & & & \\ & & & \ldots & \\ & & & & x^{n-r-1}g(x) \end{pmatrix}$$

  Mivel $n-r = k$ lineárisan független többszöröse van $g(x)$-nek: $g(x), xg(x), \ldots, x^{k-1}g(x)$, ezért a kód dimenziója k.

4.2.2. Megjegyzés. Egy másik generálási módszer **szisztematikus**. Itt a generátormátrix bal oldali blokkjában egy $k \times k$ méretű egységmátrix áll. A jobb oldali blokkot úgy

kapjuk meg, hogy kiszámítjuk a x^{r+i} modulo $g(x)$ maradékjait $i = 0, 1, \ldots, k-1$-re. Ezeket (felülről lefelé haladva) beírjuk a jobb oldali részmátrixba. Ld 4.2.5.

4.2.3. Tétel. *A bináris* $\mathcal{H}_m = [n = 2^m - 1, k = n - m, d = 3]$ *Hamming kód ciklikus,* $\mathbf{H} = (1, \alpha, \alpha^2, \ldots, \alpha^{2^m-2})$, *ahol* α *egy primitív eleme* $\mathbb{GF}(2^m)$*-nak és a generátorpolinomja* $g(x) = m_{(1)}(x)$, *ahol* $m_{(1)}(x)$ *jelenti az* α^1 *minimálpolinomját.*

4.2.4. Megjegyzés. A **primitív elem** generálja a $\mathbb{GF}(2^m)$ multiplikatív csoportját, amely ciklikus. Például $\mathbb{GF}(2^4)$-nek az $\alpha = 0100$:

Construction of the field GF(16)

as a 4-tuple	as a polynomial	as a power of α	logarithm
0000	0	0	$-\infty$
1000	1	1	0
0100	α	α	1
0010	α^2	α^2	2
0001	α^3	α^3	3
1100	$1+\alpha$	α^4	4
0110	$\alpha+\alpha^2$	α^5	5
0011	$\alpha^2+\alpha^3$	α^6	6
1101	$1+\alpha+\alpha^3$	α^7	7
1010	$1+\alpha^2$	α^8	8
0101	$\alpha+\alpha^3$	α^9	9
1110	$1+\alpha+\alpha^2$	α^{10}	10
0111	$\alpha+\alpha^2+\alpha^3$	α^{11}	11
1111	$1+\alpha+\alpha^2+\alpha^3$	α^{12}	12
1011	$1+\alpha^2+\alpha^3$	α^{13}	13
1001	$1+\alpha^3$	α^{14}	14

GF(2^4) generated by $\alpha^4 + \alpha + 1 = 0$.

4.2.5. Példa. $\mathcal{H}_4 = [15, 11, 3]$ paritásellenőrző mátrixa $\mathbf{H}$.

$\mathbf{H} = (1, \alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6, \alpha^7, \alpha^8, \alpha^9, \alpha^{10}, \alpha^{11}, \alpha^{12}, \alpha^{13}, \alpha^{14})$

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

ahol $\alpha \in \mathbb{GF}(2^4)$ és kielégíti az $\alpha^4 + \alpha + 1 = 0$ egyenletet. Például $0001 \cdot 1101 = x^3(1 + x + x^3) = x^6 + x^4 + x^3$. Ennek a kapott polinomnak a fokát szeretnénk lecsökkenteni $\leqslant$ 3-ra. Ekkor használjuk a feltett egyenlőségünket, és így kapjuk: $x^2(x^4 + x + 1) + x^4 + x^2 = x^2 + x + 1 = 1110$. Ami tényleg igaz, mert $\alpha^3 \cdot \alpha^7 = \alpha^{10} = 1110$.

$c = (c_0, c_1, \ldots, c_{n-1})$ vektor kódszó akkor és csak akkor, ha $\mathbf{H}c^T = 0 \Leftrightarrow \sum_{i=0}^{n-1} c_i \alpha^i = 0 \Leftrightarrow c(\alpha) = 0$. A minimálpolinom egyik tulajdonsága, hogy akármilyen $f(x)$ polinom, amelynek

az együtthatói $\mathbb{GF}(2)$-beliek és $f(\alpha)=0$, akkor $m(x)|f(x)$. Ebből adódik, hogy $c\in\mathcal{H}_m$ akkor és csak akkor, ha $m_{(1)}(x)$ minimálpolinom osztja $c(x)$-et. Ebből következik, hogy $\mathcal{H}_m$ generátorpolinomja $g(x)=m_{(1)}(x)$, azaz

$$\mathbf{G}=\begin{pmatrix} m_{(1)}(x) & & & & & & \\ & xm_{(1)}(x) & & & & & \\ & & x^2m_{(1)}(x) & & & & \\ & & & \dots & \dots & & \\ & & & & & x^{n-m-1}m_{(1)}(x) \end{pmatrix}.$$

Példánkban:

$$\mathbf{G}=\begin{pmatrix} 1 & 1 & 0 & 0 & 1 & & & & & & & & & & \\ & 1 & 1 & 0 & 0 & 1 & & & & & & & & & \\ & & & & & \dots & \dots & \dots & & & & & & & \\ & & & & & & & & 1 & 1 & 0 & 0 & 1 \end{pmatrix},$$

ahol $\mathbf{G}$ egy 11×15 méretű mátrix. Szisztematikus módszerrel modulo $(1+x+x^4)$:

$x^4=1+x$

$x^5=x(x^4+x+1)-x^2-x=x^2+x$

$\vdots$

$x^{13}=x^9(x^4+x+1)-x^{10}-x^9=(x^6-x^5)(x^4+x+1)-x^7-x^6+x^6+x^5=(-x^3+x)(x^4+x+1)+x^4+x^3-x^2-x=x^3+x^2+1$

$x^{14}=x^{10}(x^4+x+1)-x^{11}-x^{10}=(x^7-x^6)(x^4+x+1)-x^8-x^7+x^7-x^6=(x^4-x^2)(x^4+x+1)-x^5-x^4+x^3+x^2=x(x^4+x+1)-x+x^3+x^4=x^3+1$. Ezek alapján:

$$\mathbf{G}=\left(\begin{array}{ccccccccc|cccc} 1 & 0 & \dots & & & & & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & & \dots & & & 0 & 0 & 1 & 1 & 0 \\ & & & & \dots & \dots & \dots & & & \dots & \dots & \\ & & & & & & & 1 & 0 & 1 & 0 & 1 & 1 \\ & & & & & & & 1 & 1 & 0 & 0 & 1 \end{array}\right)$$

4.2.6. Definíció. *Legyen $\mathcal{C}$ egy ciklikus kód, $g(x)$ generátorpolinommal. Ekkor $h(x)=(x^n-1)/g(x)=\sum_{i=0}^{k}h_ix^i$, ahol $h_k\neq 0$. Ezt* ***paritásellenőrző polinom****nak nevezzük.*

4.2.7. Tétel. *Egy $c(x)$ polinom akkor és csak akkor kódszó, ha $c(x)h(x)=0$.*

Bizonyítás. Legyen $c(x) = f(x)g(x)$. Ekkor $c(x)h(x) = f(x)g(x)h(x) = 0$, miután $h(x)g(x) = 0$. Visszafele: ha $c(x)h(x) = 0\ mod(x^n - 1)$, akkor $c(x)h(x) = u(x)(x^n - 1)$. Ezt átrendezve $c(x) = u(x)(x^n - 1)/h(x) = u(x)g(x)$ kapjuk, tehát $c(x)$ kódszó. □

A $c(x)h(x)$ szorzatban szereplő x^j együtthatói: $\sum_{i=0}^{n-1} c_i h_{j-i} = 0,\ j = 0, 1, \ldots, n-1$, ahol az alsó indexet modulo n vesszük. Így

$$\mathbf{H} = \begin{pmatrix} & & & h_k & \ldots & h_2 & h_1 & h_0 \\ & & h_k & \ldots & h_2 & h_1 & h_0 & \\ & & & \ldots & \ldots & & & \\ h_k & \ldots & h_2 & h_1 & h_0 & & & \end{pmatrix}$$

$$= \begin{pmatrix} & & & & & \overleftarrow{h(x)} \\ & & & & \overleftarrow{xh(x)} & \\ & & & \ldots & & \\ x^{n-k-1}\overleftarrow{h(x)} & & & & & \end{pmatrix}$$

4.2.8. Példa. (4.2.5) példánkhoz visszatérve paritásellenőrző polinomunk $h(x) = (x^15 + 1)/(x^4 + x + 1) = x^{11} + x^8 + x^7 + x^5 + x^3 + x^2 + x + 1$, így

$$\mathbf{H} = \begin{pmatrix} & & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\ & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & & \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & & \end{pmatrix}.$$

Ez megegyezik (4.2.5) példában leírt $\mathbf{H}$ paritásellenőrző mátrixszal. (Az oszlopok más sorrendben vannak.)

4.2.2. Kódolás

Legyen adva az $u(x)$ polinommal meghatározott k hosszúságú (max $(k-1)$-ed fokú polinommal leírt) üzenet, és a $g(x)$ generátorpolinom. Keressük a $c(x)$ kódszót, kódpolinomot.

A nem szisztematikus esetben már láttuk, hogy $c(x) = g(x)u(x)$.

Szisztematikus kód esetén a kódszó első k helyén az üzenet, azaz $x^{n-k}u(x)$ áll, és ezt követi a paritás rész, $c(x) = x^{n-k}u(x) + p(x)$. Mivel a kódszónak oszthatónak kell lennie $g(x)$-szel, így $x^{n-k}u(x) + p(x) = 0$ modulo $g(x)$. Tekintve, hogy $p(x)$ maximum $(n-k-1)$-ed fokú, $g(x)$ pedig $(n-k)$-ad fokú, így $p(x) = x^{n-k}u(x)$ modulo $g(x)$. Ebből azt kapjuk, hogy a kódszónk $c(x) = x^{n-k}u(x) + \left[x^{n-k}u(x)\right]$ modulo $g(x)$.

4.2.3. Mellékosztály

A ciklikus kódoknak is vannak mellékosztályai. Ehhoz először meg kell nézni az $(x^n - 1)$ szorzatra való bontását $\mathbb{GF}(q^m)$ felett. Feltesszük, hogy n és q relatív prímek, tehát n páratlan bináris esetben. $(x^n - 1)$-nek n darab különböző gyöke van: $x^n - 1 = \prod_{i=0}^{n-1}(x - \alpha_i)$. Ha ez az α egy primitív eleme $\mathbb{GF}(q^m)$-nek, akkor a következőre módosul az előző egyenlet: $x^n - 1 = \prod_{i=0}^{n-1}(x - \alpha^i)$. A testelemeket, amelyeknek ugyanaz a minimálpolinomja **konjugáltak**nak hívjuk. Például i és $-i$ konjugáltak, $x^2 + 1$ minimálpolinommal, a valós test felett. Nézzük most a $\mathbb{GF}(q^4)$-t. $\alpha, \alpha^2, (\alpha^2)^2 = \alpha^4, (\alpha^4)^2 = \alpha^8$ (és $(\alpha^8)^2 = \alpha$ megint) ugyanaz a minimálpolinomja. Ez a minimálpolinom egyik tulajdonságából következik, miszerint β-nak és β^2-nek ugyanaz a minimálpolinomja $\mathbb{GF}(2^m)$ felett. Ugyanígy $\alpha^3, \alpha^6, \alpha^12, \alpha^24) = \alpha^9$ (és $\alpha^18 = \alpha^3$ megint), és így tovább. Ezek a hatványkitevői α-nak diszjunkt osztályokat alkotnak. Ezeket **ciklikus mellékosztály**nak nevezzük. Egy ciklikus mellékosztály, amely tartalmazza s-et : $C_s = \{s, qs, q^2s, \ldots, q^{m_s - 1}s\}$ alakú, ahol m_s a legkisebb egész, amelyre igaz, hogy $sq^{m_s} \equiv s \bmod (q^m - 1)$. Az s-et a mellékosztály **reprezentáns**ának hívjuk. $m = m_1$, amely C_1 elemszámát jelöli.

Például a ciklikus mellékosztályok $n = 2^4 - 1 = 15,\ q = 2$-re:

$C_0 = \{0\}$

$C_1 = \{1, 2, 4, 8\}$

$C_3 = \{3, 6, 12, 9\}$

$C_5 = \{5, 10\}$

$C_7 = \{7, 14, 13, 11\}$

Így $m = 4$, és $x^{15} - 1$ felbomlik $\mathbb{GF}(2^4)$ felett. Ekkor α^s-nek a minimálpolinomja $m_{(s)}(x) = \prod_{i \in C_s}(x - \alpha^i)$ és $x^n - 1 = \prod_s m_{(s)}(x)$, ahol s végig fut az összes mellékosztály reprezentánson modulo n.

Visszatérve a példánkhoz, $x^{15} - 1 = m_{(0)}(x)m_{(1)}(x)m_{(3)}(x)m_{(5)}(x)m_{(7)}(x)$.

4.3. További példák

4.3.1. BCH kódok

A BCH kódokat 1959-ben találta ki Alexis Hocquenghem, francia matematikus, és tőle függetlenül 1960-ban Raj Bose és D. K. Ray-Chaudhuri. Az ő neveik kezdőbetűiből kapta a kód a nevét. A BCH kódok nagy előnye, hogy tervezésüknél pontosan meg tudjuk mondani, hogy legalább hány hibát javítson a kód. Másik előnyük, hogy gyorsan lehet őket dekódolni szindróma segítségével. Ez lehetővé teszi a kis teljesítményű hardware használatát.
BCH kódokat főként CD-k és DVD-k lejátszásánál, merev lemez és SSD-k futtatásánál, kétdimenziós bar kódok leolvasásánál és szatellit kommunikációnál használják.

4.3.1. Definíció. *Egy n hosszú ciklikus kód $\mathbb{GF}(q)$ felett **BCH kód** δ tervezett távolsággal, ha pozitív egész b-re a generátorpolinomja: $g(x) = lkkt\{m_{(b)}(x)m_{(b+1)}(x)\dots m_{(b+\delta-2)}(x)\}$.*

$g(x)$ a legkisebb fokú főpolinom $\mathbb{GF}(q)$ felett, aminek az $\alpha^b, \alpha^{b+1}, \dots, \alpha^{b+\delta-2}$ a gyöke. Ezért c akkor és csak akkor eleme a kódnak, ha $c(\alpha^b) = c(\alpha^{b+1}) = \dots, c(\alpha^{b+\delta-2}) = 0$. Ebből adódik, hogy a "paritásellenőrző mátrixa" ($\mathbb{GF}(q^m)$ felett)

$$\mathbf{H} = \begin{pmatrix} 1 & \alpha^b & \alpha^{b+1} & \dots & \alpha^{(n-1)b} \\ 1 & \alpha^{b+1} & \alpha^{2(b+1)} & \dots & \alpha^{(n-1)(b+1)} \\ & \dots & \dots & \dots & \\ 1 & \alpha^{b+\delta-2} & \alpha^{2(b+\delta-2)} & \dots & \alpha^{(n-1)(b+\delta-2)} \end{pmatrix},$$

ahol minden elem helyére megfelelő m magasságú oszlopvektort írjuk a $\mathbb{GF}(q)$ felett. (m az előző részben volt definiálva.) Miután lecseréltük őket, a sorok lesznek a kód paritásellenőrző egyenletei. Így $m(\delta-1)$ darab egyenlet van, de ezek nem biztos, hogy lineárisan függetlenek. Tehát a kód dimenziója legalább $n - m(\delta-1)$, és a minimális távolsága legalább δ.

4.3.2. Megjegyzés. Ha $n = q^m - 1$, akkor **primitív BCH kód**nak hívjuk.

4.3.3. Példa. Mint már láttuk, a bináris $\mathcal{H}_m$ Hamming kód generátorpolinomja $m_{(1)}(x)$. $m_{(1)}(\alpha) = m_{(1)}(\alpha^2) = 0$ adódik a minimálpolinom tulajdonságából. Tehát α és α^2 egymást követő gyökei, így a kód minimális távolsága legalább 3.

designed distance δ	generator polynomial $g(x)$	exponents of roots of $g(x)$	dimension $= n - \deg g(x)$	actual distance d
1	1	–	15	1
3	$M^{(1)}(x)$	1, 2, 4, 8	11	3
5	$M^{(1)}(x)M^{(3)}(x)$	1–4, 6, 8, 9, 12	7	5
7	$M^{(1)}(x)M^{(3)}(x)M^{(5)}(x)$	1–6, 8–10, 12	5	7
9, 11, 13 or 15	$M^{(1)}M^{(3)}M^{(5)}M^{(7)} = (x^{15}+1)/(x+1)$	1–14	1	15

BCH codes of length 15.

A következő példában a minimális távolság ténylegesen nagyobb, mint a tervezett. Ez a bináris (nem primitív) BCH kód $n = 23$ hosszú. Ciklikus mellékosztályai:
$C_0 = \{0\}$
$C_1 = \{1, 2, 4, 8, 16, 9, 18, 13, 3, 6, 12\}$
$C_5 = \{5, 10, 20, 17, 11, 22, 21, 19, 15, 7, 14\}$
Miután $|C_1| = 11$, 2-nek a modulo 23 multiplikatív rendje 11. Így $x^{23}+1$ szorzatokra bomlik $\mathbb{GF}(2^{11})$ felett, és α a primitív eleme. $\mathbb{GF}(2)$ felett is szorzatokra bomlik: $x^{23}+1 = (x+1)m_{(1)}m_{(5)}$, ahol $m_{(1)} = x^{11}+x^9+x^7+x^6+x^5+x+1$ és $m_{(5)} = x^{11}+x^{10}+x^6+x^5+x^4+x^2+1$.

designed distance δ	generator polynomial $g(x)$	dimension $= n - \deg g(x)$	actual distance d
1	1	23	1
3 or 5	$M^{(1)}$	12	7
7, 9, . . . , 23	$M^{(1)}M^{(5)}$	1	23

BCH codes of length 23.

Ez a kód a $\mathcal{G}_{23} = [23, 12, 7]$ Golay kód.

4.3.2. Golay kódok

A kódok egy különös osztálya a Golay kódok. 4 különböző kód tartozik bele: $\mathcal{G}_{23} = [23, 12, 7]$, $\mathcal{G}_{24} = [24, 12, 8]$ bináris kódok, és $\mathcal{G}_{11} = [11, 6, 5]$, $\mathcal{G}_{12} = [12, 6, 6]$ a 3 elemű

test feletti kódok. Nézzük meg a két bináris kódot kicsit közelebbről. Ezeket a kódokat különböző módszerekkel állíthatjuk elő. Például generátormátrixszal, vagy esetleg a $\mathcal{G}_{23}$ kódból $\mathcal{G}_{24}$-et csinálhatunk, vagy fordítva. Most a generátormátrixszal adom meg a $\mathcal{G}_{24}$-et: $\mathbf{G} = [\mathbf{I}_{12}|B]$, ahol $\mathbf{I}_{12}$ a 12×12 méretű egységmátrix, és

$$B = \begin{pmatrix} 0 & 1 & \dots & 1 \\ 1 & & & \\ \vdots & & A & \\ 1 & & & \end{pmatrix}$$

ahol az A egy 11×11 méretű $\{0,1\}$ mátrix. Az i-edik helyre 1-est írunk, ha az i kvadratikus maradék modulo 11, különben 0-át. Ezek a négyzetszámok modulo 11-ben a $0, 1, 3, 4, 5, 9$ számok. Tehát A első sora: $(1, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0)$. Ahhoz, hogy a maradék sorokat megkapjuk, az első sort el kell tolni balra. Ezt 10-szer megismételjük. Tehát

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

A $\mathcal{G}_{24}$ kód szép tulajdonsága, hogy minden kódszó súlya osztható 4-gyel. Jelölje α_i a darabszámát az i súlyú kódszavaknak. Ekkor:

i:	0	8	12	16	24
α_i:	1	759	2576	759	1

A $\mathcal{G}_{23}$ kódot megkaphatjuk, ha $\mathcal{G}_{24}$ kód valamelyik koordinátáját töröljük.

i:	0	7	8	11	12	15	16	23
α_i:	1	253	506	1288	1288	506	253	1

4.3.4. Tétel. *Legyen $\mathcal{C}$ egy bináris kód, aminek a hossza $n = 24$ és a minimális távolsága $d = 8$. Ekkor $|\mathcal{C}| \leqslant 2^{12}$. Ha $|\mathcal{C}| = 2^{12}$, akkor $\mathcal{C}$ ekvivalens $\mathcal{G}_{24}$-gyel.*

4.3.5. Tétel. *A 3 hibajavító $\mathcal{G}_{23}$ kód perfekt.*

Bizonyítás. Teljesíti a gömbpakolási-korlátot:
$\frac{2^{23}}{1+\binom{23}{1}+\binom{23}{2}+\binom{23}{3}} = \frac{2^{23}}{2^{11}} = 2^{12}$ $\square$

4.3.6. Következmény. Minden $(23, 2^{12}, 7)$ bináris kód ekvivalens $\mathcal{G}_{23}$-mal.

A Voyager–1 és a Voyager–2 űrszondák már színes képeket közvetítettek a Jupiterről és a Szaturnuszról 1979-ben és 1980-ban. A színes képek háromszoros adatmennyiséget jelentenek, így a Golay $[24, 12, 8]$ kód került felhasználásra. Ez a Golay kód csak 3 hibát képes javítani, viszont magasabb adat rátát enged meg az átvitel folyamán.

4.3.3. CRC kód

A CRC (Cyclic Renundancy Check) kódokat a hibajelzés területén használjuk. Ezt az eljárást az IEEE 802.3 (Ethernet) szabvány írja le. Az Internet Protocol is ezt használja. A **CRC kódok** bináris, ciklikus kódok, amelyeket szisztematikus módon generálunk a generátorpolinomok segítségével. Ha a dekódoló hibát észlel, értesíti a küldőt, amely ezután megismétli a kódolt üzenetet. A teljesítmény és a sebesség olyan mértékben nő folyamatosan, hogy nem kell spórolni a küldések számával. Ezt az eljárást Automatic Repeat reQuestnek (**ARQ**) nevezzük.

A következő C program egy CRC-32 generátort mutat be.

```
   #include <stdio.h>

int main();
unsigned long getcrc();
void crcgen();

unsigned long crcTable[256];
/*****************************************************************************/
int main( argc, argv )
int argc;
char *argv[ ];
{
int i;
FILE   *fp;
unsigned long crc;

crcgen();
if (argc < 2)
{
crc = getcrc( stdin );
printf("crc32 = %08lx for <stdin>\n", crc);
} else {
for (i=1; i<argc; i++) {
if ( (fp=fopen(argv[i],"rb")) == NULL ) {
printf("error opening file \"%s\"!\n",argv[i]);
} else {
crc = getcrc( fp );
printf("crc32 = %08lx for \"%s\"\n",
crc, argv[i]);
fclose( fp );
}
}
}
return( 0 );
```

```
}
/*****************************************************************************/
unsigned long getcrc( fp )
FILE *fp;
{
register unsigned long crc;
int c;

crc = 0xFFFFFFFF;
while( (c=getc(fp)) != EOF ) {
crc = ((crc>>8) & 0x00FFFFFF) ^ crcTable[ (crc^c) & 0xFF ];
}
return( crc^0xFFFFFFFF );
}

/*****************************************************************************/
void crcgen( )
{
unsigned long crc, poly;
int i, j;

poly = 0xEDB88320L;
for (i=0; i<256; i++) {
crc = i;
for (j=8; j>0; j--) {
if (crc&1) {
crc = (crc >> 1) ^ poly;
} else {
crc >>= 1;
}
}
crcTable[i] = crc;
}
}
```

Irodalomjegyzék

[1] J.H. van Lint, *Introduction to Coding Theory*, Springer, Third Edition (ábra)

[2] Györfi László, Győri Sándor és Vajda István, *Információ- és Kódelmélet*, Typotex Kiadó, 2000

[3] Jacobus H.van Lint és Gerard van der Geer, *Introduction to Coding Theory and Algebraic Geometry*, 1988

[4] Proof of Shannon's Theorem, refer to Robert Gallager. Information Theory and Reliable Communication. John Wiley and Sons, Inc. 1968

[5] F.J. MacWilliams and N.J.A Sloane. The Theory of Error-Correcting Codes. North-Holland, Publishing Company. 1978 (ábra)

[6] D. R. Shier and K. T. Wallenius (Eds.), *Applied Mathematical Modeling: A Multidisciplinary Approach*, Chapman and Hall/CRC Press, Boca Raton, FL, 1999

[7] Enes Pasalic, *Coding Theory and Applications*, University of Primorska, Koper, 2013 (ábra)

[8] Gonda János, *Hibakorlátozás*, Budapest, 2007

[9] *Chapter 3, Linear Codes*, www.mth.msu.edu/ jhall/classes/codenotes/linear.pdf

[10] http://aix1.uottawa.ca/ jkhoury/coding.htm (program)