5. Forráskódolás és hibavédő kódolás

5.1. Példa: forráskódolás

Egy (szimbólumonként kódolt) forrás legtömörebb bináris kódjában a kódszavak hossza rendre 2,3,3,3,3,4,4,4,5,5.

a) Lehet-e ez a kód egyértelműen megfejthető kód?

b) Mekkora a forrás entrópiája, ha tudjuk, hogy egyenlő a kód átlagos szóhosszúságával?

Megoldás:

a) A Kraft egyenlőtlenség egyenlőséggel teljesül, tehát lehet.

b) A feladat szövege szerint: $\sum_{i=1}^{10} p_i \cdot ld\left(\frac{1}{p_i}\right) = \sum_{i=1}^{10} p_i \cdot l_i$, ez viszont akkor bizonyosan teljesül, ha $ld(1/p_i) = l_i$,ha $i = 1...10$, azaz $p_i = 2^{-l_i}$. Így a kérdezett entrópia 49/16. Igényes megoldás megemlítené: ez (még) nem bizonyítja, hogy nincs más, ugyanilyen tulajdonságú valószínűségeloszlás.

5.2. Shannon kód szerkesztése

A forrásszimbólumokat valószínűségeik szerint csökkenő sorba rendezzük, azaz $p_1 \geq p_2 \geq \ldots \geq p_n$. Shannon szerint mármost az i-edik szimbólumot megjelenítő kódszó az $F_i \in [0,1)$ szám $l_i = ld\,1/p_i$ bitre csonkolt értéke, ahol

$$F_i = \sum_{k=1}^{i-1} p_k .$$

Az alábbi táblázat egy 6 szimbólumos forrás kódjának elkészítését mutatja.

szimb./i	x1/1	x2/2	x3/3	x4/4	x5/5	x6/6
p_i	0.4	0.25	0.12	0.1	0.08	0.05
l_i	2	2	4	4	4	5
F_i	0.0	0.4	0.65	0.77	0.87	0.95
bináris F_i	0.00000	0.01100	0.10100	0.11000	0.11011	0.11110
kódszó	**00**	**01**	**1010**	**1100**	**1101**	**11110**

A Shannon kód gyakran rövidíthető. Példánkban látható, hogy az utolsó szimbólumhoz tartozó kódszó utolsó eleme elhagyható, a kód ettől még prefix (mentes) marad. Ha sort tudunk rá keríteni - elég időnk van - célszerű kiszámítani a kód átlagos szóhosszát, s összevetni az entrópiával [Az entrópia értéke 2.235, az átlagos kódszóhossz pedig 2.75].

Azt, hogy ezzel az eljárással prefix (azaz prefix-mentes) kódot állítunk elő, egyszerűen lehet bizonyítani. A kódszavak elé egy bináris pontot képzelve valamennyi kódszó megfelel egy 0-1 közötti számnak. Vegyük észre, hogy az l hosszú a szám (kódszó) csak azoknak a b kódszavaknak (számoknak) lehet az előtagja, amelyekre

$$b < a + 2^{-l}.$$

Az l_{i-1} hosszúra csonkolt F_{i-1} tehát csakis olyan számoknak lehet az előtagja, amelyekre

$$b < csonkolt(F_{i-1}) + 2^{-l_{i-1}} \leq F_{i-1} + 2^{-l_{i-1}} \leq F_{i-1} + p_{i-1} = F_i .$$

Tehát: az l_{i-1} hosszú i-1-edik kódszó nem lehet előtagja sem az i-ediknek, sem pedig a többieknek. Másrészről a magasabb indexű kódszavak sem lehetnek előtagjai alacsonyabb indexűeknek, hiszen a kódszavak hossza monoton nemcsökkenő sorozatot alkot.

5.3. Jellegzetes zéhá példa

Egy diszkrét, emlékezetnélküli, véletlen forrás szimbólumait a P = {0.5, 0.25, 0.15, 0.1} valószínűségeloszlás szerint szolgáltatja. Pistike alkot egy kódot, amelyben a kódszavak a szimbólumok fenti sorrendjében a következőek: (01), (10), (011), (1011).

a) Egyértelműen dekódolható-e a fenti kód? Indokolja válaszát!

b) A fenti kódhosszúságokkal lehet-e prefixmentes kódot konstruálni?

c) Határozza meg a várható kódszóhosszat! Milyen messze esik ez az érték a tömöríthetőség elvi alsó határától? (Adja meg az eltérést %-ban!)

Megoldás:

a) Tekintve, hogy az első kódszó előtagja a harmadiknak, s a második is a negyediknek, az a gyanúnk ébredhet, hogy az egyértelmű megfejthetőséggel is baj lehet. Valóban, ha például az adó kétszer egymás után a harmadik kódszót adja: 011 011, akkor a kódolt jelnek ezt a szakaszát akár 01 1011-nek is olvashatjuk. Pistike kódja tehát nem egyértelműen megfejthető kód. Megjegyezzük, sok olyan kód létezik, ami noha nem prefix (mentes), ennek ellenére egyértelműen megfejthető.

b) A Kraft egyenlőtlenség teljesül a 2, 2, 3, 4 számokra, hiszen $1/4+1/4+1/8+1/16 < 1$, ezért e szóhosszakkal lehet prefix kódot szerkeszteni, pl. (00), (01), (100), (1100).

c) A várható kódszóhossz: $L = 0.5\times2 + 0.25\times2 + 0.15\times3 + 0.1\times 4 = 2.05$

A tömöríthetőség alsó határát az eloszlás entrópiája adja, ez most

$H(P) = 0.5\times\mathrm{ld}\,(1/ 0.5) + 0.25\times\mathrm{ld}(1/ 0.25) + 0.15\times \mathrm{ld}(1/ 0.15) + 0.1\times \mathrm{ld}(1/ 0.1)$.

Nekem így 1.7428 jött ki.

5.4. Hibavédelem blokk kóddal

Adott egy lineáris kód a kódszavaival: $c_1 = 00000$, $c_2 = 10100$, $c_3 = 01110$, $c_4 = 11010$.

a) Adja meg a kód generátormátrixát!

b) Határozza meg az egyhibás átvitelhez tartozó szindrómákat!

c) Létrehozható -e egyetlen kódszó módosításával olyan lineáris kód, amely minden egyhibát javít?

Megoldás:

a) Mivel a kódszavak száma 4, az üzenetvektorok mérete k=2, így a generátormátrixnak két sora (és természetesen 5 oszlopa) van. Bázisvektornak csak c_1 nem választható, a többiek közül bármelyik kettő megfelel. Pl. $G = \{c_2, c_3\} = \{01110, 10100\}$. Ez a generátormátrix ráadásul szisztematikus kódot állít elő.

b) Állítsuk elő a paritásellenőrző mátrix transzponáltját: $H^T = \{001, 010, 100, 110, 100\}$. E mátrix sorai éppen az egyhibás hibamintákhoz tartozó szindrómákat tartalmazzák. Látható, hogy van két egyező sor, tehát van két olyan egyhibás helyzet, amelyeket ez a kód nem tud megkülönböztetni (első és harmadik pozíció hibája).

c) Nem. A csupa nulla kódszó eleme kell legyen a kódnak, hiszen a kód lineáris, tehát az nem módosítható. A maradék három kódszó bármelyikében bármit változtatunk, valamelyik másik kódszónak is változni kell, ha a kód linearitását meg akarjuk őrizni. Ha pl. a paritásellenőrző mátrix transzponáltjának első sorát módosítjuk, legyen az most (101), akkor módosul c_2, új értéke $c_2 = 10101$, de változik c_3 is, hiszen az c_1 és c_2 összege kell legyen.

5.5. Példa: hibavédelem blokk kóddal

A bináris, lineáris és szisztematikus (23,12) Golay kód 6 hiba jelzésére képes.

a) Mekkora lehet e kód minimális távolsága?

b) Hány hiba javítására lehet alkalmas ez a kód?
c) Hány kódszava van ennek a kódnak?
d) Hányféle szindróma képződhet e kód alkalmazásánál? Hány szindróma jelez egy-, két- és háromhibás hibamintát?
e) Felismerhető-e, ha a valamelyik kódszót négy hiba éri?
f) Mely elemei hibásodhattak meg a kódszónak, ha a dekódolásnál számolt szindróma

$$s = (\overset{1}{0}\overset{2}{0}\overset{3}{1}\overset{4}{0}\overset{5}{0}\overset{6}{0}\overset{7}{0}\overset{8}{0}\overset{9}{1}\overset{10}{0}\overset{11}{0}) \ ?$$

Megoldás:
[A Golay-kód egy olyan perfekt kód, amely több hibát (nevezetesen 3-at) javít. A Golay-kódok kapcsán érdekességként érdemes megemlíteni, hogy az 1977-beli Voyager expedíció (melynek célja a Jupiter és a Szaturnusz vizsgálata volt), a G_{24} bináris Golay-kódot használta. A képet itt 800*800 darab 8 bites képpontra osztották. A Voyager II. szonda is a Golay-kóddal küldte képeit, de több évvel elindulása után, mikor rájöttek, hogy a szondát továbbirányíthatják az Uránusz, majd a Neptunusz felé, akkor a CD-n és újabban a DVD-n is használt Reed-Solomon kódra tértek át. Ezt azért tették, mert a még távolabbról jövő egyre gyengülő jelekhez több hiba kijavítására volt szükség.]
a) 6 hiba garantáltan akkor jelezhető, ha a kódtávolság (legalább) 7.
b) Ha a kódtávolság 7, akkor 3 hiba garantáltan javítható.
c) Az üzenetvektorok 12 bitesek, tehát a kódszavak száma $2^{12} = 4096$.
d) A szindróma 23-12=11 bites, tehát 2048 féle szindróma van. 23 féle egyhibás helyzet lehetséges, $23 \cdot 22 / 2 = 253$ féle kéthibás helyzet van, a háromhibás lehetőségek száma pedig $23 \cdot 22 \cdot 21 / (2 \cdot 3) = 1771$. [Láthatjuk, hogy az összes szindrómavektort elhasználtuk.]
e) A hibás kódszó ténye felismerhető, az azonban, hogy a hibás kódszó négy vagy három hiba következtében jött-e létre, már nem állapítható meg, ugyanis ez a kód perfekt kód. Ez azt jelenti, hogy a legfeljebb három hibás kódszavak teljesen kitöltik a teret ($1+23+253+1771=2048$), s így egy kódszó bármely negyedik szomszédja biztosan harmadik szomszédja valamelyik másik kódszónak. [A perfekt kód jelentése az lenne, hogy a kódszavai egyenlő távolságra vannak egymástól. Tulajdonképpen ennek lesz a következménye az, hogy nem is tudunk négyhibás eseteket detektálni, hiszen azok megfelelnek egy háromhibás esetnek.]
f) Feltételezzük, hogy a paritásellenőrző mátrix $\boldsymbol{H} = \left(\boldsymbol{B}^T \boldsymbol{I}\right)$ alakú. A szindróma úgy keletkezik, hogy a vett sorozatot szorozzuk e mátrix transzponáltjával, s ez végső soron egyenlő lesz az $\boldsymbol{eH}^T$ szorzattal, ahol $\boldsymbol{e}$ a hibaminta. Rá lehet jönni, ha a hibaminta 12+3=15. és 12+9=21. eleme 1 értékű, akkor éppen a megadott szindróma keletkezik, hiszen ezeket az elemeket a mátrix egységmátrix almátrixa szorozza.
Megjegyzés: rá lehet mutatni arra is, hogy a paritásellenőrző mátrix ezen megválasztása egy lehetőség, de nem szükségszerű. Az a fontos tulajdonsága, hogy a kódszavakat transzponáltjával szorozva csupa nulla szindróma (tünetmentesség!) keletkezik, megmarad, ha a transzponált mátrix oszlopait csereberéljük. Ekkor azonban nincs egységmátrix a megfelelő pozíciókban. [Továbbá egyértelmű, hogy a kódnak azon jellege miatt, hogy a javítható hibákhoz tartozó szindrómák kitöltik a rendelkezésre álló teret, *nem szabad lennie* másik olyan *javítható* hibának, ami ezt a szindrómát adja.]

5.6. Hibajavítás lineáris kóddal

Vizsgáljuk azt a bináris (8,4) kódot, amelynek generátormátrixa

$$\boldsymbol{G} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}.$$

Állapítsuk meg, szisztematikus-e ez a kód, és állítsuk elő a paritásellenőrző mátrixát! Próbáljuk meg kitalálni, hány hiba jelzésére/javítására lehet alkalmas! Képezzük azt a generátormátrixot, amely szisztematikusan állítja elő ugyanezt a kódot! Képezzük az $u = 0 \quad 1 \quad 1 \quad 0$ üzenethez tartozó kódszót, s tételezzük fel, hogy e kódszó első bitje meghibásodik! Demonstráljuk a (táblázatos) hibajavítás folyamatát!

Megoldás:
A mátrix baloldalán elhelyezkedő almátrix nem egységmátrix, tehát a kód nem lehet szisztematikus.
A mátrix sorai mind kódszavak, s legtöbbjük csak három 1-es elemet tartalmaz (a súlyuk 3). E kódszavak távolsága a csupa 0-t tartalmazó kódszótól (amely minden lineáris kódnak eleme) mindössze 3. Következésképpen e kód minimális távolsága sem lehet háromnál nagyobb. Hogy nem is kisebb, ez csak akkor látszik, ha valamennyi kódszót előállítjuk, s kikeressük közülük a minimális súlyút. [Mivel ez lineáris kód, elegendő egy kiválasztott kódszótól való távolságot vizsgálni, ez pedig legegyszerűbb a csupa 0-t tartalmazó kódszóhoz képest tenni.] Az a kód, amelynek a kódtávolsága 3, 2 hibát jelezni, 1 hibát javítani tud.
A mátrix két utolsó sorát összeadva a balodali egységmátrix létrehozható és ebből már könnyen megadhatjuk a paritásellenőrző mátrixot:

$$\boldsymbol{G}_{sz} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \qquad \boldsymbol{H} = \begin{bmatrix} 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Az $\boldsymbol{u}$ üzenethez tartozó kódszó:

$$\boldsymbol{c}(\boldsymbol{u}) = \boldsymbol{u} \cdot \boldsymbol{G}_{sz} = 0 \quad 1 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 .$$

A meghibásodott kódszó:

$$\hat{\boldsymbol{c}}(\boldsymbol{u}, \boldsymbol{e}) = 1 \quad 1 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 .$$

Ezt szorozzuk a paritásellenőrző mátrix transzponáltjával, hogy előállítsuk a szindrómát:

$$\boldsymbol{s} = \hat{\boldsymbol{c}} \cdot \boldsymbol{H}^T = 0 \quad 0 \quad 1 \quad 1 .$$

Már csak az lehet kérdés, melyik egyhibás hibavektornak ez a szindrómája. Természetesen annak, amelynek az első eleme 1. Érdemes megfigyelni, hogy az a kéthibás hibavektor, amelynek az *utolsó* két eleme 1, ugyanezt a szindrómát szolgáltatja.

5.7. Hibajelzés polinomkóddal

Tekintsük a (7,3) bináris polinomkódot, amelynek generátorpolinomja

$$g(x) = x^4 + x^3 + x^2 + 1 \; !$$

Állapítsuk meg, ciklikus-e az a kód, amelyet ez a generátorpolinom állít elő! Kövessük végig a szisztematikus kód kódszavának előállítását, feltéve, hogy az üzenetvektor

$$\boldsymbol{u} = 0 \quad 1 \quad 0 \; !$$

Vizsgáljuk meg a hibajelzés folyamatát is, a kódszó első bitjének meghibásodását feltételezve!

Megoldás:
A kód ciklikussága tekintetében az a meghatározó, hogy a generátorpolinom osztója-e az x^7+1 polinomnak, vagy sem. Az osztást elvégezve zérus a maradék (a hányados egyébként x^3+x^2+1), tehát a kód ciklikus. Az üzenetvektornak megfelelő polinom

$$u(x)=x \text{ . [pontosabban: } 0\cdot x^2+1\cdot x^1+0\cdot x^0]$$

A szisztematikus kódszavakat (kódpolinomokat) úgy képezzük, hogy a megfelelő mértékben "fel"tolt (x^{n-k}-val megszorzott) üzenetpolinomot olyan függelékkel egészítjük ki, amely biztosítja, hogy az eredmény osztható legyen a generátorpolinommal. E függelék természetesen

$$p(x)=rem\frac{x^{n-k}\cdot u(x)}{g(x)}.$$

A számolást elvégezve $p(x)=0\cdot x^3+1\cdot x^2+1\cdot x+1$ adódik. Így a keresett kódszó:

$$\boldsymbol{c}=\ 0\quad 1\quad 0\quad 0\quad 1\quad 1\quad 1\ .$$

Az első bitjében hibás kódszónak megfelelő polinom:

$$\hat{c}(x)=x^6+x^5+x^2+x+1\ .$$

Képeznünk kell osztási maradékát a generátorpolinommal, s ez most

$$s(x)=x^3+x^2+x\neq 0.$$

A szindróma nem zéruspolinom, tehát történt hiba.

5.8. Tömörítés a gyakorlatban

A gyakorlatban alkalmazott tömörítő eljárások másként működnek, de hatékonyságukat aszimptotikusan esetükben is a forrás entrópiája határozza meg. Ennek bizonyítása nem túl egyszerű, így erről lemondunk, de az eljárások alapötletét ismertetjük.
Tekintsük egy bináris forrás valamely hosszú, *n* bites üzenetét! Tördeljük ezt az üzenetet balról jobbra haladva olyan szeletekre, amelyek valamelyik megelőző szelet és egyetlen követő bit együtteséből állanak, és nem egyeznek meg egyik megelőző szelettel sem! A szeleteket a sorszámukkal jelölve világos, hogy a dekódolás annyiból fog állani, hogy az 1, 2, 3, stb. sorszámú szeletekhez tartozó bitsorozatokat egymás után másoljuk. Az egyes szeleteknek megfelelő bitsorozatokat egy táblázat adja meg, amely megmondja, hogy az egyes szeletek melyik szelet milyen bittel való kiegészítéseként keletkezett. Maga a kódolt üzenet tkp. éppen ez a táblázat. Az, hogy eképpen az üzeneteinket tömöríteni lehet, azon múlik, mekkora a táblázat, hány szeletre tördelendő az üzenet. Legyen ez a szám $c(n)$! Ekkor a táblázat elemei $ld(c(n))$ bites kódszavakkal azonosíthatóak, illetve a táblázat rovataiban $ld(c(n))$ bittel egy korábbi szeletre, s egy bittel az aktuális függelékre hivatkozunk. Összesen tehát a táblázat rovataiban $c(n)\cdot\left(ld(c(n))+1\right)$ bit foglal helyet. Bizonyították, hogy "tisztességes", *H* entrópiájú források üzeneteire

$$\frac{c(n)\cdot\left(ld(c(n))+1\right)}{n}\to H\ ,$$

ha az üzenetek hossza tart a végtelenhez.
Példa: tekintsük a 000111000100010111100110101000101000000001 üzenetet!
Végezzük el a szeletelést, állítsuk elő a kódolt szöveget jelentő táblázatot!

Megoldás:

0|00|1|11|000|10|001|01|111|0011|010|100|0101|0000|0001|

A táblázat:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	00	1	11	000	10	001	01	111	0011	010	100	0101	0000	0001
0,0	**1,0**	**0,1**	**3,1**	**2,0**	**3,0**	**2,1**	**1,1**	**4,1**	**7,1**	**8,0**	**6,0**	**11,1**	**5,0**	**5,1**

A kódolt üzenet tkp. a táblázat harmadik sora. A gyakorlatban persze még meg kell oldani néhány apró problémát (pl. hogyan lehet kitalálni a kódolás elején, mekkora legyen a szeletazonosító kódszavak mérete, mit tegyünk az utolsó szelettel, stb.), ezekre számos megoldás elképzelhető.