Kapitola 1

Vektorové priestory v informatike

Teoretické vlastnosti a pojmy, ktoré sme doteraz naštudovali v teórii vektorových priestorov nám pomôžu pochopiť myšlienky, ktoré sa využívajú pri technických aplikáciách ako sú napríklad kompresia, kódovanie a šifrovanie. Na zjednodušených príkladoch uvedieme myšlienky a postupy, ktoré sa skutočne v praxi používajú.

1.1 Kompresia

Kompresia dát znamená takú ich transformáciu, ktorá zabezpečí zmenšenie objemu dát, pri ktorom sa podarí zachovať podstatná informácia obsiahnutá v pôvodných dátach.
Použijeme princíp priemetu vektora do vektorového podpriestoru s menšou dimenziou. Konkrétny postup spočíva v transformácii nameraných hodnôt do koeficientov nejakej vopred dohodnutej bázy. Báza je zvolená tak, aby čo najlepšie vyjadrovala všetky možné procesy, ktoré do nej budeme premietať. Po premietnutí do vhodnej bázy ostanú zachované podstatné informácie, ktoré pomôžu pôvodný proces dobre odhadnúť. Pôvodný proces už z takto komprimovaného procesu nedokážeme presne zrekonštruovať. Preto už pred kompresiou nastavíme parametre kompresie (=dimenziu podpriestoru do ktorého budeme proces premietať) tak, aby vyhovovala náročnosti zákazníka.

Úloha 1.1.1 *Kompresia zvuku je napríklad transformácia procesu (zvukového súboru) z formátu* `wav` *do formátu* `mp3`. *Aká báza sa používa pre kompresiu zvuku?*

Úloha 1.1.2 *Kompresia obrazu je napríklad transformácia bitmapy do formátu* `jpg`. *Aká báza sa používa pre kompresiu obrazu?*

Úloha 1.1.3 *Zistite, aká kompresia je používaná pre video súbory. Dá sa aj táto kompresia vyjadriť ako priemet do podpriestoru?*

1.2 Kódovanie

Kódovanie dát znamená takú ich transformáciu, ktorá zabezpečí detekciu alebo opravenie chýb pri prenose sieťou. K prenášaným dátam je pripojená informácia navyše. Jedná sa o opačný proces ako pri kompresii, teda transformáciu procesu do nadpriestoru. Kódovanie môžeme deliť podľa toho, či chybu chceme iba detekovať, alebo aj zistiť na ktorom mieste sa vyskytla a opraviť ju. Na základe takéhoto delenia rozlišujeme kódy detekčné a samoopravné.

Detekčné kódy

Pri reprezentácii dát v počítači sa používajú N-tice núl a jednotiek. Jednu takúto N-ticu nazveme slovo. Na najjednoduchšie zabezpečenie, ktoré umožňuje detekovanie chyby, sa používa zabezpečenie paritou. Spočíva v pridaní jedného bitu ku každému prenášanému slovu tak, aby celkový počet jednotiek v zabezpečenom slove bol párny. Po prenesení sieťou sa v prenesenom slove spočítajú jednotky. Ak je ich počet nepárny pri prenose nastala určite chyba.

Príklad 1.2.1 *Urobte matematický model, ktorý popisuje zabezpečenie paritou ako transformáciu vektora do nadpriestoru.*

Riešenie:
Úlohu vysvetlíme na slovách, ktoré budú pozostávať z N znakov (0 alebo 1). Tieto N-tice tvoria vektorový priestor $\mathcal{S}$ so skalármi 0 a 1. Operácie sčítania a násobenia skalárov 0 a 1 sú definované ako sčítanie a násobenie modulo 2, (tabuľka 1.1).

$\oplus_2$	0	1
0	0	1
1	1	0

$\otimes_2$	0	1
0	0	0
1	0	1

Tabuľka 1.1: Tabuľky sčítania a násobenia modulo 2

Súčet vektorov $\mathbf{f} = (f_0, f_1, \ldots, f_{N-1})$ a $\mathbf{g} = (g_0, g_1, \ldots, g_{N-1})$ v priestore $\mathcal{S}$ je definovaný predpisom $\mathbf{f} + \mathbf{g} = (f_0 \oplus_2 g_0, f_1 \oplus_2 g_1, \ldots, f_{N-1} \oplus_2 g_{N-1})$.
Vektor $\mathbf{f} = (f_0, f_1, \ldots, f_{N-1})$ patriaci do N-rozmerného vektorového priestoru $\mathcal{S}$ môžeme zapísať ako lineárnu kombináciu vektorov jednotkovej bázy

$$\mathcal{E} = \{(1,0,0,0,\ldots,0),(0,1,0,0,\ldots,0),\ldots,(0,0,0,0,\ldots,1)\}$$

$$\mathbf{f} = f_0(1,0,0,0,\ldots,0) + f_1(0,1,0,0,\ldots,0) + \cdots + f_{N-1}(0,0,0,0,\ldots,1)$$

Vytvorme zabezpečený vektor $\mathbf{f}^\star$ ako prvok $N+1$-rozmerného vektorového priestoru $\mathcal{S}^\star$. Bude to vektor $\mathbf{f}^\star = (f_0, f_1, \ldots, f_{N-1}, z)$, kde z je podľa parity číslo 0 alebo 1. Zabezpečené kódové slová tvoria N-rozmerný vektorový podpriestor $\mathcal{K}$ vektorového priestoru $\mathcal{S}^\star(N+1$-rozmerného). Uvedené tvrdenie platí, ak ľubovoľná lineárna kombinácia dvoch zabezpečených kódových slov je zase zabezpečené kódové slovo. (Poprosíme čitateľa, aby toto tvrdenie overil).

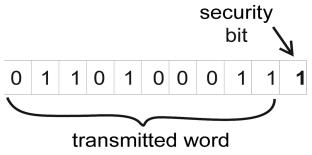
Bázou podpriestoru $\mathcal{K}$, ktorý pozostáva zo slov kódu (=zabezpečených slov) je množina

$$\mathcal{B}_\mathcal{K} = \{(1,0,0,0,\ldots,0,1),(0,1,0,0,\ldots,0,1),\ldots,(0,0,0,0,\ldots,1,1)\}$$

Ak vektor

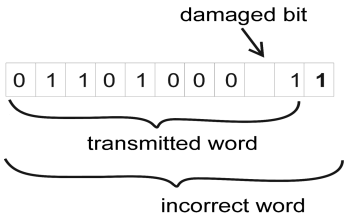
$$\mathbf{f}^\star = f_0(1,0,0,0,\ldots,0,1) + f_1(0,1,0,0,\ldots,0,1) + \cdots + f_{N-1}(0,0,0,0,\ldots,1,1)$$

po prenose sieťou nie je prvkom podpriestoru $\mathcal{K}$, tak hovoríme, že bola detekovaná chyba.
Príklad prenášaného slova a jeho zabezpečenia je na obrázku 1.1.



Obrázok 1.1: Zabezpečenie slova paritou

Na obrázku 1.2 vidíme príklad poškodeného slova, teda slova, ktoré nepatrí do podpriestoru kódových slov $\mathcal{K}$.



Obrázok 1.2: Detekcia chyby pri zabezpečení paritou

□

Úloha 1.2.1 *Navrhnite spôsob, ako zistiť, že zvolený vektor* $\mathbf{f}^\star$*, ktorý je prvkom* $N+1$*-rozmerného vektorového priestoru* $\mathcal{S}^\star$ *je vektorom podpriestoru* $\mathcal{K}$ *všetkých zabezpečených kódových slov. Pri riešení sa môže využiť skutočnosť, že vektor* $\mathbf{v}^\star = (1,1,\ldots,1)$ *je kolmý na všetky vektory podpriestoru* $\mathcal{K}$*.*

Pri zabezpečení paritou je možné detekovať chybu, ale nie je to zaručené. Chyba nebude zistená, ak sa zmenia 2, alebo párny počet bitov. Po detekovaní chyby je potrebné poškodené slovo preniesť znovu, pretože táto metóda nedáva odpoveď na to, ktorý bit bol poškodený.

Iným príkladom kódu je **cyklický kód**, ktorého názov sa odvodzuje od vlastnosti cyklického opakovania sa zvyškov po delení generujúcim polynómom.

Myšlienku zabezpečenia delením dohodnutým deliteľom môžeme ukázať na delení celými číslami. Dohodnime sa napríklad, že na zabezpečenie čísla 124 pre prenos je potrebné ho upraviť na číslo, ktoré bude deliteľné číslom 7.

$$124 : 7 = 17 \quad (\text{zvyšok } 5) \tag{1.2.1}$$

Upravíme číslo 124:

$$124 - 5 = 119$$

Číslo 119 už bude deliteľné bezo zvyšku. Toto číslo prenesieme sieťou. Ak po prenose bude na strane prijímača číslo deliteľné 7, prenos bol bezchybný a vieme, že prenesené číslo 119 bolo skutočne aj vyslané. Problémom ostáva, že 119 nie je pôvodné číslo 124. Ako 119 mohli byť vyslané aj čísla 120, 121, 122, 123 alebo 125. Pri tomto kódovaní nastala zmena pôvodného slova a preto sa nedozvieme, presne aké slovo bolo vyslané. Také kódy, pri ktorých sa pôvodné slovo zmení, sa nazývajú **konvolučné kódy**.

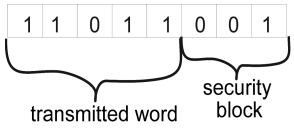
Kódy, pri ktorých sa ku pôvodnému slovu iba pridá zabezpečovacia časť (blok), sa nazývajú **blokové kódy**.
Ak chceme zabezpečiť číslo 124 tak, aby sme ho neporušili, pripíšeme zaň ešte jednu cifru 124x. Číslo x bude také, aby bolo číslo 124x deliteľné siedmimi.

$$124x : 7 = 1246 : 7 = 178 \quad \text{zvyšok} \quad 0 \tag{1.2.2}$$

Číslo $x = 6$ nájdeme tak, že za 124 pripíšeme 9, potom vydelíme $1249 : 7 = 178$ zvyšok 3. Nakoniec odpočítame $1249 - 3$ a dostaneme zabezpečené slovo 1246. Toto číslo je skutočne deliteľné číslom 7 bezo zvyšku.

Ukážkou blokového kódu je cyklický kód z nasledujúceho príkladu, ktorý sa v praxi naozaj používa.

Príklad 1.2.2 (Cyklický kód s generujúcim polynómom, CRC) *Nech slovo, ktoré má byť zakódované je pätica pozostávajúca z núl a jednotiek, napríklad* 11011. *Za toto slovo pridáme tri čísla (v tomto prípade sú to čísla* 001*) tak, aby polynóm, reprezentovaný celým zabezpečeným slovom bol deliteľný nejakým vopred dohodnutým generujúcim polynómom* $q(x) = x^3 + x + 1$. *Zabezpečené slovo je na obrázku* 1.3.



Obrázok 1.3: Blokový kód, ktorý zabezpečí slovo 11011 blokom 001

Zabezpečené slovo bude polynóm s koeficientami 1, 1, 0, 1, 1, 0, 0, 1 *pri jednotlivých mocninách* x, *konkrétne*

$$1 \cdot x^7 + 1 \cdot x^6 + 0 \cdot x^5 + 1 \cdot x^4 + 1 \cdot x^3 + 0 \cdot x^2 + 0 \cdot x^1 + 1 \cdot x^0 = x^7 + x^6 + x^4 + x^3 + 1$$

Sčítanie a delenie polynómov definujeme tak, aby koeficienty polynómov boli vždy z množiny $\{0, 1\}$. *Pri súčte polynómov budú koeficienty pri zodpovedajúcich mocninách súčtom modulo* 2. *Pri algoritme delenia rovnako použijeme násobenie a sčítanie modulo* 2.
Pre uvedené polynómy potom naozaj paltí

$$\left(x^7 + x^6 + x^4 + x^3 + 1\right) : \left(x^3 + x + 1\right) = x^3 + x^2 + x + 1 \quad (\textit{zvyšok} \quad 0)$$

Ako sa vypočítajú hodnoty 001 *v zabezpečovacom bloku? Ako sa detekuje chyba v prenesenom slove?*

Riešenie:
Prenášanému slovu 11011 zodpovedá polynóm

$$f(x) = 1 \cdot x^4 + 1 \cdot x^3 + 0 \cdot x^2 + 1 \cdot x^1 + 1 \cdot x^0 = x^4 + x^3 + x + 1$$

Podobne ako pri príklade s celými číslami 1.2.1, zistíme zvyšok po delení vopred určeným generujúcim polynómom $q(x) = x^3+x+1$. Tento zvyšok potom odpočítame, aby zabezpečený polynóm bol deliteľný polynómom $q(x)$ bezo zvyšku. Prepis do tvaru polynómu navyše umožňuje oddeliť informačnú a zabezpečovaciu časť kódového slova.
Vynásobením polynómu $f(x)$ polynómom x^k dostaneme polynóm $f^{\circledast}(x)$, ktorý bude mať nulové koeficienty pri mocninách $x^0, x^1, \ldots, x^{k-1}$ a koeficienty pri vyšších mocninách budú zodpovedať koeficientom polynómu $f(x)$.

$$f^{\circledast}(x) = 1 \cdot x^{k+4} + 1 \cdot x^{k+3} + 0 \cdot x^{k+2} + 1 \cdot x^{k+1} + 1 \cdot x^{k+0}$$

Zodpovedajúce slovo polynómu $f^{\circledast}(x)$ je $11011\underbrace{00\ldots0}_{k}$.

Hodnota k v polynóme x^k závisí od veľkosti zvyškov po delení generujúcim polynómom. V prípade polynómu $q(x) = x^3 + x + 1$ môžu byť zvyšky po delení

polynóm	slovo
0	0
1	1
x	10
$x+1$	11
x^2	100
x^2+x	110
x^2+x+1	111

Z tvaru všetkých zvyškov vidíme, že na zabezpečenie stačia tri miesta. Preto polynóm $f(x) = x^4 + x^3 + x + 1$ (teda slovo 11011) vynásobíme x^3.
Po vynásobení dostaneme polynóm

$$f^{\circledast}(x) = x^7 + x^6 + x^4 + x^3$$

Zodpovedajúce slovo k polynómu $f^{\circledast}(x)$ je 11011000.
Tento polynóm však nie je deliteľný bezo zvyšku generujúcim polynómom $q(x) = x^3+x+1$. Aby sme zistili, aké hodnoty má mať zabezpečovací blok, napodobíme postup podľa výpočtu 1.2.2.

$$\left(x^7 + x^6 + x^4 + x^3\right) : \left(x^3 + x + 1\right) = x^4 + x^3 + x^2 + x + 1 \quad (\text{zvyšok } 1)$$

Zvyšok 1 po delení generujúcim polynómom odpočítame (odpočítať binárne číslo modulo 2 znamená pripočítať toto číslo) od polynómu $f^{\circledast}(x)$ a dostaneme zabezpečený polynóm $f^{\star}(x)$.

$$f^{\star}(x) = f^{\circledast}(x) - 1 = f^{\circledast}(x) \oplus_2 1 = x^7 + x^6 + x^4 + x^3 + 1$$

Polynóm $f^{\star}(x)$ je skutočne bezo zvyšku deliteľný generujúcim polynómom $q(x) = x^3 + x + 1$ a navyše, jeho binárny zápis má informačnú časť oddelenú od zabezpečovacej časti.

Kódové slovo bude obsahovať chybu ak polynóm $f^{\star}(x)$ prislúchajúci tomuto slovu nie je deliteľný generujúcim polynómom $q(x) = x^3 + x + 1$. □

Úloha 1.2.2 *Dokážte, že pre ľubovoľné polynómy $f(x)$, $q(x)$ (z priestoru polynómov ako v príklade 1.2.2), polynóm $z(x)$ (zvyšok po delení $f(x) : q(x)$) a číslo k (ktoré udáva stupeň polynómu $q(x)$) platí:*

$$q(x) \quad \textit{je deliteľom polynómu} \quad f(x) \cdot x^k + z(x)$$

Úloha 1.2.3 *Ako by sa zmenil algoritmus zabezpečovania slova generujúcim polynómom, keby sa na konci slova nepridalo k núl, ale k jednotiek?*

Úloha 1.2.4 *Pri CRC kódovaní je potrebné deliť polynómy zodpovedajúce kódovým slovám. Slová pozostávajú z 0 a 1, preto by mohli byť považované za zápisy čísel v dvojkovej sústave. Je možné delenie polynómu polynómom nahradiť podielom čísel v dvojkovej sústave, ktoré robí kalkulačka? Ak áno, vysvetlite prečo, ak nie, nájdite príklad, v ktorom nebude výsledok pri týchto spôsoboch delenia rovnaký.*

Úloha 1.2.5 *Aké slová sa skrývajú za skratkou CRC?*

Príklad 1.2.3 *Urobte matematický model, ktorý popisuje cyklický kód s generujúcim polynómom ako transformáciu vektora do nadpriestoru.*

Riešenie:
Použijeme slová a generujúci polynóm ako v príklade 1.2.2. Naďalej budeme používať súčin a súčet modulo 2. Slová sú prvky vektorového priestoru $\mathcal{S}_5$ všetkých 5-zložkových vektorov zložených z 0 a 1. Tieto slová chceme preniesť nejakým prostredím a predtým zakódovať tak, aby boli deliteľné generujúcim polynómom. Bázu $\mathcal{B}_5$ priestoru $\mathcal{S}_5$ tvoria vektory

$$\mathbf{e}_0 = (1,0,0,0,0) \qquad \mathbf{e}_1 = (0,1,0,0,0) \qquad \mathbf{e}_2 = (0,0,1,0,0)$$

$$\mathbf{e}_3 = (0,0,0,1,0) \qquad \mathbf{e}_4 = (0,0,0,0,1)$$

Každý vektor $\mathbf{f} \in \mathcal{S}_5$ možno napísať ako lineárnu kombináciu vektorov bázy $\mathcal{B}$. Napríklad

$$\begin{aligned}&\mathbf{f} = (1,1,0,1,1) = \\ &= 1 \cdot (1,0,0,0,0) + 1 \cdot (0,1,0,0,0) + 0 \cdot (0,0,1,0,0) + 1 \cdot (0,0,0,1,0) + \\ &+ 0 \cdot (0,0,0,0,1) = 1 \cdot \mathbf{e}_0 + 1 \cdot \mathbf{e}_1 + 0 \cdot \mathbf{e}_2 + 1 \cdot \mathbf{e}_3 + 1 \cdot \mathbf{e}_4\end{aligned}$$

Vektorový priestor vnoríme ako podpriestor $\mathcal{S}$ do vektorového priestoru $\mathcal{S}_8$. Bázou $\mathcal{B}$ vektorového podpriestoru $\mathcal{S}$ budú 8 zložkové vektory, ktoré budú rozšírením vektorov bázy $\mathcal{B}_5$ a budú zodpovedať polynómom, ktoré sú deliteľné generujúcim polynómom $q(x) = x^3 + x + 1$.

$$\mathbf{b}_0 = (1,0,0,0,0,0,0,1) \quad \mathbf{b}_1 = (0,1,0,0,0,1,0,1) \quad \mathbf{b}_2 = (0,0,1,0,0,1,1,1)$$

$$\mathbf{b}_3 = (0,0,0,1,0,1,1,0) \quad \mathbf{b}_4 = (0,0,0,0,1,0,1,1)$$

Zabezpečiť vektor $\mathbf{f} = (1,1,0,1,1)$ cyklickým kódom znamená vyjadriť ho v báze $\mathcal{B}$, teda

$$\mathbf{f}^\star = 1 \cdot \mathbf{b}_0 + 1 \cdot \mathbf{b}_1 + 0 \cdot \mathbf{b}_2 + 1 \cdot \mathbf{b}_3 + 1 \cdot \mathbf{b}_4 = (1,1,0,1,1,0,0,1)$$

Vektoru $(1,1,0,1,1,0,0,1)$ zodpovedá polynóm $x^7+x^6+x^4+x^3+1$, ktorý je naozaj bezo zvyšku deliteľný polynómom generujúcim polynómom $q(x) = x^3+x+1$.

V prípade, že pri prenose komunikačným prostredím, teda transformácii vektora $\mathbf{f}$ na vektor $\mathbf{f}'$ nastane chyba a výsledný polynóm nebude deliteľný polynómom $q(x)$, tak vektor $\mathbf{f}' \notin \mathcal{S}$. □

Úloha 1.2.6 *Ukážte, že ak sú dva polynómy deliteľné polynómom $q(x)$, tak aj každá ich lineárna kombinácia je deliteľná polynómom $q(x)$.*

Úloha 1.2.7 *Navrhnite, ako pomocou podpriestoru kolmého na podpriestor $\mathcal{S}$, je možné overiť, či vektor $\mathbf{f}' \in \mathcal{S}$.*

Samoopravné kódy

Pri transformácii do nadpriestoru väčšieho rozmeru sa dá zistiť nielen to, či prenesené slovo je správne, alebo chybné, ale napríklad aj to ako asi vyzeralo pôvodné slovo. Ak by sme chceli k chybnému slovu $\mathbf{f}'$ nájsť slovo $\mathbf{f}^\star$, ktoré malo byť prijaté, môžeme napríklad spočítať vzdialenosť vektora $\mathbf{f}'$ od všetkých vektorov priestoru $\mathcal{S}$. Vektor s najmenšou vzdialenosťou budeme považovať za vyslaný vektor $\mathbf{f}^\star$. Výhodná pre tento výpočet sa ukázala Hammingova vzdialenosť:

$$\mathrm{d}(\mathbf{f},\mathbf{g}) = \text{počet miest, na ktorých sa vektory navzájom líšia}$$

Úloha 1.2.8 *Zovšeobecneným kódovaním nazveme také vnorenie priestoru kódových slov do nadpriestoru, v ktorom sa jednotlivé slová od seba čo najviac líšia. Stačí, keď táto požiadavka bude splnená v nejakej vhodnej metrike. Oprava chybného slova bude spočívať v nájdení najbližšieho slova kódu. Navrhnite kódovanie, ktoré spĺňa uvedené podmienky.*

1.3 Šifrovanie

Pri využívaní informačných sietí je potrebné časť prenášanej informácie utajiť pred možnosťou prečítania neoprávnenou osobou. Princíp, ktorý sa pri tomto utajovaní používa je zmena bázy.

Na strane vysielača je proces $\mathbf{f} = (f_0, f_1, \ldots, f_{N-1})$, tento budeme považovať za vektor koeficientov v báze $\mathcal{B}$, teda

$$\mathbf{f} = (f_0, f_1, \ldots, f_{N-1})_{\mathcal{B}} = \sum_{n=0}^{N-1} f_n \cdot \mathbf{b}_n = (c_0, c_1, \ldots, c_{N-1})_{\mathcal{E}}$$

$$\mathbf{f}\mathbb{B} = \mathbf{c}\mathbb{E}$$

$$\mathbf{f} = \mathbf{c}\mathbb{E}\mathbb{B}^{-1}$$

Maticu $\mathbb{B}$ nazývame kódovacia (encoding) matica a maticu $\mathbb{B}^{-1}$ nazývame dekódovacia (decoding) matica.

Bezpečnostné kódy a elektronický podpis sú dva spôsoby šifrovania, ktoré sa líšia tým, ako je správa šifrovaná. Niekedy správu nemôže prečítať nikto okrem adresáta, alebo naopak, prečítať ju môže každý, ale zašifrovať len vlastník kľúča.

Treba poznamenať, že šifrovanie založené na lineárnej transformácii je ľahko prelomiteľné. Preto sa v praxi používajú iné, nelineárne transformácia. Myšlienka je však podobná: úloha zašifrovania je jednoduchšia ako úloha hľadania kľúča.

Bezpečnostné kódy

V prípade bezpečnostných kódov je známy verejný kľúč, teda spôsob ako správu zašifrovať. Na nájdenie tohoto spôsobu by bolo treba viac času, než ako dlho má byť informácia utajená. Dôvody prečo má byť správa utajená si čitateľ určite dokáže predstaviť sám, my sa sústredíme na spôsob, akým môže byť správa zakódovaná.

Príklad 1.3.1 *Vektor* (a, b, c) *šifrujeme napríklad tak, že ho vynásobíme maticou* $\mathbf{B}$.

$$\mathbb{B} = \begin{pmatrix} 2 & 1 & 4 \\ 3 & 2 & 0 \\ 4 & 1 & 2 \end{pmatrix}$$

Dostaneme vektor (slovo) $(2a + 3b + 4c, a + 2b + c, 4a + 2c)$*, ktoré sa dešifruje tak, že sa vynásobí maticou*

$$\mathbb{B}^{-1} = \begin{pmatrix} -0.22 & -0.11 & 0.44 \\ 0.33 & 0.67 & -0.67 \\ 0.28 & -0.11 & -0.06 \end{pmatrix}$$

Koľko slov a im prislúchajúcich zašifrovaných slov je treba, aby sme dokázali dešifrovať každé ďalšie slovo. Ako to treba urobiť?

Riešenie:
Na to, aby sme dokázali vypočítať aké bolo pôvodné slovo ak poznáme zašifrované slovo, je potrebné poznať maticu $\mathbb{B}^{-1}$.

Na prelomenie šifry treba zostaviť sústavu 9 rovníc o 9 neznámych, ktoré sú prvkami matice $\mathbb{B}^{-1}$.

Na to treba poznať 3 vektory (slová) v pôvodnom a zašifrovanom tvare. Predpokladajme, že poznáme slová (vektory) a ich šifry:

$$(1, 2, 3) \mapsto (20, 8, 10)$$

$$(3, 4, 1) \mapsto (22, 12, 14)$$

$$(-1, 2, 1) \mapsto (8, 4, -2)$$

Hľadáme neznámu maticu

$$\mathbb{B} = \begin{pmatrix} b_{1\,1} & b_{1\,2} & b_{1\,3} \\ b_{2\,1} & b_{2\,2} & b_{2\,3} \\ b_{3\,1} & b_{3\,2} & b_{3\,3} \end{pmatrix}$$

Sústava rovníc, ktorej riešením bude matica $\mathbb{B}$ je

$$\begin{array}{rrrcr}
1\cdot b_{1\,1}+ & 2\cdot b_{2\,1}+ & 3\cdot b_{3\,1} & = & 20\\
1\cdot b_{1\,2}+ & 2\cdot b_{2\,2}+ & 3\cdot b_{3\,2} & = & 8\\
1\cdot b_{1\,3}+ & 2\cdot b_{2\,3}+ & 3\cdot b_{3\,3} & = & 10\\
3\cdot b_{1\,1}+ & 4\cdot b_{2\,1}+ & 1\cdot b_{3\,1} & = & 22\\
3\cdot b_{1\,2}+ & 4\cdot b_{2\,2}+ & 1\cdot b_{3\,2} & = & 12\\
3\cdot b_{1\,3}+ & 4\cdot b_{2\,3}+ & 1\cdot b_{3\,3} & = & 14\\
-1\cdot b_{1\,1}+ & 2\cdot b_{2\,1}+ & 1\cdot b_{3\,1} & = & 8\\
-1\cdot b_{1\,2}+ & 2\cdot b_{2\,2}+ & 1\cdot b_{3\,2} & = & 4\\
-1\cdot b_{1\,3}+ & 2\cdot b_{2\,3}+ & 1\cdot b_{3\,3} & = & -2
\end{array}$$

□

Elektronický podpis

Opačná situácia než pri bezpečnostných kódoch nastáva pri elektronickom podpise. Tu je neverejný (súkromný) kľúč, ktorým sa podpis vytvára, ale každý, kto chce, môže jeho správnosť skontrolovať použitím verejného kľúča. Oba kľúče si vygeneruje užívateľ a verejný kľúč si nechá potvrdiť certifikačnou autoritou. Okrem autorizácie dokumentu, podpisujúci elektronickým podpisom zabezpečí dokument aj voči neoprávneným zásahom, pretože elektronický podpis funguje len s neporušeným dokumentom.

V prenášanej správe sa určitým spôsobom vyhodnotia všetky jej znaky a z nich sa vytvorí číslo (message digest), ktoré sa následne zašifruje pomocou súkromného kľúča. Prijímateľ správy ju potom dešifruje verejným kľúčom, ktorý je voľne dostupný na nejakom dohodnutom mieste alebo ešte lepšie pripojený ku správe vo forme certifikátu. Certifikát je vlastne elektronicky podpísaný dokument certifikačnou autoritou, na ktorom je verejný kľúč a identifikácia osoby, ktorej daný kľúč patrí. Z dôvodu utajenia nie je technicky možné z verejného kľúča odvodiť (vygenerovať) súkromný kľúč.

Uvedieme ešte pekný príklad, ktorý je inšpirovaný myšlienkou z knihy [**?**].

Príklad 1.3.2 *Dvaja priatelia, vášniví šachisti, ktorí hrávajú šach bez šachovnice, sa rozhodli, že si takýmto spôsobom zahrajú poker. Presnejšie, úlohou je navrhnúť, ako hrať kartovú hru cez telefón, tak, aby pri nej bolo možné rozdať karty, striedavo ťahať z kopy, pričom každý z hráčov pozná svoje a nepozná súperove karty. Nebudeme riešiť všetky komplikácie, ktoré pri hre museli vyriešiť, sústredíme sa iba na prvú z nich. Rozdávanie kariet. Akým spôsobom môže hráč súperovi povedať, aké karty má na ruke, tak aby to ostalo počas hry utajené?*

Riešenie:
Predstavme si zobrazenie, ktoré každej z 52 kariet s ktorými sa hrá poker priradí postupne čísla $1, 2, 3, 4, 5, \ldots, 52$. Jednotlivým farbám potom koeficienty $29, 31, 37, 41$. Keď hráč vytiahne napríklad karty $Z7, Z9, SK, GA$ pošle ako informáciu o svojich kartách číslo 1782, ktoré vypočíta ako

$$29 \cdot 3 + 29 \cdot 7 + 41 \cdot 19 + 31 \cdot 23 = 1782$$

Keď ako odpoveď od súpera dostane číslo 1524, nevie vypočítať aké karty má na ruke súper. Po hre však môžu spoločne skontrolovať, či naozaj súper hral s kartami, ktoré uviedol počas hry. □

Úloha 1.3.1 *V knihe* [**?**] *je myšlienka z príkladu* 1.3.2 *realizovaná pomocou dvoch šifier (=kľúčov). Tieto kľúče majú tú vlastnosť, že pri šifrovaní aj dešifrovaní je možné ľubovoľne meniť ich poradie. Platí*

$$K_1(K_2(X)) = K_2(K_1(X)) \qquad K_1^{-1}(K_2^{-1}(X)) = K_2^{-1}(K_1^{-1}(X)) \qquad (1.3.1)$$

Poker cez telefón sa hrá potom tak, že si hráči posielajú dva razy zašifrovaný zoznam kariet a podľa potreby dešifruje zoznam kariet raz jeden, raz druhý hráč. Skúste simulovať takéto šifrovanie pomocou schránky s dvomi zámkami a hru odohrať tak, že si budete posúvať karty v uzamknutej schránke.

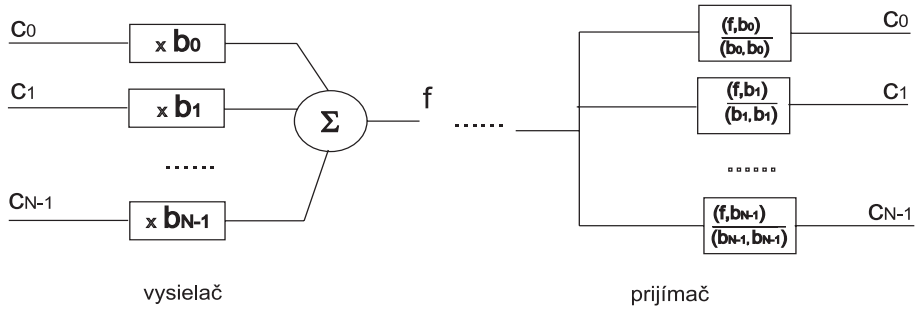
Úloha 1.3.2 *Násobenie maticou vlastnosť* 1.3.1 *nemá, lebo násobenie matíc nie je komutatívne, preto na úlohu šifrovania kariet treba použiť nelineárne šifry. Nájdite príklad takých matíc (kľúčov)* $K_1 = \mathbb{A}$ *a* $K_2 = \mathbb{B}$*, pre ktoré neplatí rovnosť v* 1.3.1*. Hľadáme také matice, pre ktoré*

$$K_1(K_2(X)) \neq K_2(K_1(X)) \qquad \textit{pre matice} \qquad (\mathbf{x} \cdot \mathbb{A}) \cdot \mathbb{B} \neq (\mathbf{x} \cdot \mathbb{B}) \cdot \mathbb{A}$$

1.4 Modulácia

V snahe, čo najlepšie využiť kapacitu prenosového kanála, používa sa pri prenose dát mechanizmus, ktorý dovoľuje tým istým kanálom prenášať rôzne dáta. Tento mechanizmus sa nazýva modulácia a spočíva vo vtlačení určitého príznaku prenášaným dátam tak, aby sa po prenose dali tieto dáta od seba bezpečne oddeliť, demodulovať.

Modulácia teda spočíva v transformácii procesu do tvaru, ktorý dovoľuje prenos. Matematický princíp, ktorý využíva modulácia je vnorenie procesu do nadpriestoru. Jednotlivé hodnoty, ktoré chceme preniesť prevedieme na vektory nadpriestoru tak, že ich vynásobíme bázovými vektormi patriacimi nejakej báze vektorového priestoru. Vynásobené vektory sčítame a takto skonštruovaný vektor (proces) prenesieme spoločným komunikačným prostredím. Po prenose, na strane prijímača proces demodulujeme tak, že urobíme jeho priemety do jednorozmerných podpriestorov generovaných jednotlivými bázovými vektormi. Schéma postupu pri modulácii a demodulácii je nakreslená na obrázku 1.4.



Obrázok 1.4: Schematický náčrt princípu modulácie a demodulácie

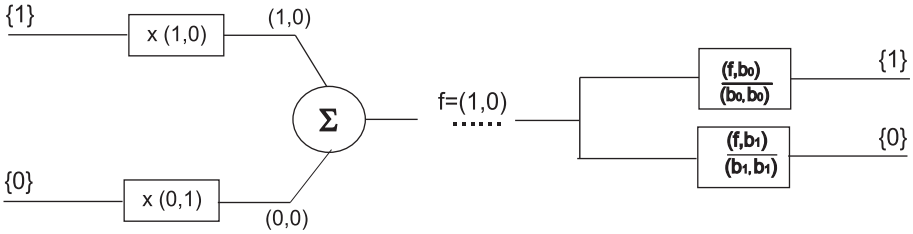
Príklad 1.4.1 *Na vstupe A je jedna hodnota z množiny $\{0,1\}$. Na vstupe B je jedna hodnota z množiny $\{0,1\}$. Popíšte a zakreslite priebeh modulácie a demodulácie týchto hodnôt pomocou jednotkovej bázy dvojrozmerného vektorového priestoru.*

Riešenie:
Hodnotu c_A z množiny $\{0,1\}$ na vstupe A vynásobíme vektorom $(1,0)$. Nech je táto hodnota 1.
Hodnotu z c_B množiny $\{0,1\}$ na vstupe B vynásobíme vektorom $(0,1)$. Nech je táto hodnota 0.
Vektory sčítame a cez spoločné komunikačné prostredie prenesieme niektorý z vektorov $(0,0),(1,0),(0,1),(1,1)$, v našom prípade to bude vektor (1,0).
Demodulátor na strane prijímača potom urobí priemet preneseného vektora do dvoch podpriestorov. Do podpriestoru generovaného vektorom $(1,0)$ pre výstup A a do podpriestoru generovaného vektorom $(0,1)$ pre výstup B.

$$c_A = \frac{((1,0),(1,0))}{((1,0),(1,0))} = 1$$

$$c_B = \frac{((1,0),(0,1))}{((0,1),(0,1))} = 0$$

Schéma postupu pri modulácii a demodulácii je nakreslená na obrázku 1.5. □



Obrázok 1.5: Schematický náčrt modulácie a demodulácie pre dve vstupné hodnoty

Úloha 1.4.1 *Pri prenose dvoch hodnôt z množiny $\{0,1\}$, tak ako je popísaná v príklade* 1.4.1*, je komunikačným prostredím prenášaný dvojzložkový vektor. Aký je vlastne význam modulácie, keď namiesto dvoch hodnôt* 1 *a* 0 *je prenášaný vektor* $(1,0)$*? Nestačilo by len povedať, že prvá zložka vektora bude hodnota zo vstupu A a druhá zložka vektora bude hodnota zo vstupu B? Vymyslite nejaké zdôvodnenie!*

1.5 Optimálny prijímač

V úlohe modulácie sme riešili technickú úlohu ako cez jedno komunikačné prostredie prenášať naraz väčší počet správ. Teraz budeme riešiť úlohu, ako upraviť dáta pre prenos tak, aby aj v prípade porúch počas prenosu, boli tieto na strane prijímača správne zrekonštruované. V tejto úlohe predpokladáme, že počas

prenosu kanálom budú dáta náhodne zmenené. K prenášaným dátam sa pridá šum.

Na vstupe kanála dáta upravíme tak, aby na výstupe bolo možné správne identifikovať, čo bolo vyslané. Úprava na vstupe bude spočívať vo vnorení hodnôt do nadpriestoru tak, aby boli v tomto nadpriestore jednotlivé prenášané prvky od seba dostatočne vzdialené. Aj po prenose potom bude možné zistiť, čo bolo vyslané. Vysielanú hodnotu vynásobíme bázovým vektorom. Počas prenosu sa proces zašumí. Po prenose urobíme priemet procesu do podpriestoru generovaného bázovým vektorom. Priemetu porovnáme s množinou prípustných výsledkov a vyberieme najbližší z nich.

Príklad 1.5.1 *Na vstupe kanála je jedna hodnota z množiny* $\{0,1\}$. *Popíšte a zakreslite spracovanie a optimálny príjem pomocou vektora* $(1,1,1)$.

Riešenie:
Hodnotu c z množiny $\{0,1\}$ na vstupe vynásobíme vektorom $\mathbf{b} = (1,1,1)$. Ak je táto hodnota 1, budeme prenášať vektor $\mathbf{f} = 1 \cdot \mathbf{b} = (1,1,1)$, ak je táto hodnota 0, budeme prenášať vektor $\mathbf{f} = 0 \cdot \mathbf{b} = (0,0,0)$.
Vektor prenesieme cez komunikačný kanál, kde naň pôsobí šum a preto dostaneme niektorý z vektorov

$$(0,0,0),(1,0,0),(0,1,0),(1,1,0),(0,0,1),(1,0,1),(0,1,1),(1,1,1)$$

Nech to v našom prípade to bude vektor $\mathbf{f}' = (1,0,1)$.
Optimálny prijímač potom urobí priemet preneseného vektora do podpriestoru generovaného vektorom $\mathbf{b} = (1,1,1)$.
Koeficient priemetu $\tilde{\mathbf{f}}'$ vektora $\mathbf{f}'$ je

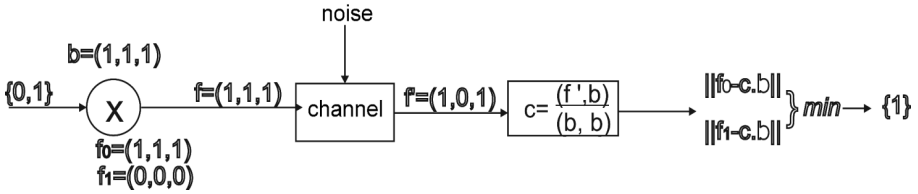
$$c = \frac{(\mathbf{f}', \mathbf{b})}{(\mathbf{b}, \mathbf{b})} = \frac{((1,0,1),(1,1,1))}{((1,1,1),(1,1,1))} = 0.67$$

Porovnajme teraz výsledný priemet $\tilde{\mathbf{f}}' = (0.67, 0.67, 0.67)$ s dvomi prípustnými možnosťami vyslaného procesu. Mohol to byť $\mathbf{f}_0 = (1,1,1)$ alebo $\mathbf{f}_1 = (0,0,0)$.

$$\|\tilde{\mathbf{f}}' - \mathbf{f}_0\| = \|(-0.33, -0.33, -0.33)\| = \sqrt{(-0.33)^2 + (-0.33)^2 + (-0.33)^2} = 0.58$$

$$\|\tilde{\mathbf{f}}' - \mathbf{f}_1\| = \|(0.67, 0.67, 0.67)\| = \sqrt{0.67^2 + 0.67^2 + 0.67^2} = 1.16$$

Porovnaním veľkosti odchýliek $\|\tilde{\mathbf{f}}' - \mathbf{f}_0\|$ a $\|\tilde{\mathbf{f}}' - \mathbf{f}_1\|$ vidíme, že vyslaný bol vektor $\mathbf{f}_0$ a teda pôvodná vyslaná hodnota bola 1. Schéma použitia takéhoto optimálneho prijímača je nakreslená na obrázku 1.6. □



Obrázok 1.6: Schéma práce optimálneho prijímača

Úloha 1.5.1 *Navrhnite optimálny prijímač pre prenos 8 rôznych hodnôt. Koľko rozmerný vektorový priestor treba použiť?*

Poznámka:
Optimálny prijímač a samoopravné kódy sú dve technológie, ktoré využívajú rovnaký princíp vnorenia vektora, predstavujúceho prenášané slovo, do nadpriestoru. Komunikačným prostredím sú potom prenášané vektory, ktorých vzájomná vzdialenosť je dostatočne veľká na to, aby ich bolo možné aj po chybnom prenose správne vyhodnotiť. To znamená nahradiť najbližšími vektormi pôvodného priestoru. Optimálny prijímač a samoopravné kódy teda používajú rovnaký princíp v rôznych vektorových priestoroch, s rôznymi metrikami.

1.6 Riešenie neriešiteľnej sústavy rovníc

Metódu priemetu do podpriestoru možeme využiť aj na nájdenie približného riešenia sústavy rovníc, ktorá nemá riešenie. Nájdeme čo najlepšiu aproximáciu riešenia, teda také hodnoty x a y, ktoré dajú po dosadení čo najpresnejší odhad pravej strany.

Príklad 1.6.1 *Sústava rovníc* 1.6.1 *nie je riešiteľná. Určite také hodnoty x a y, pre ktoré má ľavá strana sústavy najmenšiu odchylku od pravej strany.*

$$\begin{aligned} 2 \cdot x + 3 \cdot y &= 3 \\ 1 \cdot x + 4 \cdot y &= 4 \\ 2 \cdot x + 3 \cdot y &= 2 \\ 4 \cdot x + 1 \cdot y &= 1 \\ 3 \cdot x + 2 \cdot y &= 3 \\ 2 \cdot x + 2 \cdot y &= 2 \end{aligned} \tag{1.6.1}$$

Riešenie:
Sústavu rovníc 1.6.1 môžeme napísať v tvare

$$x \cdot \mathbf{u} + y \cdot \mathbf{v} = \mathbf{z}$$

kde $\mathbf{u} = (2, 1, 2, 4, 3, 2)$, $\mathbf{v} = (3, 4, 3, 1, 2, 2)$ a $\mathbf{z} = (3, 4, 2, 1, 3, 2)$.

Hľadáme také koeficienty x_0 a y_0, pre ktoré bude platiť

$$\tilde{\mathbf{z}} = x_0 \cdot \mathbf{u} + y_0 \cdot \mathbf{v}$$

a zároveň

$$\|\mathbf{z} - \tilde{\mathbf{z}}\| \to min$$

To nastane vtedy, keď bude

$$(\mathbf{z} - \tilde{\mathbf{z}}) \perp \mathbf{u} \quad \text{a} \quad (\mathbf{z} - \tilde{\mathbf{z}}) \perp \mathbf{v}$$

Po úpravách

$$\begin{aligned} \langle \mathbf{z} - \tilde{\mathbf{z}}, \mathbf{u} \rangle = 0 \quad \text{a} \quad \langle \mathbf{z} - \tilde{\mathbf{z}}, \mathbf{v} \rangle = 0 \\ \langle \mathbf{z}, \mathbf{u} \rangle = \langle \tilde{\mathbf{z}}, \mathbf{u} \rangle \quad \text{a} \quad \langle \mathbf{z}, \mathbf{v} \rangle = \langle \tilde{\mathbf{z}}, \mathbf{v} \rangle \\ \langle \mathbf{z}, \mathbf{u} \rangle = \langle x_0 \cdot \mathbf{u} + y_0 \cdot \mathbf{v}, \mathbf{u} \rangle \quad \text{a} \quad \langle \mathbf{z}, \mathbf{v} \rangle = \langle x_0 \cdot \mathbf{u} + y_0 \cdot \mathbf{v}, \mathbf{v} \rangle \\ \langle \mathbf{z}, \mathbf{u} \rangle = x_0 \cdot \langle \mathbf{u}, \mathbf{u} \rangle + y_0 \cdot \langle \mathbf{v}, \mathbf{u} \rangle \quad \text{a} \quad \langle \mathbf{z}, \mathbf{v} \rangle = x_0 \cdot \langle \mathbf{u}, \mathbf{v} \rangle + y_0 \cdot \langle \mathbf{v}, \mathbf{v} \rangle \end{aligned}$$

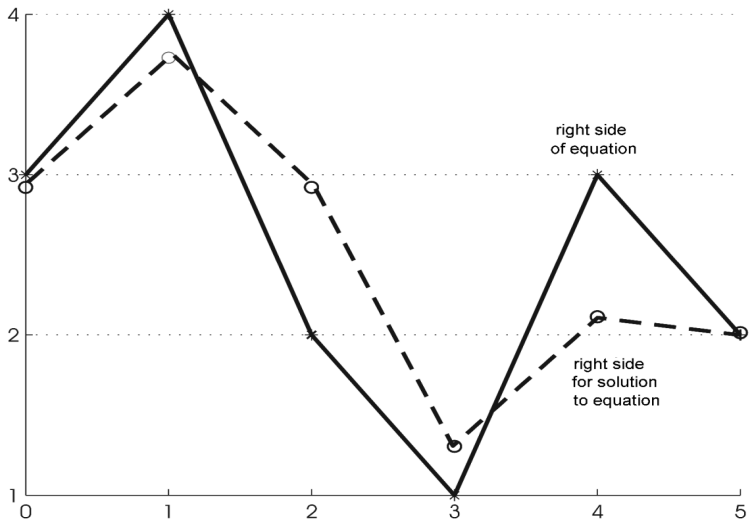
$$31 = 38x_0 + 30y_0 \quad \text{a} \quad 42 = 30x_0 + 43y_0$$

Riešením poslednej sústavy sú koeficienty $x_0 = 0.1$ a $y_0 = 0.91$. Ich dosadením dostaneme približne pravú stranu sústavy 1.6.1 rovnú $\tilde{\mathbf{z}}$, jej hodnoty sú vykreslené na obrázku 1.7.

$$\tilde{\mathbf{z}} = (2.92, 3.73, 2.92, 1.31, 2.11, 2.01)$$

Chyba odhadu pravej strany je 1.35 a približné riešenie sústavy je

$$(x, y) = (0.1, 0.91)$$



Obrázok 1.7: Pravá strana sústavy rovníc a jej priemet do podpriestoru všetkých riešení sústavy 1.6.1

□