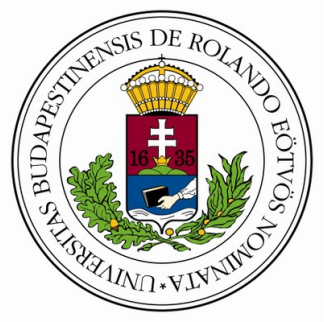
Eötvös Loránd Tudományegyetem
Természettudományi Kar

LDPC kódok

Szakdolgozat

Pongráczné Gazdag Brigitta
Matematika BSc
Matematikai elemző szakirány

Témavezető:
Héger Tamás
Tudományos munkatárs
Számítógéptudományi Tanszék



Budapest, 2019

Tartalomjegyzék

Köszönetnyilvánítás

Szeretnék köszönetet mondani témavezetőmnek, Héger Tamásnak a szakdolgozatom készítése során nyújtott minden segítségért.

Továbbá köszönöm a rengeteg segítséget és támogatást férjemnek, aki nélkül nem tudtam volna befejezni a tanulmányaimat.

1. Bevezetés

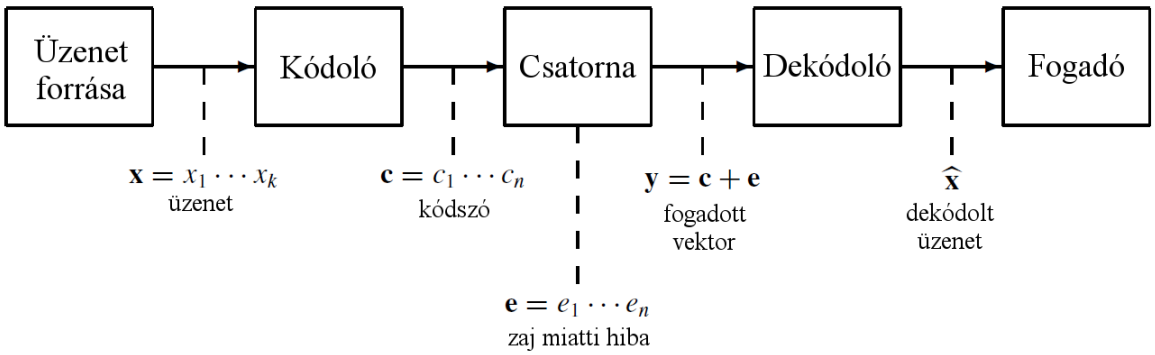
A hibajavító kódolás a számítástudomány egy mára önállóvá vált ága. A hírközlésben és az informatikában fontos feladat a továbbított információ hibáinak felismerése, és - lehetőség szerint - a hibák javítása.

Minden üzenet, amit egy rendszer egy másiknak továbbít, szükségképpen áthalad valamilyen csatornán, azaz egy üzenettovábbításra szolgáló rendszeren. Azonban ha a csatorna zajos, akkor kisebb a kapacitása, vagyis kevesebb információ jut át rajta sérülés nélkül. Ekkor az üzenetünket ugyanis torzíthatja a zaj, ami valójában nem más, mint az üzenethez adódó felesleges, véletlenszerű jelek összessége.

A hibajavító kódolás feladata egyrészt ezeket a sérüléseket minimalizálni, másrészt az információ célba érkezése után a lehető legpontosabban és leggyorsabban rekonstruálni az eredeti üzenetet. Éppen ezért a hibajavító kódok meghatározó tulajdonságai a kódolás és a dekódolás műveletigénye, valamint a hibafelismerő és hibajavító képességük. Elvárjuk tőlük, hogy relatíve sok hibát kiküszöböljenek úgy, hogy az üzenet hosszát ne növeljék meg túlságosan. Emellett a kódnak mindig igazodnia kell a csatornához, hiszen különböző csatornáknak más-más jellemző hibáik lehetnek. Ilyen tipikus hiba lehet például két egymás melletti jel felcserélése, vagy a hibák egy helyre csoportosulása.

Az algebrai kódelmélet mára a hétköznapi élet részévé vált. Hibajavító kódokat használunk, mikor CD-n, vagy merevlemezen adatot tárolunk el. Felhasználja őket a televíziós és rádiós műsorszórás, a mobiltelefonok, GPS-ek, QR kódok. Alkalmazzuk gyakorlatilag minden olyan területen, ahol elektronikus úton adatot továbbítunk.

1. ábra. A kódolt információ útja a feladótól a vevőig [3]



A dolgozat megírásához felhasznált szakirodalom sorszámai az egyes fejezetek elején vannak feltüntetve, illetve a felhasznált ábrák címében külön is.

A 2. fejezet tételei és definíciói a [15] és [9] jegyzetekből származnak. A kapacitás kiszámítása saját munka, a kölcsönös információ deriválását a https://www.wolframalpha.com/ segítségével végeztem el.

A 3. fejezet Általános fogalmak alfejezet tételeihez és definícióihoz a [13] és [11] forrásokat használtam. A fejezet további részében a [14], [10] és [6] forrásokra támaszkodom.

A 4. fejezetben a felsorolt szabványok a [12] cikkben vannak listázva. A bevezetés és a definíciók a [1], [2] és [4] alapján íródtak.

Az 5. fejezethez a [1], [2] és [6] cikkeket használtam fel.

A 6. fejezet az egyes alfejezetekben megjelölt forrásokat követik.

A 7. fejezet a példákkal és ábrákkal együtt a [3] könyvet és a [4] cikket követi. A fejezet bevezetéséhez felhasználtam még a [1] és [2] forrásokat.

A 8. fejezet a [4] cikk illetve a Csapdák alfejezet a [7] cikk felhasználásával íródott.

2. Csatorna kapacitása

Amikor adatot továbbítunk egy csatornán keresztül, akkor szeretnénk, ha az információ célba is érne. Azonban a valóságban a kommunikációs csatornák olyanok, hogy a rajtuk áthaladó adatok egy része elveszhet, vagy módosulhat.

Shannon 1948-ban publikált munkájában [1] bebizonyította, hogy minden csatornának létezik úgynevezett kapacitása, mely megmondja, hogy legfeljebb mennyi információt képes megbízhatóan továbbítani.

Shannon azt is megmutatta, hogy az adatvesztés minimalizálható, ha ügyesen kódoljuk az üzenetet. Ilyen „ügyes" kódok az LDPC kódok is, melyek segítségével úgy közelíthetjük meg a kapacitást, hogy közben a kódolás és a dekódolást is viszonylag hatékonyan elvégezhető művelet marad.

A Shannon-tétel kimondásához először felidézünk néhány alapfogalmat a valószínűségszámítás témaköréből. [15] [9]

2.1. Definíció.
Legyen $(\Omega, \mathcal{A}, P)$ valószínűségi mező. Az $X : \Omega \to \mathbb{R}$ függvény valószínűségi változó, ha

$$\{\omega \in \Omega : X(\omega) \leq x\} \in \mathcal{A}$$

minden $x \in \mathbb{R}$.
X diszkrét valószínűségi változó, ha értékkészlete diszkrét.
Az X eloszlását egyértelműen meghatározzák a $P(X = x_i) = P(x_i)$ valószínűségek, ahol $i = 1, 2, ...$ értékeket vesz fel.

2.2. Definíció.
Legyenek X, Y diszkrét valószínűségi változók, x, y események. Ekkor

[1] Claude E. Shannon: A Mathematical Theory of Communication. Bell System Technical Journal. **27**(3), 379–423, 1948. július.

1. $P_{X,Y}(x,y) = P(X = x, Y = y)$ az együttes eloszlás,
2. $P_X(x) = P(X = x)$ és $P_Y(y) = P(Y = y)$ a peremeloszlások,
3. $P_{Y|X}(y,x) = P(Y = y|X = x)$ és $P_{X|Y}(x,y) = P(X = x|Y = y)$ a feltételes eloszlások.

2.1. Tétel.
Legyenek X, Y diszkrét valószínűségi változók, ekkor

$$P_{X,Y}(x,y) = P_{Y|X}(y,x)P_X(x) = P_{X|Y}(x,y)P_Y(y).$$

A Shannon-tétel kimondásához szükségünk van néhány új fogalomra is.

2.3. Definíció.
Legyen I a bemeneti ábécé, O a kimeneti ábécé, $i \in I$ és $o \in O$ betűk. Legyen $P(i,o)$ annak a valószínűsége, hogy o megérkezett a címzetthez feltéve, hogy i-t elküldtük. Az (I, O, P) hármast csatornának nevezzük.

2.4. Definíció.
Legyen X a bemenetet, Y a kimenetet jelentő valószínűségi változó. A bináris szimmetrikus csatorna (BSC, azaz Binary Symmetric Channel) olyan csatorna, melyre $I = O = \{0,1\}$, hibavalószínűsége $0 \leq p \leq \frac{1}{2}$, és teljesülnek a következők:

1. $P_{Y|X}(0,0) = P_{Y|X}(1,1) = 1 - p$,
2. $P_{Y|X}(0,1) = P_{Y|X}(1,0) = p$.

2.5. Definíció.
Az X és Y diszkrét valószínűségi változók kölcsönös információja

$$I(X,Y) = \sum_{y \in Y} \sum_{x \in X} P_{X,Y}(x,y) \log \frac{P_{X,Y}(x,y)}{P_X(x)P_Y(y)}.$$

A log jelölés a természetes alapú logaritmust jelenti. (Választhatjuk másnak is a logaritmus alapját, ez csak egy konstans szorzóbeli eltérést eredményez.)

2.2. Tétel (Shannon-tétel)**.**
Legyen X a bemenetet, Y a kimenetet jelentő valószínűségi változó. Egy kommunikációs csatorna kapacitása, vagyis az időegység alatt a csatornán maximálisan továbbítható információ

$$C = \sup_{P_X(x)} I(X,Y).$$

2.1. Példa.
A Shannon-tétel alapján kiszámítható a bináris szimmetrikus csatorna kapacitása.

A kapacitáshoz X és Y kölcsönös információjának szuprémumát kell meghatároznunk, amihez először az együttes eloszlásokra van szükség.

Legyen $0 \leq r \leq 1$ annak a valószínűsége, hogy a bemenet $X = 0$, ekkor $r - 1$ valószínűséggel lesz $X = 1$. Tehát

$$P_X(0) = r \quad \text{és} \quad P_X(1) = 1 - r.$$

A feltételes eloszlások ismertek a bináris szimmetrikus csatorna definíciója miatt, így a 2.1 tétel alapján az együttes eloszlások

1. $P_{X,Y}(0,0) = r(1-p)$,
2. $P_{X,Y}(0,1) = rp$,
3. $P_{X,Y}(1,0) = (1-r)p$,
4. $P_{X,Y}(1,1) = (1-r)(1-p)$.

Ebből a P_Y peremeloszlás

1. $P_Y(0) = r(1-p) + (1-r)p = r - rp + p - rp = r - 2rp + p$
2. $P_Y(1) = rp + (1-r)(1-p) = rp + 1 - p - r + rp = 1 - r + 2rp - p$.

Az előzőeket behelyettesítve a kölcsönös információ

$$\begin{aligned}
&I(X,Y) = \\
&= r(1-p)\log\left(\frac{r(1-p)}{r(r-2rp+p)}\right) + (1-r)p\log\left(\frac{(1-r)p}{(1-r)(r-2rp+p)}\right) + \\
&+ rp\log\left(\frac{rp}{r(1-r+2rp-p)}\right) + (1-r)(1-p)\log\left(\frac{(1-r)(1-p)}{(1-r)(1-r+2rp-p)}\right) = \\
&= r(1-p)\log\left(\frac{1-p}{r-2rp+p}\right) + (1-r)p\log\left(\frac{p}{r-2rp+p}\right) + \\
&+ rp\log\left(\frac{p}{1-r+2rp-p}\right) + (1-r)(1-p)\log\left(\frac{1-p}{1-r+2rp-p}\right),
\end{aligned}$$

ezt a kifejezést kell maximalizálni r-ben. Ha létezik maximum, akkor ez szuprémum is. A szélsőértéket úgy találjuk meg, hogy a deriváltat 0-val tesszük egyenlővé.

$$\begin{aligned}
I(X,Y)' = -p\log\left(\frac{1-p}{-2pr+p+r}\right) + \log\left(\frac{1-p}{-2pr+p+r}\right) - p\log\left(\frac{p}{-2pr+p+r}\right) + \\
+p\log\left(\frac{1-p}{2pr-p-r+1}\right) - \log\left(\frac{1-p}{2pr-p-r+1}\right) + p\log\left(\frac{p}{2pr-p-r+1}\right) = 0
\end{aligned}$$

A derivált ekvivalens átalakításokkal egyszerűbb alakra hozható.

$$I(X,Y)' = (2p-1)\log\left(\frac{-2pr+p+r}{2pr-p-r+1}\right) = 0$$

Ha $(2p-1) = 0$, akkor $p = \frac{1}{2}$, ami azt jelenti, hogy nincs összefüggés az X bemenet és az Y kimenet között. Ebben az esetben a kölcsönös információ 0, ami $I(X,Y)$ minimuma. Így feltehetjük, hogy $(2p-1) \neq 0$, és

$$\log\left(\frac{-2pr+p+r}{2pr-p-r+1}\right) = 0.$$

Ebből kifejezzük az r valószínűséget, így azt kapjuk, hogy $r = \frac{1}{2}$. Tehát akkor tudunk optimális információmennyiséget továbbítani, ha az X bemenet $\frac{1}{2}$ valószínűséggel 0, és ugyanekkora valószínűséggel 1. Ha r-et visszahelyettesítjük a kölcsönös információ képletébe, abból megkapjuk a csatorna kapacitását.

$$C = \sup_{P_X(x)} I(X,Y) = \frac{1-p}{2}\log(2-2p) + \frac{p}{2}\log(2p) + \frac{p}{2}\log(2p) + \frac{1-p}{2}\log(2-2p) =$$
$$= (1-p)\log(2-2p) + p\log(2p) = \log(2) + p\log(p) + (1-p)\log(1-p)$$

Mivel a csatorna bináris, praktikusabb a természetes alapú helyett a kettes alapú logaritmussal számolni. Ezzel az áttéréssel a kapacitást bitben kapjuk meg, míg a természetes alapú logaritmust használva a mértékegység nat (natural unit of information) marad.

$C = 1 + p\log_2(p) + (1-p)\log_2(1-p)$ bit a bináris szimmetrikus csatorna kapacitása.

3. Lineáris kódok

A lineáris kódok a hibajavító kódok egy nagy családja, melyet általában két csoportra osztunk, a konvolúciós kódok és a blokk-kódok csoportjára. Ez utóbbiba tartoznak az alacsony sűrűségű paritásellenőrző, vagy LDPC kódok, melyekkel a dolgozatban foglalkozunk.

Ebben a fejezetben a lineáris kódolás témakörében használt fontosabb fogalmakat tekintjük át.

3.1. Általános fogalmak

Először néhány alapvető fogalommal kell megismerkednünk. [13] [11]

3.1. Definíció.
Legyen Q egy ábécé, azaz betűk halmaza, Q^m az m elemű sorozatok halmaza, Q^m elemeit szavaknak nevezzük.

A kódolás egy $\phi : Q^m \to Q^n$ injektív függvény. $\phi(Q^m) = C \subseteq Q^n$ a ϕ értékkészlete, elemei a kódszavak.

A Q^m értelmezési tartomány elemeiből épülnek fel az üzenetek, ezért Q^m elemeit az üzenet koordinátáinak vagy bitjeinek is nevezzük.

3.1. Példa.
Kódolás az üzenetbitek duplázása.
Például a 010001 üzenet kódolva 001100000011.
Itt $Q = \{0, 1\}$, $m = 1$, $n = 2$, $\phi(a) = aa$, $C = \{00, 11\} \subseteq Q^2$.

3.2. Definíció.
Legyen Q ábécé, $\phi : Q^m \to Q^n$ kódoló függvény, ekkor $C \subseteq Q^n$-t egy (n, m) paraméterű kódnak nevezzük, n a kód hossza, m a kód dimenziója.

Egy kód megadásához elégséges csak a C halmazt meghatározni a kódoló függvényt nélkül. A gyakorlatban persze nagyon fontos, hogy a ϕ függvény hatékony legyen, ezért szerencsésebb ezt is megadni.

3.3. Definíció.
Legyenek $v, w \in Q^n$ kódszavak. v és w Hamming-távolsága azoknak a koordinátáknak a száma, ahol a két kódszó eltér egymástól.

3.2. Példa.
Legyen $v = 00111\mathbf{01}01$, $w = 00111\mathbf{10}01$, ekkor v és w Hamming-távolsága 2.

3.4. Definíció.
Egy kód d minimális távolsága a különböző kódszavak Hamming-távolságainak minimuma, vagyis a két legközelebbi kódszó távolsága.

3.5. Definíció.
Egy $C = (m, n)$ kód rátájának nevezzük az $R = \frac{m}{n}$ számot.

A kód rátája tehát megmutatja, hogy egy üzenetben a vektor hosszához viszonyítva mennyi hasznos információt továbbítunk. Nyilván azt szeretnénk, hogy a ráta nagy legyen (kapacitáshoz közeli), mert hosszú, mégis kevés adatot tartalmazó üzeneteket küldeni nem gazdaságos. Azonban ha a ráta értéke nagyobb, mint a csatorna kapacitása, akkor dekódoláskor nem áll rendelkezésünkre elég segédkoordináta ahhoz, hogy felismerjük és kijavítsuk az üzenet értékes részében keletkezett hibákat. Így a kód nem javíthat tetszőlegesen sok hibát anélkül, hogy a hossza miatt ne romlana a hatékonysága. A feladat tehát olyan kódokat találni, melyek rátája közel van a csatorna kapacitásához, de a hibajavításhoz szükséges redundáns bitek aránya a lehető legkisebb. A hibajavítással a következő fejezet foglalkozik.

3.2. Hibajavító képesség

Hibajavító kódokat több szempont alapján is összehasonlíthatunk. Amikor ezt a hibajavító képesség alapján tesszük, akkor azt mondjuk, hogy egy kód jobb hibajavító képességűnek számít a másiknál, ha több hibát tud felfedezni, és kijavítani anélkül, hogy jelentősen megnövelné az üzenet hosszát.

Ahhoz, hogy sok hibát javítson egy kód, nagy minimális távolságúnak kell lennie. Ezzel szemben a kódolt üzenet akkor nem lesz túl hosszú, ha a kód rátája nagy, vagy másképp megfogalmazva, ha $|m-n|$ kicsi. Ez a két követelmény ellentmond egymásnak, ezért nem választhatjuk a kód minden tulajdonságát tetszőlegesen kedvezőre. A paraméterek közötti összefüggést a Hamming-korlát adja meg. [13] [11] [14]

Amikor egy kódszó átmegy a csatornán, esetleg módosulhat néhány koordinátája. Ekkor a legjobb taktika az, ha megkeressük a kódszavak közül azt, amelyik a legkevesebb helyen tér el a kapott szótól. Ez a legbiztosabb dekódoló módszer. Megtehetjük például azt, hogy végignézzük az összes kódszót, és megnézzük, melyiktől tér el a legkevesebb helyen a módosult kódszó. Azonban ha sok a kódszó, akkor ez nyilván túl időigényes folyamat.

A dekódolásban épp az a kihívás, hogy olyan algoritmust kell találnunk, ami esetleg nem mindig a legközelebbi kódszót adja, de általában igen, és gyorsan teszi ezt. Később látni fogjuk, hogy az LDPC kódokhoz van ilyen dekóder.

A dekódolás feladata tehát úgy visszaadni az üzenetet, hogy közben a lehető legtöbb hibát kijavítsa. A hibák javíthatósága pedig függ a kód fentebb már megemlített tulajdonságaitól.

3.6. Definíció.
Legyen $e \geq 1$ egész szám. A C kód e-hibajelző, ha egy kódszót legfeljebb e helyen megváltoztatva az eredmény nem lehet kódszó.

3.1. Tétel.
Legyen a C kód minimális távolsága d. A C kód pontosan akkor e-hibajelző, ha $e < d$.

Bizonyítás.
A kód minimális távolsága d, emiatt vehetünk olyan v és w kódszavakat, melyek távolsága éppen d. Ez azt jelenti, hogy v-t d helyen megfelelően megváltoztatva megkaphatjuk w-t. Ekkor v-ből egy másik kódszót kapunk, tehát v-t legfeljebb $e = d - 1$ helyen módosíthatjuk, ha meg akarjuk tartani az e-hibajelző tulajdonságot. □

3.7. Definíció.
A C kód e-hibajavító, ha bármely két kódszót legfeljebb e helyen megváltoztatva nem kaphatjuk eredményül ugyanazt a kódszót.

3.2. Tétel.
Legyen a C kód minimális távolsága d. A C kód pontosan akkor e-hibajavító, ha $2e < d$.

Bizonyítás.
Legyenek v és w kódszavak, $v \neq w$, minimális távolságuk d.

Az e-hibajavító tulajdonság pontosan azt jelenti, hogy ha a v' kódszó legfeljebb e helyen változott, akkor még egyértelműen rekonstruálható az eredeti v. Azonban ha e-t $\frac{d}{2}$-nél nagyobbnak választjuk, akkor v' v-től vett távolsága nagyobb lesz, mint a w-től vett távolsága. Ekkor javítás után nem v-t kapjuk vissza, hanem w-t.

Abban az esetben, ha e-t $\frac{d}{2}$-nek választjuk, nem dönthető el egyértelműen, hogy v vagy w volt-e az eredeti kódszó, mert v' egyenlő távolságra van tőlük. □

Az, hogy egy kód e hibát javít, szemléletesen azt jelenti, hogy ha a kódszavak köré e sugarú n dimenziós gömböket írunk, akkor a gömbök diszjunktak lesznek. Ha a gömbök átfedés nélkül kitöltik a teret, akkor a kódot perfektnek nevezzük.

A gömbök térfogatára ad felső becslést a Hamming-korlát, amit épp ezért gömbkitöltési korlátnak is szokás hívni.

3.3. Tétel (Hamming-korlát)**.**
Legyen $C = (n, m)$ kód a q-elemű Q ábécé felett, és legyen C minimális távolsága d. Vezessük be az $e = \lfloor (d-1)/2 \rfloor$ jelölést. Ekkor

$$q^m V_q(n, e) \leq q^n,$$

ahol

$$V_q(n, e) = \sum_{i=0}^{e} \binom{n}{i} (q-1)^i.$$

Bizonyítás.
Q felett q^n darab n hosszú szót tudunk alkotni (azaz $|Q^n| = q^n$). Minden szó köré írunk egy $e = \left\lfloor \frac{d-1}{2} \right\rfloor$ sugarú gömböt. Ezek páronként diszjunktak, hiszen sugaruk legfeljebb $\frac{d-1}{2}$, tehát $|Q^n| = q^n \geq m V_q(n, e)$. □

A $V_q(n, e)$ összeg éppen egy e sugarú gömbben található szavak száma.

3.8. Definíció.
Ha a $C = (n, m)$ kódra egyenlőséggel áll fenn a Hamming-korlát, akkor C perfekt kód.

3.3. Lineáris kódok mátrixai

A lineáris kódokat két mátrix segítségével írjuk le, ezek a generátor- és a paritásellenőrző mátrix. [13] [11] [14]

3.9. Definíció.
Ha $Q = \mathbb{F}_q$ a q elemű véges test és C altere a Q^n vektortérnek, akkor C-t lineáris kódnak nevezzük.

Egy lineáris kódot megadhatunk például úgy, hogy meghatározzuk a C vektortér egy bázisát.

3.10. Definíció.
Legyen C egy lineáris kód $\mathbb{F}_q$ felett. Ekkor C tetszőleges bázisát egy mátrix soraiba írva a kapott $G \in \mathbb{R}^{m \times n}$ mátrixot C generátormátrixának nevezzük.

Ha C egy lineáris (n, m) kód $\mathbb{F}_q$ felett, akkor a kódszavak az $\mathbb{F}_q^n$ vektortérnek egy m-dimenziós alterét alkotják, így a C kód egy G generátormátrixa $\mathbb{F}_q^{m \times n}$-beli. Tetszőleges $w \in \mathbb{F}_q^m$ szó esetén $v = wG$ kódszó, tehát az üzenetek kódolása egyszerűen kivitelezhető mátrixszorzás segítségével. G sorai tehát generálják a kódszavakat, hiszen minden kódszó a mátrix sorainak valamely lineáris kombinációja.

Megjegyezzük, hogy a generátormátrix természetesen nem egyértelmű, hiszen C-ben többféle bázist is választhatunk.

3.4. Tétel.
Legyenek v és w kódszavai, G generátormátrixa a C lineáris kódnak. Ekkor

$$v = wG.$$

3.3. Példa.
A 3.1 példában szereplő duplázó kód lineáris.
Általánosan, ha a szavak m hosszú vektorok, akkor a duplázás generátormátrixa $G = (I_m, I_m)$.

Idézzük fel a szokásos skaláris szorzást: ha $v = (v_1, \ldots, v_n)$ és $w = (w_1, \ldots, w_n)$, akkor $\langle v, w \rangle := v_1 w_1 + \ldots + v_n w_n$. Ez $\mathbb{F}_q^n$-beli vektorokra ugyanúgy értelmezhető, mint valósakra, és ugyanúgy fennáll, hogy $\mathbb{F}_q^n$ egy altérének dimenziója és a merőleges (ortogonális) kiegészítőjének dimenziója összeadva n. Megjegyezzük, hogy a valós esettől eltérően ez a két altér metszhet nemtriviálisan is.

3.11. Definíció.
Legyen C egy lineáris (n, m)-kód $\mathbb{F}_q$ fölött. Legyen $k = n - m$, és legyen $H \in \mathbb{F}_q^{k \times n}$ a C altér ortogonális kiegészítő alterének egy bázisából mint sorvektorokból álló mátrix. Ekkor H-t a C kód paritásellenőrző mátrixának nevezzük.

Vegyük észre, hogy definíció szerint $HG^T = 0$.

3.5. Tétel.
Legyen C egy lineáris (n, m)-kód $\mathbb{F}_q$ fölött, $k = n - m$, és legyen C paritásellenőrző mátrixa $H \in \mathbb{F}_q^{k \times n}$. Ekkor $w \in \mathbb{F}_q^n$ pontosan akkor kódszó, ha $Hw^T = 0$ teljesül.

Bizonyítás.
Tegyük fel, hogy w kódszó. Ekkor merőleges H soraira, mert G sorainak lineáris kombinációja. Így Hw eredménye k hosszú nullvektor. Fordítva: Tegyük fel, hogy v nem kódszó, ekkor nem merőleges H minden sorára, ezért Hw nem lehet a nullvektor. □

3.12. Definíció.
A $C = (n, m)$ lineáris kódot szisztematikusnak nevezzük, ha a kódszavak első m koordinátája megfelel az üzenetnek.

Szisztematikus kód esetén a generátormátrix $G = (I_m, B)$, ahol I_m az $m \times m$ méretű egységmátrix, B pedig egy $m \times (n-m)$ méretű mátrix. A v üzenethez tartozó kódszó felépítése a következő:

$$w = (v_1, v_2, ..., v_m, w_{m+1}, w_{m+2}, ..., w_n).$$

A kódszó első m koordinátáját üzenetszegmensnek, az utolsó $n - m$ koordinátáját paritásszegmensnek nevezzük.

3.6. Tétel.
Legyen a $C = (n, m)$ lineáris kód szisztematikus, ekkor a C kód H paritásellenőrző mátrixa felírható a következő alakban:

$$H = (-B^T, I_{n-m}),$$

ahol I_{n-m} az $(n - m) \times (n - m)$ méretű egységmátrix, B a generátormátrix szisztematikus alakjában szereplő mátrix.

Bizonyítás.
Legyen $w = (v_1, ..., v_m, w_{m+1}, ..., w_n)$ az üzenetvektor, és tegyük fel, hogy $H = -B^T, I_{n-m}$ alakú. Az üzenetvektort behelyettesítve:

$$Hw^T = H(vG)^T = HG^T v^T = 0,$$

ahonnan $HG^T = 0$ adódik. Ekkor G és H szisztematikus alakját használva a következőt kapjuk:

$$HG^T = (A, I^{n-m})(I_m, B)^T = A + B^T = 0.$$

Tehát $A = -B^T$. □

Egy C lineáris kódnak általában több generátormátrixa és paritásellenőrző mátrixa is lehet.

3.4. Tanner gráf

Minden bináris lineáris kód paritásellenőrző mátrixa reprezentálható gráfként is, ezt Tanner gráfnak nevezzük. A paritásellenőrző mátrix és a Tanner gráf kölcsönösen egyértelműen megfeleltethető egymásnak úgy, hogy a mátrix minden sora megfelel egy bitcsúcsnak, és minden oszlopa megfelel egy ellenőrzőcsúcsnak.

Ez az ábrázolás könnyebben átláthatóvá teszi a kódolás és a dekódolás folyamatát, mint a mátrixokon végzett műveletek, ezért lineáris kódokat sokszor érdemesebb ebben a formában vizsgálni. [6]

3.13. Definíció.
Legyen $H \in \mathbb{R}^{k \times n}$ egy bináris paritásellenőrző mátrix, $T = \{t_1, \ldots, t_k\}$ a bitcsúcsok halmaza, $|T| = k$, $S = \{s_1, \ldots, s_n\}$ az ellenőrzőcsúcsok halmaza, $|S| = n$. Legyen a $t_i \in T$ bitcsúcs szomszédos az $s_j \in S$ csúccsal pontosan akkor, ha $H_{i,j} = 1$. A $TG = (T, S)$ páros gráfot Tanner gráfnak nevezzük.

3.7. Tétel.
Legyen $TG = (T, S)$ a C kód Tanner gráfja. A $t \in T$ bitcsúcs foka pontosan annyi, ahány paritásellenőrzésben szerepel a t. bit. Az $s \in S$ ellenőrzőcsúcs foka pontosan annyi, ahány bit szerepel az s. paritásellenőrzésben.

Paritásellenőrzésnek a H mátrix soraival vett szorzást nevezzük.

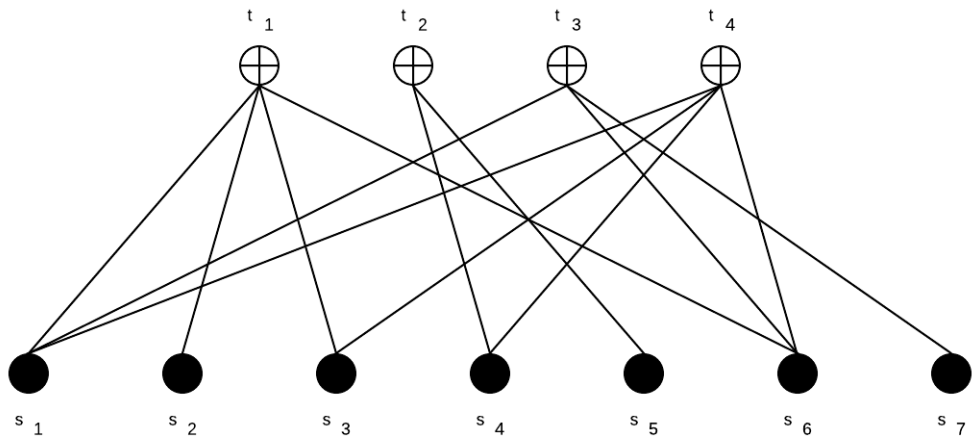
Bizonyítás.
A s csúcs akkor szomszédos a t csúccsal, ha a t. bit szerepel az s. paritásellenőrzésben. Ez igaz minden $s \in S$-re, így minden paritásellenőrzés eggyel növeli t fokát.
Fordítva ugyanígy minden bit ellenőrzése eggyel növeli az s ellenőrzőcsúcs fokát. □

3.4. Példa.
Legyen a paritásellenőrző mátrix

$$H = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \end{bmatrix}$$

A H-nak megfelelő Tanner gráf a következő:

2. ábra. Tanner gráf



3.5. Példák lineáris kódokra

A következő fejezetben néhány alapvető lineáris blokk-kódot mutatunk be érintőlegesen. [14] [10]

Hamming-kódok

A Hamming-kódok a hibajavító kódok egy könnyen kódolható és dekódolható csoportja, az első hibajavító kódok között alkalmazták őket. Egyszerű felépítésük miatt azonban nem teljesítenek jól, ha a csatorna hibái jellemzően egy helyre tömörülnek (ilyen hiba például egy karcolás a merevlemezen). Ennek ellenére a Hamming-kódok ma is használatosak, és sok bonyolultabb lineáris kód alapjául is szolgálnak.

3.14. Definíció.
Az 1-hibajavító, 2-hibajelző perfekt lineáris kódokat Hamming-kódoknak nevezzük.

3.8. Tétel.
Legyen $p \geq 2$ egész, $n = 2^p - 1$, $k = 2^p - 1 - p$. Ekkor a $C = (n, k)$ lineáris kód Hamming-kód.

3.5. Példa.
A $C = (7, 4)$ bináris Hamming-kód esetében $r = 3$, $n = 7$, $k = 4$. A kód minimális távolsága $d = 3$, paritásellenőrző mátrixa

$$H = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Reed-Solomon kódok

A Reed-Solomon kódok jelentősségét az adja, hogy MSD-kódok, azaz minimális távolságuk a lehető legnagyobb az adott vektortér feletti lineáris kódok között. Ezt a maximumot a Singleton-korlát adja meg.

3.9. Tétel (Singleton-korlát)**.**
Legyen $C = (n, m)$ lineáris kód minimális távolsága d, ekkor

$$d \leq n - m + 1.$$

3.15. Definíció.
Ha a $C = (n, m)$ lineáris kód minimális távolsága d, és C-re egyenlőséggel teljesül a Singleton-korlát, akkor a kódot MDS-kódnak (maximum distance separable) nevezzük.

Reed-Solomon kódokat többféleképpen létrehozhatunk, egy lehetséges konstrukció a következő:

$$C(n, m) = p(x_1), p(x_2), ..., p(x_n),$$

ahol p legfeljebb m-edfokú polinom $\mathbb{F}_q$ felett, $x_1, ..., x_n$ pedig $\mathbb{F}_q$-beli pontok.

Vagyis kiértékeljük a p polinomot az $x_1, ..., x_n$ helyeken, és az így kapott értékek lesznek a kódszavak. Az x_i pontokat praktikus okokból általában prímhatványoknak választjuk.

Golay-kódok

A Golay-kódok közé négy lineáris kód tartozik, ezek közül kettő bináris, kettő ternáris, azaz $\mathbb{F}_3$ felett definiált.

A bináris Golay-kódok áttörést jelentettek a hibajavító kódok között, hiszen segítségükkel sokkal nagyobb adatmennyiséget lehetett feldolgozni ugyanannyi idő alatt, mint a korábban használatos kódokkal. Hatékonyságuk miatt így kódolták például a Voyager űrszondák által küldött adatokat is.

Turbo kódok

A Turbo kódok voltak az első olyan hibajavító kódok, amelyek teljesítménye jól meg tudta közelíteni az elméleti maximumot, a Shannon-korlátot. Bár az LDPC kódok elkezdték kiszorítani őket, még ma is használjuk őket például műholdas kommunikáció során.

A Turbo kódok családja többféle kódot tartalmaz, ezek között akadnak blokk-kódok, konvolúciós kódok, valamint a kettő keveréke is.

Felépítésüket tekintve a Turbo kódok eredetileg két másik kódból állnak össze. Minden üzenetkoordinátánál véletlenszerűen kiválasztjuk az egyik kódot, és ezzel kódoljuk az aktuális koordinátát, így mindkét kóddal alkotunk kódszavakat.

4. LDPC kódok és gyakorlati jelentősségük

A Low-Density Parity-Check, azaz alacsony sűrűségű paritásellenőrző kódok olyan lineáris hibajavító blokk-kódok, melyek paritásellenőrző mátrixa „ritka", vagyis kevés nemnulla elemet tartalmaz. Ez általában azt jelenti, hogy a mátrixnak sokkal kevesebb nemnulla eleme van, mint ahány 0 van benne. Ennél pontosabban a sűrűség fogalmának bevezetésével határozhatjuk meg, hogy mennyire legyen ritka a mátrix. [12] [4] [1] [2]

Az LDPC kódokat 1963-ban Robert Gallager mutatta be a doktori értekezésében, azonban sokáig nem ismerték fel valódi jelentősségüket, főleg a nagy számítási komplexitásuk miatt. Az 1990-es években kerültek ismét előtérbe.

Teljesítményük rendkívül jó, egyes véletlen módon generált kódoké jól közelíti az elméleti határt, így újbóli felfedezésük óta fokozatosan kiszorítják eddig használt konkurenseiket, például a Turbo kódokat. Széleskörűen alkalmazzák őket a gyakorlatban, főleg olyan területeken, ahol különösen fontos, hogy az üzenet minél gyorsabban továbbítható és dekódolható legyen. Népszerűségüket növeli, hogy egy LDPC kód általában többféle zajos csatornán is jól teljesít, így sok területen felhasználható.

Az LDPC kódok mára számos szabványba is bekerültek, többek között a második generációs műsorszóró (DVB-T2, S2, C2) szabványba, a WiFi (IEEE 802.11n) szabványba, a WiMAX (IEEE802.16e) szabványba, illetve a 10 gigabites Ethernet szabványba is.

A dekódolás sebességét nagyban befolyásolja az adott kód paritásellenőrző mátrixának sűrűsége, ahol a sűrűséget a mátrix oszlopainak és sorainak súlyával definiáljuk.

4.1. Definíció.
A $v \in \mathbb{R}^n$ vektor súlya az a szám, ahány nemnulla koordinátája van v-nek.

Az alapján, hogy egységes-e a paritásellenőrző mátrix sorainak és az oszlopainak súlya, megkülönböztetünk reguláris és irreguláris LDPC kódokat.

4.2. Definíció.
Legyenek $\rho \in \mathbb{N}$ és $\gamma \in \mathbb{N}$ rögzítettek. A C LDPC kód reguláris, ha C paritásellenőrző mátrixának minden sora ρ súlyú, és minden oszlopa γ súlyú. Ekkor a C kódot (ρ, γ)-regulárisnak is nevezzük.

4.3. Definíció.
A C LDPC kód irreguláris, ha nem reguláris.

4.4. Definíció.
Legyen a C kód (ρ, γ)-reguláris, $H \in \mathbb{R}^{k \times n}$ a C paritásellenőrző mátrixa. Ekkor a H mátrix δ sűrűsége

$$\delta := \frac{\rho}{n} = \frac{\gamma}{k}.$$

A gyakorlatban használt LDPC kódok esetén általában elvárjuk a jobb teljesítmény érdekében, hogy a δ sűrűség $\frac{6}{2^8}$ és $\frac{6}{2^{13}}$ közé essen. A dekódoló algoritmus hatékonyabbá tehető úgy is, ha a H paritásellenőrző mátrix olyan, hogy a hozzá tartozó Tanner gráf bősége nem kisebb egy $p \in \mathbb{N}$ paraméternél, tehát nincs benne p-nél kisebb kör. p-t általában 6-nak, de legalább 4-nek választjuk.

Bár elméleti szempontból ez nem elvárás, praktikus okokból az alkalmazott LDPC kódok jellemzően binárisak. Ezek a kódok általában több ezer sorból és oszlopból álló G generátormátrixszal, kódszavaik relatíve nagy minimális távolsággal rendelkeznek.

5. Kódolás

LDPC kódok esetében a kódolás folyamata tipikusan úgy történik, hogy valamilyen eljárással konstruálunk egy H paritásellenőrző mátrixot, amiből aztán kiszámítható a kód G generátormátrixa. [6] [1] [2]

G-ből egy mátrixszorzással megkapjuk a kódszavakat.

Legyen w az üzenetvektor, ekkor a hozzá tartozó v kódszó a

$$v = wG$$

szorzással kapható meg.

Azok a módszerek, melyekkel LDPC kódokat tudunk létrehozni, nem a generátormátrixot, hanem a paritásellenőrző mátrixot adják meg (vagy a vele ekvivalens Tanner gráfot), ezért első lépésként meg kell keresnünk H ortogonális kiegészítőjét, azaz a generátormátrixot. Szisztematikus kódok esetén G-t könnyen meghatározhatjuk H-ból, hiszen

$$H = (-B^T, I_{n-m}) \quad \text{alakú, amiből} \quad G = (I_m, B).$$

Nem szisztematikus kódok esetében G megtalálása nehezebb feladat.

A generátormátrixos szorzással más probléma is felmerülhet az előbbin kívül. Mivel a G mátrix a paritásellenőrző mátrixszal ellentétben nem feltétlenül ritka, így a kódolás általában jóval kevésbé hatékonyan zajlik, mint a dekódolás. Kivételnek számítanak bizonyos geometriai megközelítéssel létrehozott LDPC kódok, melyekkel a következő fejezetben foglalkozunk.

6. Konstrukciók

LDPC kódok előállítására számos módszer ismert. Léteznek gráf alapú, kombinatorikai és geometriai konstrukciók, pszeudovéletlen algoritmusokkal generált kódok és más típusú kódokból származtatott kódok is. A következő fejezet a gyakoribb eljárásokat tekinti át.

Ahhoz, hogy létrehozzunk egy LDPC kódot, tulajdonképpen a kód paramétereit határozzuk meg. Ezek a paraméterek:

1. A kód n hossza és m dimenziója. Ez a két paraméter meghatározza a G generátormátrix és a H paritásellenőrző mátrix méretét is, hiszen $G \in \mathbb{R}^{m \times n}$,és $H \in \mathbb{R}^{k \times n}$, ahol $k = n - m$ a paritásellenőrzések száma. Mivel a hosszabb kódok jobban teljesítenek, mint a rövidek, n-t érdemes nagynak választani.

2. A paritásellenőrző mátrix sorainak és oszlopainak súlya. Ha a paritásellenőrző mátrix helyett a Tanner gráf segítségével építjük fel a kódot, akkor ez a lépés azt jelenti, hogy meghatározzuk az egyes csúcsok fokszámát.

3. A paritásellenőrző mátrix felépítése. Véletlenszerű generálás esetén természetesen a H mátrixnak sem lesz semmilyen jellegzetes struktúrája, más módszerek esetén azonban lehet összefüggés a sorok, vagy az oszlopok között. Tanner gráffal reprezentált kód esetén ez a lépés a gráf felépítését határozza meg, azaz megmondjuk, hogy az egyes bitcsúcsok melyik paritásellenőrző csúcsokkal szomszédosak.

6.1. Definíció.
Ha az LDPC kódot nem valamilyen véletlen konstrukcióval építjük fel, akkor tehát a paritásellenőrző mátrixának lesz egy jellemző mintázata. Az ilyen kódokat strukturált kódoknak nevezzük.

6.1. Véletlen konstrukciók

A gyakorlatban igen népszerűek a paritásellenőrző mátrixot véletlen módon megválasztó kódgeneráló módszerek. Ezek előnye, hogy az így elkészített LDPC kódok rendkívül gyorsan dekódolhatóak. [6] [8]

MacKay-Neal-féle konstrukció

A MacKay és Neal által javasolt módszer egyszerű algoritmust ad ugyan a paritásellenőrző mátrix megválasztására, de nem vesz figyelembe olyan fontos szempontokat, mint például a könnyű dekódolhatóság.

Az algoritmus soronként állít elő egy bináris $H \in \mathbb{R}^{k \times n}$ mátrixot. Tegyük fel, hogy a mátrix sorainak súlya ρ, oszlopainak súlya γ. A H mátrix $i.$ sorának meghatározásához véletlen módon generálunk egy ρ súlyú vektort. Kiegészítjük a már meglévő $1., 2., ..., i-1.$ sorokból álló H_{i-1} mátrixot az $i.$ sorral. Ekkor az $i.$ sor megfelelő, ha

1. a H_i mátrix minden oszlopának súlya legfeljebb γ,

2. H_i bármely két sorának legfeljebb egy közös 1-es komponense van. Ezzel a feltétellel kizárjuk a 6-nál kisebb köröket a mátrixhoz tartozó Tanner gráfban.

Ha az i. sor nem teljesíti a fenti feltételeket, akkor új vektort generálunk a helyette, majd erre is végrehajtjuk az ellenőrzést. A k. sort akkor tekintjük megfelelőnek, ha

1. a $H_k = H$ mátrix minden oszlopának súlya pontosan γ,

2. H bármely két sorának legfeljebb egy közös 1-es komponense van.

Mackay a fenti algoritmus mellett javasolt további eljárásokat is a paritásellőrző mátrix létrehozására. Néhány példa:

1. Kiindulunk a $k \times n$-es nullamátrixból, majd véletlenszerűen kiválasztjuk a mátrix néhány elemét. A kiválasztott nullákat egyesekre cseréljük.

2. Meghatározzuk a mátrix oszlopainak γ súlyát, majd véletlenszerűen generálunk n darab k hosszú γ súlyú oszlopvektort, amiket egymás mellé írunk.

3. A fenti algoritmust módosítjuk úgy, hogy 6-nál nagyobb köröket is ki tudjunk zárni.

Gallager-féle konstrukció

Gallager javaslata szerint a H paritásellenőrző mátrixot olyan alakban kell keresnünk, hogy $\frac{k}{\gamma} = \frac{n}{\rho} = p$ teljesüljön, ahol $p \in \mathbb{N}$ egy előre meghatározott konstans. Mivel H sűrűsége az definició szerint $\delta := \frac{\rho}{n} = \frac{\gamma}{k} = \frac{1}{p}$, a p paramétert kellően nagynak választva a H mátrixhoz tartozó kód biztosan LDPC kód lesz.

Konstrukció permutáló mátrixokkal

6.2. Definíció.
Az $M \in \mathbb{R}^{n \times n}$ bináris mátrixot permutáló mátrixnak nevezzük, ha minden sorában és minden oszlopában pontosan egy 1-es szerepel.

Ez az algoritmus egy $H_0 \in \mathbb{R}^{j \times l}$, $j < k$, $l < n$ bináris paritásellenőrző mátrixot használ kiindulási pontnak, feltéve, hogy H_0 sorainak súlya ρ, oszlopainak súlya γ. Ezt bővítjük permutáló mátrixokkal.

A H_0 mátrixban minden 1-est helyettesítünk egy véletlenszerűen választott $p_1 \times p_1$-es permutáló mátrixszal, és minden 0 elemet helyettesítünk a $p_1 \times p_1$-es nulla mátrixszal. Az így kapott H_1 mátrixot a következő iterációban ismét bővítjük egy véletlenszerűen választott $p_2 \times p_2$-es permutáló mátrixszal, illetve a nulla mátrixszal ($p_1, p_2, ... > 1$ egészek). A bővítéseket addig ismételjük, míg egy $H_n = H \in \mathbb{R}^{k \times n}$ mátrixot kapunk eredményül.

Ez az eljárás a H_0-hoz tartozó LDPC kódból állít elő egy hosszabb kódot anélkül, hogy a kezdeti H_0 mátrix sorainak és oszlopainak súlyát módosítaná.

6.2. Strukturált konstrukciók

A strukturált, algebrai és geometriai alapú módszerekkel generált LDPC kódok némileg lassabban dekódolhatóak, mint a véletlen módon generált kódok. Ennek ellenére széles körben alkalmazzák ezeket az eljárásokat is, ugyanis segítségükkel olyan paritásellenőrző mátrixok generálhatók, melyek hatékonyabb kódolást tesznek lehetővé. Ennek a tulajdonságuknak köszönhetően a strukturált kódok remekül megfelelnek olyan esetekben, amikor a nagy sebességű dekódolhatóság helyett fontosabb szempont a könnyű kódolhatóság, míg fordított esetben a véletlen algoritmusokat érdemes előnyben részesíteni.

A következő fejezetben ismertetünk néhány ilyen eljárást. [6] [11] [4] [5]

Konstrukció elcsúsztatott egységmátrixokkal

Ez az algoritmus egy olyan $H \in \mathbb{R}^{k \times n}$ paritásellenőrző mátrixból indul ki, melyben a sorok súlya ρ, az oszlopok súlya γ. Legyen $p \leq (\rho - 1)(\gamma - 1) + 1$ tetszőleges, és $l \in \{0, ..., p-1\}$ tetszőleges, valamint legyen I_p a $p \times p$-es egységmátrix.

Jelölje $J_{p,l}$ az a mátrixot, melyet úgy kapunk, hogy I_p első l oszlopát leválasztjuk, és a mátrix utolsó oszlopa mögé írjuk.

6.1. Példa.

$$J_{3,1} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad \text{és} \quad J_{3,2} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

A H mátrixot a következőképpen definiáljuk:

$$H = \begin{bmatrix} I_p & I_p & I_p & ... & I_p \\ I_p & J_{p,1} & J_{p,2} & ... & J_{p,\rho-1} \\ I_p & J_{p,2} & J_{p,4} & ... & J_{p,2(\rho-1)} \\ I_p & J_{p,3} & J_{p,6} & ... & J_{p,3(\rho-1)} \\ \vdots & \vdots & \vdots & ... & \vdots \\ I_p & J_{p,\gamma-1} & J_{p,2(\gamma-1)} & ... & J_{p,(\gamma-1)(\rho-1)} \end{bmatrix}$$

Ekkor $k = p\gamma$, $n = p\rho$. Minden sor (oszlop) súlya ρ (γ), hiszen a $J_{p,l}$ mátrix minden sora (oszlopa) pontosan egy 1-est tartalmaz.

Konstrukciók véges geometriák használatával

6.3. Definíció.

Illeszkedési struktúrának egy $\mathbf{G} = (P, B, I)$ hármast nevezünk, ahol P és B két diszjunkt

halmaz, I pedig egy P és B elemei közti reláció, azaz $I \subset P \times B$. P elemeit pontoknak, B elemeit blokkoknak, az I relációt pedig illeszkedési relációnak nevezzük. $\mathbf{G}$ véges illeszkedési struktúra, ha a P és B halmazok végesek.

6.4. Definíció.
Legyen $\mathbf{G} = (P, B, I)$ véges illeszkedési struktúra, ahol $|P| = n$ és $|B| = k$. Ekkor

1. $\mathbf{G}$ elsőfajú mátrixának nevezzük azt a $H^{(1)} \in \mathbb{R}^{k \times n}$ bináris mátrixot, ahol

$$H^{(1)}_{i,j} = \begin{cases} 1, & \text{ha a } \ j \in P \text{ pont illeszkedik az } i \in B \text{ blokkra} \\ 0, & \text{egyébként.} \end{cases}$$

2. $\mathbf{G}$ másodfajú mátrixának nevezzük azt a $H^{(2)} \in \mathbb{R}^{n \times k}$ bináris mátrixot, ahol

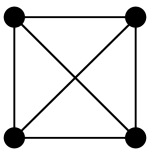
$$H^{(2)}_{i,j} = \begin{cases} 1, & \text{ha a } \ i \in P \text{ pont illeszkedik az } j \in B \text{ blokkra} \\ 0, & \text{egyébként.} \end{cases}$$

   Ez tehát $H^{(1)}$ transzponáltja.

3. $\mathbf{G}$ elsőfajú LDPC kódja a $H^{(1)}$ mátrix által generált n-hosszú LDPC kód, azaz $H^{(1)}$ magtere.

4. $\mathbf{G}$ másodfajú LDPC kódja a $H^{(2)}$ mátrix által generált k-hosszú LDPC kód, azaz $H^{(2)}$ magtere.

6.2. Példa.
Egy véges illeszkedési struktúra 4 ponttal és 6 egyenessel, és a hozzá tartozó első és másodfajú mátrixok:



$$H^{(1)} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \quad \text{és} \quad H^{(2)} = H^{(1)T} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

6.5. Definíció.
A $\mathbf{G}$ illeszkedési struktúra parciális lineáris tér, ha $\mathbf{G}$ bármely két blokkjának legfeljebb egy közös pontja van.

6.6. Definíció.
A $\mathbf{G}$ illeszkedési struktúra lineáris tér, ha $\mathbf{G}$ bármely két blokkjának pontosan egy közös pontja van.

Láttuk, hogy a $H^{(1)}$ és $H^{(2)}$ mátrixok a definíciójukból adódóan egymás transzponáltjai, az általuk generált kódok mégis igen különbözőek lehetnek, általában a paritásellenőrző mátrixaikon kívül nincs közöttük semmiféle kapcsolat.

A dekódoló algoritmus sebességét javíthatjuk, ha a $H^{(1)}$ vagy $H^{(2)}$ mátrixhoz tartozó Tanner gráf bőségét legalább 6-nak választjuk. Ehhez elég ha $\mathbf{G}$ bármely két blokkjának legfeljebb egy közös pontja van, tehát ha $\mathbf{G}$ (parciális) lineáris tér.

A lineáris terek, avagy vektorterek közül leggyakrabban a 2 hatvány elemszámú véges testek feletti projektív (PG, azaz projective geometry) és affin terek (EG, azaz Euclidean geometry) tulajdonságait használjuk fel kódgeneráláshoz.

Megjegyezzük, hogy bár a felhasznált szakirodalom az „Euclidean geometry" kifejezést használja, azonban ez a struktúra itt nem az euklideszi tereket (azaz a skaláris szorzattal ellátott vektortereket) jelenti. A helyes magyar fordítás az affin tér.

6.7. Definíció.
Legyen $\mathbb{F}_q$ a q elemű véges test, V egy $n+1$ dimenziós vektortér $\mathbb{F}_q$ felett, $q, n \in \mathbb{N}$. A $PG(n, q)$ n dimenziós, q-adrendű projektív tér az a tér $\mathbb{F}_q$ felett, melynek k dimenziós alterei pontosan a V vektortér $k + 1$ dimenziós alterei.

6.8. Definíció.
Legyen V egy n dimenziós vektortér $\mathbb{F}_q$ felett. Az n-dimenziós q-adrendű affin tér (jele a magyar szakirodalomban: $AG(n, q)$) pontjai ugyanazok, mint V pontjai, alterei pedig V altereinek eltoltjai (azaz $AG(n.q)$ minden k-dimenziós altere előáll úgy, mint V egy k-dimenziós alterének eltoltja).

Legyen V egy n dimenziós vektortér $\mathbb{F}_q$ felett, és V-nek a k dimenziós lineáris altereit és azok eltoltjait hívjuk legyenek a tér k dimenziós alterei. Az így kapott ponthalmazt és alterek rendszerét hívjuk q-adrendű n-dimenziós affin geometriának, jele AG(n,q).

Attól függően, hogy affin, vagy projektív térből származik a kód, megkülönböztetünk EG-LDPC és PG-LDPC kódokat. Ezek lehetnek első-, vagy másodfajúak aszerint, hogy a 6.4 definícióban szereplő mátrixok közül melyiket használtuk kódgenerálásra.

Az elsőfajú EG-LDPC és PG-LDPC kódok ciklikusak, ennek köszönhetően az LDPC kódok között egyedülálló módon a kódszavak hosszához képest lineáris időben kódolhatóak. Emellett kis kódhosszúság esetén hibajavító képességük is kiemelkedően jó.

6.9. Definíció.
Egy lineáris kód ciklikus, ha abból, hogy $w = (w_1, w_2, ..., w_n)$ kódszó, következik, hogy $w' = (w_n, w_1, ..., w_{n-1})$ is kódszó.

6.3. Hibrid LDPC kódok

Az LDPC kódok a hibajavító kódok egy alapvetően rendkívül jól használható osztálya, mégis létezik néhány olyan hibrid kód, amely az LDPC kódokat valamilyen más hibajavító kóddal ötvözi. Ilyenkor igyekszünk kihasználni ennek a másik kódnak az előnyös tulajdonságait. Például ha azt szeretnénk, hogy az új kód jobban illeszkedjen a csatorna sajátosságaira, akkor eszerint módosítjuk az LDPC kódunkat.

Többféle olyan hibrid kódfajta is létezik, aminek valamilyen LDPC kód az alapja. Ezek közül a reguláris RA kódokat mutatjuk be. [6]

RA kódok

Az RA, azaz Repeat-Accumulate kódok a bináris hibajavító kódok egy különleges osztálya, melyek irreguláris LDPC kódokból és Turbo kódokból származnak. Lineáris időben kódolhatóak a kódszavak n hosszúságához képest, emellett rendelkeznek az LDPC kódok jó tulajdonságaival is.

Egy (ρ, γ)-reguláris RA kód szisztematikus módon állítható elő, vagyis a kódszavak $w = (v, p)$ alakúak, ahol v az üzenetvektor, p pedig a paritásellenőrzésekből álló vektor. Az algoritmus a következő:

1. Először az m hosszú v üzenet minden koordinátáját lemásoljuk ρ-szor, ezzel egy ρm hosszú vektort hozunk létre.

$$w = (\underbrace{v_1, ..., v_1}_{\rho}, \underbrace{v_2, ..., v_2}_{\rho}, ..., \underbrace{v_m, ..., v_m}_{\rho})$$

   A ρ paraméter értéke általában 2 vagy 3.

2. Véletlenszerűen permutáljuk w koordinátáit.

3. Az új w vektor koordinátáit γ hosszú blokkokra osztjuk, majd minden blokkban összeadjuk a koordinátákat (a kételemű test feletti összeadással). Most ezeket az összegeket válaszjuk w koordinátáinak, tehát w egy $\frac{\rho m}{\gamma}$ hosszú vektor lesz. A paramétereket úgy választjuk meg, hogy $\frac{\rho m}{\gamma}$ egész legyen.

4. Előállítjuk a k hosszú p vektort. Az első koordináta $p_1 = 0$, ezután az i-edik koordináta úgy áll elő, hogy összeadjuk az előző paritásbitet w i-edik koordinátájával, vagyis

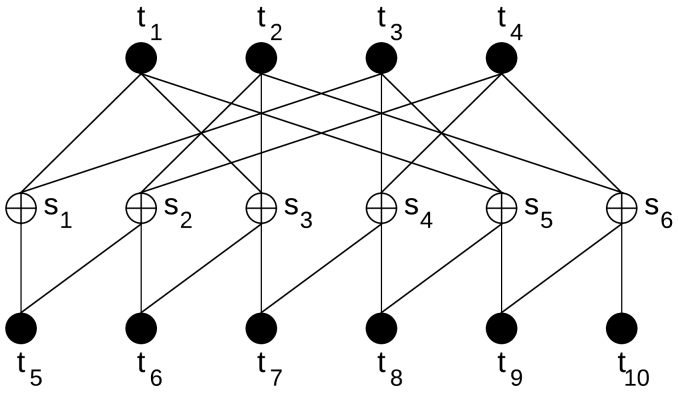
$$p_i = p_{i-1} + w_i, \quad i = 2, ..., k.$$

   Mivel p előállításakor $\frac{\rho m}{\gamma}$ bitet használtunk fel, így p hossza $k = \frac{\rho m}{\gamma}$.

5. Végül w ismét új értéket kap úgy, hogy a v vektort kibővítjük p-vel. Így w egy $k+m = n$ hosszú kódszó lesz.

Egy (ρ, γ)-reguláris RA kód gráfjában minden bitcsúcs ρ ellenőrzőcsúcshoz csatlakozik.

3. ábra. Egy RA kód Tanner gráfja [6]



7. Dekódolás

LDPC kódok használata esetén egyértelműen a dekódolás a legnagyobb számításigényű és komplexitású feladat, ezért egy helyesen megválasztott dekódoló módszer jelentősen felgyorsítja az adattovábbítás folyamatát.

Az dekódolási algoritmusok alapvetően két nagy csoportba sorolhatóak; egyrészt a kemény, vagy Bit Flipping (bitváltó), másrészt a lágy, vagy Message Passing (üzenet továbbító) eljárásokra. A kemény dekódolás alacsonyabb komplexitású, gyorsabb algoritmust eredményez, de ennek következtében kevésbé pontos. A kemény dekódoló módszerek minden üzenetkoordináta esetén kiválasztják a legvalószínűbb lehetséges értéket, és ezt rendelik a koordinátához. A lágy dekódolás esetén viszont konkrét értékek helyett valószínűség-eloszlásokkal dolgozunk, tehát a lehetséges értékek mellett a koordináták azt az információt is hordozzák, hogy melyik érték

mekkora valószínűséggel lesz a helyes megoldás. Mindkét eljárás iteratív, minden iterációban javítjuk az aktuális hibákat.

A dekódoló algoritmusok ezen két nagy csoportján belül többféle eljárás is használatos. Ezek többnyire egymás finomításai, azaz kisebb módosításoktól eltekintve egy alapötletre támaszkodnak. Az, hogy mikor melyik módszert célravezető használni, függ a kód paramétereitől, a csatornától és a gyakori hibák jellegétől is.

A dolgozatban a kemény és a lágy dekódoló eljárások közül is a Gallager által javasoltat mutatjuk be. [3] [4] [1] [2]

7.1. Kemény dekódolás

Tegyük fel, hogy zajos csatornán keresztül egy adott (n, m) bináris LDPC kód v kódszavát továbbítjuk, és legyen a w vektor a beérkezett üzenet, $H \in \mathbb{R}^{k \times n}$ a paritásellenőrző mátrix. A w vektor segítségével kiszámítható az $s = Hw^T$ vektor, ezt szindrómavektornak nevezzük. Minden beérkezett w_i üzenetbit legfeljebb k szindrómaelemben vesz részt, mivel az i-edik bit k paritásellenőrzésben szerepel.

A Gallager-féle kemény dekódoló algoritmus lépései a következők:

1. Kiszámítjuk az s szindrómavektort, és megkeressük a vektor nemnulla komponenseit. Ezek a komponensek jelzik, hogy mely paritásellenőrzések adtak hibás eredményt.

2. A beérkezett n bit mindegyikére kiszámítjuk, hogy hány rossz eredményt adó paritásellenőrzésben vesz részt.

3. A legtöbb rossz eredményt adó paritásellenőrzésben részt vevő bitet megváltoztatjuk. Ha több ilyen bit is adódik, akkor mindegyiket megváltoztatjuk. Legyen w most az így kapott új vektor.

4. Addig ismételjük az eljárás 1-3. pontjait, amíg $Hw^T = 0$ eredményt nem kapunk, vagy amíg az iterációk száma el nem ér egy előre meghatározott értéket. Előbbi esetben az üzenetet dekódoltnak tekintjük, a dekódolt érték az új w vektor. Utóbbi esetben a dekódolás sikertelen.

A következő példákban a fenti algoritmust használva próbáljuk visszakapni a w üzenetből a v kódszót, ha a paritásellenőrző mátrix a következő:

$$H = \left[\begin{array}{cccc|cccc|cccc|cccc}
1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
\hline
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\
\hline
1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0
\end{array}\right]$$

7.1. Példa.
Legyen $w = [0100101000100000]$ a beérkezett üzenet. Ekkor $Hw^T = [101011001001]^T$. A vektor hat komponense nemnulla, tehát hat paritásellenőrzés helytelen eredményt adott. Ezek a H mátrix 1., 3., 5., 6., 9. és 12. sorához tartoznak. Az 1. és a 10. bit ezek közül három helytelen eredményt adó paritásellenőrzésben szerepel, a többi bit legfeljebb kettőben, így ezt a két bitet kell módosítanunk. Ekkor a $w = [1100101001100000]$ vektort kapjuk.

A következő iterációban ismét elvégezzük a mátrixszorzást, ami a $Hw^T = 0$ eredményt adja, azaz w egy kódszó.

7.2. Példa.
Legyen most $w = [1100101000000000]$, ekkor $Hw^T = [000001101001]^T$. A 6., 7., 9. és 12. sorhoz tartozó paritásellenőrzések nem adtak helyes eredményt, ezekben a 6., 10., 11. és 15. bitek kétszer szerepelnek, míg a többi bit legfeljebb egyszer. Ezeket a biteket megváltoztatva $w = [1100111001100010]$.

A következő iterációban $Hw^T = [010101101001]^T$ a mátrixszorzás eredménye, ekkor a 2., 4., 6., 7., 9. és 12. sorokhoz tartozó ellenőrzések jeleznek hibát. A 6. és a 15. bitet kell módosítanunk, mivel ezek háromszor szerepelnek, míg a többi bit legfeljebb kétszer. Az új vektor a $w = [1100101001100000]$, amire $Hw^T = 0$, tehát sikeresen dekódoltuk az üzenetet.

7.3. Példa.
Legyen végül $w = [0100101001001000]$. A szorzás a $Hw^T = [101100100100]^T$ eredményt adja. Öt rossz eredményt adó paritásellenőrzést kapunk, ezekben a 2., 3., 7., 11., 12., 13. és 15. bitek kétszer szerepelnek, a többi nullaszor, vagy egyszer. Mindegyik érintett bitet megváltoztatjuk, így $w = [0010100001110010]$ lesz az új vektorunk, azaz $Hw^T = [111111111111]^T$. Ez azt jelenti, hogy az összes paritásellenőrzés helytelen eredményű, és mindegyik bit három

ellenőrzésben szerepel. w minden elemét módosítva azonban ismét a $Hw^T = [111111111111]^T$ oszlopvektort kapjuk, így a következő iterációban a bitcserék után visszakapjuk az eredeti $w = [0100101001001000]$ vektort. A következő lépésekben az első és a második iteráció váltakozik ciklikusan, ebben a példában tehát nem dekódolható az üzenet a kemény dekódoló algoritmussal.

A kemény dekódoló algoritmusnak létezik egy másik változata is, melyben egyszerre mindig csak egy bitet cserélünk fel. Ha több bit is felcserélhető, akkor mindig a legkisebb indexűt választjuk.

7.2. Lágy dekódolás

Kemény dekódolás esetén az egyes bitekhez 0 vagy 1 értéket rendelünk, a lágy dekódoló algoritmus használatakor azonban valószínűség-eloszlásokat rendelünk minden bithez, ezek határozzák meg, hogy melyik értéket mekkora valószínűséggel veszik fel az egyes bitek. Bináris esetben ez egy $(p_1, p_2) \in [0,1] \times [0,1]$ valós számpár, a bit értéke p_1 valószínűséggel 0, p_2 valószínűséggel 1, és $p_1 + p_2 = 1$.

Tegyük fel ismét, hogy zajos csatornán keresztül egy adott (n, m) bináris LDPC kód v kódszavát továbbítjuk, és legyen a beérkezett üzenet a w vektor.

Az összeg-szorzat algoritmus lépései a következők:

1. Először ellenőrizzük, hogy az eloszlásokat a megfelelő konkrét értékekkel helyettesítve kódszót ad-e megoldásul a paritásellenőrző mátrixszal való szorzás. Ha igen, készen vagyunk. Ha nem, akkor a következő lépésekben újra az eloszlásokkal dolgozunk.

2. Elkészítjük a kód H paritásellenőrző mátrixához tartozó Tanner gráfot. Legyen T a bitcsúcsok halmaza, S az ellenőrzőcsúcsok halmaza. Legyen $p_{t,s}$ a $t \in T$ bitcsúcsból az $s \in S$ ellenőrzőcsúcsba üzenetként továbbított eloszlás, $q_{s,t}$ pedig az s csúcsból a t csúcsba küldött eloszlás. Legyen továbbá $T(s)$ az s csúccsal szomszédos bitcsúcsok halmaza, $S(t)$ pedig a t csúccsal szomszédos ellenőrzőcsúcsok halmaza. A t bitcsúcsban eltároljuk a w_t eloszlást.

3. Minden s ellenőrzőcsúcsban kiszámítjuk a beérkezett eloszlások konvolúcióit, majd az eredményül kapott eloszlásokat visszaküldjük a bitcsúcsoknak. Ekkor

$$q_{s,t} = \bigotimes_{\substack{u \in T(s) \\ u \neq t}} p_{u,s} \, .$$

   A $\otimes$ szimbólum eloszlások konvolúcióját jelöli, azaz ha a és b valószínűség-eloszlások, akkor

$$a \otimes b(0) = a(0)b(0) + a(1)b(1) \quad \text{és} \quad a \otimes b(1) = a(0)b(1) + a(1)b(0) \, .$$

4. A következő lépésben kiszámítjuk az ellenőrzőcsúcsoktól újonnan kapott eloszlások Hadamard-szorzatait, majd az eredményeket normáljuk, hogy ismét eloszlásokat kapjunk. Ezeket az új $p_{t,s}$ értékeket küldi vissza minden $t \in T$ bitcsúcs az $s \in S$ ellenőrzőcsúcsoknak. Kiszámításuk a következőképp történik:

$$p_{t,s} = \frac{1}{N} w_t \prod_{\substack{v \in S(t) \\ v \neq s}} q_{v,t} \text{ , ahol } \quad N = w_t(0) \prod_{\substack{v \in S(t) \\ v \neq s}} q_{v,t}(0) + w_t(1) \prod_{\substack{v \in S(t) \\ v \neq s}} q_{v,t}(1) \, .$$

   Ezzel párhuzamosan az összes t bitcsúcsban tárolt w_t eloszlást is frissítjük. Ezek új értéke:

$$w_t' = \frac{1}{N} w_t \prod_{v \in S(t)} q_{v,t} \text{ , ahol } \quad N = w_t(0) \prod_{v \in S(t)} q_{v,t}(0) + w_t(1) \prod_{v \in S(t)} q_{v,t}(1) \, .$$

5. Az előző lépéseket addig ismételjük, míg az első pontban kódszót kapunk a mátrixszorzás eredményéül, vagy amíg az iterációk száma el nem ér egy előre meghatározott küszöböt. Előbbi esetben az eredményül kapott vektor a dekódolt üzenet, utóbbi esetben a dekódolás sikertelen.

A következő példában a lágy algoritmus használatát mutatjuk be ábrák segítségével.

7.4. Példa.
Legyen a beérkezett üzenet

$$w = [(0.9, 0.1), (0.6, 0.4), (0.1, 0.9), (0.1, 0.9), (0.8, 0.2), (0.8, 0.2), (0.6, 0.4)],$$

a paritásellenőrző mátrix

$$H = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

A w üzenethez a kemény dekódoló algoritmus a $w' = [0011000]$ bináris vektort rendelné.

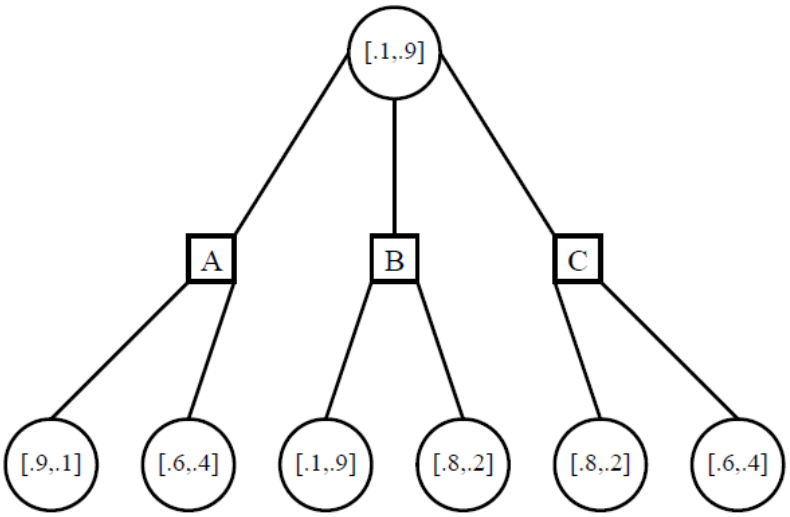
$$Hw'^T = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 0 \end{bmatrix}^T = [101]^T \neq 0$$

w tehát nem lehet kódszó.

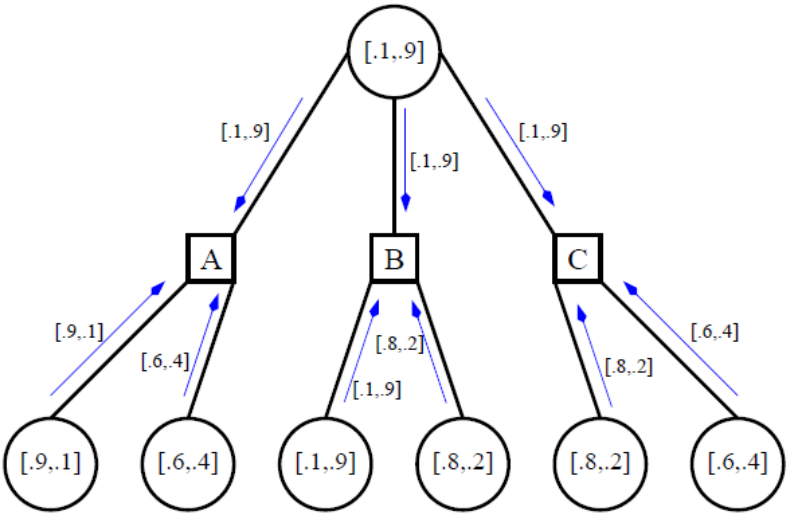
A 4. ábrán a paritásellenőrző mátrixhoz tartozó Tanner gráf látható. A 7 bitcsúcsban eltároljuk a hozzájuk tartozó eloszlásokat, ezeket kapják meg az A, B, C ellenőrzőcsúcsok az 5.

ábrán.

4. ábra. Inicializálás, kezdeti eloszlások eltárolása a bitcsúcsokban [4]

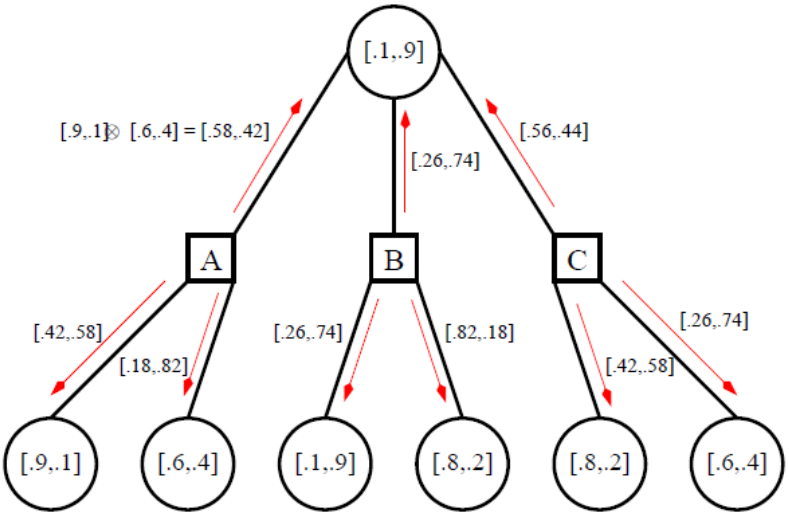


5. ábra. Üzenetküldés a bitcsúcsokból az ellenőrzőcsúcsokba [4]



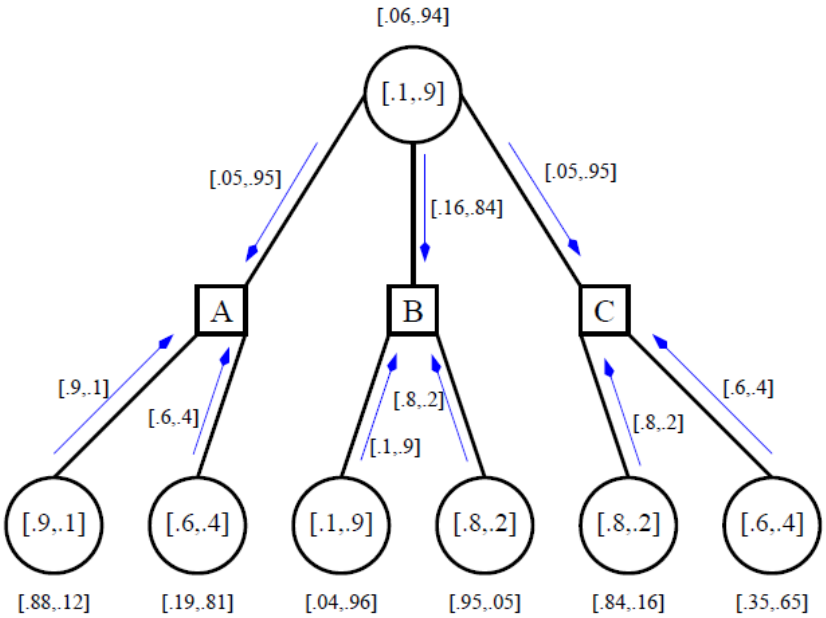
A 6. ábrán az ellenőrzőcsúcsok kiszámítják a kapott eloszlások konvolúcióit, majd továbbítják azokat a megfelelő bitcsúcsoknak.

6. ábra. Konvolúciók kiszámítása és visszaküldése a bitcsúcsokba [4]

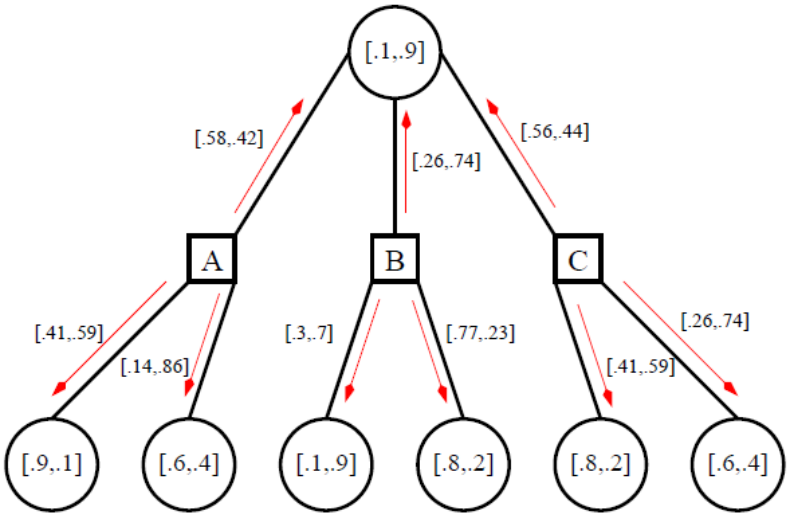


Ezután a 7. ábrán a bitcsúcsok újra üzenetet küldenek az ellenőrzőcsúcsoknak, ezek tartalmazzák az előzőleg beérkezett eloszlásokból számított Hadamard-szorzatokat. Ugyanekkor a bitcsúcsokban eltárolt eloszlásokat is frissítjük az algoritmus 4. lépésében ismertetett módon, ezzel elérkezünk a következő iterációhoz. A 8. ábrán újra az ellenőrzőcsúcsok üzennek a bitcsúcsoknak.

7. ábra. Hadamard-szorzatok kiszámítása és küldése az ellenőrzőcsúcsokba, kezdeti eloszlások frissítése [4]

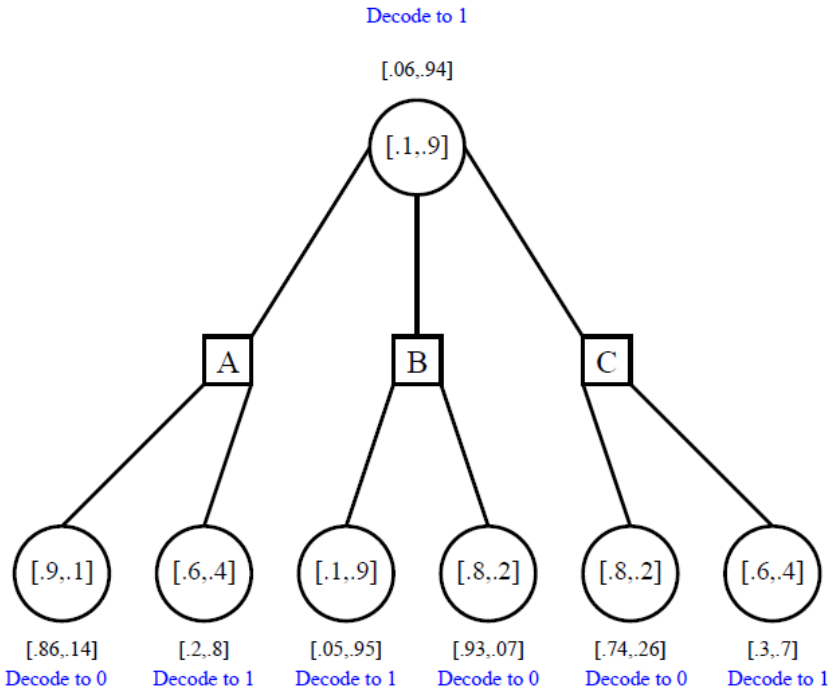


8. ábra. Üzenetküldés a bitcsúcsokból a 2. iterációban [4]



A második iteráció végén a
$w' = [(0.86, 0.14), (0.2, 0.8), (0.06, 0.94), (0.05, 0.95), (0.74, 0.26), (0.93, .07), (0.3, 0.7)]$ vektor adódik, ez a kemény algoritmust használva a $w'' = [0111001]$ bináris vektornak felel meg.

9. ábra. Eloszlások frissítése a 2. iteráció végén [4]



Ha ekkor elvégezzük a Hw''^T szorzást, akkor a nullvektort kapjuk eredményül, tehát w kódszó, így leáll az algoritmus.

Megjegyezzük, hogy ebben a példában már a 7. ábra alapján ugyanazt az eredményt kaphattuk volna, mint végül a 9. ábrára hagyatkozva. A 7. ábra eloszlásai ugyanazokra az értékekre

kerekíthetők, mint a 9. ábrán szereplő eloszlások, az algoritmus viszont csak az iterációk végén ellenőrzi a leállási feltételt.

8. LDPC kódok teljesítménye

Az LDPC kódok széleskörű alkalmazhatóságának fő oka, hogy rendkívül jól kihasználják a csatorna adottságait, amin a kódolt üzenetet küldjük.

Felfedezésük előtt a Turbo kódok különlegességének számított, hogy csaknem teljes mértékben továbbítani tudják azt az optimális információmennyiséget, melyet a Shannon-tétel határoz meg. Mint azóta kiderült, az LDPC kódok szintén rendelkeznek ezzel a képességgel, sőt, jobban meg is közelítik a maximumot, mint a Turbo kódok.

Az LDPC kódok teljesítményvizsgálatakor egyrészt azt elemezzük, hogy mennyire vagyunk közel a Shannon-korláthoz, másrészt azt, hogyan változik a kód viselkedése, ha változtatjuk a csatorna tulajdonságait. [4]

8.1. Ábrázolás

Az LDPC kódok teljesítményét általában vízesés diagramokon ábrázoljuk. A kód minőségét ebben az esetben úgy jellemezzük, hogy a bithibaarányt ábrázoljuk a jel-zaj viszony függvényében.

8.1. Definíció.
A bithibaarány (BER, azaz bit-error ratio) a beérkezett üzenetben (azaz a dekódolás eredményében, ami küldött kódszótól esetleg eltérő kódszó) lévő hibás betűk számának és az összes beérkező betű számának a hányadosa.

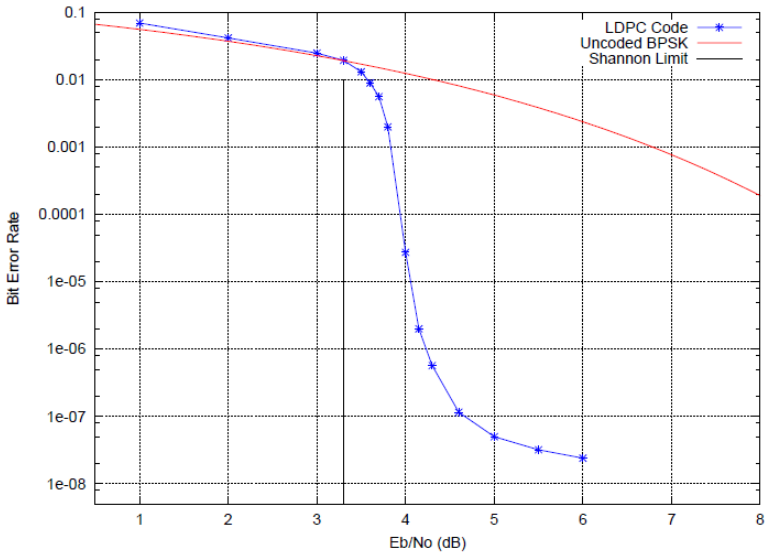
8.2. Definíció.
A jel-zaj viszony (SNR, azaz signal-to-noise ratio) a csatorna minőségét jellemző mennyiség. A csatorna által egységnyi idő alatt továbbított üzenetmennyiség és zajmennyiség hányadosa. Magas SNR esetén a csatorna hatékonyan továbbítja az üzenetet.

Amikor egy LDPC kódot vagy dekódert vizsgálunk, jellemzően rengeteg tesztfutást végzünk egymás után. Ennek egyik fő oka, hogy LDPC kódok vizsgálatára nem létezik olyan elméleti módszer, ami pontosan megjósolná a kód viselkedését. A Shannon-tétel csak azt mondja ki, hogy megfelelően megválasztott kódolással tetszőlegesen közel kerülhetünk a csatorna kapacitásához, de nem ad semmilyen támpontot ahhoz, hogy hogyan találhatunk ilyen tulajdonságú kódokat. Így a teljesítményvizsgálat egyetlen megbízható módja a tesztelés. Emellett sok esetben igen kicsi a hibák valószínűsége a dekódolás során, ezért szignifikáns eredményt csak nagy mennyiségű adat kielemzésével kaphatunk.

A tesztekhez modellezünk egy csatornát előre meghatározott jel-zaj viszonnyal. Ezután véletlenszerűen előállítunk egy üzenetet, amit kódolunk. A csatorna jel-zaj viszonyát figyelembe véve szintén véletlenszerűen zajt generálunk, ami módosítja az üzenetet. Az így keletkezett vektort tekintjük a beérkezett üzenetnek, és ezt dekódoljuk. Végül ellenőrizzük, hogy az eredeti üzenet és a dekódolt üzenet hány helyen tér el egymástól. Kellően nagy számú tesztelés alapján kapjuk meg az adott SNR melletti BER értéket.

A tesztfolyamatot többször egymásután elvégezzük ugyanazzal a kóddal és dekóderrel, de különböző zajszintekkel.

10. ábra. Egy LDPC kód teljesítménye különböző zajszintek mellett [4]



A vízesés diagramon általában háromféle görbét ábrázolunk. Egyrészt a Shannon-korlátot, amit itt egy függőleges egyenes jelez. A pirossal jelzett görbe a kódolás nélküli hibaarányt jelenti, ez tehát egyfajta referencia. A kék görbe a kódolt üzenetek hibaarányát reprezentálja. Ha a kékkel jelölt görbe kellően meredek, azaz közel van a Shannon-korláthoz, akkor a kód jól közelíti a csatorna kapacitását, tehát jó teljesítményűnek tekinthetjük.

A diagramon látható, hogy a jel-zaj viszony növelésével a hibaarány gyorsan csökken, majd ellaposodik a görbe. A lapos részt hibapadlónak hívjuk. Az, hogy a görbe meredeksége csökken ezen a részen azt jelenti, hogy a jel-zaj viszony növelése (azaz a csatorna átviteli minőségének javulása) esetén a kód és a dekódoló algoritmus nem javítja lényegesen hatékonyabban a hibákat. Ebben a tartományban tehát nem praktikus a vizsgált LDPC kódot használni, hiszen a hibák aránya nem fog jelentősen tovább csökkenni. A kód olyan SNR értékek mellett működik hatékonyan, amelyeknél a grafikonja meredek, ez a terület a vízesés tartomány.

A hibapadló elhelyezkedése azonban nem csak az előbb említett ok miatt fontos. A bithibaarány tengelyt vizsgálva láthatjuk, hogy a görbe meredeksége csak viszonylag alacsony BER értékek esetén kezd csökkenni, tehát a kód akkor is jól javítja a hibákat, ha azok előfordulási

valószínűsége kicsi. Épp ezért az iparban alkalmazott LDPC kódok létrehozásakor törekszünk arra, hogy a hibapadló minél alacsonyabb BER értéknél kezdődjön.

LDPC kódok teljesítményének javítására többféle módszer is létezik. Nem csak a hibapadló elhelyezkedését változtathatjuk meg, hanem például némiképp javítható a dekódolás sebessége, és a hibajavító képesség is a következő módszerekkel:

1. Érdemes a kód n hosszát nagynak választani, mivel a hosszabb kódok jobban teljesítenek.
2. Szintén elősegíti a gyorsabb dekódolhatóságot, ha a kódhoz tartozó Tanner gráf bősége nagy.
3. Több hibát javíthat a kód, ha növeljük a minimális távolságát.
4. Egy jól megválasztott speciális dekódoló algoritmussal megnyújthatjuk a kód vízesés tartományát, így a hibapadló alacsonyabb bithibaaránynál jelentkezik.

8.2. Csapdák

AZ LDPC kódok hibapadlójának helye összefügg a Tanner gráf szerkezetével. Egyes részgráfok okozhatják a dekódolás sikertelenségét, ezek a csapdák. [7]

8.3. Definíció.
Legyen $TG = (T, S)$ a C kód Tanner-gráfja. Legyen $1 \leq \tau \leq |T|$ és $1 \leq \sigma \leq |S|$. Egy (τ, σ) csapda olyan $t \in T$ bitcsúcsokból álló halmaz, melyre teljesül, hogy

1. elemszáma τ,
2. a halmazt alkotó t bitcsúcsok és a velük szomszédos s ellenőrzőcsúcsok TG-nek olyan részgráfját feszítik, melyben a páratlan fokú ellenőrzőcsúcsok száma σ, a páros fokú ellenőrzőcsúcsok száma tetszőleges.

8.4. Definíció. Eleminek nevezzük azt a (τ, σ) csapdát, melyre teljesül, hogy

1. elemszáma τ,
2. pontosan σ olyan ellenőrzőcsúcs tartozik hozzá, melynek foka 1,
3. a többi ellenőrzőcsúcs foka 2.

Megfigyelhető, hogy dekódoláskor néha akkor is elérjük az előre meghatározott maximális iterációszámot, ha viszonylag kevés hibás bitet kell javítanunk. A jelenséget a csapdák okozzák.

Ha az üzenetben továbbítás közben egy (τ, σ) csapda τ bitje megsérül, akkor dekódolás közben σ helytelen eredményű paritásellenőrzést kell kijavítanunk. Ez alapesetben csak annyit jelentene, hogy hosszabbodik a folyamat, hiszen újabb iterációkra van szükség a javításhoz.

Azonban ha nem jól választjuk meg a dekódoló eljárást, akkor az algoritmus ezen a ponton „csapdába esik", minden újabb iterációban helytelenül módosítja a hibás biteket. Adhatja mindig ugyanazt a rossz eredményt, vagy oszcillálhat több hibás érték körül is.

Az összeg-szorzat algoritmus és a hozzá hasonló lágy módszerek különösen érzékenyek a csapdákra, mert nem globálisan kezelik az üzenetet dekódolás közben, hanem a Tanner gráf ágaiban szétosztva a biteket kisebb részekre bontják a problémát. Amikor egy csapda bitjeihez ér a dekódolás, akkor valamelyik paritásellenőrző csúcs minden bitcsúcsból helytelen valószínűségeloszlást kap, így egyik iterációban sem jutunk jó eredményre. A dekódolás végül sikertelen lesz.

A megfigyelések alapján a hibapadló elhelyezkedését leginkább az elemi csapdák befolyásolják.

A csapdák okozta problémák kiküszöbölése egyelőre nem teljesen megoldott feladat. Egyes dekódoló eljárások kevésbé érzékenyek rájuk általánosan, míg más algoritmusok le tudnak kezelni egy bizonyos méretű csapdát, de eltérő méretűekben elakad a dekódolás. Egy egyszerű, de időigényes megoldás lehet, ha több dekóderrel is elvégezzük a tesztelési folyamatot.

9. Összefoglalás

A dolgozat a hibajavító kódolással, azon belül pedig a lineáris blokk-kódok csoportjába tartozó Low-Density Parity-Check, vagy LDPC kódokkal foglalkozott.

Az általános témájú bevezetés után végigvettük a lineáris kódokhoz kapcsolódó alapvető fogalmakat, és példákat is adtunk. A következő fejezetekben bemutattuk az LDPC kódokat, majd néhány lehetséges konstrukcióval és két alapvető dekódoló módszerrel foglalkoztunk. A dolgozat végén ismertettük azokat az eljárásokat, melyekkel megvizsgálhatjuk az LDPC kódok teljesítményét.

Az LDPC kódolás témaköre rengeteg nyitott problémát és további kutatási témát rejt. Ilyenek például a parciális lineáris terekre épített kódok témája, a nagy dimenziójú kódok minimális távolságának becslése, vagy annak a problémának a kiküszöbölése, hogy nagyon alacsony bithibaarány mellett a kódok jellemzően már a hibapadló tartományába esnek.

Ezekkel és más hasonló problémákkal is foglalkoznak jelenleg világszerte a hibajavító kódokkal dolgozó mérnökök, matematikusok és informatikusok. Munkájuknak köszönhetően valószínűleg nem kell már sokat várnunk egy újabb áttörésre például az űrkutatás területén, melyhez jó eséllyel hozzájárulhatnak majd az LDPC kódok is.

10. Irodalomjegyzék

Hivatkozások

[1] Robert G. Gallager *Low-Density Parity-Check Codes*, Monograph, M.I.T. Press, 1963

[2] D. MacKay and R. Neal, *Good codes based on very sparse matrices, in Cryp- tography and Coding*, 5th IMA Conference, 1995

[3] W. Cary Huffman, Vera Pless, *Fundamentals of Error-Correcting Codes*, Cambridge University Press, 2003

[4] Jan De Beule, Leo Storme, *Current Research Topics in Galois Geometry*, Nova Science Publishers, Inc., 2011

[5] Yu Kou, Shu Lin, Marc P.C. Fossorier, *Low Density Parity Check Codes Based on Finite Geometries: A Rediscovery and New Results*, IEEE Transactions on Information Theory, 2001

[6] Shu Lin, Daniel J. Costello, *Error Control Coding*, Prentice-Hall, 2004

[7] Qin Huang, Qiuju Diao, Shu Lin, Khaled Abdel-Ghaffar, *Trapping Sets of Structured LDPC Codes*, IEEE International Symposium on Information Theory Proceedings, 2011

[8] Maximilian Stark, *Information Optimum Design of Discrete LDPC Decoders for Irregular Codes*, Hamburg University, 2017

[9] Thomas M. Cover, Joy A. Thomas, *Elements of Information Theory*, John Wiley and Sons, New York, 1991

[10] William E. Ryan, Shu Lin, *Channel Codes: Classical and Modern*, Cambridge University Press, 2009

[11] Szőnyi Tamás, *Szimmetrikus Struktúrák*, Typotex, 2014

[12] Kollár Zsolt, *LDPC kódolás - Kutatási beszámoló*, BME Villamosmérnöki és Informatikai Kar Szélessávú Hírközlés és Villamosságtan Tanszék, 2013

[13] Kiss Emil, *Algebra 3 jegyzet*, (http://ewkiss.web.elte.hu/html/bboard/07o.el/tem.html)

[14] Láng Csabáné, *Hibajavító kódolás jegyzet*, (http://compalg.inf.elte.hu/ zslang)

[15] Varga László, *Valószínűségszámítás 1 jegyzet*, (http://vargal4.elte.hu)