Információ- és kódelmélet

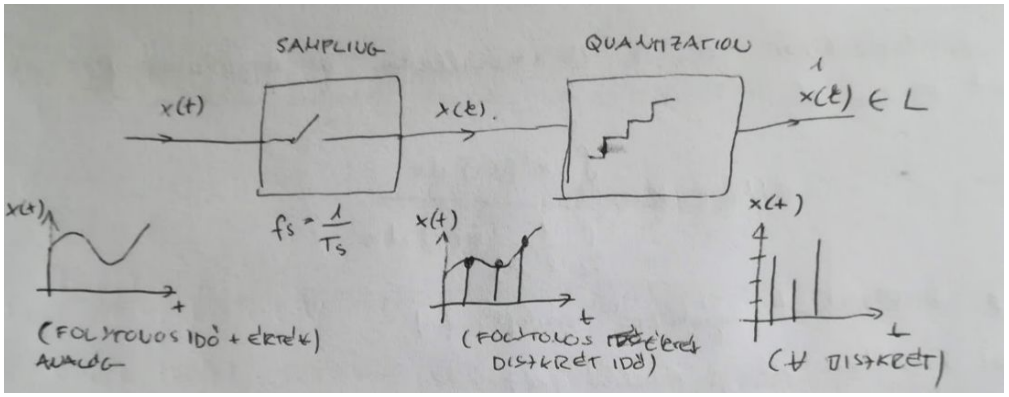
Híradástechnika szigorlat

Kidolgozott tételek - mese

2020

Hunglish

1. Describe the discrete memoryless source model (sampling, optimal Lloyd Max quantization ...etc.).



- Analóg jel digitálissá alakítása: Van először a folytonos jeled amit elkezdesz **mintavételezni** adott időközönként. Ezután az idő értékek diszkrétek lesznek, de a folytonosak az értékek. Ezután jön a **kvantálás**, ahol ezeket az értékeket egy meghatározott diszkrét értékhalmazra képezzük le (kerekítünk bizonyos értékekre)

Mintavételezés

- **Memoryless:** A memóriamentes azt jelenti, hogy az egymást követő üzenetek függetlenek.
- **Sávszélesség:** x(t) sávszélessége B, ha így áll elő Fourier transzformáltjából:

$$x(t) = \int_{-B}^{B} X(f) e^{j2\pi ft} df$$

- A mintavételezés akkor **veszteségmentes**, ha az eredeti jel visszaállítható a mintavételezett jelből.
- **Mintavételi tétel:** Ha a mintavételi frekvencia legalább akkora mint a sávszélesség kétszerese $f_s \geq 2B$. Akkor a mintavétel veszteségmentes, tehát a jel visszaállítható. A visszaállítás képlete:

$$x(t) = T \sum_k x_k \, h(t - kT) \quad h(t) = \frac{\sin(2\pi Bt)}{2\pi Bt}$$

- Érdemes megjegyezni, hogy valójában pillanatszerű érték vételezés nem létezik, ezért nagyon rövid időre vonatkozó mintákat veszünk, amit átlagolunk.
- Túlmintavételezés esetén, vagyis amikor $f_s > 2B$ több adat keletkezik, a továbbításhoz nagyobb sávszélesség kell. Akkor miért nem a lehető leggazdaságosabb $f_s = 2B$ -frekvenciát választjuk?
    - Azért mert ahhoz egy ideális aluláteresztő szűrő kéne, olyanunk meg nincs.

Kvantálás

- A **jel-zaj viszony** (Signal-to-Noise Ratio, SNR) két teljesítmény jellegű mennyiség hányadosaként azt fejezi ki, hogy hogyan viszonyul a jel teljesítménye a háttérzajhoz. (P: átlagos teljesítmény, A: amplitúdó négyzetes átlaga)

$$\mathrm{SNR} = \frac{P_{\mathrm{jel}}}{P_{\mathrm{zaj}}} = \left(\frac{A_{\mathrm{jel}}}{A_{\mathrm{zaj}}}\right)^2, \qquad \mathrm{SNR}^{\mathrm{dB}} = 10 \log_{10}\left(\frac{P_{\mathrm{jel}}}{P_{\mathrm{zaj}}}\right) = 20 \log_{10}\left(\frac{A_{\mathrm{jel}}}{A_{\mathrm{zaj}}}\right)$$

Jelölések a következőkben: $[-C, C]$ *- jelszintek intervalluma,* N *- kvantálási szintek száma,* $L = \{l_1, ..., l_N\}$ *- kvantálási szintek,* $\Delta = \{\Delta_1, ..., \Delta_N\}$ *- kvantálási intervallumok,* $\{x_0, ..., x_N\}$ *- intervallum végpontok.*

Ekvidisztáns kvantáló

- Ilyenkor tulajdonképpen egyenlő részekre osztjuk fel a kvantálási intervallumot.

$$\Delta = x_{k+1} - x_k = \frac{2C}{N} \forall k = 1...N \ .$$

- Ekvidisztáns kvantáló esetén az SQNR:

$$SQNR = \frac{3}{2} 2^{2n}$$

- (csak teljesen kivezérelt jel esetében igaz)

Logaritmikus kvantáló

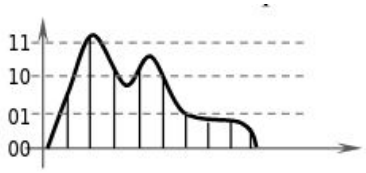
- Az ekvidisztáns kvantáló esetében néha kaphatunk különböző SNR-t (emberi hang)
- A kvantáló logaritmikus, ha a kvantálási szintek és intervallumok eloszlása logaritmikus.
- Logaritmikus kvantáló SNR-je:

$$\mathrm{SNR} = K' \frac{\int_{-C}^{C} x^2 p(x)\,\mathrm{d}x}{\int_{-C}^{C} \frac{1}{\ell'^2(x)} p(x)\,\mathrm{d}x}$$

- itt az l: osztópontokat meghatározó fv, p: ismeretlen valségi eloszlású jel

Differenciális kvantáló

- Itt az a lényeg, hogy felhasználjuk az előző minta tartalma számít a jelenlegi szempontjából.
- Egy megvalósítás, hogy azt mutatja, hogy egy jel csökken vagy nő az előzőhöz képest.



hagyományos kvantálás: 01 11 10 10 10 01 01 01 00

differenciális kvantálás: 1 1 0 0 1 0 0 0 0

- Lehet egy másik megvalósítás, hogy a beérkezett jelből ismerve, hogy mi a következő jelek valószínűsége, a valószínűbbhöz rövidebb kódot rendelünk.

		$\mathbb{P}$	buta kód	okos kód
Kérek egy jó pohár sö…	…rt!	0,999	00	0
	…tét gépolajat!	0,0005	01	10
	…rétet a puskámba!	0,0003	10	110
	…ntésből maradt lét!	0,0002	11	111

Lloyd-Max algoritmus

- Optimális kvantálót szeretnénk fogni.
- Mese: A nagyobb valószínűségi helyeken a kvantálási intervallumokat kisebbeknek választjuk.
- Ez azt jelenti, hogy egy olyan kvantálót megvalósító függvényt keresünk, ami a maximális SNR-t eredményezi nekünk.

$$\ell_{\mathrm{opt}} : \max_{\ell(x)} \mathrm{SNR}$$

- Az algoritmus lényege, hogy két kiindulási halmaz van:
  - $\Delta = \{\Delta_1, ..., \Delta_N\}$: kvantálási szintek közti különbségek
  - $Q = \{Q_1, ..., Q_N\}$: kvantáló osztópontjai
- Az algoritmus a hűségkritérium megfelelő mértékű csökkenéséig fut:

$$J(\Delta, Q) \coloneqq \sum_{i=1}^{N} \int_{\Delta_i} (x - q_i)^2 p(x)\ \mathrm{d}x$$

- Az algoritmus lépései a következők

- Ha az adott lépésben az optimális Q ismert, akkor

$$\Delta_{\ell,\mathrm{opt}} \coloneqq \{x\colon (x-q_\ell)^2 < (x-q_i)^2, \forall i\ i \neq \ell\}$$

- Ha az adott lépésben az optimális Δ ismert, akkor

$$q_{\ell,\mathrm{opt}} \coloneqq \frac{\int_{\Delta_\ell} x\, p(x)\, \mathrm{d}x}{\int_{\Delta_\ell} p(x)\, \mathrm{d}x} = \mathbb{E}(x|x \in \Delta_\ell)\ \forall \ell$$

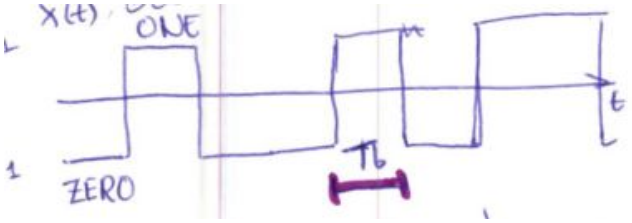
- Ez egy rekurzív algoritmus, mely az alábbi útvonalon halad:

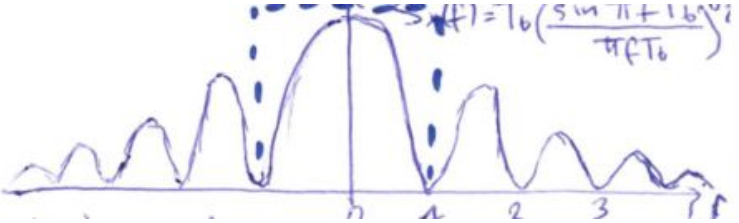
$$\Delta(0), Q(0) \to \Delta_{\mathrm{opt}}(1), Q(0) \to \Delta_{\mathrm{opt}}(1), Q_{\mathrm{opt}}(1) \to \Delta_{\mathrm{opt}}(2), Q_{\mathrm{opt}}(1) \to \cdots$$

2. Derive the Nyquist criterion for ISI free communication over band limited channels.

Szimbólum közti interferencia

- Intersymbol Interference (ISI)
- Egy négyzetes jel az időtartományban egy szinuszos függvénynek felel meg a frenvencia tartományban.





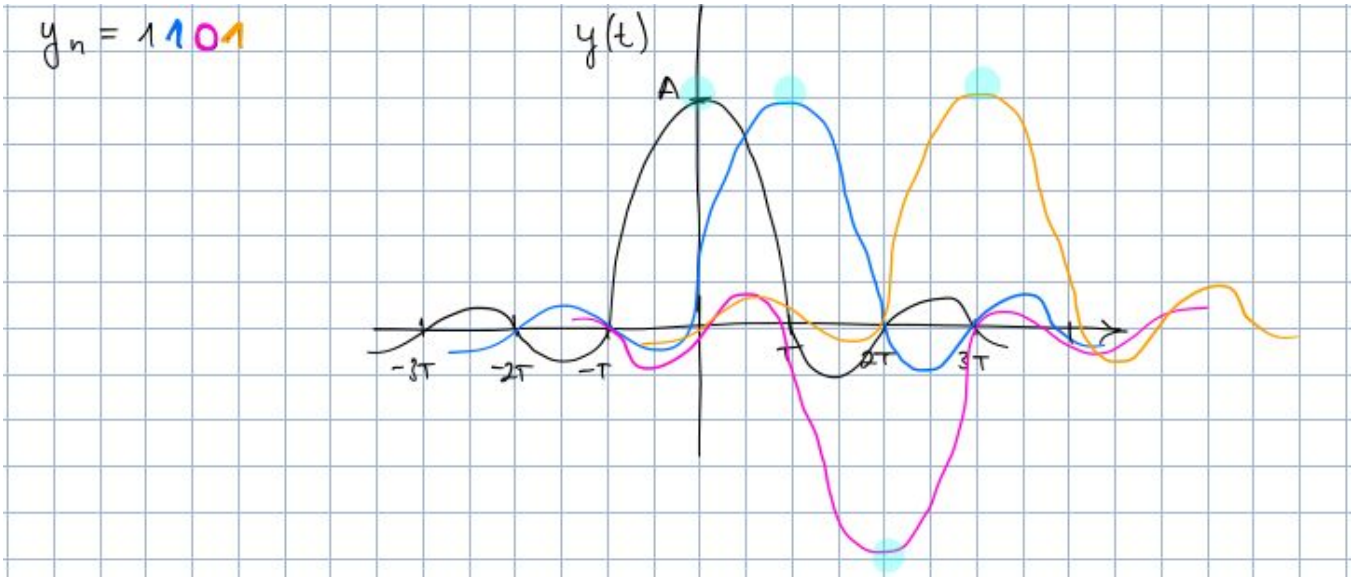
- Végtelen sáv szélességet igényelne (????)
- A sávszélesség limitálásához szűrőt használunk
- Minden impulzus átlapolódik

Nyquist

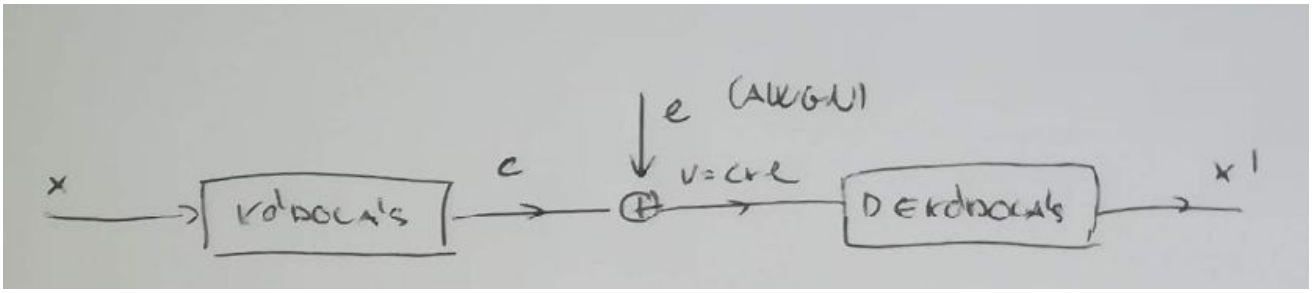
- Nájkviszt észrevette, hogy bár az impulzusok átlapolódhatnak a szimbólumperiódusra, elkerülhető az ISI, ha ezeknek az értéke 0 a többi impulzus közepén.
- A nyquist kritérium kimondja, hogy nincs ISI akkor és csak akkor, ha:

$$\frac{1}{T}\sum_{k=-\infty}^{+\infty} H(f-\frac{k}{T}) = 1,$$

- Legjellemzőbb impulzus: emelt koszinusz.
- Nem értem am. (najó kb)
- a kis genyó hullámok hatnak egymásra, azt akarjuk hogy nulla legyen mindenki más a kék pöttyöknél)



3. Describe the memoryless channel model (AWGN channel and BSC), derive the bit error probability as a function of the signal-to-noise ratio.



- Egy csatorna memóriamentes, ha a kapott szimbólum csak azon a szimbólumon dependál, ami abban a very same pillanatban lett elküldve.

$$P(v_k = y_i | c_k = x_i, c_{k-1} = x_{i-1}, ...) = P(v_k = y_i | c_k = x_i)$$

BSC: Binary Symmetric Channel

- Egy olyan kommunikációs csatorna modell, amelyben bináris adatokat küldünk és annak a valsége, hogy egy bit rosszul érkezik meg csak a bittől függ.
- A BSC-ket a Bit Error Probability (bithibaarány) jellemzi, amelyet úgy definiálunk, hogy

$$P_b = P(v = 1 | c = 0) = P(v = 0 | c = 1).$$

AWGN: Additive White Gaussian Noise

- Az AWGN csatorna egy olyan kommunikációs modell, amelyben a beérkező jel a küldött jel és egy normál eloszlás szerinti random változó összege.
- Jellemzi az σ^2 DEVIANCE (tudja a fasz hogy ez mi magyarul lol - szórásnégyzet)
- Egy AWGN csatorna SNR-je:

$$SNR_{[\mathrm{dB}]} = 10 \cdot \log\left(\frac{P_{signal}}{\sigma^2}\right).$$

- BSC bithibaaránya az SNR függvényében:

$$P_b = \Phi\left(-0.5 \cdot \sqrt{\frac{10^{\frac{1}{10} SNR_{[\mathrm{dB}]}}}{P_{signal}}}\right)$$

- Bizonyítás:

$$P(v = 0 | c = 1) = P(e < -0.5) = \Phi\left(\frac{-0.5}{\sigma}\right) \text{ and}$$
$$P(v = 1 | c = 0) = P(e > 0.5) = 1 - \Phi\left(\frac{0.5}{\sigma}\right) = \Phi\left(\frac{-0.5}{\sigma}\right).$$

$$P_b = P(c = 0) \cdot P(v = 1 | c = 0) + P(c = 1) \cdot P(v = 0 | c = 1) = \Phi\left(\frac{-0.5}{\sigma}\right).$$

$$SNR_{[\mathrm{dB}]} = 10 \cdot \log\left(\frac{P_{signal}}{\sigma^2}\right) \Rightarrow \sigma = \sqrt{\frac{P_{signal}}{10^{\frac{1}{10}SNR_{[\mathrm{dB}]}}}}$$

- Végén visszahelyettesíted az σ-t a P_b-be.

4. Define and describe the properties of entropy, joint entropy, conditional entropy and mutual information.

Entrópia

- Entrópia: bizonytalanság mértéke.
- Mese: Egy valószínűségi változó entrópiája úgy fogható fel, mint az általa reprezentált véletlen kísérlet kimenetelének bizonytalansága. Így például nulla az entrópiája a biztosan bekövetkező eseményt reprezentáló 1 valószínűségű konstans valségi változónak. Maximális az entrópiája a legbizonytalanabb kiementelű egyenletes eloszlású valségi változónak.
- Egy szimbólum információtartalma: (ld: log2)

$$\mathrm{I}(x) = \mathrm{ld}\,\frac{1}{p(x)}$$

- Az entrópiája egy forrásnak az átlagos információja az elküldött üzeneteknek:

$$\mathrm{H}(X) = \mathbb{E}(\mathrm{I}(x)) = \sum_{x \in X} p(x) \cdot \mathrm{I}(x) = \sum_{x \in X} p(x)\,\mathrm{ld}\,\frac{1}{p(x)}$$

- Az entrópia tulajdonságai: (N az üzenet hossza)
  1. $0 \leq H(X) \leq ld\ N$
  2. $\forall X\ \forall g(X) \quad H(g(X)) \leq H(X)$

Együttes entrópia

- Két forrás együttes entrópiája:

$$\mathrm{H}(X,Y) = \sum_{x}\sum_{y} p(x,y)\,\mathrm{ld}\,\frac{1}{p(x,y)}$$

- Az entrópia tulajdonságai:

1. $H(X,Y) \geq \max(H(X), H(Y))$
2. $H(X_1, \ldots, X_n) \geq \max(H(X_1), \ldots, H(X_n))$
3. $H(X,Y) \leq H(X) + H(Y) \longrightarrow H(X,Y) = H(X) + H(Y) \Leftrightarrow$ X és Y független
4. $H(X_1, \ldots, X_n) \leq H(X_1) + \ldots + H(X_n)$

Feltételes entrópia

- Mennyi bizonytalanság marad átlagosan X-re nézve, ha van lehetőségünk Y előzetes megfigyelésére.
- Két forrás feltételes entrópiája:

$$\mathrm{H}(X|Y) = \sum_{x}\sum_{y} p(x,y)\,\mathrm{ld}\,\frac{1}{p(x|y)}$$

- Az együttes entrópia előáll az alábbiak szerint

$$\mathrm{H}(X,Y) = \mathrm{H}(X) + \mathrm{H}(Y|X) = \mathrm{H}(Y) + \mathrm{H}(X|Y)$$

- Független források esetén:

$$\mathrm{H}(X,Y) = \mathrm{H}(X) + \mathrm{H}(Y)$$

- Feltételes entrópia tulajdonságai:

1. $H(Y|X) \leq H(Y)$
2. $H(X,Y) = H(X|Y) + H(Y|X) + I(X,Y)$ ← kölcsönös információ
3. $H(X,Y) = H(X) + H(Y) - I(X,Y)$
4. $I(X,Y) \leq H(X)$
5. $H(Y|X) = H(X,Y) - H(X) + H(Y)$

$p(x)$

$\begin{cases} H(Y|X) = H(X,Y) - H(X) \\ H(X|Y) = H(Y,X) - H(Y) \end{cases}$

Kölcsönös információ

- Kullback-Leibeler távolsága két valségi eloszlásnak:

$$\mathbb{D}(p(x)||q(x)) = \sum_x p(x) \operatorname{ld} \frac{p(x)}{q(x)}$$

- Kölcsönös információ:

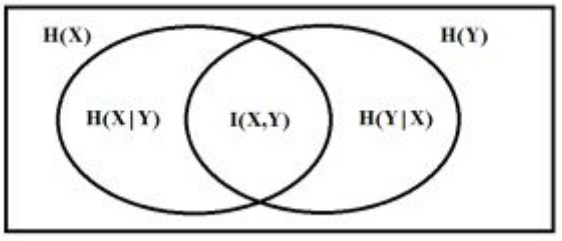
$$\mathrm{I}(X,Y) = \mathbb{D}(p(x,y)||p(x)\cdot(y))$$

$$\mathrm{I}(X,Y) = \sum_x \sum_y p(x,y) \operatorname{ld} \frac{p(x,y)}{p(x)p(y)}$$

- Kölcsönös információ tulajdonságai:
  - 1. Nagyobb egyenlő mint 0 (akkor nulla, ha X, Y függetlenek)
  - 2. Kommutatív
- Igaz még, hogy:

$$\mathrm{I}(X,Y) = \mathrm{H}(X) - \mathrm{H}(X|Y) = \mathrm{H}(Y) - \mathrm{H}(Y|X)$$

- Az összefüggéseket a legkönnyebb ez az ábra alapján megjegyezni:



5. Define the typical set of an IT source (AEP) and derive its properties.

- Információ elméletben a nagy számok törvényének megfelelő fogalom az Asymptotic Equipartition Property (AEP).
- A nagy számok törvénye kimondja, hogy egymástól független egyenlő eloszlású véletlen változóknál az

$$\frac{1}{n} \sum_{i=1}^{n} X_i$$

  érték közel van a várható értékhez kellően nagy n esetén.
- Az AEP azt állítja, hogy

$$\lim_{n\to\infty} \frac{1}{n} \operatorname{ld} p(x_1, ..., x_n) = -\operatorname{H}(X)$$

  Ahol x_1, x_2 stb. egyenlő eloszlású valségi változók, a $p(x_1, \ldots, x_n)$ pedig az x_1, x_2 stb.sorozat valószínűsége.
- Ennek következménye, hogy a $p(x_1, \ldots, x_n)$ valség közel lesz az 2^{-nH} értékhez.
- Ez lehetővé teszi számunkra, hogy szétosszuk az összes szekvenciát tartalmazó halmazt kétfelé
  - a tipikus halmazra, amelyben a “sample entropy” közel van az igazi entrópiához,
  - meg a nem tipikus halmazra, amiben minden más van.
- A könyv így fogalmaz: “Almost all events are almost equally surprising”
- Az X AEP IT source-hoz tartozó tipikus halmaz formális definíciója:

$$A = \{\mathbf{x} = (x_1, x_2, ..., x_n) \mid 2^{-n\operatorname{H}(X)-\varepsilon} \leq p(\mathbf{x}) \leq 2^{-n\operatorname{H}(X)+\varepsilon}\}$$

Tipikus halmaz tulajdonságai

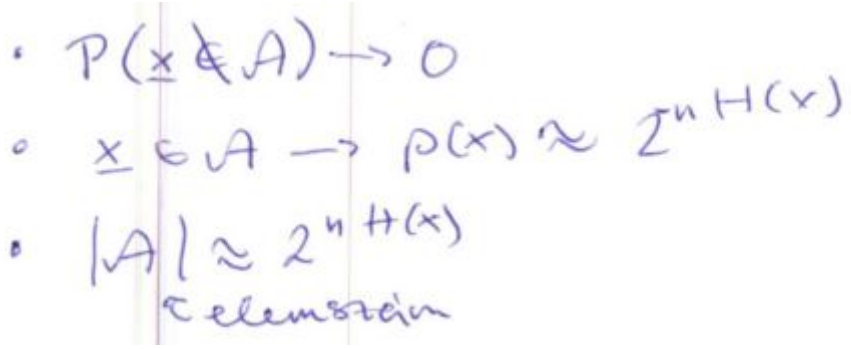
1. Annak valószínűsége. hogy egy random vektor a halmazban van:

$$(1-\varepsilon) \leq P(\mathbf{x} \in A) \leq 1$$

2. Tipikus halmaz mérete:

$$(1-\varepsilon) \cdot 2^{n\operatorname{H}(X)-\varepsilon} \leq |A| \leq 2^{n\operatorname{H}(X)+\varepsilon}$$

Konklúziók

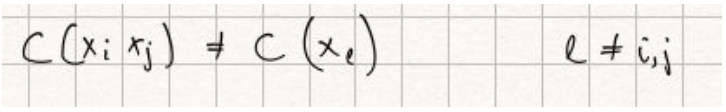


6. Define the properties of uniquely decodable codes, and discuss the source coding theorem.

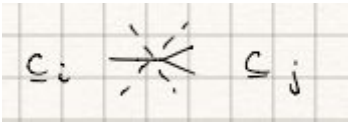
- Mese: Ez a téma a forráskódoláshoz kapcsolódik, ami más néven adattömörítés. Ennek célja, hogy egy üzenetet, amely egy véges halmaz (forrás ABC) elemeiből álló véges sorozat, egy másik véges halmaz (kódábécé) elemeiből álló sorozattal reprezentálhatjuk úgy, hogy ez a reprezentálás a lehető legrövidebb és belőle az üzenet visszaállítható legyen.
- Az X itt a forrásábácét az Y* pedig kódszavak halmaza. A kettő között egy függvényt kódnak nevezünk. X elemeiből alkotott sorozatok az üzenetek / közlemények.
- Az $f : X \to Y^*$ **egyértelműen dekódolható**, ha minden kódsorozat csak egy üzenetet kódol, pontosabban, ha u, v üzenetekre

  $$f(u_1)f(u_2)..f(u_k) = f(v_1)f(v_2)..f(v_m)$$

  akkor és csak akkor, ha $u = v$.
- Ez a Petra által írt definíció. A Lina jegyzetében szereplő defi: (értelmezésem szerint egy kódszó sem áll elő másik kettőből, de ezt nem 100%-ig értem)



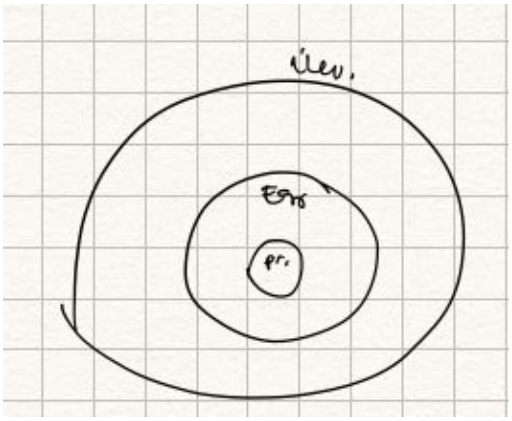
- **Prefix kód:** ha a lehetséges kódszavak közül egyik sem folytatása a másiknak, vagyis bármely kódszó végéből bármekkora szegmenst levágva nem kapunk másik kódszót. Egy prefix kód egyben egyértelműen dekódolható is. ($\not<$: does not precede)



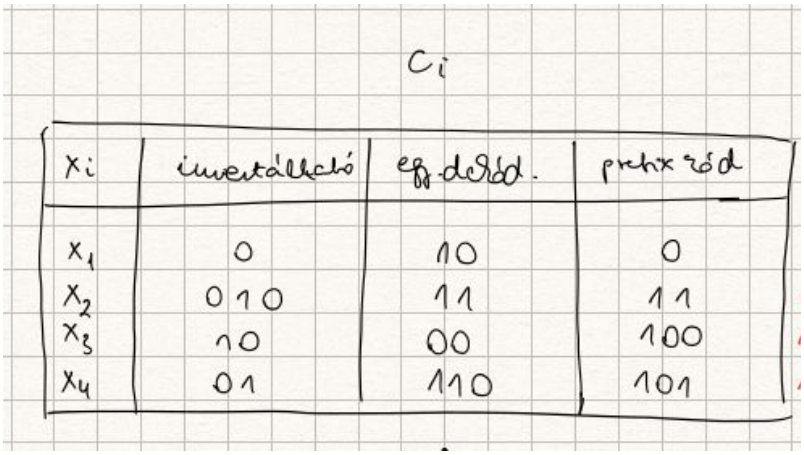
- **Invertálható kód:** (ez sajnos nem tűnik számomra logikusnak)

$$\mathbf{x_1 \neq x_2 \rightarrow C(x_1) \neq C(x_2)}$$

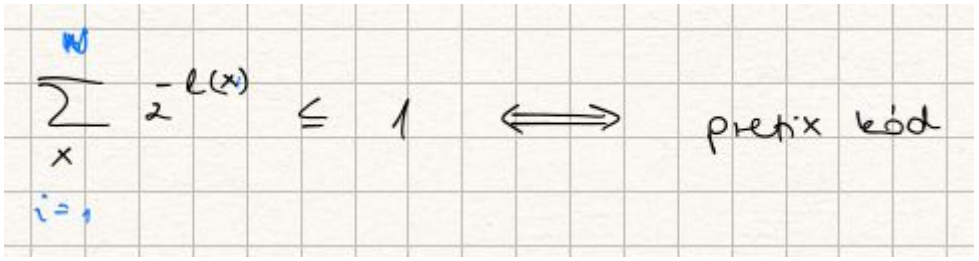
- **Prefix, invertálható és egyértelműen dekódolható kódok viszonya**



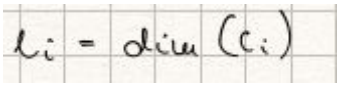
- **Példa**



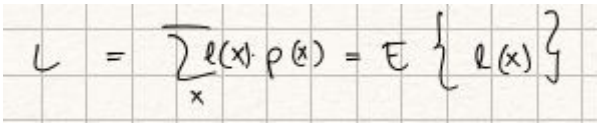
- **Kraft (McMillian) egyenlőtlenség:**



- Itt az l(x) a kódszó hosszát jelenti:



- Az átlagos kódhossz:



Forráskódolás alaptétele

- Meghatározza a lehetséges adattömörítés határait.
- Azt mutatja meg, hogy lehetetlen úgy tömöríteni adatot, hogy az átlagos kódhossz kevesebb mint a forrás entrópiája, vagy legalábbis infót fogunk veszíteni.

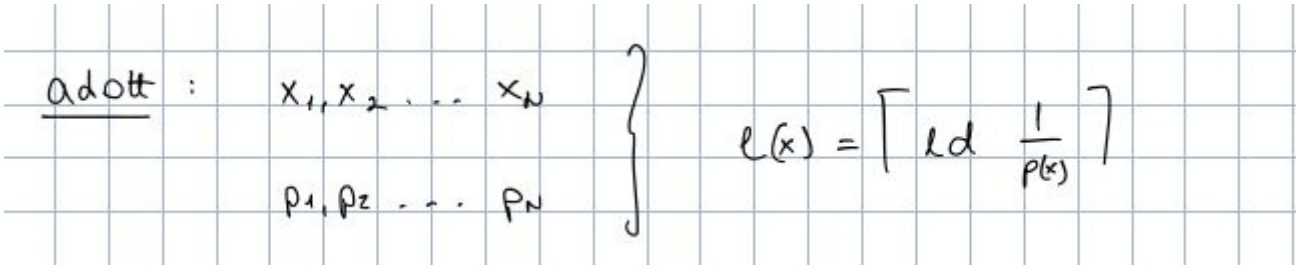
$$\mathrm{H}(X) \leq L = \sum_{x} p(x) \cdot l(x)$$

- *N független és egyenletes eloszlású véletlen valváltozó H(X) entrópiával NH(X) bitre tömöríthető információvesztés nélkül.*

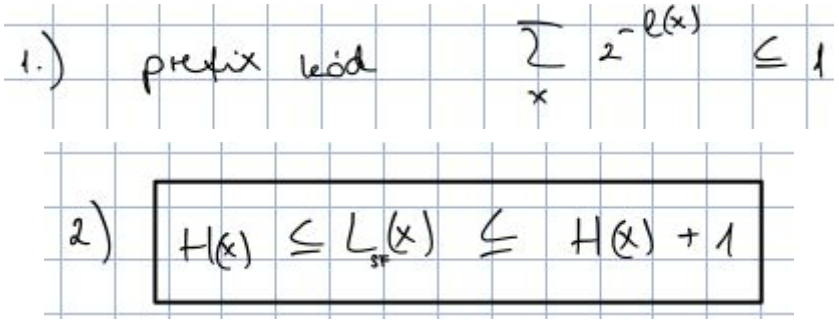
7. Describe the Shannon-Fano, Huffman, and arithmetic coding and discuss their performance.

Shannon-Fano

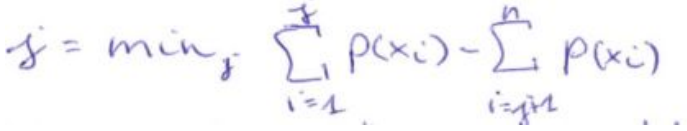
- A Shannon Fano kódok kódhossza:



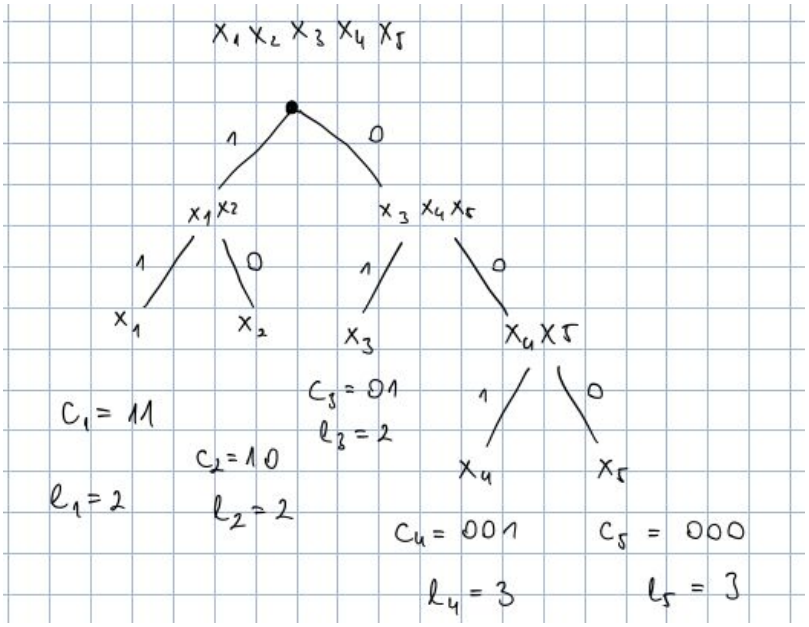
- Tulajdonságok:



- Így kell csinálni az algoritmust:
  - A valségeket megfogod és csökkenő sorrendbe rendezed.
  - Felosztod őket kétfelé úgy, hogy a lehető legkisebb legyen a két csoport között a különbség a valség összegben.



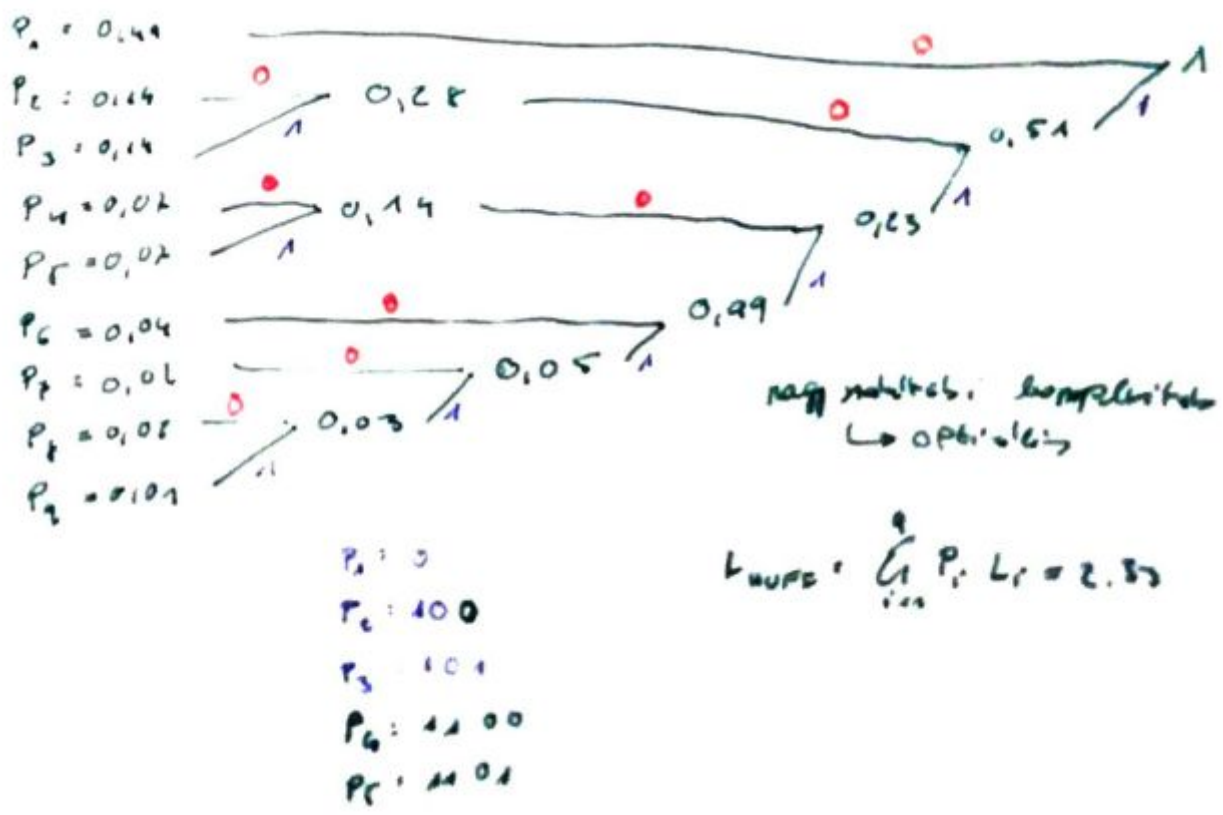
  - A bal oldalónak 1-es értéket, a jobb oldalinak nullát adunk.
  - Addig ismételjük ezt, amíg minden szimbólumnak értéket nem adtunk.
  - Azt illik tudni, hogy ennek nem feltétlenül optimális kód lesz a vége.



Huffman

- Optimális kód megvalósítására alkalmas

- Kritériumok az optimális kódhoz
  - $p_1 < p_2 < \ ... \ < p_N$
  - $l_N = l_{N-1}$ és az utolsó biten különbözőek
  - $p_i > p_j \rightarrow l_i \leq l_j$
- Bináris fákat használunk itt is.
- A két legkisebb valószínűség összevonásával addig redukáljuk a problémát, amíg az triviális nem lesz, vagyis amíg összesen két valószínűségünk marad. Ezt n-2 lépésben érhetjük el. Ezután az összevonások megfordításával, mindig a megfelelő kódszó kétféle kiegészítésével újabb n-2 lépésben felépítjük az optimális kódot, a Huffman-kódot.
- O(n^2) komplexitása van a cuccnak.



Adaptív Huffman

- A fenti algoritmus két lépésben hajtható végre. Először meghatározzuk a forrásbetűk relatív gyakoriságát ~ valószínűségét, majd ennek felhasználásával elvégezzük a tényleges kódolást.
- Nem mindig engedhető meg azonban olyan nagy mértékű késleltetés, hogy csak az összes bemeneti adat megérkezése után kezdünk hozzá a kimenet előállításához, másrészt a kétmenetes beolvasás akkor is lassítja az algoritmust, ha a bemenet már rendelkezésre áll.
- Emiatt a gyakorlatban egymenetes algoritmust célszerű használni.
- Egy forrásbetűt az előző forrásbetűk gyakorisága alapján kódolunk, se ezzel együtt lépésenként változik maga a kód. Tehát az aktuális forrásbetű kódolását egy az előzőleg feldolgozott forrásbetűkre optimális kóddal hajtjuk végre
- Ez az Adaptív huffman.

Shannon-Fano-Elias kód

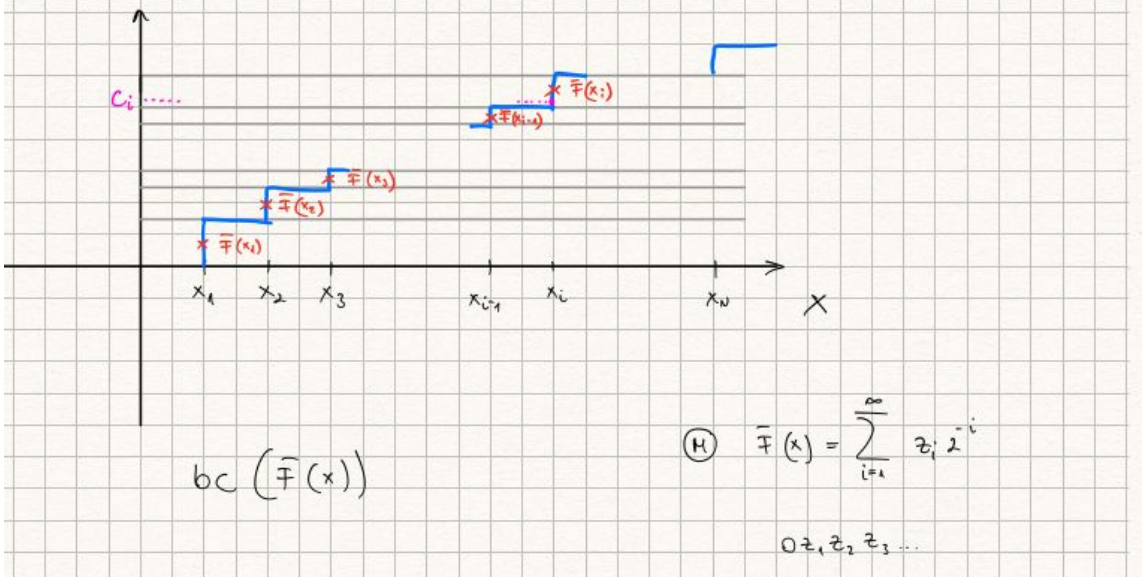
- Prefix kódok használatával akár 1 bitet is veszíthetünk a tömörítés alsó korlátjához képest. Ezen a blokkonkénti kódolással lehet segíteni. Azt azonban láthatjuk, hogy minél nagyobb a blokk méret amit választunk, annál kisebb lesz a veszteség VISZONT bejön a késleltetés mint tényező, ami exponenciálisan nő a blokk mérettel tehát szopó.
- Aritmetikai kódolás: valós időben történik a kódszó előállítása és visszafejtése, nem lép fel késleltetés akkor sem, ha nagyon hosszú blokkokat kódolunk.

- Az SFE egy aritmetikai kódolás (Ha jól értem)
- Felső határ:

$$H(x) \leq L_{SFE}(x) \leq H(x) + 2$$

módosítjuk

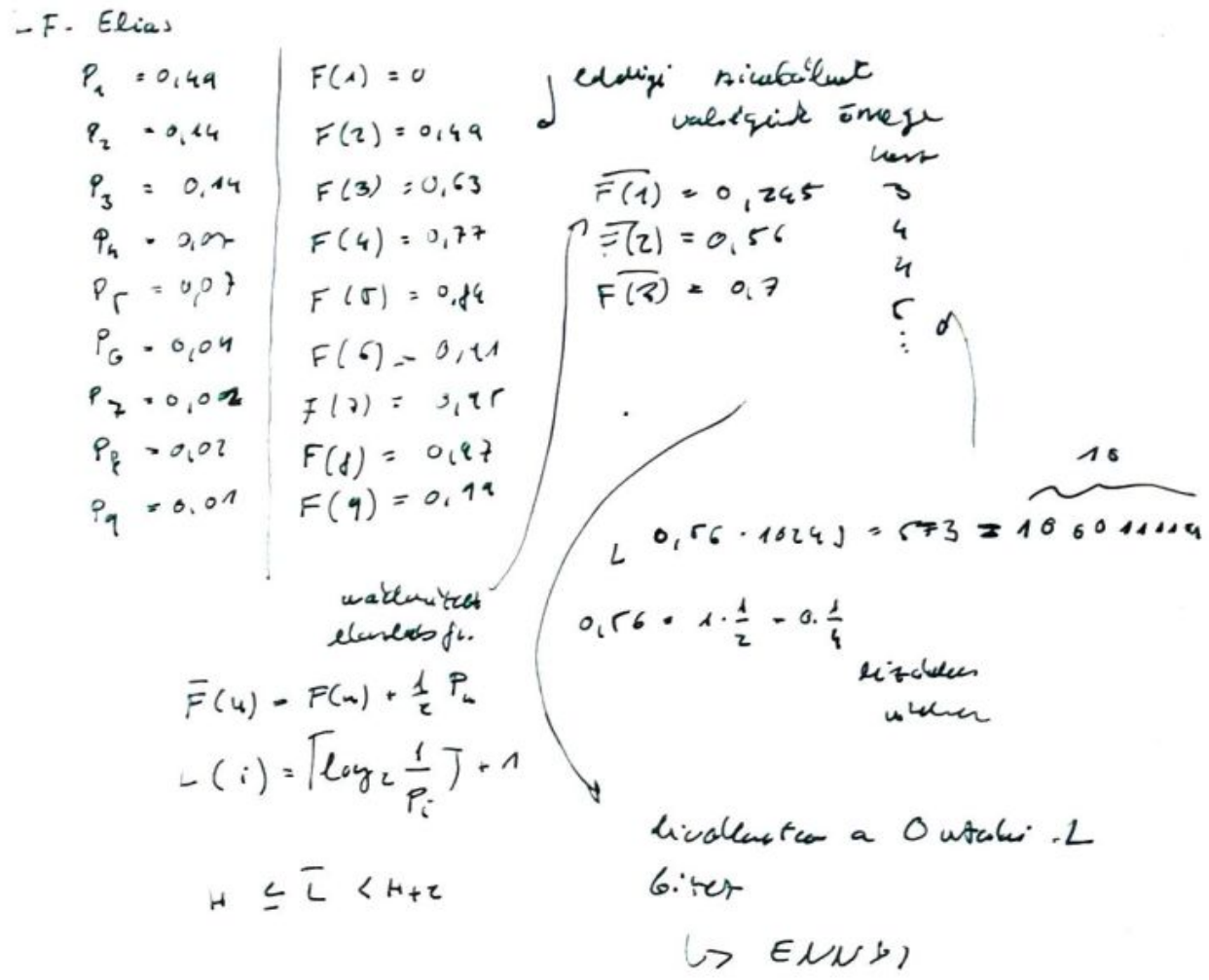
$$\bar{F}(x) = \sum_{x \geq a} p(a) \quad \rightarrow \quad \bar{F}(x) = \sum_{x > a} p(x) + \frac{1}{2} p(x)$$



$bc\left(\bar{F}(x)\right)$

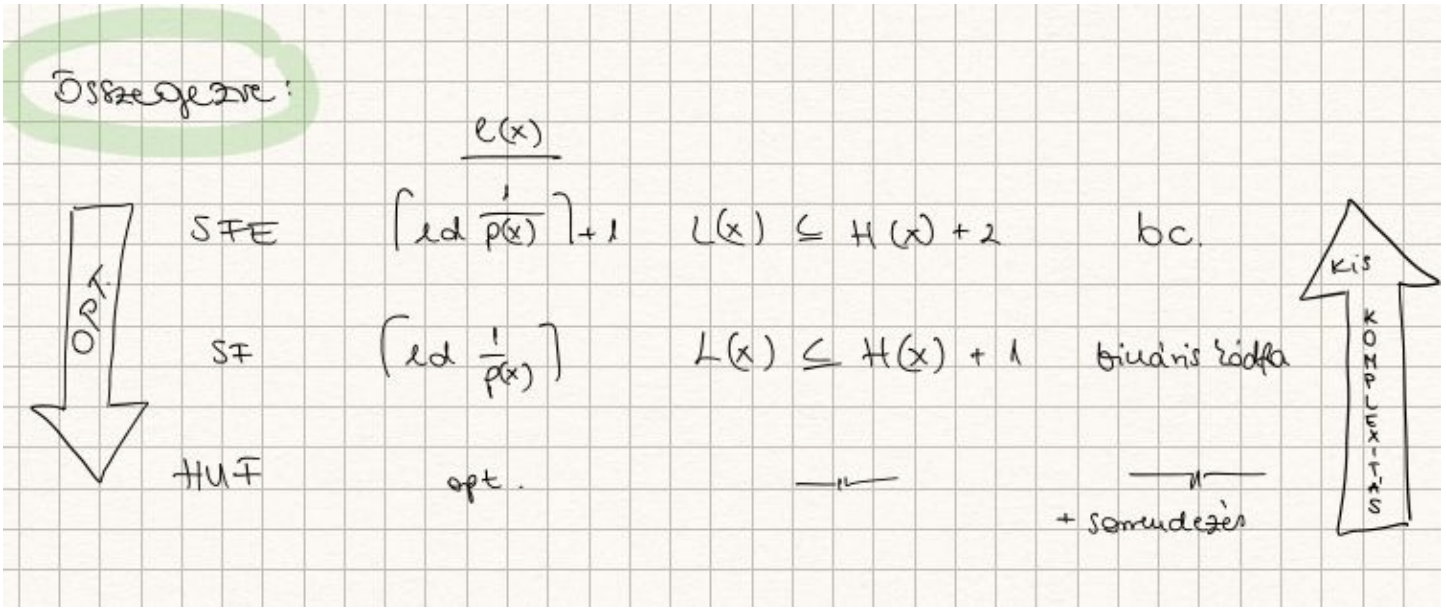
(H) $\bar{F}(x) = \sum_{i=1}^{\infty} z_i 2^{-i}$

$0 z_1 z_2 z_3 \ldots$



Összehasonlítás

- És akkor a három hogy hasonul egymáshoz
- Dióhéjban: Ahogy nő a teljesítőképesség úgy csökkent az implementálhatóság. Komplexitás vs teljesítőképesség.

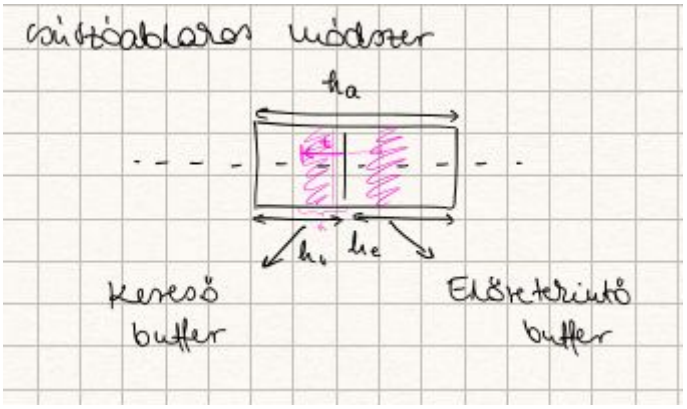


8. Describe the LZ based compression algorithm.

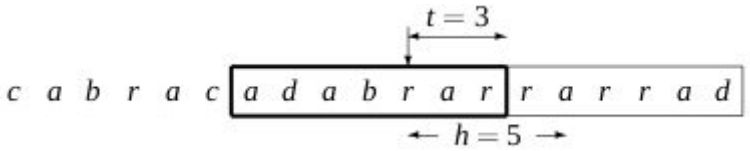
- Eddigi kódoknál az adó és vevő között átvitt bitek két csoportra oszlanak:
  - Először átvisszük a blokk kódot leíró információt. Ez egy állandó költség, mely független az üzenet tényleges hosszától.
  - Majd átvisszük az üzenet kódszavait.
- Eddig azt mondtuk, hogy az üzneet végtelen hosszú. Ilyen módon a kódok aszimptotikus viselkedését tekintve az állandó költség fajlagosan nullához tart, tehát elhanyagolható. A gyakorlatban azonban véges forrásokkal van dolgunk.
- Véges források esetén az állandó költség akár nagyobb is lehet mint az üzenet kódszavainak hosszúsága.
- Ezt elkerülendő, jó lenne, ha rendelkezésünkre állna egy olyan technika, amelynek nincs állandó költsége, de aszimptotikusan ugyanolyan jó tömörítési arányt ér el, mint a blokk kódok.
- Az állandó költség abból adódik, hogy a kódot a forráson előzetesen elvégzett statisztikai vizsgálatok alapján hozzuk létre, tehát ezek az adatok szükségesek a kód leírásához.
- Ehelyett menet közben fogunk információt gyűjteni a forrás szimbólumokról, vagyis az aktuális szimbólumot az ezt megelőző szimbólumok alapján kódoljuk. Az ilyen kódokat adaptív kódoknak nevezzük. pl Adaptív Huffman
- Ilyen adaptív kódok a Lempel Ziv kódok is

LZ77

- 1977-ben publikálták nahát
- A kódoló a forrás szimbólum sorozatát egy h_a hosszú csúszó ablakon keresztül vizsgálja. Az ablak két részből áll: egy kereső pufferből, amely a legutóbb kódolt h_k darab kódolandó szimbólumot tartalmazza, és egy előretekintő pufferből, mely a következő h_e darabot. $h_a = h_k + h_e$.
- A kódoló a kereső pufferben megkeresi az előretekintő puffer első szimbólumával megegyező szimbólumokat. Ehhez egy hátrafelé haladó mutatót használ.
- Megnézi, hogy a megtalált pozíciókkal kezdődően a kereső pufferben lévő szimbólumok milyen hosszan egyeznek meg az előretekintő puffer szimbólumaival, és a találatok közül azt választja ki, amelytől kezdve a leghosszabb az egyezés.



- A kódoló ezután küld egy <t, h, c> (MARIHUÁNA LOL) hármast, ahol
  - t: kereső pufferben megtalált szimbólum távolsága az előretekintő puffertől
  - h: a kereső és az előretekintő puffer egyező szimbólumainak legnagyobb hosszúsága
  - c: az első előretekintő pufferben lévő nem egyező karakter kódszava
    - hogy elkerüljük azt, hogy az előretekintő puffer szimbólumait nem találjuk a kereső pufferben. Ilyenkor t és h értéke 0.
- Ez egy nagyon egyszerű adaptív algoritmus, amihez nem kell ismernünk/tudnunk semmit a forrásról. Ráadásul a hatékonysága aszimptotikusan közelíti az optimális algoritmusét.
- Feltételezzük, hogy a kódolt szimbólumok periodicitása kisebb, mint a kereső buffer hossza -> nem igaz ezért van LZ78
- Példa: https://www.youtube.com/watch?v=cSyK2iCqr4w



LZ78

- A kódoló és a dekódoló is szótárt épít az előzőleg előfordult sorozatokból.

a kódoló kimenete	szótár index	szótár bejegyzés
$\langle 0, f(d)\rangle$	1	d
$\langle 0, f(a)\rangle$	2	a
$\langle 0, f(b)\rangle$	3	b
$\langle 3, f(a)\rangle$	4	ba
$\langle 0, f(c)\rangle$	5	c
$\langle 1, f(a)\rangle$	6	da
$\langle 3, f(b)\rangle$	7	bb
$\langle 2, f(c)\rangle$	8	ac
$\langle 6, f(b)\rangle$	9	dab
$\langle 4, f(c)\rangle$	10	bac
$\langle 9, f(b)\rangle$	11	$dabb$
$\langle 8, f(d)\rangle$	12	acd
$\langle 0, f(e)\rangle$	13	e
$\langle 13, f(c)\rangle$	14	ec
$\langle 1, f(e)\rangle$	15	de
$\langle 14, f(d)\rangle$	16	ecd
$\langle 13, f(e)\rangle$	17	ee

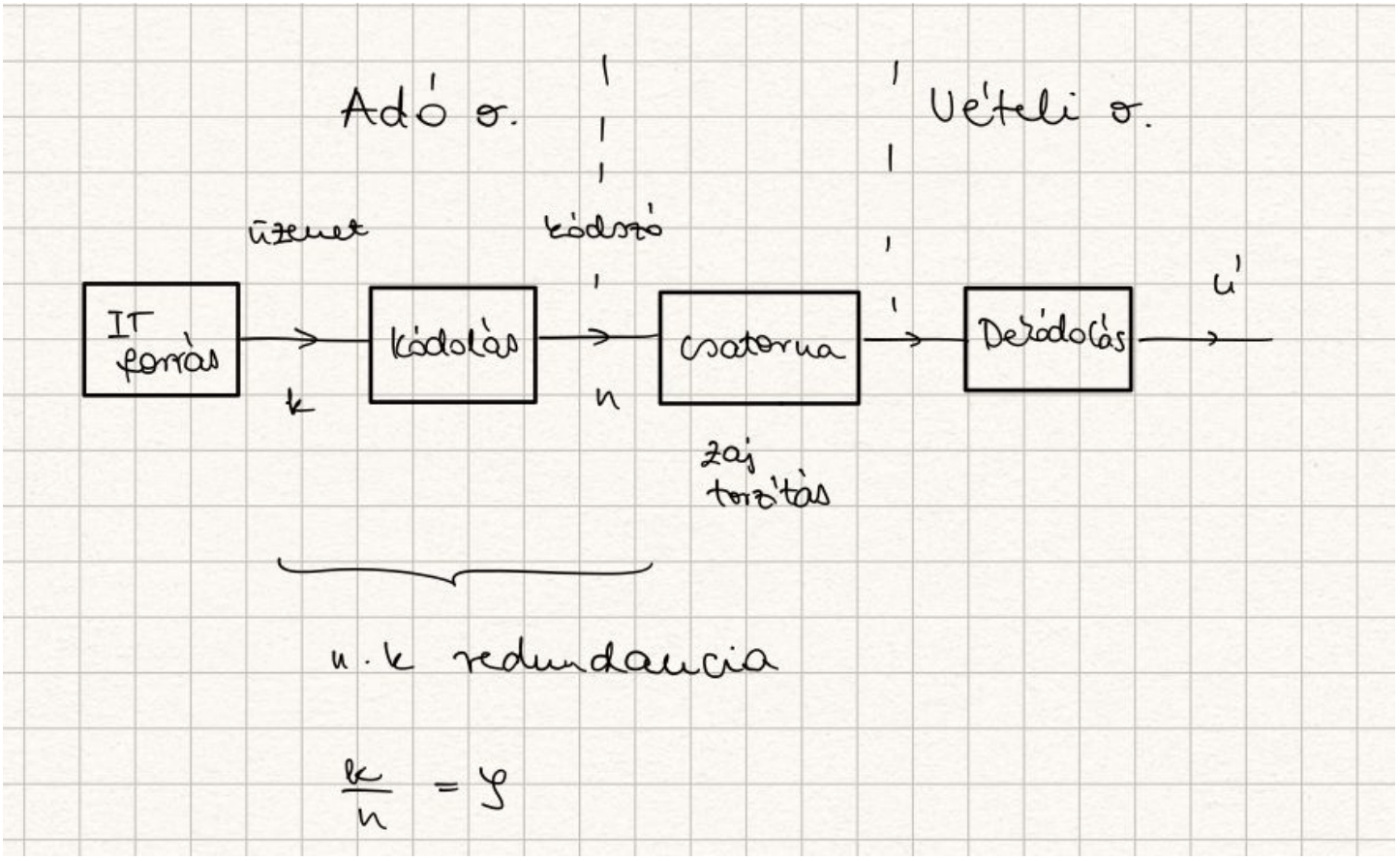
- A kódoló megkeresi a forrásszimbólumok aktuális pozíciójától kezdődő leghosszabb egyezést a szótárban.
- Átküld egy <i, c> párt:
  - i: az egyező karaktersorozat szótárbeli indexe
  - c: az első nem egyező karakter kódja
- Majd felveszi a szótárba az i. indexű karaktersorozat és a c karakter konkatenációjaként kapott sztringet. Ha nem talál egyező karakter sorozatot akkor a <0,c>-t.
- Az algoritmus egyik hibája, hogy a szótár folyamatosan korlát nélkül növekszik. A gyakorlatban egy bizonyos határon túl gátat szabunk a növekedésnek:
  - rendszeresen eltávolítjuk a felesleges vagy ritkán használt szimbólumokat
  - vagy egy fix idő után fix szótárasként működik tovább az eljárás.
- https://www.youtube.com/watch?v=qVmpPk1C2wc

LZW

- Terry Welch módosítja az LZ78-at úgy hogy megtakarítható legyen az <i, c> -ből a c elküldése.
- Szükséges, hogy a szótában már a kiinduló állapotban is szerepljen az összes egybetűs szimbólum a forrásábécéből.
- A kódolás során az aktuális pozíciótól kezdve addig olvassuk be a forrásszimbólumot a p pufferbe, amíg a sorozat szerepel a szótárban. Ha az a karakter az első olyan amelyre pa (konkatenált) nincs benne a szótárban, akkor átküldjük a p sorozat indexét, a pa sorozatot felvesszük a szótárba és az a karaktertől kezdve folytatjuk az eljárást.
- Pl. a GIF formátum LZW-t használ.

9. Describe the channel coding theorem, and define the channel capacity and elaborate on its calculation for symmetric channels.

Bevezetés

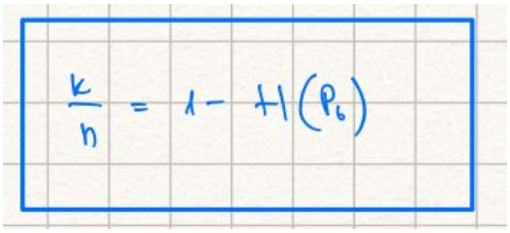


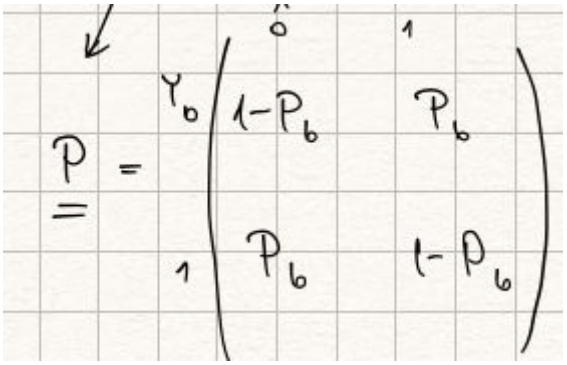
- Cél: Zajos csatornán megbízható infó átvitel
- csatorna ~ átviteli közeg
- két probléma:
  - csatorna bemeneti ABC /= forrás ABC
  - átvivő közeg zajos, tehát a bemenet nem feltétlenül egyezik meg a kimenettel
- ezeket a problémákat hivatott megoldani a csatorna kódolás, elvi lehetőségeit a csatornakódolási tétel mutatja meg
- a beadott üzenetek bizonyos valséggel meghibásodnak
- a csatornakódolás alapvető problémája: hogyan lehet egy megbízhatatlan eszközön (zajos csatorna) üzenetet átküldeni úgy, hogy minél jobban kihasználjuk a csatornát
- ennek a kihasználtságnak elvi határa a csatornakapacitás
- k/n az adatátviteli sebesség (amúgy R-rel van jelölve máshol)
- ezt az értéket szeretnénk értelemszerűen maximalizálni. tehát mi az a max k/n, amin még megbízható a komm?
  - Shanon tétel:

$$C \geq \frac{k}{n}$$

BSC eset

- Binary Symmetric Channel ugyebár
- Itt a $H(P_b)$ az entrópia, P_b pedig a bit error rate (annak a valsége, hogy egy bit átfordul)



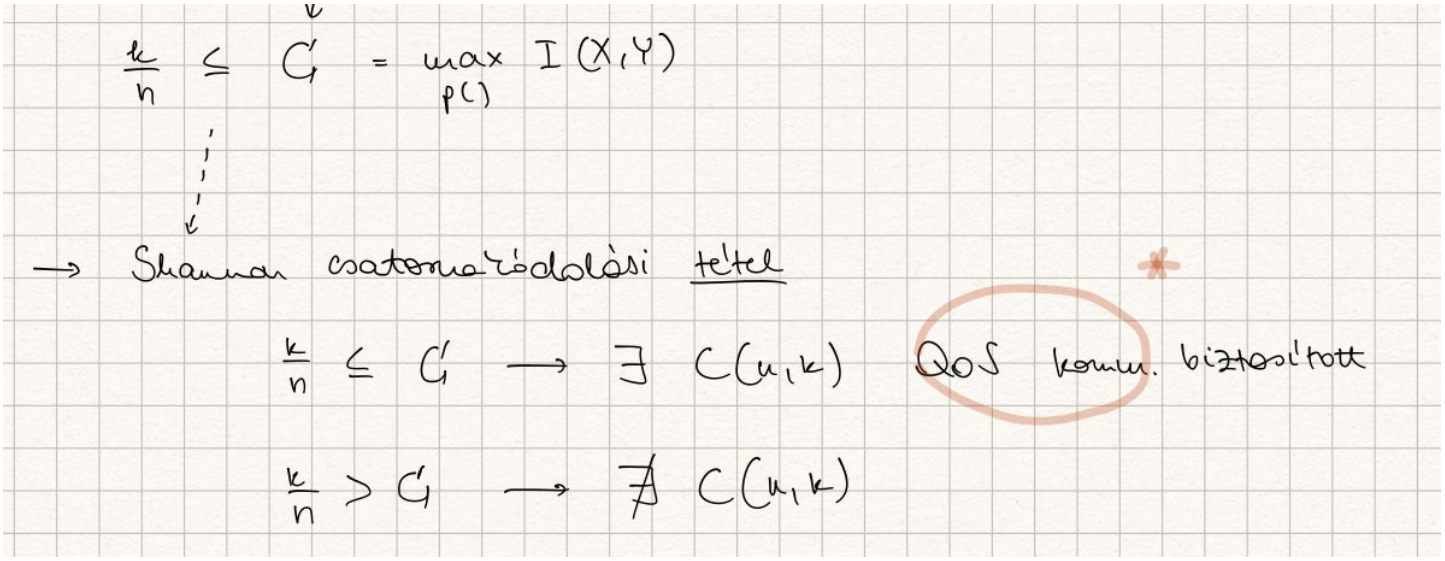


Csatorna, csatornakapacitás

- A diszkrét csatorna leírható a bemeneti ABC-vel a kimeneti ABC-vel és az átmenet valószínűségekkel.
- Az ABC-k legalább két elemű véges halmazok.
- Mindig feltételezzük, hogy nincs szinkronizáció hiba, tehát a szimbólumok nem vesznek el útközben, max sérülnek.
- A csatorna akkor diszkrét memóriamentes, ha a kimeneti szimbólumának eloszlása mindig csak az aktuális bemenettől függ.
- Meghatározza az adatátviteli sebességet
- minden diszkrét memóriamentes csatorna esetén

$$C = \max_{p(x)} I(x,y) = \max_{p(x)} \{\mathrm{H}(Y) - \mathrm{H}(Y|X)\}$$

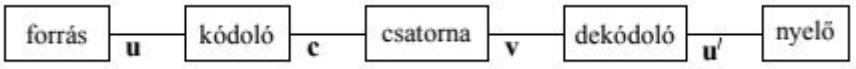
Csatornakódolási tétel

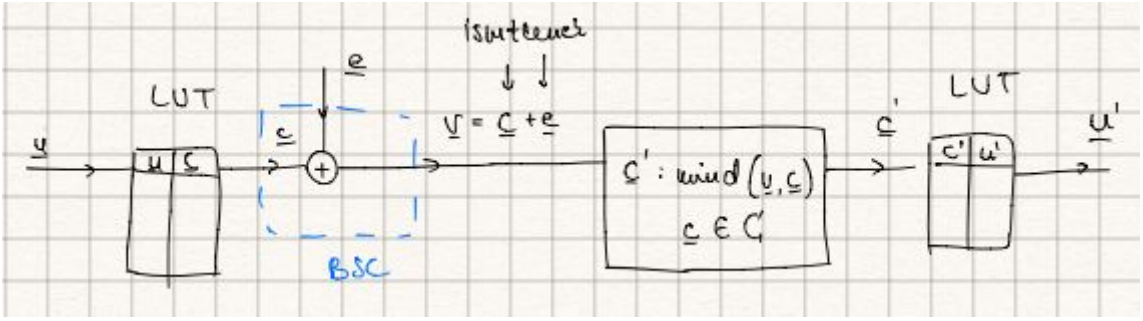


10. Define and explain the relationship between the following properties and parameters of error correcting codes

1. minimum code distance;
2. code-length and message-length versus performance (Singleton and Hamming bounds);
3. general algorithmic complexity of coding with tables

Bevezetés





- u: üzenet vektor, c:kód vektor, v: vett vektor, csatorna kimenete, f: kódoló fv, g: dekódoló fv
- a forrásabc-ből (F) kiveszünk egy k hosszú u vektort és leképezzük egy n hosszú c vektorba (ez a kódabc/csatorna bemeneti abc-ből van: Q). A csatorna kimenete egy szintén n hosszú v vektor lesz (Q-beli). Abban az m pillanatban, amikor a c_m != v_m, hiba történt.
    - hibák száma: t = d(c, v) a hibákat tartalmazza, ahol c_i != v_i , ezt nevezzük Hamming távolságnak

$d(\mathbf{c}, \mathbf{v})$ valóban távolság, hiszen

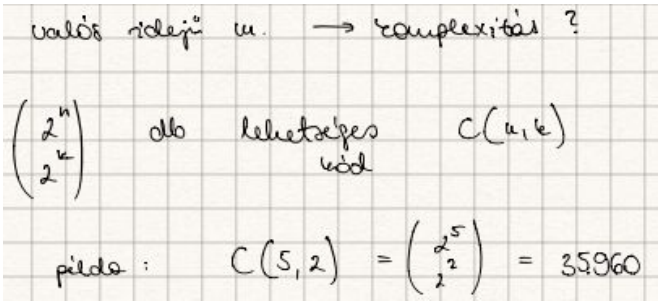
$$d(\mathbf{c}, \mathbf{v}) \geq 0, \qquad \text{és igaz a háromszög-egyenlőtlenség:}$$

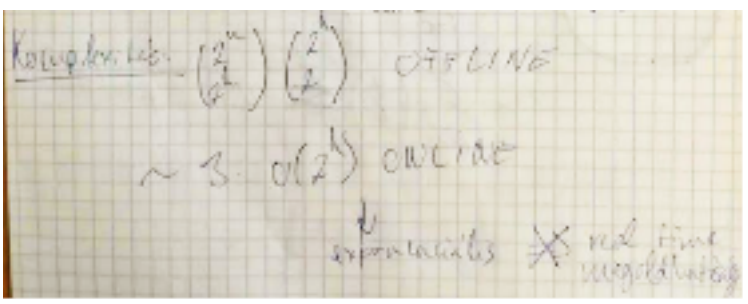
$$d(\mathbf{c}, \mathbf{v}) = d(\mathbf{v}, \mathbf{c}), \qquad d(\mathbf{c}, \mathbf{v}) \leq d(\mathbf{c}, \mathbf{w}) + d(\mathbf{w}, \mathbf{v}).$$

    - dekódolásnál azt csináljuk, hogy megkeressük a v-hez azt a c' kódszót, ami a Hamming távolság szerint a legközelebb van hozzá. Az a trükk, hogy úgy kéne meghatározni ezt a függvényt, hogy ne kelljen az összes d(c,v)-t kiszámolni. Ha c' = g(v), akkor

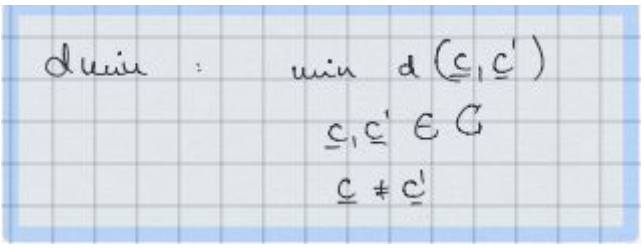
$$d(\mathbf{c}', \mathbf{v}) = \min_{\mathbf{c} \in C} d(\mathbf{c}, \mathbf{v}).$$

    - **táblázatos dekódolás**: ha mégis felírnánk az összes ilyen kódszót, akkor egy táblázatba lehetne rendezni, de ez egy nagy komplexitású feladat (pazarlás). **Komplexitása = q^n** db üzenetből áll, ahol q = |Q|.
      Előadás példa komplexitásra:

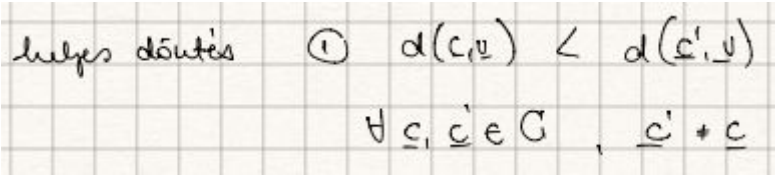


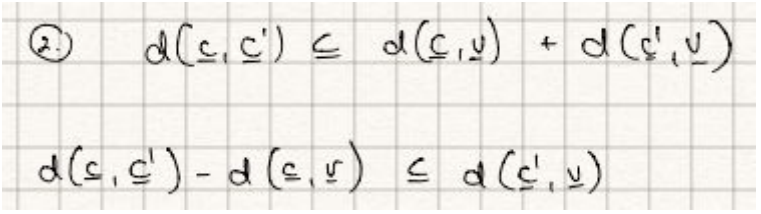


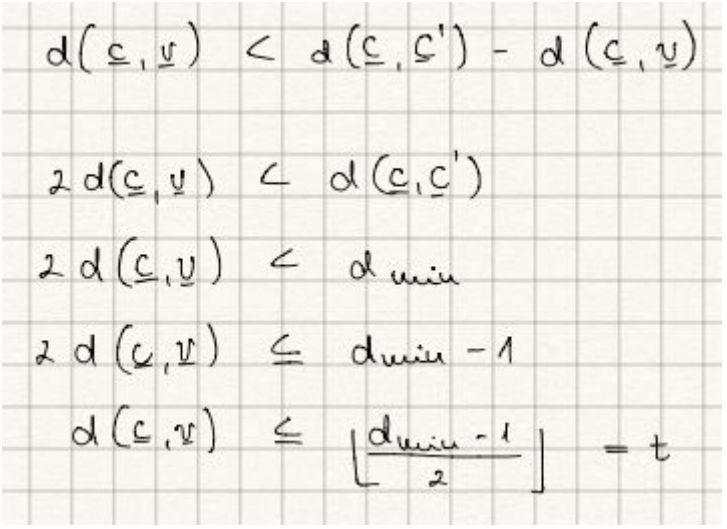
- Az f kódoló fv leglényegesebb tulajdonsága a kódtávolság, amit d_{min}-nel jelölünk. Ez a C kód egy paramétere, amit úgy lehet kiszámolni, hogy vesszük a Hamming távolságát az eredeti kódszónak és a dekódoló függvényben kapott kódszónak és kiválasztjuk a legkisebbet.
    - Magyarabbul: végignézzük, hogy melyik dekódolt kódszó tér el legkevésbé az eredeti kódszótól és ahány bitben eltér, annyi lesz a **minimális kód távolság**.



- Hibajelzésnek nevezzük azt, amikor nem azt kérdezzük, hogy hány hiba van, hanem hogy van-e hiba? Nyilván egy v vett szó esetén akkor tudjuk a hibázást észrevenni, ha v nem kódszó, amire garancia, hogy ha c küldött kódszó esetén: $d_{min} > d(v, c) \Rightarrow d_{min} > t$
    - tehát egy d_{min} kódtávolságú kód minden, legfeljebb d_{min}-1 számú hibát jelezni tud.
    - helyes döntés: v vett szóból a c küldött kódszó egyértelműen visszaállítható legyen, vagyis minden más c' kódszóra igaz lesz az (1) képlet
    - Hibajavítás képlete a háromszög egyenlőtlenség alapján jön ki:







Singleton korlát, Hamming korlát (lila keretben van, a kékek speciális esetek):

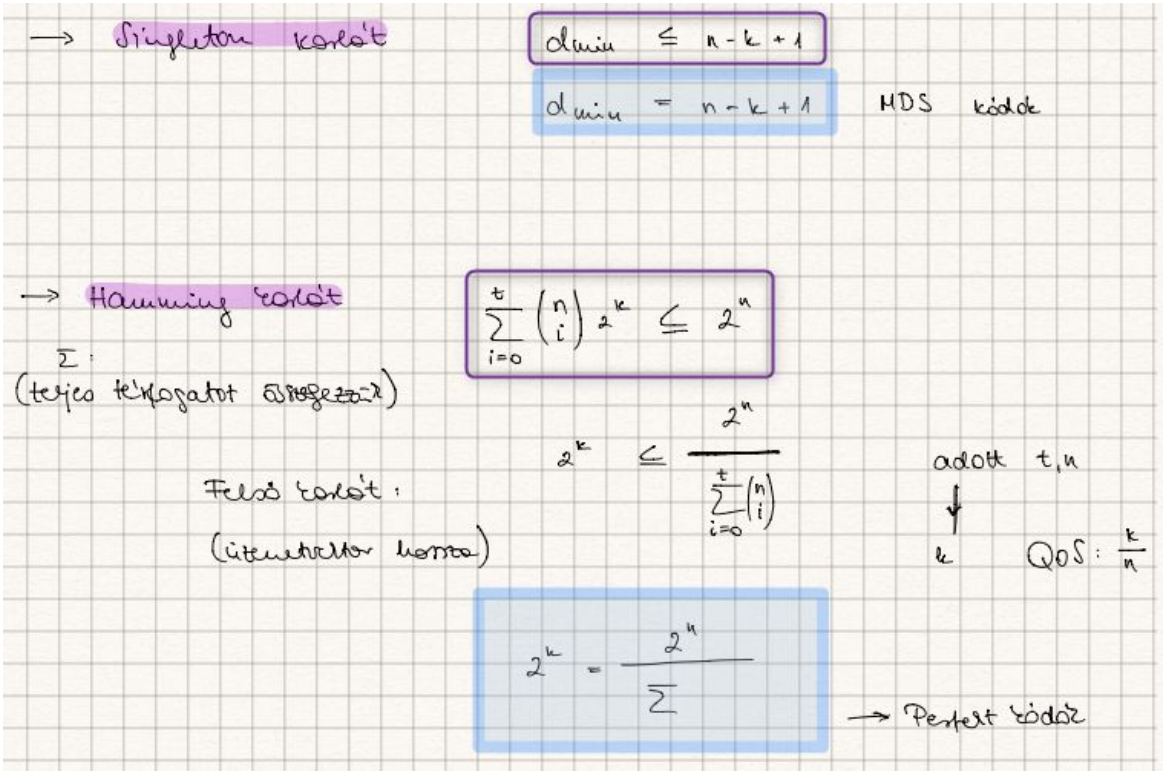
4.1. tétel (Singleton-korlát). *Egy M kódszóból álló, n hosszú és d_{min} kódtávolságú kódra*

$$M \leq q^{n-d_{min}+1}.$$

BIZONYÍTÁS: Legyen k egy természetes szám, melyre

$$q^{k-1} < M \leq q^k.$$

Jellegzetes esetben $M = q^k$, vagyis a kódoló k hosszú forrásszegmensekhez rendel n hosszú vektorokat. Azt mondjuk ilyenkor, hogy a kódunk (n,k) paraméterű. Ebben az esetben a Singleton-korlát alakja

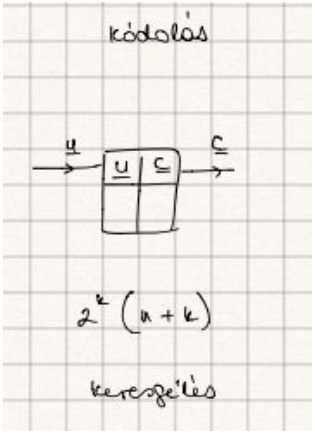


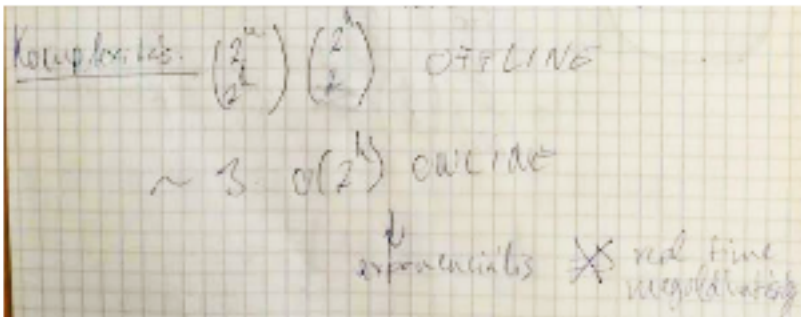
4.2. tétel (Hamming-korlát). *Ha egy (n,k) paraméterű kód t hibát tud javítani, akkor*

$$\sum_{i=0}^{t} \binom{n}{i} (q-1)^i \leq q^{n-k}. \qquad (4.3)$$

- MDS: Maximum distance separable

General algorithmic complexity of coding with tables



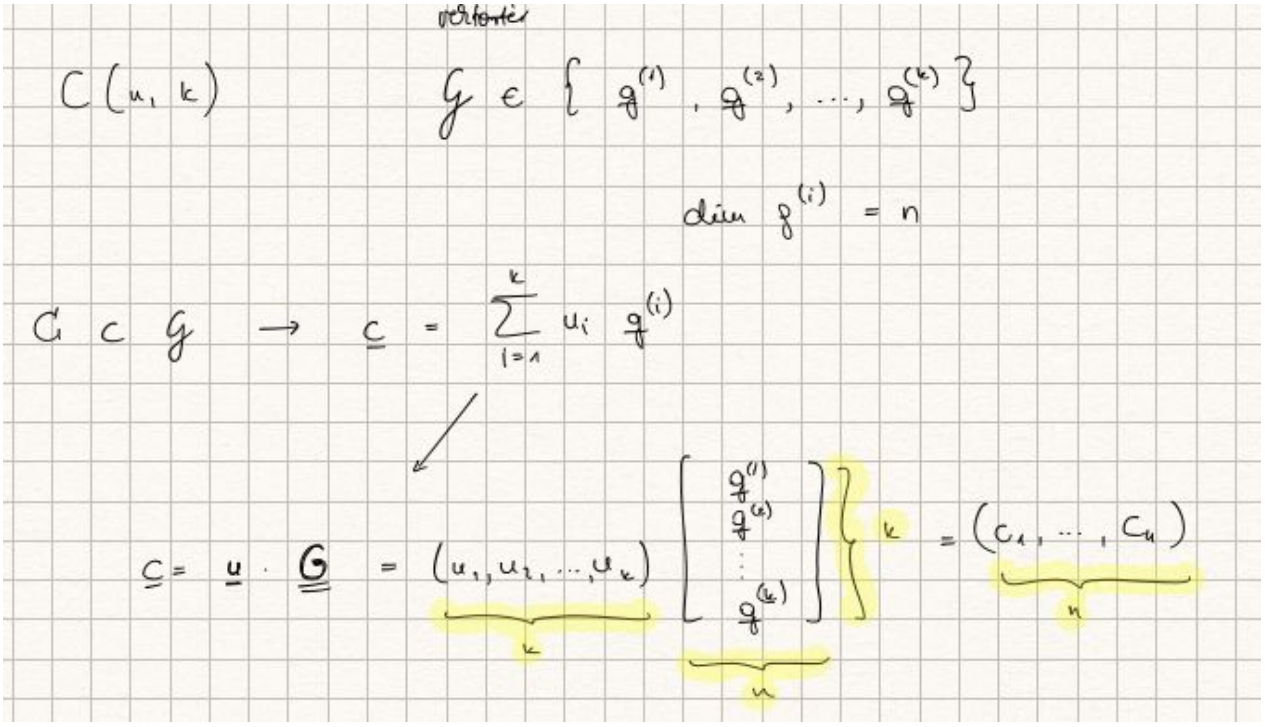


11. Introduce the concept of linear block coding and explain the meaning of systematic codes, generator matrix and parity check matrix (and their relationship), algorithmic complexity of linear block coding (including detection).

- Az általános look up tables kódolás nem túl hatékony: azért, hogy ezen javítsunk bevezethetjük a lineáris blokk kódolást.

Generátor mátrix

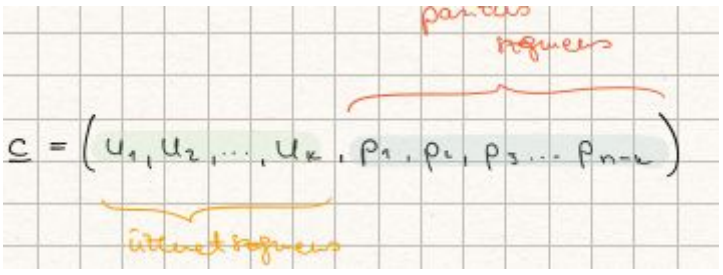
4.3. definíció. *Egy C kód* ***lineáris****, ha a C halmaza lineáris tér, azaz ha minden* $\mathbf{c}, \mathbf{c}' \in C$*-re* $\mathbf{c} + \mathbf{c}' \in C$.



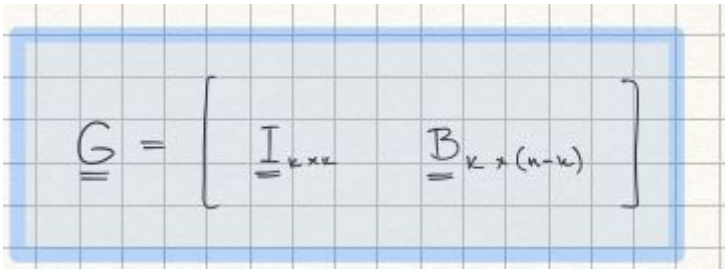
- G halmaz elemeiből soronként képzünk egy mátrixot, ez a generátor mátrix.
- Az u az üzenetvektor.
- Ezt a generátor mátrixot összeszorozva az üzenet vektorral megkapjuk a kódvektort.

Szisztematikus kódok

- Egy kódot szisztematikusnak hívunk leírható a üzenet bitjeivel és még pár kiegészítő bittel, valahogy így:



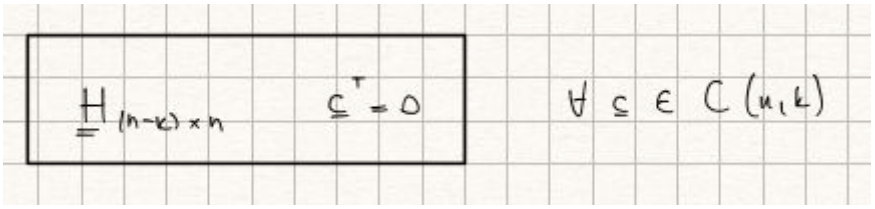
- Ebben az esetben a G generátor mátrix előáll az alábbiak szerint:



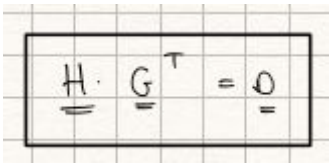
Ahol ugye I az egységmátrix, B pedig egy mátrix.

- Dekódolásnál csak el kell hagyni az üzenet végét. (értelemszerűen)

Paritásellenőrző mátrix



- H segítségével tehát meg tudjuk állapítani, hogy egy vett szó valóban kódszó e.
- Arra jó, hogy csökkentsük a dekódolás komplexitását.
- Ugyanazon C generátor és paritás mátrixa között fennáll az alábbi összefüggés:

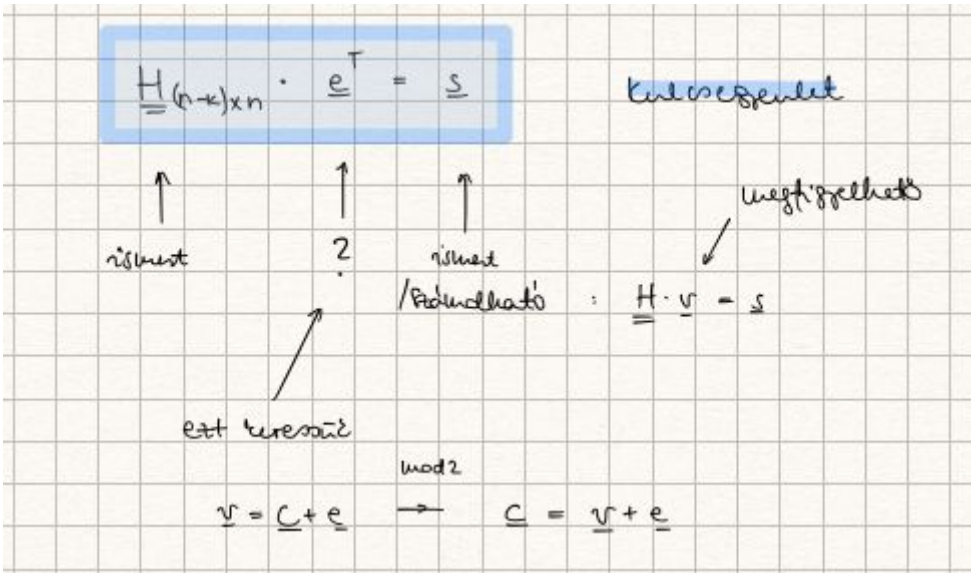


- Ezt a paritás ellenőrző mátrixot ugye úgy legóztuk össze, hogy ennek meg a vége az egységmátrix, az eleje pedig a B transzponáltja. (szisztematikus kódoknál)

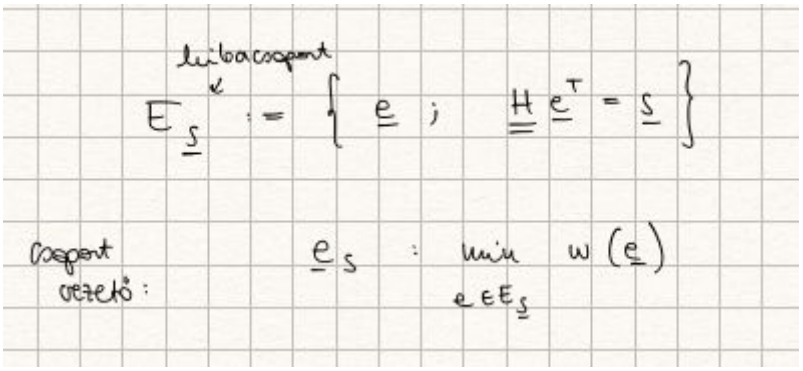
$$\boldsymbol{H} = \left[\boldsymbol{A}_{(n-k)\times k} | \boldsymbol{I}_{(n-k)\times(n-k)}\right], \text{ where } \boldsymbol{A} = \boldsymbol{B}^T$$

Szindróma vektor, hibajavítás

- Itt az az official definíció a könyvben, hogy s=eH^T
- Ha a kódszó c és a vett szó v, akkor e = v - c (hibavektor)



- Általában pont ezzel fogunk dekódolni: a vett szóból kiszámoljuk a szindrómát, amiből az e-t, amit kivonva a v-ből megkapjuk a kódszóra vonatkozó becslést.
- A trükk az, hogy erre az egyenletre több megoldás is van (az e-re), ezért vannak ez az úgynevezett hibacsoport: E_s (2^k eleme van egygébként)
- Ebből kivesszük a leggyakoribb /legkisebb súlyú e-t és azt használjuk. (ez a csoportvezető)



Extra definíciók

4.9. definíció. *Egy* **c** *vektor* ***súly****a a koordinátái között levő nem nulla elemek száma, jelölése* $w(\mathbf{c})$.

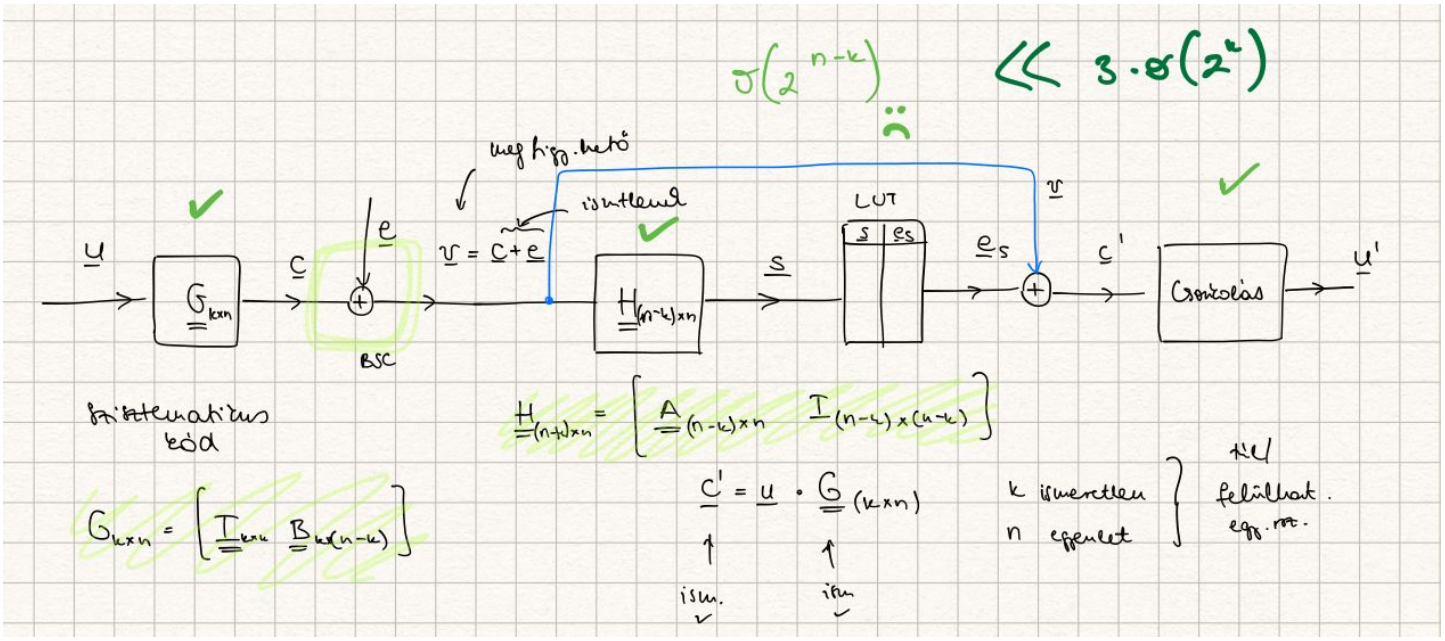
4.4. tétel. *Ha* C *lineáris kód, akkor a kódtávolsága megegyezik a minimális súlyával, azaz*

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

Algoritmikus komplexitás

- encoding, decoding ~ G-vel szorzás

Algorithmic complexity of detection (offline): $\mathcal{O}(2^{n-k})$.
Algorithmic complexity of encoding and decoding (online): $\mathcal{O}(n \cdot k)$.
Algorithmic complexity of the whole scheme: 1 LUT (detection) + 2 matrix-vector multiplication + truncation.



Példa feladat a tételhez

Pl: $C(5,2)$ $\quad \underline{\underline{G}} = \left[\begin{array}{cc:ccc} 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \end{array}\right] = [\underline{\underline{I}} : \underline{\underline{B}}]$

$$\underline{\underline{H}}_{3\times 5} = \left[\begin{array}{cc:ccc} 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{array}\right] = [\underline{\underline{B}}^T : \underline{\underline{I}}]$$

Tfh. $\underline{u} = (1,0) \rightarrow \underline{c} = \underline{u} \cdot \underline{\underline{G}} = (1\,0\,1\,1\,0)$

Tfh. $\underline{e} = (0\,0\,0\,0\,1) \rightarrow \underline{v} = \underline{c} + \underline{e} = (1\,0\,1\,1\,0) + (0\,0\,0\,0\,1)$

$$\underline{\underline{H}} \cdot \underline{v}^T = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \underline{s}$$

$\downarrow$

$\neq \underline{0}$ hibát jelez

$$E_{(001)} = \left\{ \underline{e} \mid \underline{\underline{H}} \cdot \underline{e}^T = (0\,0\,1)^T \right\} = \left\{ \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}; \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}; \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}; \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{pmatrix} \right\}$$

(2^k elem)

$\uparrow$ Csop. vezető

$\underline{e}_s$

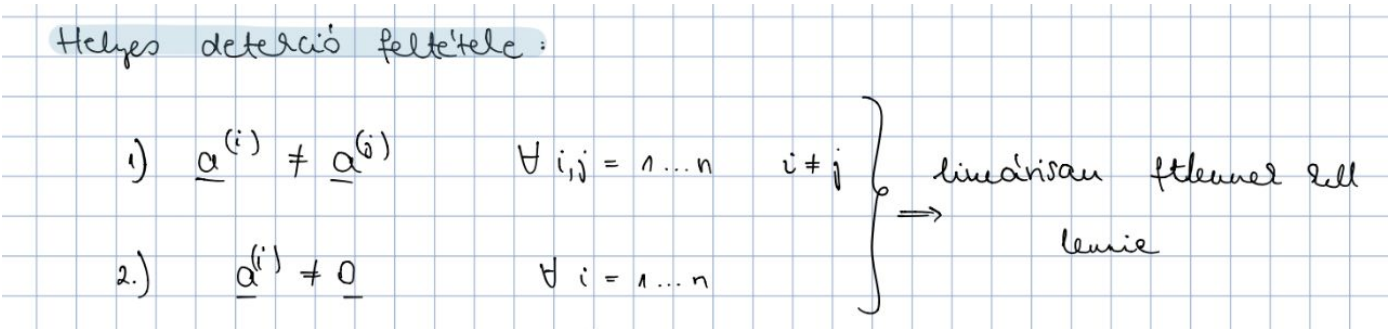
(*) itt vettük a ródszavakat

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

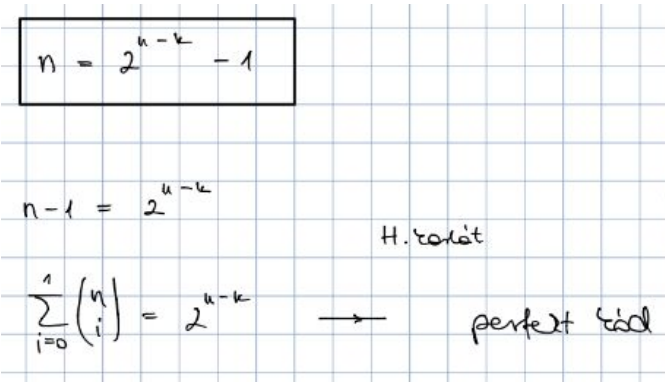
$\underline{c}' = \underline{v} + \underline{e}_s = (1\,0\,1\,1\,1) + (0\,0\,0\,0\,1) = (1\,0\,1\,1\,0)$ ✓

12. Give the construction of binary Hamming codes (define the corresponding matrices and the error correcting capability)

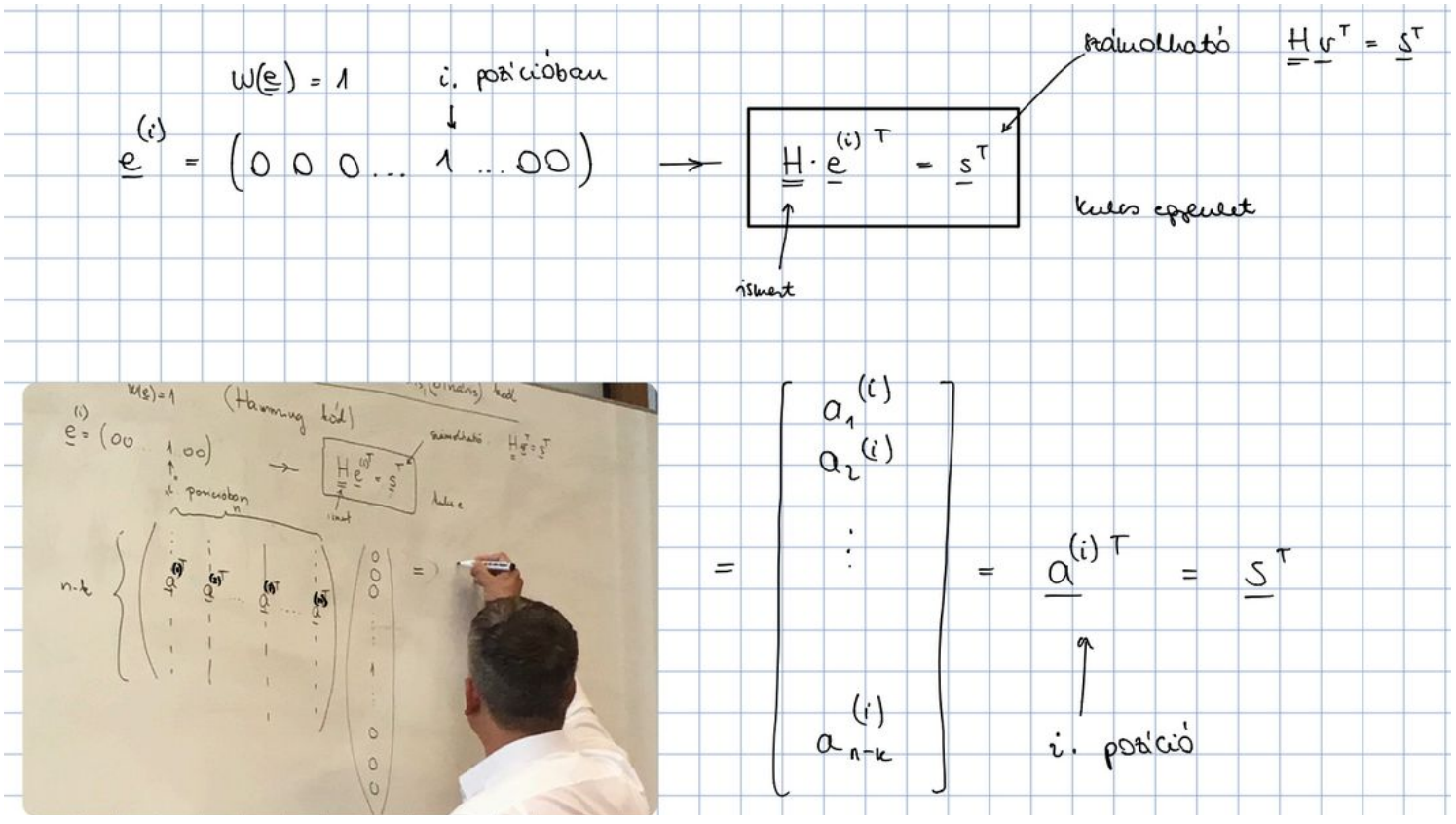
- A Hamming kód egy olyan bináris, lineáris kód, ami 2 hibát tud jelezni és 1-et javítani.
  - 1 hibát tud javítani: $1 \geq d(\mathbf{c}, \mathbf{v})$ esetén biztos meg tudjuk mondani v-ből c-t.
- Egy hiba detektálása esetén a szindróma vektor megtalálható a H paritásellenőrző mátrixban, az egyik oszloppal megegyezik.
  - oszlopai lineárisan függetlenek
  - $n = 2^{n-k}-1$ féle oszlopa lehet a H-nak

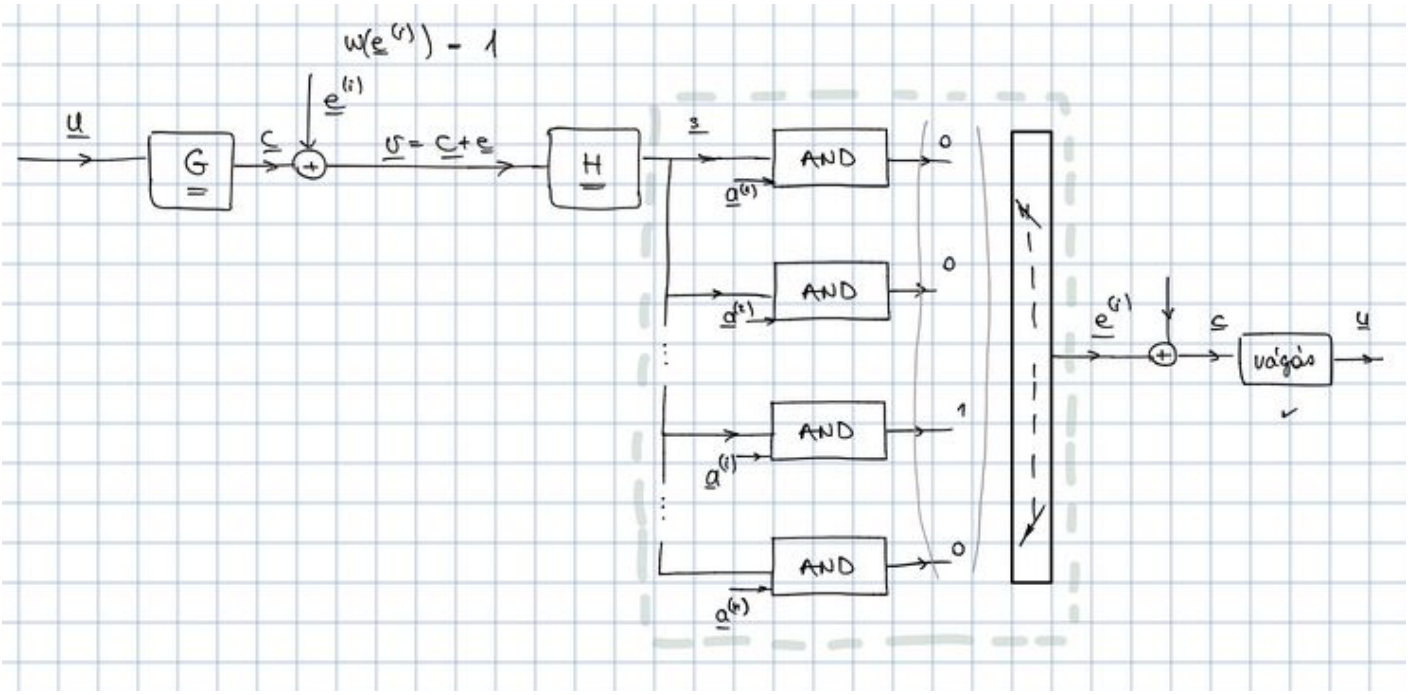


- A Hamming kódok perfekt kódok, mert teljesül a Hamming korlát.



Hamming kód felépítése





Paritásellenőrző mátrix:

Legyen a majdani kód paritásmátrixa:

$$\mathbf{H} = (\mathbf{a}_1^T, \mathbf{a}_2^T, \ldots, \mathbf{a}_n^T).$$

$$\underset{=}{H}_{(n-k)\times n} = \left(\underset{=}{A}_{(n-k)\times k} \mid \underset{=}{I}_{(n-k)\times(n-k)} \right) \qquad \underset{=}{A} = \underset{=}{B}^T \quad \text{(bináris)}$$

Generátor mátrix:

$$\underset{=}{G}_{k\times n} = \left(\underset{=}{I}_{k\times k} \mid \underset{=}{B}_{k\times(n-k)} \right)$$

Multiple error correction

Multiple error correction

Theorem. If a Hamming-code can correct every t error, $\boldsymbol{A}$ must have at least $2t$ independent column vectors.

Detection rule for $t = 2$: the syndrome vector matches with one of the columns (single error) or with one of the two-element-sums of the column vectors. This is highly inefficient to check.

13.Describe the Reed Solomon codes (generator matrix, parity check matrix, performance) + elég jó a sztori Rudi kidolgozásában

Mese

- Ahhoz, hogy t hibát (2 v több) ki tudjunk javítani, új dolgokat kell bevezetni: quáris kódok → GF(q), polinomok GF(q) felett, Shift Registerek, lineáris ciklikus kódok, ETA (mami kedvence)
- Mivel nagyobb domainbe lépünk, nagyobb az esély a hibázásra.

Bevezetés

- Az RS kódok a lineáris kódok egyik leggyakrabban használt fajtája
- **Galois Field GF(q)**-ban vagyunk, ahol

$$GF(q) = \{0, 1, 2, ..., q-1\}$$

  - q prím szám
  - Összeadás és szorzás (IKEA) műveletek vannak értelmezve **modulo q** artimetikával
  - Egy elem **rendje**:

$$ord(\alpha) = \min_{1 \leq k}\{\alpha^k = 1\}$$

  - minden GF-ben van legalább 1 primitív elem
  - Alfa GF(q) **primitív eleme**, ha $\alpha, \alpha^2, ..., \alpha^{q-1}$ expand the field.

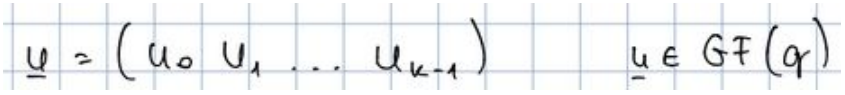
    AZAZ, $ord(\alpha) = q - 1$
  - GF(q) feletti **polinom és vektor reprezentáció**:

$$a(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2 + \cdots + a_n \cdot x^n$$

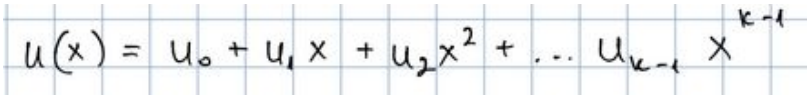
$$a(x) \leftrightarrow \boldsymbol{a} = (a_0, a_1, \ldots, a_n).$$

Reed Solomon kód konstrukció

- **C_{RS}(n,k)**, GF(q) felett, ahol **n = q - 1**
- az üzenetet vektoros reprezentációban írjuk fel:



- ehhez rendeljük az üzenetpolinomot:



- Alfák: GF(q) különböző elemei, ahol n <= q

$\alpha_0, \alpha_1, \dots \alpha_{n-1} \in GF(q)$

$\alpha_i \neq \alpha_j \qquad i,j = 0 \dots n-1$

$\alpha_i \neq 0 \qquad i \neq j$

$\boxed{n = q-1}$

legyenek $GF(q)$

- Az u üzenethez tartozó n hosszú c kódszavakat előállítjuk
  - felírjuk a vektoros és polinom reprezentációját:

$$\underline{c} = (c_0\ c_1 \dots c_{n-1})$$

$$c(x) = c_0 + c_1 x + c_2 x^2 + \dots c_{n-1} x^{n-1}$$

  - Alfát behelyettesítve és pl az n-1-re kifejtve így néz ki:

$$\begin{aligned} c_0 &= u(\alpha_0) \\ c_1 &= u(\alpha_1) \\ c_2 &= u(\alpha_2) \\ &\vdots \\ c_{n-1} &= u(\alpha_{n-1}). \end{aligned}$$

$$\vdots$$

$$c_{n-1} = u(x)\Big|_{x=\alpha_{n-1}} = u_0 + u_1 \alpha_{n-1} + u_2 \alpha_{n-1}^2 + \dots u_{k-1} \alpha_{n-1}^{k-1}$$

- Belátjuk, hogy a Reed Solomon kód lineáris és felírjuk a generátormátrixát

$$\underline{c} = \underline{u} \cdot \underline{\underline{G}}$$

Generátor mátrix: k x n

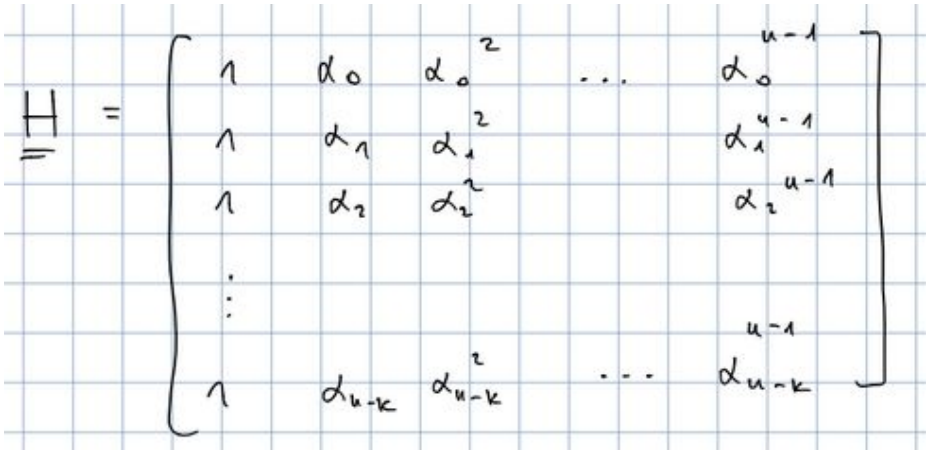
$$\underline{\underline{G}} = \begin{bmatrix} 1 & 1 & 1 & & 1 \\ \alpha_0 & \alpha_1 & \alpha_2 & & \alpha_{n-1} \\ \alpha_0^2 & \alpha_1^2 & \alpha_2^2 & \dots & \alpha_{n-1}^2 \\ \vdots & \vdots & \vdots & & \vdots \\ \alpha_0^{k-1} & \alpha_1^{k-1} & \alpha_2^{k-1} & & \alpha_{n-1}^{k-1} \end{bmatrix}$$

- Másik konstrukcióban: *Legyen α a GF(q) egy nem 0 eleme, melynek rendje m, $m \geq n$*
  És a fent említett konstrukcióban az alfákat módosítjuk a következő módon:

Ekkor a generátormátrix:

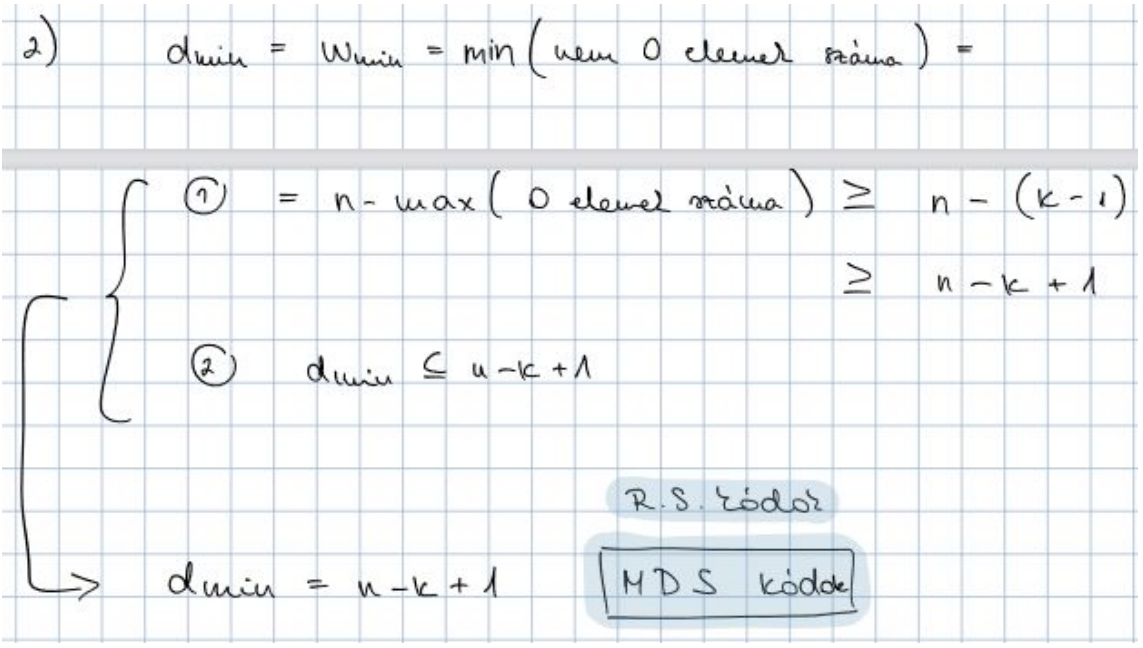
$$\mathbf{G} = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \alpha & \alpha^2 & \cdots & \alpha^{n-1} \\ 1 & \alpha^2 & \alpha^4 & \cdots & \alpha^{2(n-1)} \\ \vdots & & & \ddots & \vdots \\ 1 & \alpha^{k-1} & \alpha^{2(k-1)} & \cdots & \alpha^{(k-1)(n-1)} \end{pmatrix}$$

Paritásellenőrző mátrix: (n-k) x n



Kódtávolság (d_{min}), MDS bizonyítás

Proof. As $\deg(u(x)) = k-1 \Rightarrow u(x)$ has maximum $k-1$ roots. This means, that $\forall \boldsymbol{c}$ codeword has maximum $k-1$ nonzero components.



Tehát a Reed Solomon kód maximális távolságú

- Ebből következik, hogy n - k hibát tud jelezni
- t = $\left\lfloor \frac{n-k}{2} \right\rfloor$ hibát tud javítani

Performance

The only problem: complexity of the lookup table used: $\mathcal{O}(q^{n-k})$.

14. Describe the steps of the Error Trapping Algorithm for error detection in the case of cyclic codes

- Lineáris, ciklikus kódokat vizsgálunk

4.21. definíció. *Egy*

$$\mathbf{c} = (c_0, c_1, \ldots, c_{n-1})$$

vektor ciklikus eltoltja a

$$S\mathbf{c} = (c_{n-1}, c_0, \ldots, c_{n-2}).$$

S-et a ***ciklikus eltolás*** *operátorának nevezzük.*

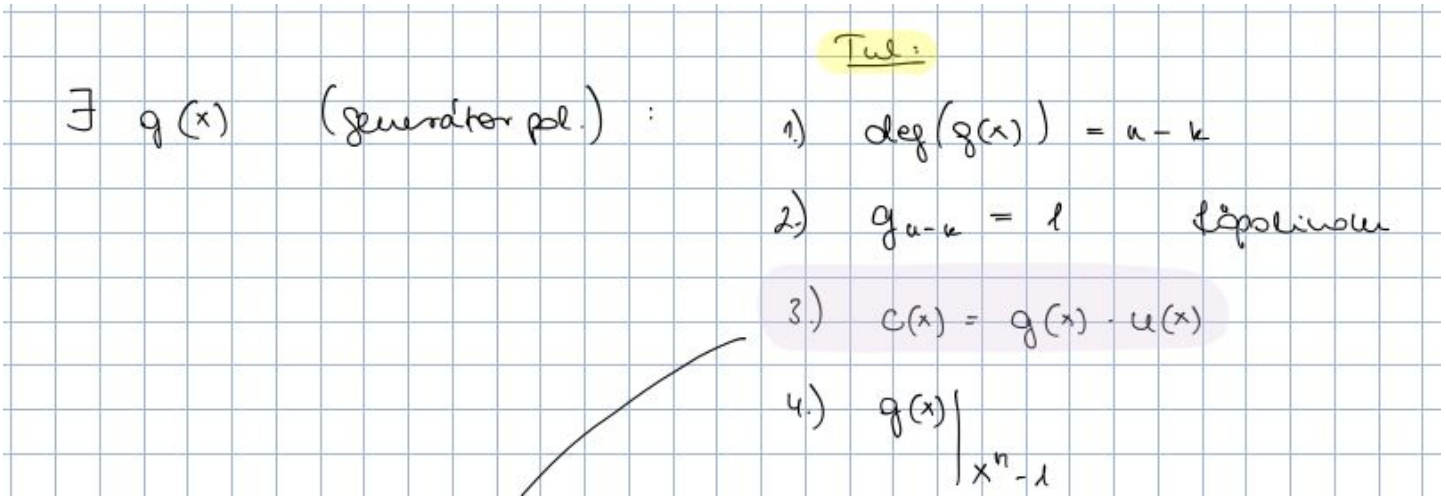
4.22. definíció. *A C kódot* ***ciklikus****nak nevezzük, ha bármely kódszó ciklikus eltoltja is kódszó.*

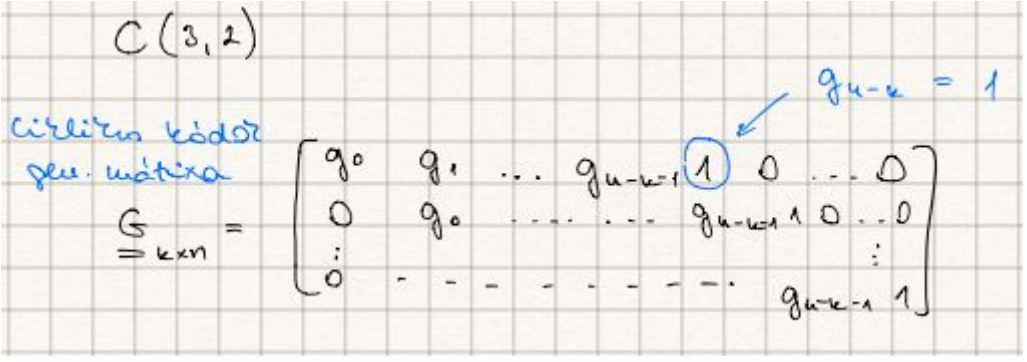
4.23. definíció. *Rendeljünk polinomot az egyes kódszavakhoz a következő módon:*

$$\mathbf{c} = (c_0, c_1, \ldots, c_{n-1}) \mapsto c(x) = c_0 + c_1 x + \cdots + c_{n-1} x^{n-1},$$

ekkor a $\mathbf{c}$ *kódszónak megfeleltetett* $c(x)$ *polinomot* ***kódszópolinom****nak vagy röviden kódpolinomnak nevezzük. A kódszópolinomok halmazát* $C(x)$*-szel jelöljük.*

- Itt ezeket a polinomokat általában GF(q)-ban nézzük ugye
- Lineáris ciklikus kódokra - ez a generátor polinom





Theorem 6: *If the generator polynomial g(x) of C has degree n-k then C is an [n,k]-cyclic code. If* $g(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_{n-k} x^{n-k}$ *then a generator matrix for C is the* $k \cdot n$ *matrix,*

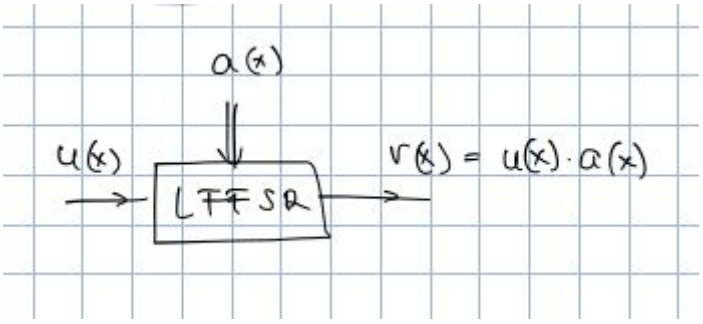
$$\begin{bmatrix} g(x) \\ x\,g(x) \\ x^2 g(x) \\ \vdots \\ x^{k-1} g(x) \end{bmatrix} = \begin{bmatrix} a_0 & a_1 & a_2 & \cdots & a_{n-k} & 0 & 0 & \cdots & 0 \\ 0 & a_0 & a_1 & \cdots & a_{n-k-1} & a_{n-k} & 0 & \cdots & 0 \\ 0 & 0 & a_0 & \cdots & a_{n-k-2} & a_{n-k-1} & a_{n-k} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & a_{n-k} \end{bmatrix}.$$

Előismeretek

- Úgy általában arra jók ezek a faszom shift regiszterek, hogy csökkentik a komplexitást, ha jól értem.

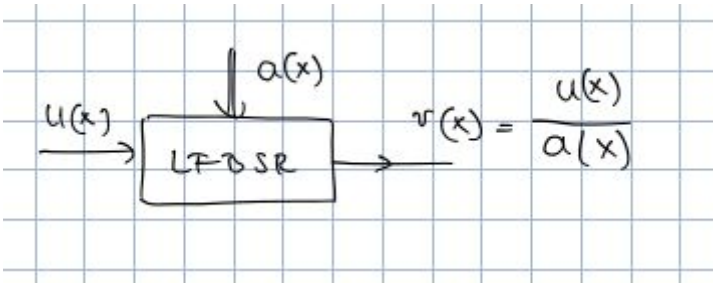
LFFSR

- Linear Feedforward Shift Register
- Arra jó, hogy polinomokat szorozzunk össze



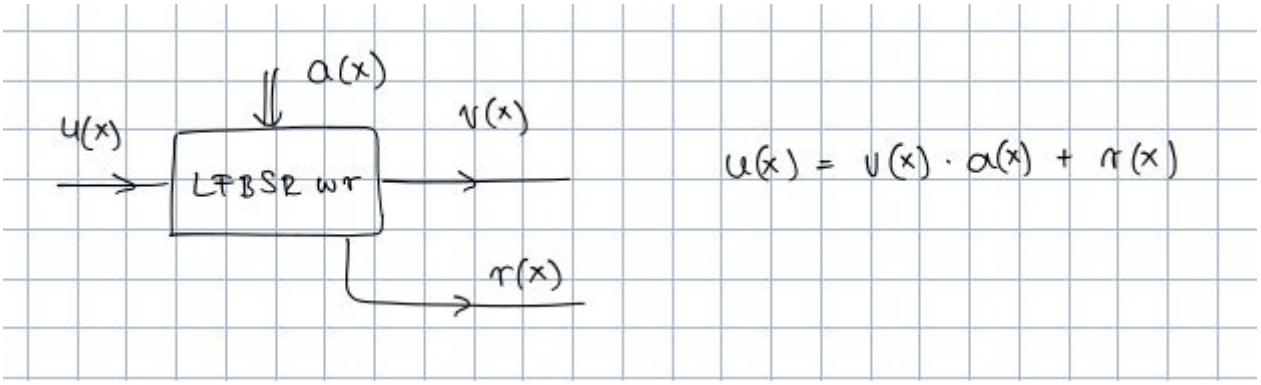
LFBSR

- Linear Feedback Shift Register
- Arra jó, hogy polinomokat osszunk vele (maradékmentesen)

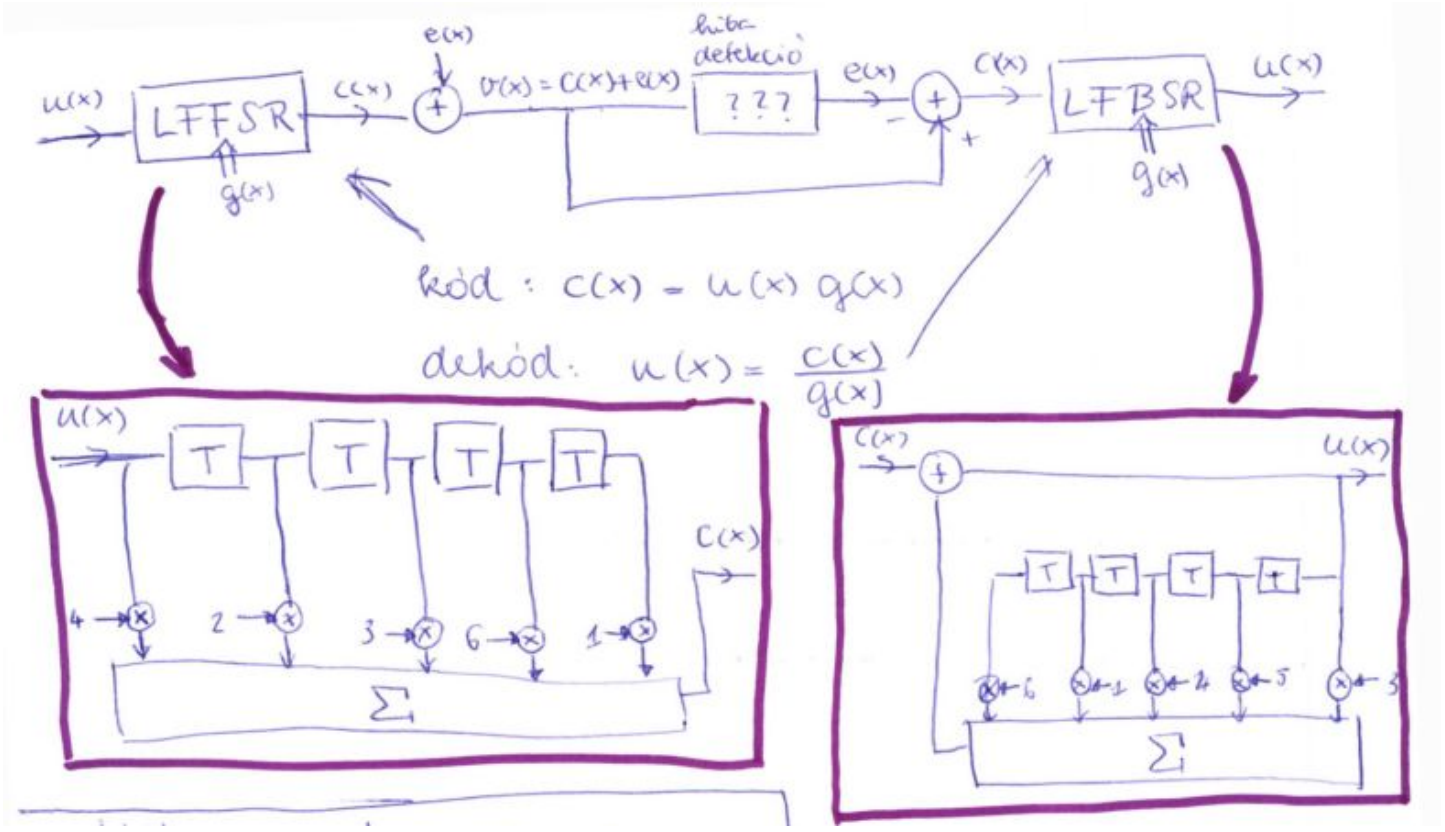


LFBSRwr

- LFBSR maradékkal



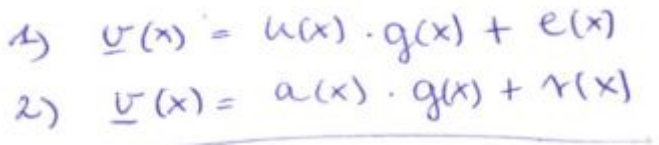
Így nézki ez a kódolás most hogy tudjuk mi a generátor polinom meg a shift regiszterek:



Error Trapping Algorithm

http://www-math.ucdenver.edu/~wcherowi/courses/m7823/cyclicII.pdf szevasz mami

- Az ETA egy módszer, amivel azonosítani tudunk ciklikus kódhibákat, anélkül, hogy LUT-okat használnánk.
- Csak burst errorokat képes felismerni.
    - Eddig minden olyan hibákat javított, amíg random oszlottak el a kódban a burst error lényege, hogy csomósodva van.
- Alap ötlet:



- Tehát ha a megkapott v(x)-et osztjuk g(x)-szel akkor e(x)-et megkapjuk osztási maradékként. Ebből következik, hogy

$$deg(e(x)) \leq n - k - 1.$$

- Mivel azt feltételezzük, hogy burst error, azért azt mondjuk hogy az első és utolsó hibás biz közötti táv max. n-k.
- Na most ebből valahogy ez következik: (i_0 az első hiba pozija)

$$e(x) = x^{i_0} r(x)$$

- És akkor itt az egyenletből ez jövöget kifelé:

$$v(x) \mod g(x) = e(x) \mod g(x)$$

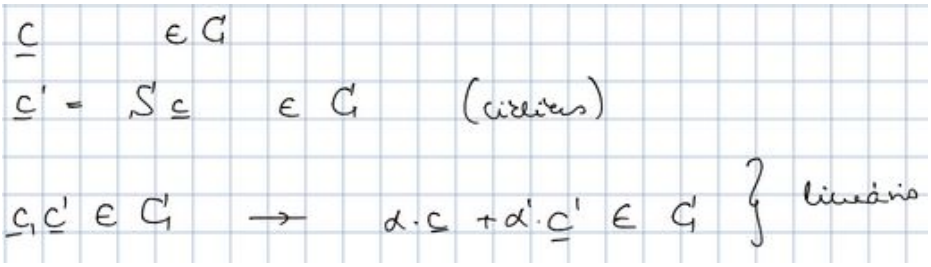
$$x^{-i} v(x) = a(x) g(x) + r(x)$$
$$x^{-i} e(x) = b(x) g(x) + r(x)$$

- NA ok ezt majd megérti jobban akinek 6 anyja van. :D

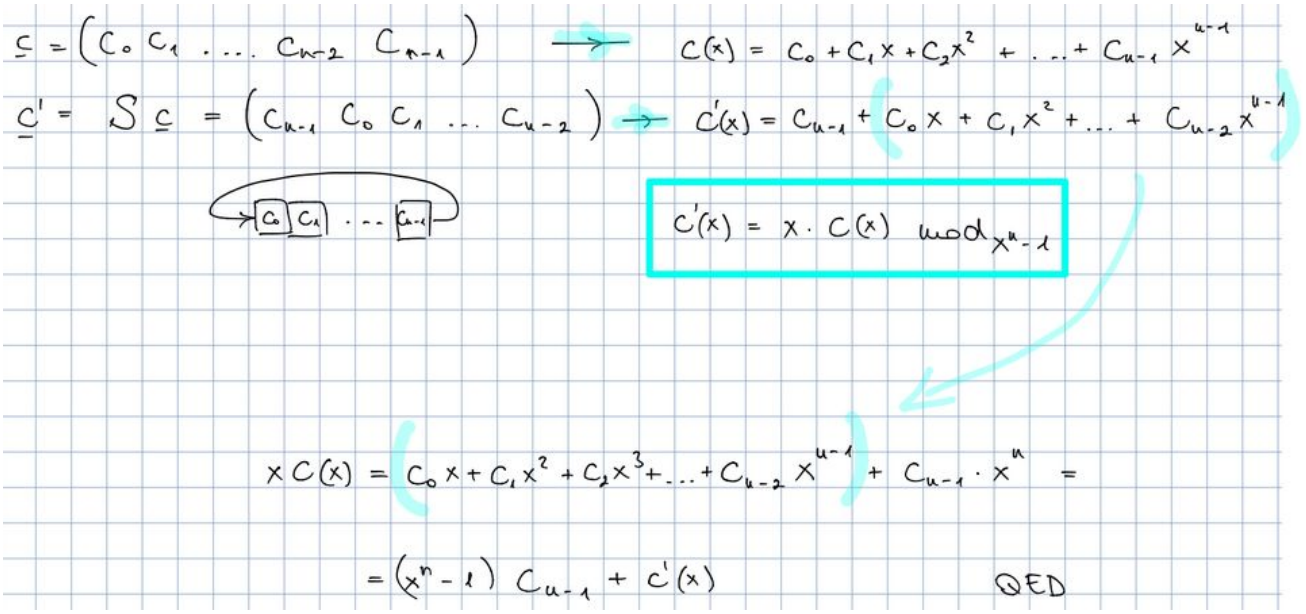
15. Describe the cyclic RS codes (generator polynom, parity check polynom, implementation)

Ciklikus kódok

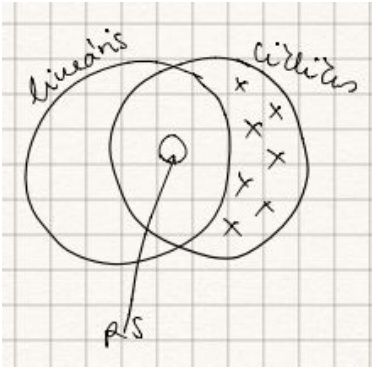
- C kódot ciklikusnak nevezzük, ha bármely kódszó ciklikus eltoltja is kódszó



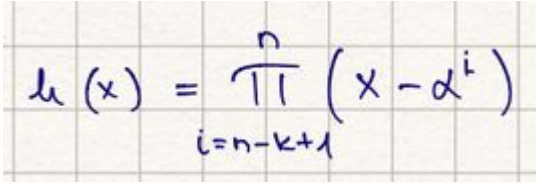
- Ciklikus eltoltja a c kódnak és c(x) polinomnak:

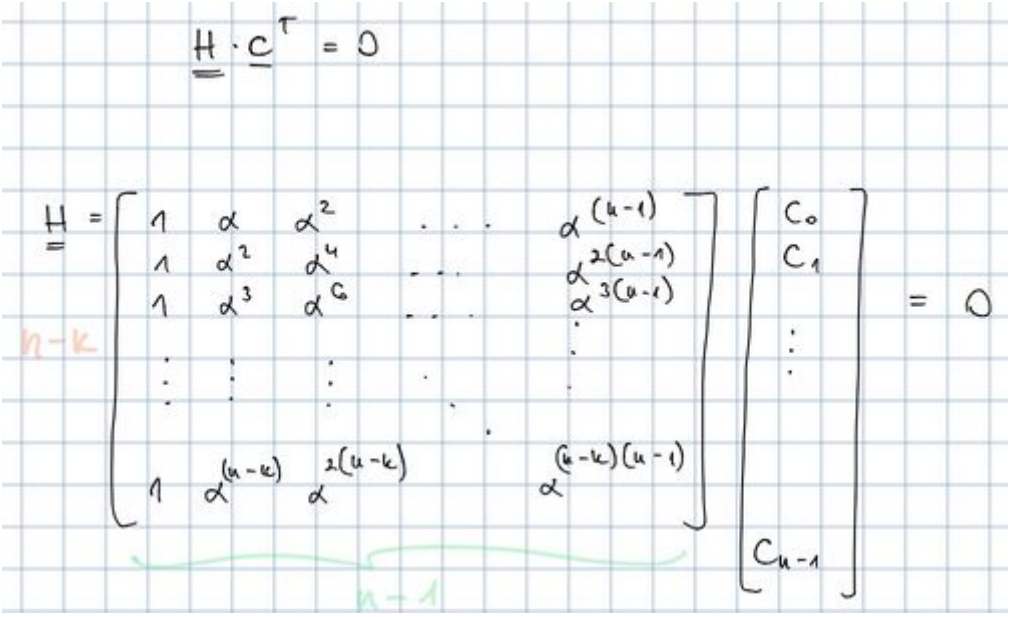


- Az RS kód MDS (optimális) és ciklikus (valós idejű implementáció)

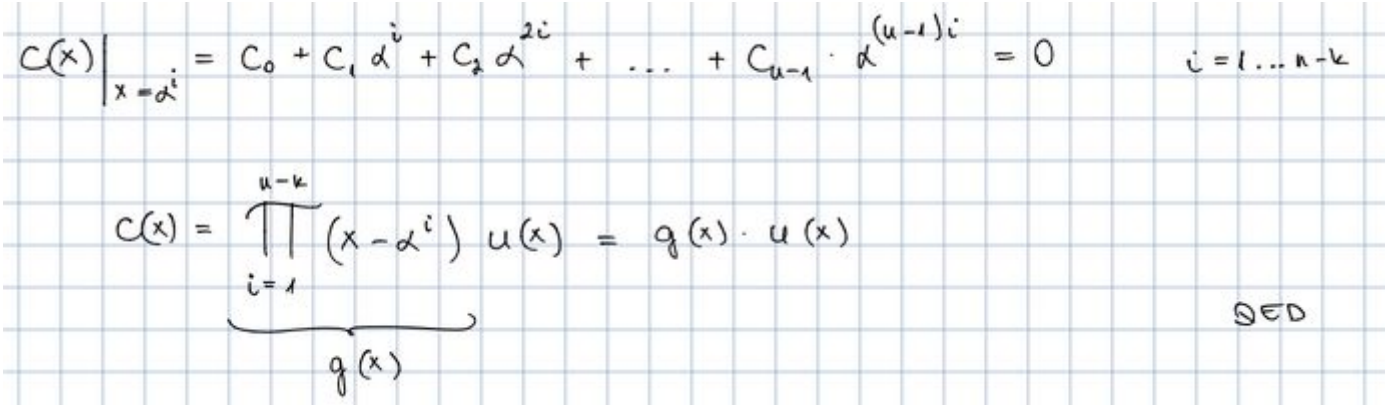


Paritásellenőrző mátrix és polinom





- A kódpolinomba behelyettesítjük az alfákat + felírjuk a generátorpolinomot



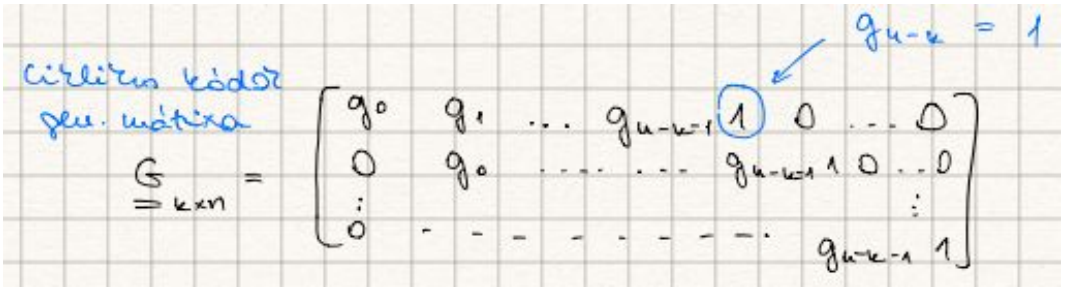
alakban. A $g(x)$ tételbeli definíciójával az tényleg egy ciklikus kód generátorpolinomja, ha megmutatjuk, hogy osztja az $(x^n - 1)$-et. Ez utóbbi azért igaz, mert $g(x)$ gyökei az $(x^n - 1)$-nek is gyökei, ugyanis

$$(\alpha^i)^n = (\alpha^i)^m = \alpha^{im} = (\alpha^m)^i = 1,$$

ezért

$$x^n - 1 = \prod_{i=1}^{n} (x - \alpha^i),$$

Generátormátrix és generátorpolinom



- g(x) és tulajdonságai:

$$g(x) = \prod_{i=1}^{n-k} (x - \alpha^i)$$

Tul:

1) $\deg(g(x)) = n - k$
2) $g_{n-k} = 1$ főpolinom
3) $c(x) = g(x) \cdot u(x)$
4) $g(x) \mid x^n - 1$

Összegezve

T) $\forall$ GF(q)-nak van legalább 1 primitív eleme

$$g(x) = \prod_{i=1}^{n-k} (x - \alpha^i)$$

$$h(x) = \prod_{i=n-k+1}^{n} (x - \alpha^i)$$

GF(2^m)

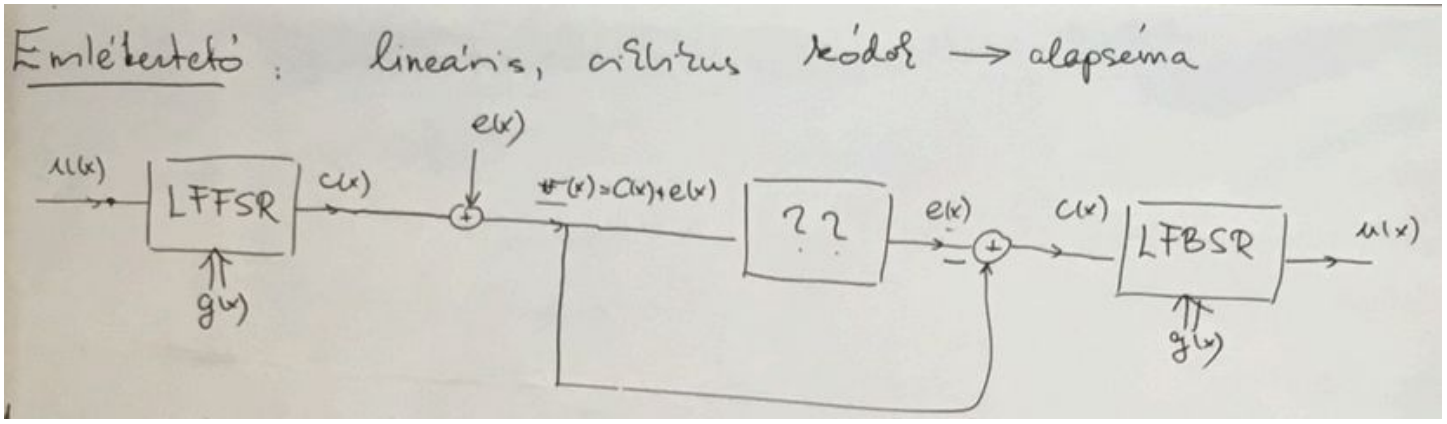
$$g(x) = \prod_{i=1}^{n-k} (x - y^i)$$

$$h(x) = \prod_{i=n-k+1}^{n} (x - y^i)$$

M) $h(x) = \dfrac{x^n - 1}{g(x)}$

$g(x) \mid x^n - 1$

Implementation



Now we arrived to a very special place. We have an optimal (MDS) code with the optimal (real-time) realization. However, we cannot be exactly happy, because the complexity of the shift registers are so high cause of the q-ary domain, it cannot be implemented on the current families of VLSI chips.

In order to make it implementable, we should introduce the fundamentals of minimal polynoms over $GF(q)$, and the Bose-Chaudhuri-Hocquenghem codes, which are not MDS, but close to optimal and actually easily implementable, because every multiplication inside of a shift register is implemented with either a wire or nothing, because the polynomials in this code has 1 or 0 as coefficients.

Implementation in binary domain

Definition. A polynom over $GF(q)$ is irreducible, if it cannot be written as a product of two lower-degree polynoms.

Theorem. Let $p(y)$ be an irreducible polynom with degree m in $GF(p)$, p prime number. Then, any element q of the field $GF(p^m)$ can be uniquely mapped to a polynom $q(y)$ over $GF(p)$, with the following operation:

$$\begin{aligned} \alpha, \beta \in GF(p^m) &\longleftrightarrow a(y), b(y) \in \mathcal{P}(GF(p)) \\ \gamma = \alpha + \beta &\longleftrightarrow c(y) = a(y) + b(y) \mod p(y) \\ \gamma = \alpha \cdot \beta &\longleftrightarrow c(y) = a(y) \cdot b(y) \mod p(y) \end{aligned}$$

Theorem. In this field, the first-order polynom y is be a primitive element.

Definition. Standard form of a polynom in $GF(p^m)$:

$$\begin{aligned} \alpha(x) &= \alpha_0 + \alpha_1 \cdot x + \cdots + \alpha_n \cdot x^n \\ \alpha(x) &= a_0(y) + a_1(y) \cdot x + \cdots + a_n(y) \cdot x^n \\ \alpha(x) &= y^{i_0} \cdot x + y^{i_1} \cdot x + \cdots + y^{i_n} \cdot x^n \end{aligned}$$

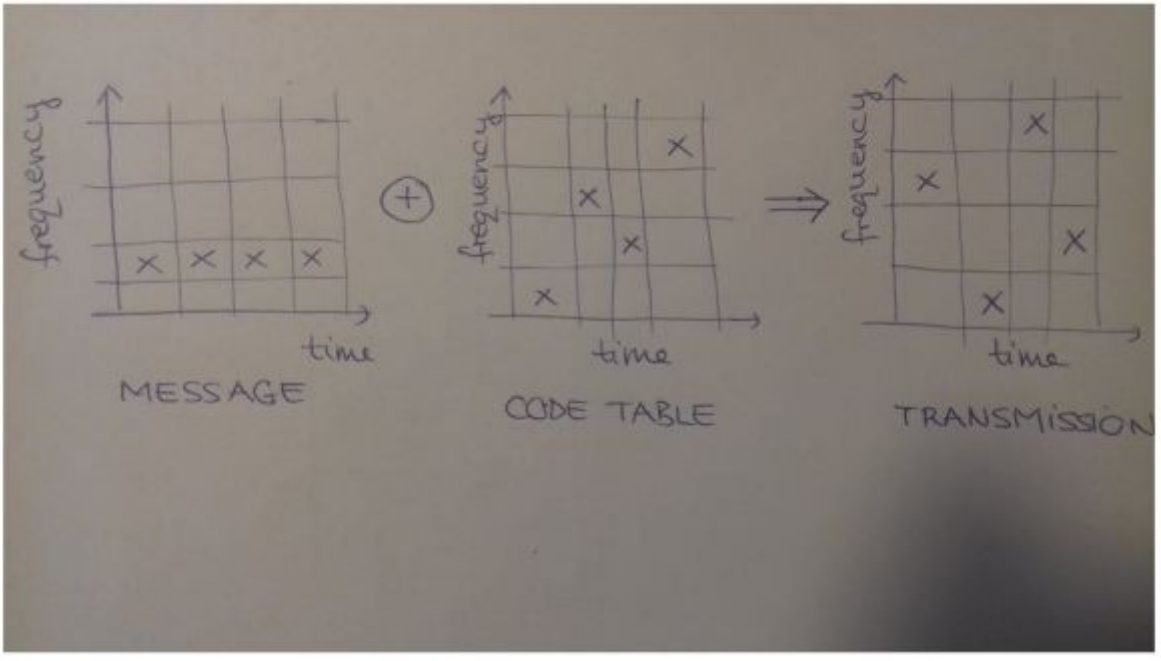
Construction of optimal binary Reed-Solomon codes: $t \to (n, k); p = 2, q = 2^m, n = 2^{m-1}$

16. Describe the CDMA/FH system and the CDMA/DS system with Walsh-Hadamard codes and with random codes

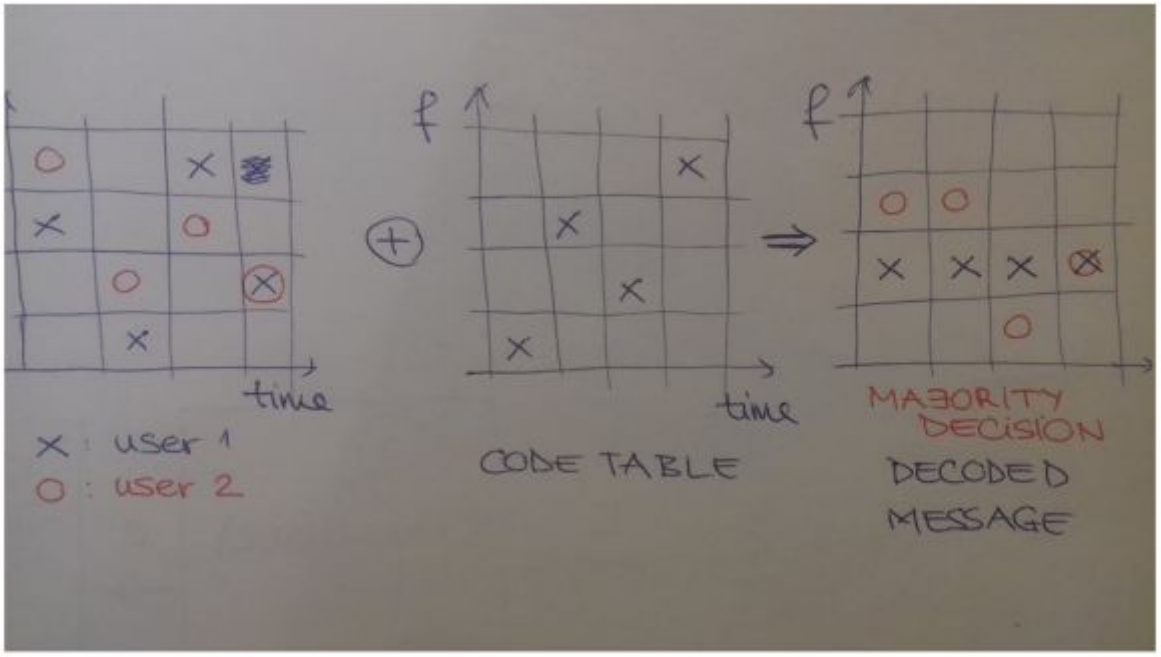
- Probléma: Vannak felhasználók, akik tetszőleges időben szeretnének adni infót egy közös csatornán. Nincsen közöttük semmilyen koordináció és a teljes sávszélességet szeretnének használni. A jeleik ütközhetnek egymással és ebből lehet nagy galiba.
- Általában felosztjuk az időintervallumokat (TDMA) vagy frenkceiosztást (FDMA) alkalmaznak és ortogonális részcsatornákra bontják a közös csatornát.
- Ötlet: egyes felhasználóknak saját kódot adunk, amelyekkel megkülönböztetjük őket egymástól. Nyilvánvaló, hogy a kódokat nem lehet akárhogy megválasztani. A kódokra ültetett üzenetek, amelyek az egyes felhasználóktól érkeznek ütköznek a csatornában (pl. összeadódnak) így érkeznek majd a vevőhöz.
- CDMA: Code Division Multiple Access

Frequency Hopping (FH)

- Itt ugye a lényeg hogy valami meghatározott random frekvencia sorozat alapján folyamatosan váltogatjuk a frekvenciát amin adunk, amit mi is meg a vevő is tudunk.
- fasza benne, hogy nem is nagyon lehet minket lehallgatni sem ilyenkor mert senki sem tudja rajtunk kívül hol jön az üzi. különböző frekiken figyelve kb csak zaj van.
- GSM pl ilyet használ, amikor hívás van kioszt egy ilyen sorozatot.
- Kód táblákat használ
- https://www.youtube.com/watch?v=CkhA7s5GIGc
- Kódolás:



- Dekódolás:



Direct Sequence

- Na ez egy másik módszer, alternatíva az FH helyett
- modulációs séma: az üzeneteket jellel moduláljuk.
- Ez a jel a signature signal: pszeudo random, rövid rádió pulzusokból áll. Egy ilyen pulzus hossza chiptime. Minél kisebb annál nagyobb lesz a sávszélesség.
- az üznetet szétdarabolja így nagyobb sávszélességet kapunk.
- ebben amúgy a csávó elmondja mint egy kisangyal:

https://www.youtube.com/watch?v=-1mxYWvfVWQ

Ezt leírtuk kicsit matekosabban is amúgy:

- Userek: i= 1,... M
- Üzenet szimbólum of user i:

$$y_i \in \{-1, 1\}$$

- Kódszó of user i: (T_sample, T_chip)

$$c_i \in \{-1, 1\}^N \qquad N = \frac{T_s}{T_c}$$

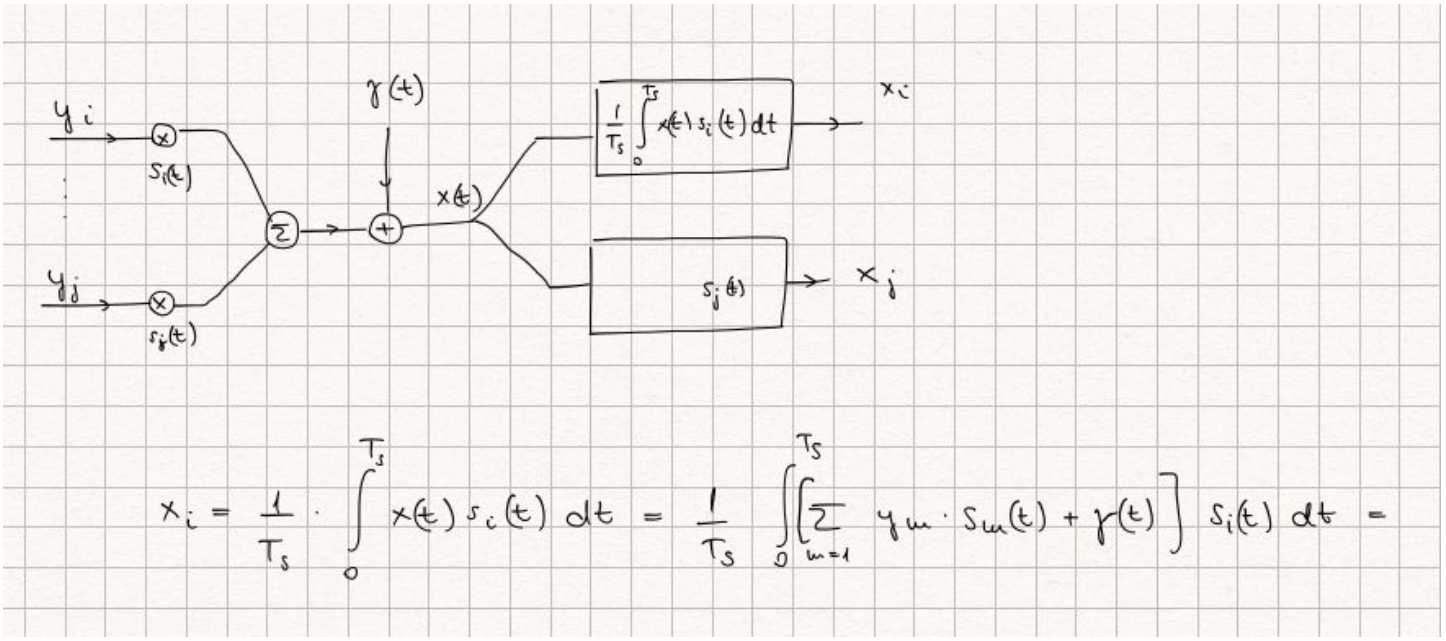
- Signature signal: (négyzetes jel)

$$s_i(t) = c_k^{(i)} \text{ for } t \in [(k-1)T_{chip}, kT_{chip}].$$

- Multiplexelt jel:

$$x(t) = \sum_{j=1}^{M} y_j \cdot s_j(t) + \nu(t)$$

- Itt aztán van ilyen nagy félelmetes ábra, de nem értem:



$$x_i = \frac{1}{T_s}\cdot\int_0^{T_s} x(t)\,s_i(t)\,dt = \frac{1}{T_s}\int_0^{T_s}\left[\sum_{m=1} y_m\cdot s_m(t) + \gamma(t)\right] s_i(t)\,dt =$$

- A lényeg, hogy észrevesszük hogy ebből az egyenletből kipottyan egy olyan tag, amit a Multi User interference okoz. Ez az amire jók az ortogonális kódok.

Walsh-Hadamard

- lehetővé teszi az $M=2^k$ ortogonális kódok konstrukcióját.
- Truly orthogonal codes: Two codes are said to be orthogonal if when they are multiplied together the result is added over a period of time they sum to zero. For example a codes 1 -1 -1 1 and 1 -1 1 -1 when multiplied together give 1 1 -1 -1 which gives the sum zero.
- Rekurzívan képezzük ezt a Walsh cuccost így:

$$\boldsymbol{C}(0) = 1.\quad \boldsymbol{C}(k+1) = \left[\begin{array}{c|c} \boldsymbol{C}(k) & \boldsymbol{C}(k) \\ \hline \boldsymbol{C}(k) & -\boldsymbol{C}(k) \end{array}\right]$$

- Ezek a kódok ortogonálisak lesznek, szóval nem lesz MUI.
- Itt is van még sok matek de nem értem igazából szóval mindegy szerintem.

Random kódos CDMA/DS

- Ha használjuk azt a Walshot az igaz, hogy nem nagyon lesz interferencia, de elég kevés user használható a high-speed data transmissionben.
- Hogy ezt kiküszöböljük random generált KVÁZI ortogonális kódokat használunk, ami követik a Bernoulli eloszlást, tehát:

$$P\left(c_m^{(i)} = 1\right) = P\left(c_m^{(i)} = -1\right) = 0{,}5$$

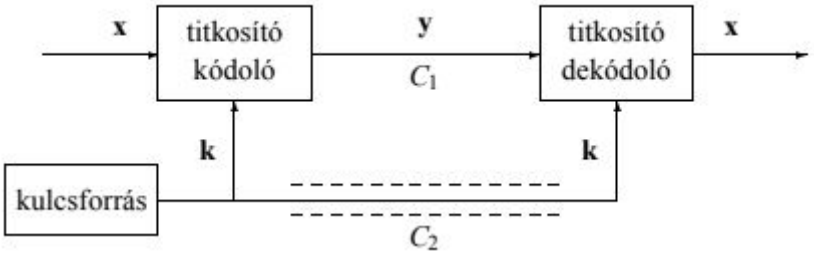
- Ez nem fogja sajnos kiküszöbölni a MUI-t, tehát szarabb lesz a bit error probability.
- Itt is van egy akkora képlet, hogy befosol, (a P_b-re) azt a képletet kell majd minimalizálni, ha optimális kódot szeretnénk.

17. Describe the OTP method and RSA algorithm for cryptography

OTP (one time pad / véletlen átkulcsolás / Vernam-féle titkosító)

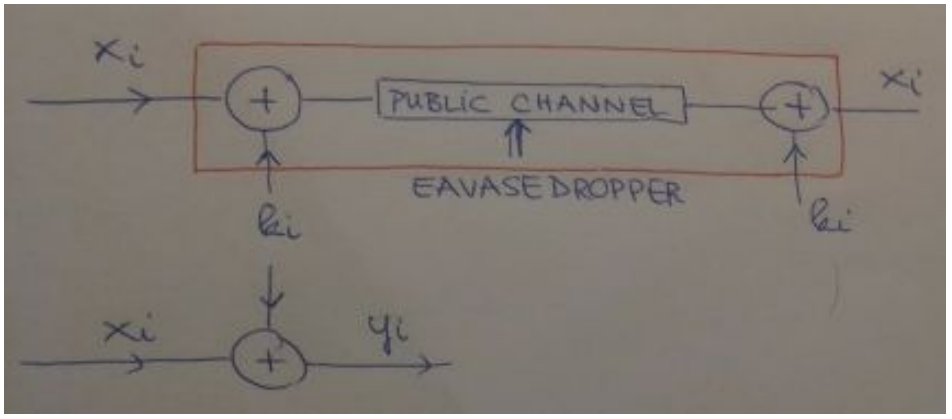
- a kulcs hossza megegyezik a kódolandó szövegével (x)
- k kulcs minden esetben automatikusan generálódik = véletlenszerűen állítódnak elő a kulcsot alkotó betűk

Konvencionális titkosító rendszer = **rejtett kulcsú** = egykulcsos:



Működése

- rejtett üzenetet a C_1 nyilvános csatornán küldjük el (pl: közhasználatú hírközlési csatorna)
- a kulcsot a C_2 titkos csatornán továbbítjuk (pl: megbízható futár kézbesít)
- ergo a kódolás és a dekódolás azonos kulcsot használ



As seen in the figure, for an eavesdropper without access to the one-time-pad, this type of encryption can be seen as a binary simmetric channel with a bit error probability equal to the probability of a bit being 1 in the key. If the key is truly random, $P(k_i = 0) = P(k_i = 1) = 0.5$.

Now, it can be easily concluded, that for a bit error probability $P_b = 0.5$, the conditional entropy of the channel is equal to $\mathrm{H}(P_b) = P_b \cdot \mathrm{ld}\,\frac{1}{P_b} + (1 - P_b) \cdot \mathrm{ld}\,\frac{1}{1-P_b} = 1$, resulting in $C = 0$ channel capacity. In another words, the mutual informaiton of the transmitted and the eavasedropped symbols is $I(X, Y) = 0$.

In conclusion, if the key is never reused, the OTP is impossible to be broken.

There are several common operations that can satisfy this property. For example, we can use modular addition for encryption, and modular subtraction for decryption, or vice versa. Or we could use e.g. bitwise XOR, which is its own inverse, and can thus be used for both encryption and decryption in the same system. Or, if we wanted, we could use e.g. multiplication in a Galois field for encryption, and multiplication by the group inverse for decryption (or, again, vice versa).

Támadás (általános, nem csak a one time padre vonatkozik):

- passzív
  - lehallgatás: a nyilvános csatornán áramló üzenetek birtokába jut

        - célja, hogy megtalálja a kapcsolatot a kinyert infók alapján és ebből algoritmikus támadást indítson = rejtjelfejtés
- aktív
    - üzenetmódosítás: kicseréli a rejtett üzeneteket
    - megszemélyesítés

Előnye
- feltörhetetlen, mert random a kulcs és csak egyszer használjuk fel (lásd “one time”)
- optimális

Hátránya
- a kulcs ugyanolyan hosszú, mint az üzenet
- csak akkor biztonságos, ha minden üzenetnek saját kulcsa van

Alkalmazása
- military, diplomatic matters, “red phone” between DC and Moscow

RSA

- Nyilvános kulcsú rejtjelezés
    - Minden felhasználónak két kulcsa van: egy publikus és egy privát
    - Publikussal titkosítjuk a nekik menő üzit priváttal dekódoljuk a nekünk jövőt
- Biztonságos adatátvitel és digitális aláírásokhoz használt.

Matematikai háttér
- Euklideszi algoritmus
    - Két szám LNKO-ját tudjuk vele kiszámolni
    - Tehát fogod b-t és elosztod c-vel maradékosan. Aztán c-t osztod az előző maradékkal maradékosan. Stb. Az utolsó nem nulla maradék b és c LNKO-ja.

5.4. tétel (Az euklidészi algoritmus). *Adott b és $c > 0$ egészekre egymás után többször alkalmazzuk a maradékos osztást, s ezzel az egyenletek következő sorozatát kapjuk:*

$$\begin{aligned} b &= cq_1 + r_1 & 0 < r_1 < c \\ c &= r_1 q_2 + r_2 & 0 < r_2 < r_1 \\ r_1 &= r_2 q_3 + r_3 & 0 < r_3 < r_2 \\ &\vdots \\ r_{n-2} &= r_{n-1} q_n + r_n & 0 < r_n < r_{n-1} \\ r_{n-1} &= r_n q_{n+1} + 0 & 0 = r_{n+1} \end{aligned}$$

A b és c számok legnagyobb közös osztója r_n, az osztási eljárás legutolsó nem nulla maradéka, azaz $(b, c) = r_n$.

- Kis Fermat
    - Ha van egy p prímünk, és egy a számunk, ami nem osztható p-vel, akkor

$$a^{p-1} \equiv 1 \pmod{p}.$$

- Fermat tétel általánosítása
    - Ha van p_1 és p_2 prímünk és a, p_1, p_2 legnagyobb közös osztója 1, akkor:

$$a^{(p_1-1)(p_2-1)} = 1 \pmod{p_1 p_2}.$$

- Euler tétel speciális esete

5.8. tétel. *Ha* $m = p_1 p_2 \cdots p_n$, *ahol* $p_1, p_2, \ldots, p_n$ *különböző prímek és* $(a, m) = 1$, *akkor*

$$a^{\phi(m)} = 1 \pmod{m}, \tag{5.14}$$

ahol $\phi(m) = (p_1 - 1)(p_2 - 1) \cdots (p_n - 1)$ *az Euler-függvény. (Ez a tétel az Euler-tétel speciális esete.)*

- Egyéb modulos dolgok:

5.8. definíció. *Azt mondjuk, hogy b modulo m inverze a-nak ha*

$$ab = 1 \pmod{m}.$$

5.5. tétel. *Az a modulo m inverze akkor és csak akkor létezik, ha* $(a, m) = 1$. *Ha létezik inverz, akkor az egyértelmű az m-nél kisebb pozitív egészek között.*

Az RSA algoritmus lépései

1. Válasszunk két random nagy prímet: p_1, p_2
2. Számoljuk ki az alábbi paramétereket

$$m = p_1 p_2 \qquad \phi(m) = (p_1 - 1)(p_2 - 1)$$

3. Válasszunk egy véletlenszerű e-t, úgy hogy

$$1 \leq e < \phi(m) \qquad (\phi(m), e) = 1$$

4. Számoljuk ki e inverzét modulo fi(m)

$$d = e^{-1} \pmod{\phi(m)}$$

5. Az m, e egészeket nyilvánosságra hozzuk (nyilvános kulcs), és a d, p_1, p_2 értékeit titokban tartjuk (privát kulcs). Tehát

$$\mathbf{k}^p = (m, e) \quad \mathbf{k}^s = (d, p_1, p_2)$$

6. A titkosító kódolás az 1 <= x < m nyílt üzenetre egy 1 <= y < m rejtett üzenetet ad, ahol
   a. Kódolás

$$y = x^e \pmod{m},$$

   b. Dekódolás

$$x = y^d \pmod{m}$$

- Ugye érdemes megjegyezni, hogy az egész azon a problémán alapul, hogy feltételezzük, hogy ez egy nehéz probléma. Sejtések alapján az m prímtényezőkre bontásának nehézségével egyezik meg. Most jelenleg úgy tudjuk hogy ez lehetetlen belátható időn belül megoldani, ha elég nagy a két prím, de simán lehet hogy valami nagyon okos ember megoldja ezt a problémát.