Infokommunikáció – Forráskódolás és hibatűrő kódolás

1 Forráskódolás

Jelölje $X = \{x_1, x_2, \ldots, x_n\}$ a forrásábécét, azaz a forrás által előállított betűk (szimbólumok) halmazát, és X^* a forrásábécé betűiből előállított szavakat, az üzeneteket. Hasonlóképp jelölje $Y = \{y_1, y_2, \ldots, y_s\}$ a kódábécét, és Y^* a kódábécé betűiből előállított szavakat, a kódszavakat. Ekkor az $f: X \to Y^*$ leképezést kódnak nevezzük.

Definíció Az f kód egyértelműen dekódolható, ha minden véges kódbetűsorozat legfeljebb egy üzenet kódolásával áll elő.

Definíció Az f kód prefix(mentes), ha a lehetséges kódszavak közül egyik sem folytatása a másiknak. Egy prefix(mentes) kód egyértelműen dekódolható.

Lemma (McMillan) Minden egyértelműen dekódolható $f: X \to Y^*$ kódra

$$\sum_{i=1}^{n} s^{-|f(x_i)|} \leq 1,$$

ahol s a kódábécé elemszáma és $|f(x_i)|$ jelöli az x_i szimbólumot leképező $f(x_i)$ kódszó hosszát.

Lemma (Kraft) Ha $l_1, l_2, \ldots, l_n$ pozitív számokra

$$\sum_{i=1}^{n} s^{-l_i} \leq 1,$$

akkor létezik olyan f prefixmentes kód, hogy $|f(x_i)| = l_i$, $i \in \{1, \ldots, n\}$.

Definíció Legyen p_k a k-dik szimbólum előfordulásának valószínűsége. Ekkor az f kód átlagos kódszóhosszán a $\lambda = \sum_{i=1}^{n} p_i\,|f(x_i)| = \sum_{i=1}^{n} p_i\,l_i$ értéket értjük.

Definíció Egy (emlékezetnélküli és stacionáris) forrás entrópiáján a $H(P) = \sum_{i=1}^{n} p_i\,log_2\frac{1}{p_i} = \sum_{i=1}^{n} p_i\,\mathrm{ld}\frac{1}{p_i}$ értéket értjük.

Tétel Tetszőleges egyértelműen dekódolható f kódra

$$\lambda \geq H(P)\,.$$

Tétel Létezik olyan f prefixmentes kód, amelyre

$$\lambda \leq H(P) + 1$$

A továbbiakban feltesszük, hogy a kódábécé mérete 2, azaz kódszavaink bináris "számok". Természetesen az eredmények általánosíthatóak nagyobb méretű kódábécék esetére is, erre itt a szűkös keretek miatt nem térünk ki.

Vegyük észre, hogy ha a Kraft-egyenlőtlenségben szereplő l_i kódszóhosszaknak megfeleltetjük az entrópia definíciójában szereplő $\mathrm{ld}\,1/p_i$ kifejezésnek, akkor az egyenlőtlenség biztos teljesülni fog, azaz létezik olyan prefix kód, amelynek a kódszavaira teljesül, hogy $l_i = \mathrm{ld}\frac{1}{p_i}$.

Definíció (Shannon-Fano kód szerkesztése) Rendezzük a forrásszimbólumokat valószínűségeik szerint csökkenő sorrendbe, azaz legyen $p_1 \geq p_2 \geq \cdots \geq p_n$ és legyen $w_0 = 0$, valamint $w_i = \sum_{k=1}^{i-1} p_i$, azaz w_i az $(i-1)$ legnagyobb valószínűségű érték valószínűségeinek összege. Az i-edik szimbólumot megjelenítő kódszó a w_i szám kettes számrendszerben $l_i = \left\lceil ld\frac{1}{p_i} \right\rceil$ bitre csonkolva.

Ha a forrás entrópiája kicsi (~1 bit/szimbólum), akkor az átlagos kódszóhosszra vonatkozó korlátok nem túl szorosak. Gyakorlati szempontból viszont nem mindegy, hogy egy kód átlagos szóhossza az alsó vagy a felső korlátot közelíti-e.

A korlátok azt mutatják, hogy jó hatásfokú kód nagy entrópiájú forrásoknál létezik. Lehetőségünk arra van, hogy az eredeti forrással gyakorlatilag egyenértékű, de nagy entrópiájú forrást

használjunk. Tekintsük ugyanis forrásszimbólumoknak az eredeti forrás K szimbólumos csoportjait! A kiterjesztett forrásnak n^K szimbóluma van, entrópiája pedig – a forrás emlékezetmentessége miatt – $KH(P)$. Így az eredeti K szimbólumot megjelenítő legjobb kód átlagos szóhosszára a $KH(P) \leq \lambda^{(K)} \leq KH(P) + 1$ egyenlőtlenség áll fenn. Ezzel elérhető, hogy az átlagos kódszóhossz egyetlen eredeti forrásszimbólumra vetített értéke tetszőlegesen megközelítse az eredeti forrás entrópiáját. A forrás entrópiája tehát gyakorlati szempontból teljesen meghatározza a forrás tömöríthetőségét, és ebben az értelemben jellemző a forrás üzeneteinek információtartalmára. A forráshoz igazodó legjobb kóddal megjelenített bitsorozat hossza mértéket jelenthet a forrás üzeneteinek információtartalmára.

2 Veszteségmentes és veszteséges tömörítés

A veszteségmentes tömörítési eljárások lehetővé teszik a tömörített adatból az eredeti adat pontos rekonstrukcióját. Ezeket a módszereket 3 fő csoportba sorolhatjuk: statikus módszerek, adaptív módszerek és hibrid módszerek. Statikus veszteségmentes tömörítéses módszerre példa a Huffman kódolás, míg az adaptív módszer például az LZ77, LZ78, LZW és az Adaptív Huffman kódolás. Az adaptív kódolások mind egyszer olvassák el a bemenetet.

A Huffmann-kód változó hosszúságú, optimális költségű – tehát minimális átlagos kódszóhosszú –, prefixmentes kód, amelyet David A. Huffman publikált először 1952-ben nemcsak bináris esetre, hanem rögtön az általánosítást is megadta. Az alábbiakban a bináris Huffman-kód készítésének algoritmusát adjuk meg.

A szimbólumokat gyakoriságuk szerint növekvő sorrendbe rendezzük. Ezután egy bináris fát építünk fel lépésről-lépésre a következő módon. Kiválasztjuk a sorozat két legkisebb gyakoriságú elemét, amely egy háromcsúcsú bináris fa két levele lesz (amelyeket a gyakorisággal címkézzük meg), majd ezekhez hozzárendelünk egy gyökeret, amelyet a két gyakoriság összegével címkézünk meg. Ezután a két vizsgált elemet kitöröljük a sorozatból, és azok összegét beszúrjuk az érték szerinti megfelelő helyre. Ezután folytatjuk az előző műveletet mindaddig, amíg van elem a sorozatban. Az így felépített fában a levelek az eredeti szimbólumoknak (illetve azok gyakoriságának) felelnek meg. Az eredményül kapott fában, minden csúcs esetében címkézzük meg 0-val a belőle kiinduló bal oldali élt, 1-gyel pedig a jobb oldalit. A gyökértől egy adott levélig egyetlen út halad. Ezen út éleihez rendelt 0 és 1 címkéket a fa – gyökerétől indulva – sorrendben összeolvasva, megkapjuk a levélhez rendelt szimbólum kódját és a végigjárt élek száma megadja az adott kódszó hosszát is. Az optimális prefix(mentes) kódokra teljesülnek az alábbi tulajdonságok:

- A nagyobb szimbólumgyakoriságokhoz kisebb szóhosszak tartoznak.
- A két legkisebb gyakoriságú forrásszimbólumhoz tartozó kódszó egyenlő hosszúságú, és csak az utolsó bitjükben különböznek egymástól.

Attól függően, hogy a bináris fa felépítésében egy adott lépésben melyik elem kerül balra és melyik jobbra, különböző eredményt (szimbólum-kódszó összerendelést) kaphatunk, de ez nem befolyásolja a kapott kód hatékonyságát, mint ahogy az sem, hogy az eredeti gyakorisági sorozatban lévő, vagy az összevonások során létrejövő azonos gyakoriságú elemek közül melyik kerül jobb és melyik kerül bal oldalra. Könnyen belátható, hogy amennyiben a kódszavak gyakoriságai 2 negatív hatványai, akkor a Shannon-kód és a Huffman-kód egyaránt az entrópiával megegyező átlagos kódszóhosszt eredményez.

A Lempel-Ziv kódolás eredeti két algoritmusát Lempel és Ziv javasolta 1977-ben és 1978-ban (LZ77, LZ78). Azóta a módosításokkal/variánsokkal együtt egy jelentős gyakorlati felhasználással rendelkező eljárás családdá vált. Alkalmazások: zip, gzip, stacker, GIF, V.42, compress (LZ78), stb. Az LZ78 algoritmus egy szótárat tart karban. Az algoritmus által generált kódszavak két elemből állnak: a leghosszabb illeszkedő szótárbeli elem indexe és az első nem illeszkedő szimbólum. Ha az algoritmus egy olyan szimbólumot talál, amely nincs a szótárban, akkor az index értéke 0 és a (0,

szimbólum) párt hozzáadja a szótárhoz. Az Lempel-Ziv tömörítők tehát a szótár típusú kódolók közé tartoznak, amelyek működése az adatrészek közti ismétlődő minták megfigyelésén alapul. Ezen belül az LZ tömörítő eljárások adaptív szótárt használnak. Fontos tulajdonságuk még, hogy bár nem igénylik a forrásstatisztikát, bizonyos idő alatt úgy adaptálódnak, hogy az átlagos kódszóhossz/forrásjel minimális lesz.

A veszteséges tömörítési eljárások esetén a tömörített adatból az eredeti adat nem állítható vissza pontosan, csak „elég jól". Ez a tulajdonsága határozza meg alapvetően azt, hogy milyen területen, milyen típusú és felhasználási célú adatokat tömöríthetünk ezekkel a módszerekkel. A legfontosabb felhasználási terület az emberi befogadásra szánt kép, hang és mozgókép jellegű adat tömörítése, például a valós idejű, vagy playback médiafolyam. A fő elv az, hogy a tömörítés és visszaállítás során megengedhető információvesztés, ha ez nem jár a felhasználó számára észrevehető minőségromlással. Emiatt az ilyen típusú tömörítési eljárások többnyire figyelembe veszik az emberi érzékszerveknek az emberi test anatómiai felépítéséből következő érzékenységét. Sok esetben a veszteséges tömörítési eljárások használata sokkal kisebb file-méretet eredményez, mint a veszteségmentes módszerek.

3 Hibajavító kódolás

Nézzük meg, milyen úton jut el az információ a forrástól a nyelőig:

Láthatjuk, hogy az $u \in X^*$ üzenetek a kódolóban $c \in C \subset Y^*$ kódszavakba képződnek le. A csatornán áthaladva a kódszavak megváltozhatnak, elromolhatnak, és v vett szavak formájában jelennek meg. Fontos megjegyezni, hogy a vett szavak ugyanazt a kódábécét használják, mint a kódszavak. Végezetül a vett szavak a dekódolóban u' üzenetekké alakulnak. Célunk, természetesen, a csatorna hibázásának az észlelése és a hibák esetleges javítása.

A dekódolás problémája úgy fogalmazható meg, hogy keressük azt a c kódszót, amelyet a csatorna bemenetére adhattak, ha a vevő oldalon a v vett szó jelent meg. Mivel $c \in C \subset Y^*$ és $v \in Y^*$, ezért ugyanolyan hosszúságú, ugyanabból a kódábécéből felépített vektorokról van szó. Ha $v \in C$, akkor feltételezzük, hogy nem történt hiba – ha történt is, nem tudjuk detektálni. Amennyiben viszont $v \notin C$, akkor ez hibát jelent. A hibadetektálás lehetősége nyilván akkor áll fenn, ha C valódi részhalmaza Y^*-nak, vagyis az n hosszúságú – az Infokommunikáció tárgyban gyakorlatilag minden esetben binárisnak tekintett – vektorok halmazának. Ha ezeket a vektorokat egy n-dimenziós koordinátarendszerben – tehát minden bitet egy koordinátával ábrázolunk és minden koordinátatengelyen csak két érték lehetséges, 0 és 1 – ábrázoljuk, akkor a pontok egy n-dimenziós hiperkocka csúcsai lesznek. Ha $C \neq Y^*$, akkor a hiperkocka nem minden csúcsa felel meg valódi kódszónak. Például az ismétléses kód ($0 \rightarrow 000$ és $1 \rightarrow 111$) esetében a kocka 2 csúcsa jelképez kódszót.

Mit tehetünk, ha az átviteli csatornánk hibázik és – maradva a fenti példánál – 001 és 011 bitsorozatokat veszünk? A dekódolási feladat megfogalmazásából kiindulva elgondolkodhatunk azon, hogy milyen ***kódszavak helyett*** kaptuk a fenti vektorokat. Logikus feltevés az, hogy mindegyik vektorhoz keressük meg azt a kódszót, amelyikre leginkább „hasonlít". A „hasonlóság" azonban egy elég szubjektív fogalom, az egzakt – itt: egyértelmű, megalapozott és megismételhető – döntés meghozatalához a közelség-távolság matematikai definícióját kell megfogalmaznunk.

Definíció A c_1 és c_2 *szavak* Hamming-távolsága azon koordináták száma, ahol c_1 és c_2 különböznek.

Bár a tárgyban bináris példákat hozunk, a Hamming-távolság nemcsak bináris kódokra értelmezhető. Belátható, hogy a Hamming-távolság valódi távolság, hiszen nemnegatív, szimmetrikus és teljesül rá a háromszög-egyenlőtlenség. A Hamming-távolság felhasználásával a dekódolás feladata a következőképp hangzik: Keressük azt a $c \in C$ kódszót, amelyre $c = \text{argmin}_{c \in C} d(c, v)$.

Definíció Egy *C* kód távolsága alatt a lehetséges Hamming-távolságok minimumát, vagyis a $d_{min} = min_{c_1,c_2 \in C;\, c_1 = c_2}\, d(c_1, c_2)$ értéket értjük.

Hibajelzés során a vevőben csupán detektálni szeretnénk a hibázás tényét. Nyilvánvalóan egy d_{min} kódtávolságú kód minden (d_{min}−1) hibát jelez. Hibajavítás esetén a feladatunk a hibák helybeni javítása. Egyszerű hibázás esetén $\left\lfloor \frac{d_{min}-1}{2} \right\rfloor$ hiba javítható. Ennek szemléletes leírása a Hamming-szférák segítségével történhet, vagyis az egyes kódszavak körül egy *n*-dimenziós $\left\lfloor \frac{d_{min}-1}{2} \right\rfloor$ sugarú zárt gömböt elképzelve az ezen gömbön belüli vektorokat a Hamming-gömb középpontjában álló kódszóra javítjuk.

Ebből következik, hogy ha szeretnénk egy kód hibajavító képességét növelni, akkor extra biteket kell hozzáfűznünk, amelyek a kódtávolság növelését szolgálják. Amennyiben a *k* bites üzenetekhez *n* bites kódszavakat rendelünk, akkor a kódunkat *C*(*n, k*) jelöli. Ennél általánosabban a kódok jellemzőinek és képességeinek tömör leírása a $C(n, k, d)_q$ illetve az azzal egyenértékű $[n, k, d]_q$ módon történik, ahol *n* a kódszavak hossza, *k* az üzenet hossza, *d* a kódtávolság és *q* a használt kódábécé elemszáma.

Hogy az üzenetekből hogyan képezünk ilyen kódszavakat, arról még eddig nem beszéltünk. Egy lehetséges megoldás-családot mutatunk be a továbbiakban.

4 Lineáris kódok

Definíció Egy *C* kód lineáris, ha *C* halmaza lineáris tér, azaz ha minden $c_1, c_2 \in C$-re $c_1+c_2 \in C$.

A fenti definícióból következik, hogy minden lineáris kód esetén a 0 egy kódszó. Másrészt érdemes meggondolni, hogy *C* az *n* hosszúságú vektorok terének egy altere. Ennek folyományaként megállapíthatjuk, hogy a kódtávolság meghatározására egy sokkal egyszerűbb módszer is adódik.

Definíció Egy *c* vektor w(*c*) súlya a koordinátái között levő nullától eltérő elemek száma.

Definíció Egy *C* kód minimális súlyán a $w_{min} = min_{c \in C;\, c \neq 0}\, w(c)$ értéket értjük.

Tétel Ha *C* lineáris kód, akkor a kódtávolsága megegyezik a minimális súlyával, vagyis

$$d_{min} = w_{min}\,.$$

Ezt a tételt most nem bizonyítjuk, de jelentősége abban áll, hogy lineáris kód esetén a kódtávolság meghatározásához elegendő a kódszóhalmaz elemszámával arányos számú összehasonlítást elvégezni, ahelyett, hogy a kódszóhalmaz elemszámának négyzetével arányos számú összehasonlítást végeznénk.

Lineáris kódok esetén az üzenethez a kódszavakat egy *G* generátormátrix segítségével rendeljük hozzá az alábbi összefüggés szerint (*c* és *u* sorvektorok):

$$c = uG\,.$$

Definíció Egy (*n, k*) lineáris kód szisztematikus, ha minden kódszavára igaz, hogy annak első k szimbóluma a neki megfelelő üzenet adja.

Szisztematikus kódok esetén a generátormátrix alakja speciális, mégpedig *G = (I_k, B) ,ahol I_k a k × k méretű egységmátrix, B pedig k × (n–k) méretű mátrix.* Szisztematikus kódok esetén a *c* kódszavak első *k* elemét üzenetszegmensnek, az utolsó *n*−*k* elemét pedig paritásszegmensnek nevezzük:

$$c = (\underbrace{u_1, \ldots, u_k}_{\text{üzenet}}, \underbrace{c_{k+1}, \ldots, c_n}_{paritás})$$

Definíció Ha egy *(n–k) × n* méretű *H* mátrixra $Hc^T = 0$ akkor és csak akkor, ha $c \in C$, akkor a *H*-t a *C* kód paritásellenőrző mátrixának nevezzük.

H segítségével meg tudjuk állapítani, hogy egy vett szó kódszó-e. Megmutatható, hogy ha *G* és *H*

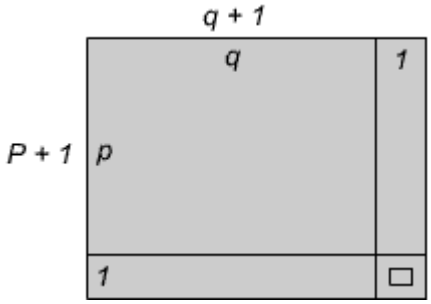
ugyanazon lineáris kód generátor- és paritásmátrixa, akkor $HG^T = 0$. Szisztematikus kódok esetén $H = (A, I_{n-k})$,ahol $A = -B^T$ (bináris esetben $A = B^T$).

Vizsgáljuk meg, milyen eredményt kapunk, ha egy nem létező kódszót szorzunk meg a paritásellenőrző mátrixszal! Legyen $c' = c + e$, ahol az e vektor egy meghibásodást jelent (hibavektor). Ekkor $c'H^T = (c + e)H^T = cH^T + eH^T = 0 + eH^T = s$, ahol $cH^T = 0$ a H definíciója miatt. A kifejezésben szereplő s vektort szindrómának nevezzük. Amikor dekódoljuk a jelet és nullától különböző szindrómát találunk, akkor abból egy meghibásodásra következtetünk (szindróma dekódolás). Ez a leképezést, a szindrómák hibamintára történő leképezését általában táblázatos formában szokták megadni. Természetesen előfordulhat, hogy különböző hibavektorokhoz azonos szindrómák tartoznak, ilyenkor a táblázatban azt a hibavektort tüntetjük fel, amelyben legkevesebb az egyesek száma.

A legegyszerűbb hibajavító kód az ismétléses kód. Ekkor az üzenet $k = 1$ méretű, és ezt ismételjük meg n-szer. Így egy $C(n, 1)$ kódot kapunk. A két különböző (0, ill. 1) üzenetnek megfelelően ez az egyszerű kód két kódszót tartalmaz a (000...0) és az (111...1) kódszavakat. A kódtávolság nyilván n, így a kód hibajavító képessége, ha n-t célszerűen páratlan értékűre választjuk, $(n-1)/2$.

A legegyszerűbb hibadetektáló kód az egyszerű paritáskód. Egy u üzenethez az üzenet bitjeinek megfelelő páros vagy páratlan paritásbitet illesztve kapjuk a kódszót (a páros paritás az elemek modulo 2 összege, a páratlan annak inverze). Így a paritáskód $C(n, n-1)$ paraméterű. Könnyen belátható, hogy a kódtávolság 2. Ugyanis, ha tetszőleges kódban egy bitet tetszőleges koordinátán megváltoztatunk, megváltozik a paritás. Két bitet változtatva visszakapjuk az eredeti (páros vagy páratlan) paritást, tehát újra egy érvényes kódszóra jutunk, amely az eredeti kódszótól két koordinátában különbözik. Mivel a kódtávolság 2, ezért egy hiba detektálására alkalmas az egyszerű paritáskód.

A kétdimenziós paritáskód készítéséhez rendezzük a k üzenetbitet egy $p \times q$ méretű U mátrixba, azaz $k = pq$. Képezzünk soronként, ill. oszloponként paritást, és az így adódó paritásbiteket írjuk a sor következő elemeként a sor végére, ill. az oszlop alá (lásd az alábbi ábrán). Ekkor egy $(p+1) \times (q+1)$ méretű C mátrixot kaphatunk, ha a mátrix még nem definiált jobb alsó sarokelemét is megadjuk. Legyen ez a sarokelem a „paritások paritása”, azaz a C mátrix utolsó sorában vagy utolsó oszlopában álló paritásbitek paritása (könnyen látható, hogy mindegyik úton azonos érték kerül a jobb alsó sarokba). A kódtávolság 4.



A kétdimenziós hibavédő kódolás jellegzetes alkalmazása a CDDA szabvány (Red Book), ahol byte szintű Reed-Solomon kódolást használunk.

Definíció (Hamming-korlát) Ha egy q elemszámú kódábécé feletti (n, k) paraméterű kód t hibát tud javítani, akkor $\sum_{i=1}^{t} \binom{n}{i} (q-1)^i \leq q^{n-k}$.

Emlékeztetőül: $t = \left\lfloor \frac{d_{min}-1}{2} \right\rfloor$.

Definíció Az olyan kódokat, ahol a Hamming-korlát egyenlőséggel teljesül, perfekt kódoknak nevezzük.

Más megfogalmazásban azt is mondhatjuk, hogy a perfekt kódok esetén a kódszavak és a köréjük írt n-dimenziós $\left\lfloor \frac{d_{min}-1}{2} \right\rfloor$ sugarú zárt gömbök teljesen kitöltik az Y^* teret (lásd még: sphere packing problem). A perfekt kódok definíciója arra utal, hogy azok valamilyen szempontból optimálisak, így természetesen adódik az a kérdés, hogy hogyan lehet perfekt kódokat generálni, egyáltalán milyen perfekt kódok léteznek. Könnyen belátható, hogy a korábban már említett $C(n, 1)$ ismétléses kód páratlan n esetén perfekt kód, mégpedig a *triviális* perfekt kódok közé tartozik, viszont a hatékonysága elég kicsi, még n=3 esetén is csak 1/3. Egyelőre megelégszünk 1 hiba javításával, de szeretnénk ennél hatékonyabb eljárást találni. Bináris kódot feltételezve a Hamming-korlátot ekkor így írhatjuk fel: $1+n=2^{n-k}$. Látható, hogy viszonylag könnyen találhatunk olyan n számot, amely 2 valamelyik hatványánál eggyel kisebb.

Definíció Az olyan perfekt kódokat, amelyek 1 hibát javítanak, Hamming-kódnak nevezzük.

Az általánosított konstrukciót ($n = 2^r-1$, $k = 2^r-1-r$ paraméterek esetére, ahol $r \geq 2$) Hamming is megadta 1950-es cikkében, viszont részletesen a $C(7, 4)$-es kóddal és annak egy páros paritásúra kiegészített változatával, a $C(8, 4)$ kóddal foglalkozott. Ezeket a kódokat szokták Hamming(7, 4) és Hamming(8, 4) kódnak is nevezni. A cikkben Hamming egy olyan kódot javasolt, amely az adatbitek között a 2^i pozíciókban elhelyezett paritásbiteket használ, amely paritásbitek pontosan azokat a pozíciójú biteket védik, amelyek pozíciójának binárisan kifejezett értékében az *i.* bit helyén 1 szerepel. Például a Hamming(7,4) kód esetén a kódszó (p_1, p_2, d_3, p_4, d_5, d_6, d_7) alakú, ahol $p_1=d_3+d_5+d_7$, $p_2=d_3+d_6+d_7$ és $p_4=d_5+d_6+d_7$. Az ennek megfelelő generátormátrix és paritásellenőrző mátrix nem szisztematikus, viszont rendelkezik azzal a tulajdonsággal, hogy a kapott szindrómavektort fordítva kiolvasva megkapjuk a hiba *pozíciójának* bináris értékét.

$$G_{Hamming(7,4)} = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \qquad H_{Hamming(7,4)} = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Természetesen belátható, hogy hibajavító-képesség megtartása mellett az üzenet-kódszó összerendelést szisztematikusan is elvégezhetjük, hiszen a szindrómaszámítás $s = eH^T$ definíciójára tekintve láthatjuk, hogy azon e vektorhoz, amely (000...1...0) alakú, azaz egyetlen 1-et tartalmaz az i-edik koordinátáján, a H mátrix transzponáltjának i-edik sora rendelődik szindrómaként. Következésképpen, ha azt szeretnénk, hogy az n különböző ilyen alakú hibavektorhoz különböző szindrómák tartozzanak, a H^T sorait egymástól és nullától különbözőnek kell választani. Ez a választás tehát garantálja, hogy a H mátrixnak megfelelő kód minden 1-hibát tartalmazó meghibásodást javítani tudjon. A H mátrixnak megfelelő kódot, illetve a G generátormátrixot úgy kaphatjuk meg egyszerűen, hogy a H mátrixot szisztematikus alakban építjük fel. A Hamming-kódok általánosíthatóak nembináris esetre is.

Marcel Golay 1949-ben egy rövid, összesen 3 hasábos megjegyzésben publikálta a róla elnevezett kódot. Tulajdonképpen két kódról beszélünk: a $[23, 12, 7]_2$ bináris perfekt kódról és a ternáris (háromelemű kódábécé feletti) $[11, 6, 5]_3$ perfekt kódról. Ezekhez jönnek még a kiterjesztések: az egy paritásbittel kiegészítetett $[24, 12, 8]_2$, illetve a ternáris (háromelemű kódábécé feletti) perfekt kód egy digittel kiegészített $[12, 6, 6]_3$. 1973-ban bizonyították be, hogy bármely prímhatvány elemszámú kódábécé feletti nemtriviális perfekt kód vagy Hamming-kód vagy Golay-kód. A kiterjesztett bináris Golay-kódot használta kezdetben a NASA a Voyager 1 és Voyager 2 küldetésben a fényképek Földre való küldése során. A küldetések célja eredetileg a Jupiter és a Szaturnusz vizsgálta volt. Mikor később kiderült, hogy az űrszondák a tervezetthez képest sokkal tovább is képesek működni és továbbirányították őket az Uránusz felé, a kódolási eljárást lecserélték Reed-Solomon kódra.