**Kódoláselmélet. (Huffmann kód, hibajavító kódok, véges testek konstrukciója. Reed-Solomon kód és dekódolása.)**

Kommunikáció során az adótól egy vevőig viszünk át valamilyen adatot egy csatornán keresztül. Az adatot az információ mennyiséggel mérhetjük, melynek mértékét az általa megszüntetett bizonytalansággal mérünk. Tegyük fel, hogy adatainkat r különböző jelből álló sorozattal kódoljuk, ekkor ha az adott jelsorozat (üzenet) p valószínűséggel fordul elő, akkor az ő információtartalma legyen $I := -log_r p$. Egy forrás által kibocsájtott átlagos információtartalom a forrás entropiája a következő mennyiség legyen: $-\sum p_i * log_r p_i$, ahol pi az i. üzenet előfordulásának valószínűsége.

Minden adatot egy adott karakterkészlet jeleiből álló sorozattal kódolunk, ugyanis ez az a forma, mellyel könnyen tudunk különféle machinációkat végrehajtani. Kódolásnak az injektív leképezéseket nevezzük. Azért csak az injektíveket tekintjük kódolásnak, mert csak ezek dekódolhatóak. Minden üzenet felbontható elemi részekre, amelyet már le tudunk kódolni jelsorozatokká, ezen elemi részeknek megfelelő kódszavak egymás után írásával kapjuk meg az üzenet ún. **betűnkénti kódolás**sal. Üzeneteinket a természetes nyelvek redundánsan közlik, melyre a zajos környezet miatt szükség is van, hiszen ha minden szavunkat legfeljebb 5 betűvel írnánk le (amit meg is tehetnénk), úgy pl. az aabaf és aadaf szavakat elég zajos helyen meg sem tudnánk különböztetni, pedig lehet, hogy teljesen mást jelentenének, ezzel szemben a kenyér és sajtosperec szavakat még zajos metrón is megkülönböztethetjük (még ha nem is értjük, de legalább tudjuk, hogy ez a kettő nem ugyanaz volt).

Ezen jelsorozatokban fellelhető redundanciát a tömörítéssel lehet csökkenteni, ami gyorsabb átvitelt is eredményezhet. Jelöljük A,B,...-vel egy adott ábécé betűiből álló halmaz, $A^+$-szal az ezen ábécé feletti legalább egy hosszú szavakat, csillagos változatában megengedjük az üres szót is. A betűnkénti kódolás egy $A \to B^+$ függvény, melyet úgy terjsztünk ki egy $A^+ \to B^+$ függvénnyé, hogy a sorozatokat alkotó betűk kódját egymás után írva azt kapjuk, mintha ezzel a függvénnyel képeztük volna le az üzenetet.

Prefix, szuffix, infix a szavak elejét, végét és közepét jelentik (mindhárom foga-

lomba belefér az egész szó maga is). Szavak nem üres halmazát prefixmentesnek nevezzük, ha benne egyik szó sem valódi prefixe a másiknak.

Kódok tulajdonságai közül néhány fontosabb.

- Egy kód **felbontható**, ha a megfelelő $A^+ \to B^+$ függvény injektív.
- Egy kód **prefixmentes**, ha a kódolás értékkészlete prefixmentes.
- Egy kód **vesszős**, ha van olyan $B^+$-beli elem, mely az értékkészlet tőle különböző elemének szuffixse, de egyiknek sem prefixe, vagy infixe. Ha ilyen vessző nincs, akkor a kód vesszőmentes.

Kód átlagos szóhossza a különböző betűk kódjának hossza szorozva az előfordulás valószínűségével, ez szummázva minden betűre.

**1. Tétel (McMillan egyenlőtlenség).** *Legyen A egy n elemű, B egy r elemű abécé, r>1, és legyen* $\varphi : A \to B^+$ *injektív függvény. Ha e leképezés által létesített betünkénti kódolás felbontható, akkor* $\sum r^{-|\varphi(a_i)|} \leq 1$. *Megfordítva, ha l1,...,ln olyan pozitív egészek, melyek megfelelnének az előbbi szummában látott kitevőknek, úgy van olyan prefix kódolása A-nak a B elemeivel, melyben az i. betű hossza li.*

E tétel szemléletesen azt fejezi ki, hogy felbontható kódban az átlagos szóhossz nem lehet túlságosan kicsi. A második része pedig azt szemlélteti, hogy felbontható kódból ugyanolyan hosszakkal készíthető prefix kód is, ha esetleg elsőre nem ilyet sikerült volna összerakni.

Adott valószínűségekhez tartozó, adott elemszámú kódoló ábécével megkonstruált minimális átlagos szóhosszúságú prefix kódot **optimális kód**nak nevezzük. Ezek azon kódok, amelyekben a McMillan egyenlőtlenség által adott szumma minél jobban megközelíti az egyet, pontosabban mivel a valós számok körében tetszőleges közelítés érhető el, azt a kódot nevezzük optimálisnak, melynek átlagos szóhossza a szumma+1-et már nem éri el, de legalább a szumma értéke.

Ilyen optimális kódot ad a **Huffman kód**, amelynek konstrukciója a következő. Rendezzük valószínűségük csökkenő sorrendjébe a betűket, majd a két legnagyobb

valószínűségű betűt vonjuk össze egy közös kódúvá. Itt lényegében az történik, hogy folyamatosan építjük a kódfát, a legnagyobb valószínűségű betűk kódját téve a legrövidebbnek, oly módon, hogy a két legkisebb valószínűségű betű lesz a kódfa legalsó szintjén, és felfelé haladva jönnek az egyre sűrűbben előforduló betűk, de egy betű az mindig egy levélen lehet.

**Hibajavításnál** az elsődleges szempont az, hogy az átviteli közeg által okozott esetleges hibákat kiszűrjük, javítsuk. Ennek alapján kétfajta hibakorlátozó kód létezik, a **hibajelző**, valamint a **hibajavító** kódok.

Egy kód t-hiba jelző, ha mindig hibát jelez, ha a küldött kódszó max t helyen hibásodott meg. Pontosan t hiba jelzőnek akkor nevezzük, ha van olyan t+1 hiba, amit már nem jelez. Hibajavításra ezt hasonlóan definiáljuk.

Ahhoz, hogy a hibajelzés és javítás között kapcsolatot teremthessünk, be kell vezetni a Hamming-távolság fogalmát. u és v kódszavak **Hamming távolság**a az egymástól eltérő helyiértékek száma. Ez valóban rendelkezik a távolságfüggvények megszokott tulajdonságaival (szimmetria, csak egyezés esetén nulla,háromszög egyenlőtlenség). u kódszó **Hamming súly**a a nem nulla bitek száma. Kód távolsága a két minimális távolságú kódszavának távolsága, súlya a minimális súlyú kódszó súlya.

Egyszerűen látható, hogy egy kód pontosan akkor t hiba jelző, ha t kisebb mint a kód távolsága, és pontosan akkor pontosan t hiba jelző, ha t megegyezik a távolság-1-gyel. Például paritásbites kód esetén a kód távolsága 2, ez a kód a hibát csak jelezni tudja, javítani nem.

**Minimális távolságú dekódolás** esetén mindig a távolságban legközelebb álló kódszóra döntünk ha hibás volt az átvitel. Több ilyen esetén valamilyen más algoritmussal dönthetünk az egyikre. Adott távolságú kódnál a távolság felének az alsó egészrésze az a határ, aminél kevesebb vagy még pont annyi hibával ez a fajta dekódolás működhet, hiszen háromszög egyenlőtlenség miatt ez a kódszó lesz a legközelebb ahhoz amit kaptunk.

Összefoglalva tehát ha egy kód távolsága d, akkor a kód d hiba jelző, pontosan d-1 hiba jelző, d/2 javító, és pontosan (d-1)/2 alsó egészrész javító.

További hibajavító kódok a mátrixos paritás bit, amikor egy adott mátrixban van az üzenet, és ennek minden sora és oszlopa után helyezünk egy paritásbitet. Ezzel már javítani is lehet, ám pl pont „négyzetesen" elhelyezkedő hibákat ezzel is csak detektálni lehet, javítani nem mindig.

**Lineáris kódok** esetében a kódszavakat egy K test feletti lineáris térből vesszük. Egy ilyen kódot e tér alterének tekintjük. Ha a tér k dimenziós, a kód távolsága d, és a test számossága q, akkor jelöljük ezt a kódot [n,k,d]-vel, (esetleg q-t is feltüntethetjük a jelölésben). Egy lineáris kód két mátrixszal jellemezhető, az ún. **generátormátrix**szal, amelyben az altér bázisvektorai szerepelnek, azaz ez egy $k \times n$ -es mátrix; valamint az **ellenőrző mátrix**szal, melynek tulajdonsága, hogy vele jobbról szorozva a kódvektorokat 0-t kapunk, míg más vektorokkal szorozva nem nullát. Bármely $K^k$-beli elemet megszorozva a generátormátrixszal, kódszót kell kapnunk.

**Reed-Solomon kódoknál** egy test feletti l dimenziós vektorteret veszünk, amihez hozzárendelünk egy polinomot, mégpedig $c \in K^l$-hez a $\sum c_i * x^i$ polinomot. Ez a hozzárendelés megfordítható természetesen. Ez egy összegtartó, injektív, valamint skalárszorost tartó leképezés. Most válasszunk a véges test multiplikatív csoportjából egy elemet, a-t, melynek rendje legalább l, jelöljük n-nel. Tudjuk, hogy ekkor a-nak az n-edik hatványa a test szorzásra nézett egységeleme, így a akárhanyadik hatványa ezen az n-ediken is az egységelem. Továbbá ezen kisebb hatványok mint különböznek, hiszen a rendje n, így azonnal következik, hogy az $a^i$-k és csak ezek a gyökei a $x^n - e$ polinomnak, vagyis felírható a gyöktényezős felbontásos alakja ennek a hatványok segítségével. Válasszuk a g polinomot ezen hatványok szorzatának, de a szorzat most csak n-k-ig menjen el, vagyis $g = \prod_{i=0}^{n-k} (x - a^i)$. Ez a polinom láthatóan osztja az $x^n - e$ polinomot. Ennek többszörösei (azaz legfeljebb k-1-edfokú nem nulla polinomokkal való szorzatai) legyenek az általa generált Reed-Solomon kód. C elemei a legfeljebb n-1-ed fokú polinomok, és van is ilyen fokú polinom, pl $x^{k-1} * g$. Belátható, hogy C egy lineáris altere a legfeljebb k-1 dimenziós K feletti térnek, valamint, hogy a C a $K^n$ k dimenziós altere. Belátható továbbá,

hogy a K feletti legfeljebb n-1-ed fokú polinomok pontosan akkor elemei e g generálta kódnak, ha g minden gyöke gyöke u-nak is. Ennek a kódnak egy ellenőrző mátrixát kaphatjuk, ha az i-edik sor j-edik elemének az $a^{(i+1)^j}$-nek választjuk, ugyanis a egy n-k-ad rendű részmátrixának a determinánsa pont ebben az esetben lesz 0, ami a megfelelő kiemelések után látszik is.

A kód egyik legfontosabb tulajdonsága, hogy a kód távolsága d=n-k+1, és mivel H rangja n-k, így a kód maximális távolságú.

Még egy fontos probléma a **véges testek megkonstruálása**. Nevezetes tétel, hogy minden $\mathbb{Z}_p$ maradékosztálygyűrű pontosan akkor test, ha p prím. További nevezetes tételek, hogy minden már véges test ezeknek a bővítésével jön létre, vagyis minden testnek valamely p prímre $\mathbb{Z}_p$ a legszűkebb részteste, ún. prímteste. Mivel egy n-ed fokú polinommal történő testbővítés után egy $p^n$ elemű test jön létre, úgy könnyen látható, hogy minden véges test elemszáma prímhatvány, valamint, hogy minden elemszámra izomorfiától eltekintve egy véges test létezik.

Tegyük fel, hogy egy q elemű testet szeretnénk bővíteni $q^n$ eleműé. Ennek lépései:

1. Keresünk egy, a q elemű test feletti irreducubilis polinomot, azaz olyat, aminke ott nincs gyöke.

2. Ennek a gyökét megjelöljük valamilyen eddig nem használt betűvel (pl i-vel).

3. Meghatározzuk az új test elemeit, amelyek i legfeljebb n-1-edfokú polinomjai lesznek (mint pl komplexeknél a bővítés másodfokúval történt, s ott i elsőfokú polinomjai lettek az új elemek, mint pl 2+4i)

4. Meghatározzuk az új test összeadó, és szorzótábláját. Összeadásnál még egyszerű a dolog, szorzásnál viszont lépten nyomon fel kell használni, hogy i annak az adott polinomnak a gyöke, különböző számolási trükkökre van szükség. (pl amikor valósat bővítettük komplexre az $x^2+1$ polinommal, akkor az i*i értékét az i*i+2=0 segítségével számoltuk ki, mivel feltettük, hogy i a polinom gyöke.)