# ELEMENTÁRNÍ TEORIE KÓDOVÁNÍ [1, 7, 10]

- jedna z nejzajímavějších součástí současné informatiky
- zabývá se jak konstrukcemi kódu, tak studiem jejich vlastností.
- vznik současné teorie kódování je v podstatě datován již od čtyřicátých let 20. století, a to pracemi Shannona, které byly věnovány teorii informace a dále hlavně Hamminga s Golayem, kteří zkonstruovali první lineární kódy.
- praktický boom pak nastal s příchodem „použitelné“ výpočetní techniky.
  - počítače běžně pracují s dvojkovou (binární) soustavou
    - nevýhodná pro lidské uvažování - neumíme v ní běžně počítat.
    - proto k převodu mezi „lidskými“ informacemi a počítači existují převodní pravidla *kódy*.

**Kód** je množina symbolů, kterými vyjadřujeme různé stavy systému.

Kód lze libovolně převádět jeden na druhý, aniž by se změnila velikost a množství informace.

# ELEMENTÁRNÍ TEORIE KÓDOVÁNÍ

Kódování lze dle účelu rozdělit na několik samostatných částí, které se v praxi mohou dokonce velmi často prolínat:

## Minimální kódování (komprese dat)

- Jedná se o způsob kódování, jehož účelem je zmenšit objem dat ve zprávě.
- Bezztrátové – představiteli této skupiny jsou komprimační programy (ARJ, ZIP, RAR, aj.)
- Ztrátové - reprezentanty jsou většinou nástroje pracující s obrazem nebo zvukem, a z nich vyplývající datové formáty (JPEG, MPG, MP3, aj.).

## Bezpečnostní kódování (samoopravné kódy)

- užívá se při přenosu informace reálným přenosovým kanálem -- může dojít k chybám
- smyslem bezpečnostního kódování je detekovat, případně i přímo opravit vzniklou chybu.
- Příkladem může být realizace zabezpečení paketů v síti internet, tzv. CRC kódem

## Kryptografické kódování (kryptografie) - šifrování

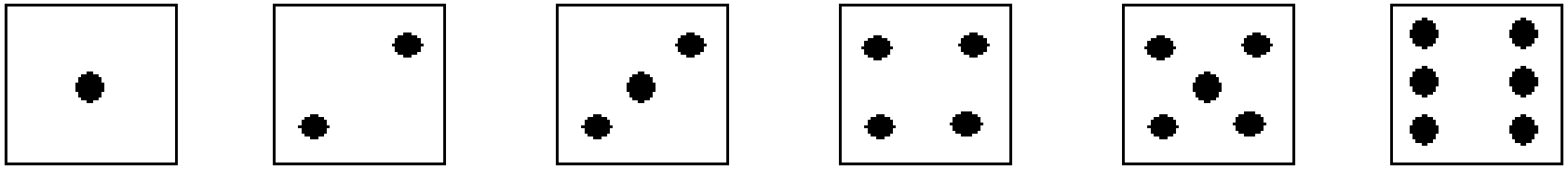
- Kódování za účelem utajení informace
- Obecně zahrnuje, jak jednoduché postupy známé z rébusů či skautských her, tak opravdu silné algoritmy k jejichž prolomení bychom potřebovaly mnoho času a špičkovou techniku

# Kódování

můžeme charakterizovat jako převod jednoho kódu na druhý a to buď pomocí tabulky anebo algoritmu.

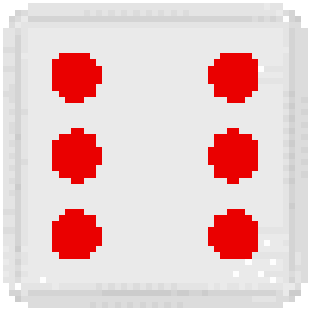
Příklad: Význam (vysvětlení) pojmu „kódování“ lze ukázat na systému hrací kostky.

Po hodu se může kostka nacházet v jednom ze šesti stavů (poloh). Stav (poloha) kostky se nejčastěji vyjadřuje pomocí množiny tečkových symbolů **(kódu):**



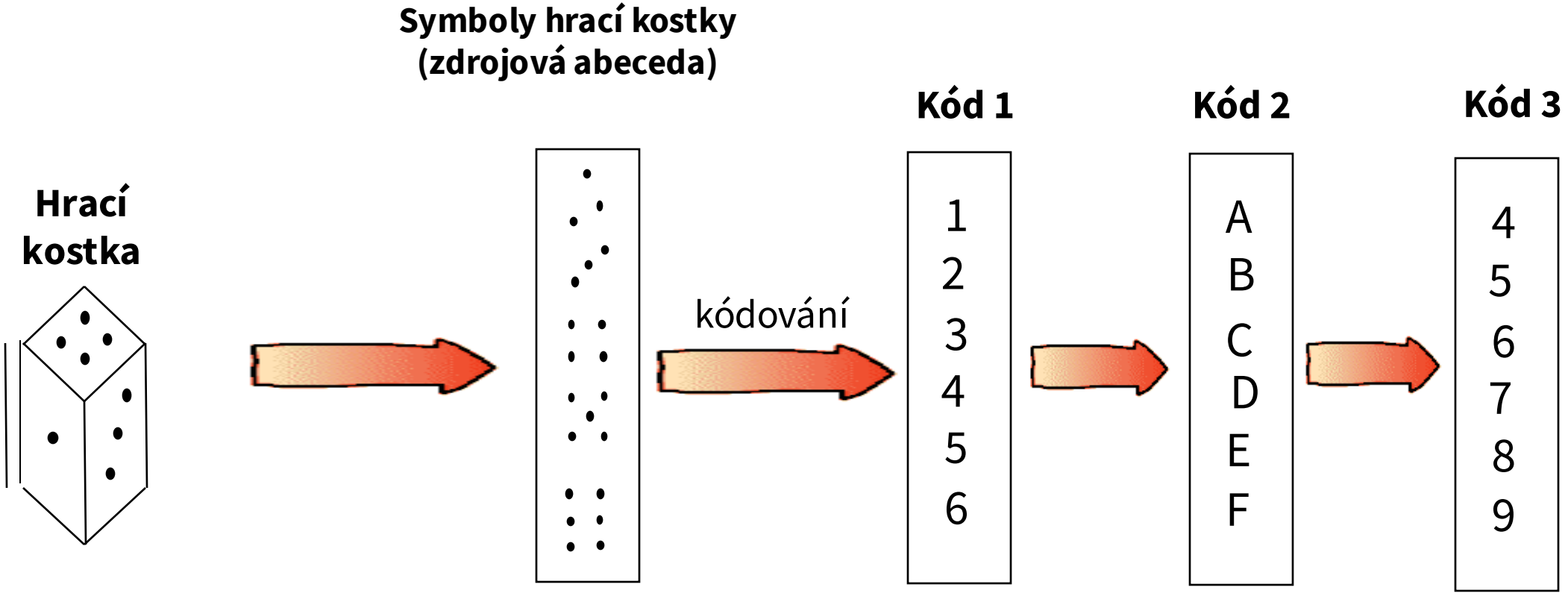
Pokud chceme zaznamenat výsledek náhodného pokusu

- neuděláme to zápisem tečkových symbolů,
- **Zapíšeme pomocí číslic (1,2,3,4,5,6)**
- **převedeme** z jedné množiny symbolů do druhé (jeden kód na druhý)



# Kódování

Vztah mezi zdrojovými symboly (zdrojovou abecedou), kódem a kódováním nám znázorňuje obrázek (ukázka kódování do třech různých kódů [11]).



**Při přenosu informací používáme dvě základní abecedy.**

## Zdrojová (abeceda zdroje)

konečná množina znaků (symbolů), kterou chceme přenést přes nějaké přenosové zařízení (přenosový kanál) - $\mathbf{A=\{ a_1, \dots , a_N \}}$

## Přenosová abeceda (abeceda přenosového kanálu)

konečná množina znaků, kterou můžeme přenášet přes nějaké přenosové zařízení

$\mathbf{Y=\{ y_1, \dots , y_R \}}$

Příklad: Přenos zprávy (textu) pomocí binárního kanálu .

Zdrojovou abecedou jsou všechna písmena a znaky české abecedy **A={ a,b T,K, 0,2,3,… }**

Přenosovou abecedou budou pouze dvě hladiny (úrovně) binárního signálu, například 10 V, 0 V. V praxi vyjadřujeme jako 0 a 1. **Y={ 0, 1 }**

Pod ***kódováním*** pak rozumíme zobrazení, které každému znaku zdrojové abecedy $a_i \in A$ přiřadí nějakou skupinu znaků přenosové abecedy $y_j \in Y$ - **kódové slovo**.

**Množina všech kódových slov se pak nazývá kód.**

**Délka kódu** – počet kódových slov daného kódu

Pokud uvažujeme konstantní délku kódového slova **m,** pak maximální délka kódu je dána vztahem:

$$L = R^m$$

R – počet prvků přenosové abecedy

- Dále se budeme zabývat pouze dvojkovými (binárními) kódy
  **=>** přenosová abeceda má jen dva prvky {0, 1}

- Délka kódu **L** se bude tedy pohybovat v rozmezí daném nerovností:

$$1 \leq L \leq 2^m$$

Kombinační možnost vzájemného přiřazení množiny kódových slov a množiny znaků zdrojové abecedy je dána permutací, tj. $2^m!$

Říkáme, že kód má $2^m!$ **kódovacích klíčů.**

**Výskyt jednotlivých znaků zdrojové abecedy = různá pravděpodobnost**

⇒ **pravděpodobnost výskytu kódových slov se liší**

⇒ **rozdílná hodnota kódovacích klíčů**

Výhodnější jsou takové kódovací klíče, které přiřazují odolnější kódové slova (kratší) nejčastěji se vyskytujícím znakům zdrojové abecedy.

**Je výhodnější nevyužívat maximální délky kódu L, nýbrž počítat (s ohledem na zvýšení bezpečnosti přenosu) pouze s délkou zkrácenou $L_z$ .**

$$L_z \leq L$$

**Kódování** však nemá pouze mechanicky nahrazovat jednu konkrétní abecedu jinou.

**Mění-li** se při kódování množina použitých symbolů i rozložení pravděpodobností, **mění** se také **entropie** na symbol zprávy.

Hlavním **cílem** je zajistit **nejmenší počet** symbolů na dané množství informace.

Důležité je, aby bylo možné ze zakódované zprávy získat zprávu zdrojovou

- *kódování musí být jednoznačně **dekódovatelné.***

**Kódování** je jednoznačně **dekódovatelné** tehdy, jestliže zobrazení, které každému znaku $a_i \in A$ přiřadí nějakou skupinu znaků $y_j \in Y$ je ***prostým zobrazením.***

Pojem slovo (kódové slovo) nad danou abecedou se liší od pojmu slova používaného v běžném hovorovém jazyce (slovem rozumíme nejmenší samostatnou jazykovou jednotku mající věcný a mluvnický význam).

Umělým jazykem (stručně jazykem) nazýváme nějakou abecedu spolu s pravděpodobnostními zákony, podle nichž vzniká z dané abecedy zpráva.

Je-li H entropie daného jazyka a $H_{max}$ je maximální entropie při použití téže abecedy pak můžeme definovat redundanci (nadbytečnost) daného jazyka jako:

$$R = 1 - \frac{H}{H_{max}}$$

Redundance evropských jazyků je větší než 0.5 a podle některých propočtů se pohybuje od 0.68 do 0.75.

# Rozdělení kódů

- **Podle použití se dělí dvojkové kódy**
  - kódy určené k matematickým operacím,
  - kódy určené k přenosu zpráv
- **Podle struktury na**
  - kódy nesystematické
  - kódy systematické
- **Podle délky na**
  - kódy rovnoměrné
  - kódy nerovnoměrné

# Rovnoměrné kódy

**Rovnoměrné kódy jsou takové kódy**, u kterých se každému znaku zdrojové abecedy přiřadí stejný počet kódových znaků.

Z toho důvodu mají tyto kódy velkou **výhodu** při dekódování.

Nerozšířenější příklady rovnoměrných kódů jsou:

- ***telegrafní kód***
- ***ASCII kód***

# Telegrafní kód – MTA2 [5]

- pětibitová abeceda, používaná zejména v dálnopisném provozu.
- označuje se jako **MTA-2** (mezinárodní telegrafní abeceda)
  - **kód Baudot**.

- každému znaku zdrojové abecedy je přiřazena konkrétní **pětice** kódových znaků 0 a 1.
- **Zdrojová abeceda** obsahuje
  - všechna písmena anglické abecedy, číslice 0, 1, ... , 9
  - a některá interpunkční znaménka.

- Vzhledem k tomu, že pětibitový kód není schopen pokrýt veškerou množinu znaků, používají se řídící znaky přepínání registru.

- Abeceda má nulovou redundanci a využívá plně celý prostor kódu.

# Jak to vlastně funguje?

Kapacita pětimístného binárního kódu je $2^5 = 32$ kódových slov.

Zdrojová abeceda obsahuje:

- Minimálně 26 písmen anglické abecedy
- 10 cifer
- nějaká interpunkční znaménka

kapacita kódu je **nedostačující**

Tento problém se řeší tak, že se **většina pětic znaků využívá dvakrát**.

K jejich rozlišení se používá vyhrazená kódová slova - tzv. ***změna registru***

- Pro písmenovou změnu - kódové slovo **11111**
- pro číslicovou změnu - kódové slovo **11011**.

**To umožňuje, abychom mohli zakódovat $2.2^5-2 =$ 62 zdrojových znaků.**

# Tabulka Mezinárodního telegrafní kódu MTA-2

| | Písmena | Číslice | Kód | | Písmena | Číslice | Kód |
|---|---|---|---|---|---|---|---|
| 1. | A | - | 11000 | 17. | Q | 1 | 11101 |
| 2. | B | ? | 10011 | 18. | R | 4 | 01010 |
| 3. | C | : | 01110 | 19. | S | , | 10100 |
| 4. | D | kdo tam? | 10010 | 20. | T | 5 | 00001 |
| 5. | E | 3 | 10000 | 21. | U | 7 | 11100 |
| 6. | F | “ | 10110 | 22. | V | ů | 01111 |
| 7. | G | ˇ | 01011 | 23. | W | 2 | 11001 |
| 8. | H | ´ | 00101 | 24. | X | / | 10111 |
| 9. | I | 8 | 01100 | 25. | Y | 6 | 10101 |
| 10. | J | zvonek | 11010 | 26. | Z | + | 10001 |
| 11. | K | | 11110 | 27. | návrat vozíku | | 00010 |
| 12. | L | | 01001 | 28. | posun o řádek | | 01000 |
| 13. | M | . | 00111 | 29. | písmenová změna | | 11111 |
| 14. | N | , | 00110 | 30. | číslicová změna | | 11011 |
| 15. | O | 9 | 00011 | 31. | mezera | | 00100 |
| 16. | P | 0 | 01101 | 32. | nepoužito | | 00000 |

# Příklad:

Pomocí tabulky mezinárodního telegrafního kódu MTA-2 zakódujte zprávu „MS DOS 7.0“

| | Písmena | Číslice | Kód | | Písmena | Číslice | Kód |
|---|---|---|---|---|---|---|---|
| 1. | A | - | 11000 | 17. | Q | 1 | 11101 |
| 2. | B | ? | 10011 | 18. | R | 4 | 01010 |
| 3. | C | : | 01110 | 19. | S | , | 10100 |
| 4. | D | kdo tam? | 10010 | 20. | T | 5 | 00001 |
| 5. | E | 3 | 10000 | 21. | U | 7 | 11100 |
| 6. | F | “ | 10110 | 22. | V | ů | 01111 |
| 7. | G | ˇ | 01011 | 23. | W | 2 | 11001 |
| 8. | H | ´ | 00101 | 24. | X | / | 10111 |
| 9. | I | 8 | 01100 | 25. | Y | 6 | 10101 |
| 10. | J | zvonek | 11010 | 26. | Z | + | 10001 |
| 11. | K | | 11110 | 27. | návrat vozíku | | 00010 |
| 12. | L | | 01001 | 28. | posun o řádek | | 01000 |
| 13. | M | . | 00111 | 29. | písmenová změna | | 11111 |
| 14. | N | , | 00110 | 30. | číslicová změna | | 11011 |
| 15. | O | 9 | 00011 | 31. | mezera | | 00100 |
| 16. | P | 0 | 01101 | 32. | nepoužito | | 00000 |

| 11111 | 00111 | 10100 | 00100 | 10010 | 00011 | 10100 | 00100 | 11011 | 11100 | 00111 | 01101 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Písmenová změna | M | S | Mezera | D | O | S | Mezera | Číslicová změna | 7 | . | 0 |

## Příklad:

Abecedou zdroje je pět stavů výrobního stroje A={Porucha, Seřízení, Volný, Pracující, Blokovaný}. Zakódujte znaky zdroje (stavy stroje) rovnoměrným kódem.

Musíme najít minimální délku kódového slova, abychom mohli použít pět různých kódových slov.

Kódová slova o délce 2 nám stačit nebudou, protože existují pouze $\mathbf{2^2 = 4}$ různé dvojice kódových znaků 0 a 1.

Je tedy třeba použít kódová slova o délce 3. V tomto případě máme k dispozici $\mathbf{2^3 = 8}$ kódových slov, ze kterých 5 využijeme pro kódování jednotlivých stavů stroje. Například takto:

| | | |
|---|---|---|
| **Porucha – 101** | **Seřízení – 010** | **Volný – 000** |
| **Pracující – 111** | **Blokovaný - 110** | |

# ASCII kód (American Standard Code for Information Interchange

- V současné době, době počítačů a počítačových sítí ztrácí dálnopisy význam.
- Telegrafní kód je již dlouho nepostačující současným potřebám přenosu dat.
- V šedesátých letech proto vznikl v USA kód vyvinutý speciálně pro mikropočítače - ***ASCII***

**Zdrojová abeceda** obsahuje kromě písmen anglické abecedy a číslic i znaky používané ve výpočetní technice (malá písmena, algebraické znaky +,–,*,=...).

Každému z $2^7$=128 znaků zdrojové abecedy se přiřadí sedmice kódových znaků 0 a 1.

Zdokonalováním ASCII kódu se vyvinulo mnoho dalších rovnoměrných kódů (např. ISO-7, ISO-8), z ISO-7 se vycházelo při sestavování osmi prvkového kódu **ISO-8.**

- Osmý paritní bit ISO-7 byl nahrazen významovým bitem umožňujícím vyjádřit malá i velká písmena např. azbuky a v české verzi písmena s háčky a čárkami.

| DEC | BIN | ZNAK (Význam) | DEC | BIN | ZNAK (Význam) | DEC | BIN | ZNAK (Význam) | DEC | BIN | ZNAK (Význam) | DEC | BIN | ZNAK (Význam) | DEC | BIN | ZNAK (Význam) | DEC | BIN | ZNAK (Význam) | DEC | BIN | ZNAK (Význam) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 00000000 | null | 32 | 00100000 | mezera | 64 | 01000000 | @ | 96 | 01100000 | ` | 128 | 10000000 | Ä | 160 | 10100000 | † | 192 | 11000000 | ņ | 224 | 11100000 | ŗ |
| 1 | 00000001 | Start of Header | 33 | 00100001 | ! | 65 | 01000001 | A | 97 | 01100001 | a | 129 | 10000001 | Ā | 161 | 10100001 | ° | 193 | 11000001 | Ń | 225 | 11100001 | Š |
| 2 | 00000010 | Start of Text | 34 | 00100010 | " | 66 | 01000010 | B | 98 | 01100010 | b | 130 | 10000010 | ā | 162 | 10100010 | Ę | 194 | 11000010 | ¬ | 226 | 11100010 | ‚ |
| 3 | 00000011 | End of Text | 35 | 00100011 | # | 67 | 01000011 | C | 99 | 01100011 | c | 131 | 10000011 | É | 163 | 10100011 | £ | 195 | 11000011 | √ | 227 | 11100011 | „ |
| 4 | 00000100 | End of Transmission | 36 | 00100100 | $ | 68 | 01000100 | D | 100 | 01100100 | d | 132 | 10000100 | Ą | 164 | 10100100 | § | 196 | 11000100 | ń | 228 | 11100100 | š |
| 5 | 00000101 | Enquiry | 37 | 00100101 | % | 69 | 01000101 | E | 101 | 01100101 | e | 133 | 10000101 | Ö | 165 | 10100101 | • | 197 | 11000101 | Ň | 229 | 11100101 | Ś |
| 6 | 00000110 | Acknowledge | 38 | 00100110 | & | 70 | 01000110 | F | 102 | 01100110 | f | 134 | 10000110 | Ü | 166 | 10100110 | ¶ | 198 | 11000110 | Δ | 230 | 11100110 | ś |
| 7 | 00000111 | Bell | 39 | 00100111 | ' | 71 | 01000111 | G | 103 | 01100111 | g | 135 | 10000111 | á | 167 | 10100111 | ß | 199 | 11000111 | « | 231 | 11100111 | Á |
| 8 | 00001000 | Backspace | 40 | 00101000 | ( | 72 | 01001000 | H | 104 | 01101000 | h | 136 | 10001000 | ą | 168 | 10101000 | ® | 200 | 11001000 | » | 232 | 11101000 | Ť |
| 9 | 00001001 | Horizontal Tab | 41 | 00101001 | ) | 73 | 01001001 | I | 105 | 01101001 | i | 137 | 10001001 | Č | 169 | 10101001 | © | 201 | 11001001 | … | 233 | 11101001 | ť |
| 10 | 00001010 | Line feed | 42 | 00101010 | * | 74 | 01001010 | J | 106 | 01101010 | j | 138 | 10001010 | ä | 170 | 10101010 | ™ | 202 | 11001010 |  | 234 | 11101010 | Í |
| 11 | 00001011 | Vertical Tab | 43 | 00101011 | + | 75 | 01001011 | K | 107 | 01101011 | k | 139 | 10001011 | č | 171 | 10101011 | ę | 203 | 11001011 | ň | 235 | 11101011 | Ž |
| 12 | 00001100 | Form Feed | 44 | 00101100 | , | 76 | 01001100 | L | 108 | 01101100 | l | 140 | 10001100 | Ć | 172 | 10101100 | ¨ | 204 | 11001100 | Ő | 236 | 11101100 | ž |
| 13 | 00001101 | Carriage return | 45 | 00101101 | - | 77 | 01001101 | M | 109 | 01101101 | m | 141 | 10001101 | ć | 173 | 10101101 | ≠ | 205 | 11001101 | Õ | 237 | 11101101 | Ū |
| 14 | 00001110 | Shift Out | 46 | 00101110 | . | 78 | 01001110 | N | 110 | 01101110 | n | 142 | 10001110 | é | 174 | 10101110 | Ģ | 206 | 11001110 | ő | 238 | 11101110 | Ó |
| 15 | 00001111 | Shift In | 47 | 00101111 | / | 79 | 01001111 | O | 111 | 01101111 | o | 143 | 10001111 | Ź | 175 | 10101111 | Į | 207 | 11001111 | Ō | 239 | 11101111 | Ô |
| 16 | 00010000 | Data Link Escape | 48 | 00110000 | 0 | 80 | 01010000 | P | 112 | 01110000 | p | 144 | 10010000 | ź | 176 | 10110000 | į | 208 | 11010000 | – | 240 | 11110000 | ū |
| 17 | 00010001 | Device control 1 | 49 | 00110001 | 1 | 81 | 01010001 | Q | 113 | 01110001 | q | 145 | 10010001 | Ď | 177 | 10110001 | Ī | 209 | 11010001 | — | 241 | 11110001 | Ů |
| 18 | 00010010 | Device control 2 | 50 | 00110010 | 2 | 82 | 01010010 | R | 114 | 01110010 | r | 146 | 10010010 | í | 178 | 10110010 | ≤ | 210 | 11010010 | “ | 242 | 11110010 | Ú |
| 19 | 00010011 | Device control 3 | 51 | 00110011 | 3 | 83 | 01010011 | S | 115 | 01110011 | s | 147 | 10010011 | ď | 179 | 10110011 | ≥ | 211 | 11010011 | ” | 243 | 11110011 | ů |
| 20 | 00010100 | Device control 4 | 52 | 00110100 | 4 | 84 | 01010100 | T | 116 | 01110100 | t | 148 | 10010100 | Ē | 180 | 10110100 | ī | 212 | 11010100 | ‘ | 244 | 11110100 | Ű |
| 21 | 00010101 | Negative Acknowledge | 53 | 00110101 | 5 | 85 | 01010101 | U | 117 | 01110101 | u | 149 | 10010101 | ē | 181 | 10110101 | Ķ | 213 | 11010101 | ’ | 245 | 11110101 | ű |
| 22 | 00010110 | Synchronous Idle | 54 | 00110110 | 6 | 86 | 01010110 | V | 118 | 01110110 | v | 150 | 10010110 | Ė | 182 | 10110110 | ∂ | 214 | 11010110 | ÷ | 246 | 11110110 | Ų |
| 23 | 00010111 | End of Transmission Block | 55 | 00110111 | 7 | 87 | 01010111 | W | 119 | 01110111 | w | 151 | 10010111 | ó | 183 | 10110111 | Σ | 215 | 11010111 | ◊ | 247 | 11110111 | ų |
| 24 | 00011000 | Cancel | 56 | 00111000 | 8 | 88 | 01011000 | X | 120 | 01111000 | x | 152 | 10011000 | ė | 184 | 10111000 | ł | 216 | 11011000 | ō | 248 | 11111000 | Ý |
| 25 | 00011001 | End of Medium | 57 | 00111001 | 9 | 89 | 01011001 | Y | 121 | 01111001 | y | 153 | 10011001 | ô | 185 | 10111001 | Ļ | 217 | 11011001 | Ŕ | 249 | 11111001 | ý |
| 26 | 00011010 | Substitute | 58 | 00111010 | : | 90 | 01011010 | Z | 122 | 01111010 | z | 154 | 10011010 | ö | 186 | 10111010 | ļ | 218 | 11011010 | ŕ | 250 | 11111010 | ķ |
| 27 | 00011011 | Escape | 59 | 00111011 | ; | 91 | 01011011 | [ | 123 | 01111011 | { | 155 | 10011011 | õ | 187 | 10111011 | Ľ | 219 | 11011011 | Ř | 251 | 11111011 | Ż |
| 28 | 00011100 | File Separator | 60 | 00111100 | < | 92 | 01011100 | \ | 124 | 01111100 | \| | 156 | 10011100 | ú | 188 | 10111100 | ľ | 220 | 11011100 | ‹ | 252 | 11111100 | Ł |
| 29 | 00011101 | Group Separator | 61 | 00111101 | = | 93 | 01011101 | ] | 125 | 01111101 | } | 157 | 10011101 | Ě | 189 | 10111101 | Ĺ | 221 | 11011101 | › | 253 | 11111101 | ż |
| 30 | 00011110 | Record Separator | 62 | 00111110 | > | 94 | 01011110 | ^ | 126 | 01111110 | ~ | 158 | 10011110 | ě | 190 | 10111110 | Í | 222 | 11011110 | ř | 254 | 11111110 | Ģ |
| 31 | 00011111 | Unit Separator | 63 | 00111111 | ? | 95 | 01011111 | _ | 127 | 01111111 | • | 159 | 10011111 | ü | 191 | 10111111 | Ņ | 223 | 11011111 | Ŗ | 255 | 11111111 | ˇ |

# Příklad:

Pomocí tabulky ASCII kódu zakódujte zprávu „FAI – Zlín 2006“

| F | A | I | |
|---|---|---|---|
| 01000110 | 01000001 | 01001001 | |
| Z | l | Í | n |
| 01011010 | 01101100 | 10010010 | 01101110 |
| 2 | 0 | 0 | 6 |
| 00110010 | 00110000 | 00110000 | 00110110 |

# Nerovnoměrné kódy

Nerovnoměrné kódy jsou takové kódy, kdy se jednotlivým znakům zdrojové abecedy přiřazují kódová slova, která mají **různý počet znaků**.

Nejznámějším a nejrozšířenějším **Morseův kód -** ***Morseova abeceda [8, 7].***

- k písmenovým symbolům (zdrojová abeceda) přiřazeny elektrické impulsy dvou délek – prezentovány znaky **.**(tečka) a **-** (čárka) .
- Vedle elektrického signálu lze použít i akustický (píšťalka,..), optický (světlo, praporky..) atd.
- Charakteristickou vlastností tohoto kódu je, že zohledňuje pravděpodobnostní strukturu zdrojové abecedy.
- To znamená, že znakům s vyšší frekvencí výskytu je přiřazeno kratší kódové slovo a naopak znakům s nižší frekvencí výskytu delší kódové slovo.
- Rychlost komunikace se pohybuje od 60 do 250 znaků za minutu

# Morseův kód – Morseova abeceda

Přestože se kód prezentuje v podobě teček a čárek **není binární**

Jedná se o *ternárni´kód*

- *krátký impuls (tečka)*
- *dlouhý impuls (čárka)*
- *mezera* - ve vysílání jakoby „neslyšíme“, ale je nezbytně nutný ke správnému dekódování kódu
- Kódovou abecedou je tedy: **y= {krátký impuls, dlouhý impuls, mezera}**

Obecně v tomto kódu nebyla stanovena maximální délka slova nebo dokonce počet jednotlivých prvků v kódovém slově.

Pro kvalitní a bezchybný příjem zpráv je třeba dodržet pravidla vysílání:

1. Určující jednotkou času je délka tečky
2. Jedna čárka trvá stejně jako tři tečky
3. Pauza v rámci jednoho znaku má stejnou délku jako jedna tečka
4. Pauza mezi jednotlivými znaky má stejnou délku jako jedna čárka

Kódovou abecedou je tedy: **y= {impuls o délce *t*, impuls o délce *3t*, mezera o délce *t*}**

# Morseův kód – Morseova abeceda

Pro kvalitní a bezchybný příjem zpráv je třeba dodržet pravidla vysílání:

1. **Určující jednotkou času je délka tečky**
2. **Jedna čárka trvá stejně jako tři tečky**
3. **Pauza v rámci jednoho znaku má stejnou délku jako jedna tečka**
4. **Pauza mezi jednotlivými znaky má stejnou délku jako jedna čárka**

Kódovou abecedou může tedy být : $y = \{y_1, y_2, y_3\}$

kde:

**$y_1$ – impuls o délce 1 časové jednotky + mezera (pausa) o délce 1 časové jednotky**

**$y_2$ – impuls o délce 3 časových jednotek + mezera (pausa) o délce 1 časové jednotky**

**$y_3$ – mezera (pausa) o délce 3 časových jednotek**

# TABULKA MORSEOVY ABECEDY

| Písmena | Kód | Písmena | Kód | Písmena | Kód | Písmena | Kód |
|---|---|---|---|---|---|---|---|
| A | .- | K | -.- | U | ..- | 5 | ..... |
| B | -... | L | .-.. | V | ...- | 6 | -.... |
| C | -.-. | M | -- | W | .-- | 7 | --... |
| D | -.. | N | -. | X | -..- | 8 | ---.. |
| E | . | O | --- | Y | -.-- | 9 | ----. |
| F | ..-. | P | .--. | Z | --.. | 0 | ----- |
| G | --. | Q | --.- | 1 | .---- | pozor | -.-.-. |
| H | .... | R | .-. | 2 | ..--- | nerozumím | ........ |
| I | .. | S | ... | 3 | ...-- | konec | ...-.- |
| J | .--- | T | - | 4 | ....- | rozumím | ----.- |

Příklad:

**Pomocí tabulky Morseové abecedy zakódujte zprávu „AHOJ LIDI“**

- **Do Morseovy abecedy**
- **Do kódové abecedy**

**Řešení:**

| A | H | O | J | | L | I | D | I |
|---|---|---|---|---|---|---|---|---|
| . - | . . . . | - - - | . - - - | | . - . . | . . | - . . | . . |
| $y_1 y_2 y_3$ | $y_1 y_1 y_1 y_1 y_3$ | $y_2 y_2 y_2 y_3$ | $y_1 y_2 y_2 y_2 y_3$ | $y_3 y_3$ | $y_1 y_2 y_1 y_1 y_3$ | $y_1 y_1 y_3$ | $y_2 y_1 y_1 y_3$ | $y_1 y_1 y_3$ |

**ZDE JE MOŽNO SI VYZKOUŠET MORSEOVU ABECEDU**

# Konstrukce nerovnoměrných kódů

Ke konstrukci nerovnoměrných kódů se využívají **grafy.**

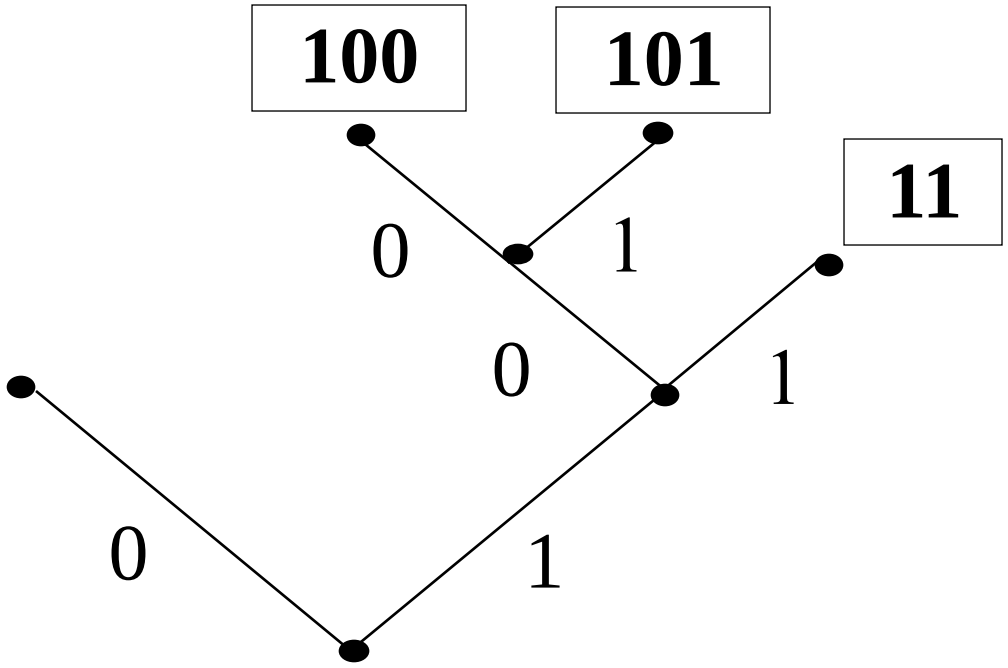
- V teorii grafů se takovýto graf nazývá ***strom.***

Strom obsahuje ***uzly*** a z každého uzlu stromu vycházejí dvě ***hrany.***

Posloupnost za sebou následujících hran se nazývá ***cesta.***

Po dohodě, že při pohybu po stromě od jednoho uzlu k druhému přiřadíme:

- **směrem doprava 1**
- **směrem doleva 0**



každé cestě začínající v kořeni stromu odpovídá nějaké kódové slovo a každému kódovému slovu je jednoznačně přiřazena nějaká cesta začínající v kořeni stromu.

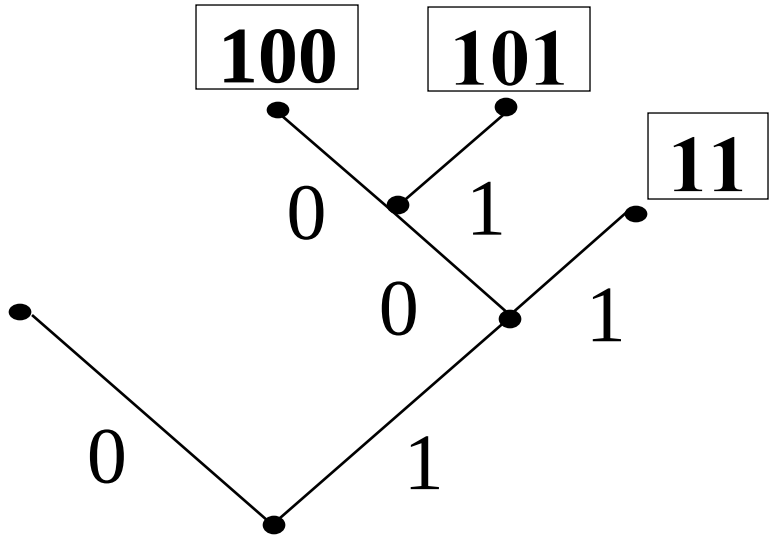
**Platí:**

Přiřadíme-li zdrojovému znaku $a_i$ nějakou cestu začínající v kořeni stromu, pak žádné jiné zdrojové slovo $a_j \neq a_i$ nemůže mít přiřazenou cestu, která by obsahovala celou cestu přiřazenou znaku $a_i$.

Toto pravidlo se označuje jako ***vlastnost předpony - P*** (*prefix property*) a takovéto kódy se nazývají ***P-kódy***.

Tyto kódy jsou jednoznačně **dekódovatelné!**

Jejich dekódování spočívá v postupném ověřování patří-li přijaté kódové slovo (nebo jeho část) do kódu, nebo ne.



# Dekódovatelnost prefixových kódů [1]

Při konstrukci prefixových kódů stačí znát délky kódových slov, které chceme použít. Pomocí známých délek lze snadno sestavit prefixový kód.

Délky určuje tzv. **Kraftova nerovnost**

Uvažujme, že při binárním kódování použijeme pro zdrojové znaky $\{a_1, a_2,\ldots,a_N\}$ kódová slova s délkami $\{d_1, d_2,\ldots,d_N\}$.

Prefixový kód lze sestavit právě tehdy když platí Kraftova nerovnost:

$$2^{-d_1} + 2^{-d_2} \ldots 2^{-d_n} \leq 1$$

$$\sum_{i=1}^{N} 2^{-d_i} \leq 1$$

***Mc Millan*** (1956) dokázal, že pro každý jednoznačně dekódovatelný kód platí Kraftova nerovnost – McMillanova věta

## Příklad:

Sestrojte prefixový kód pro zdrojovou abecedu, která obsahuje 7 písmen, jsou-li dány délky jednotlivých kódových slov takto: 2, 2, 3, 3, 4, 5, 5.

***Řešení:***

1) Zjistíme, zda lze takový jednoznačně dekódovatelný kód sestrojit.

Využijeme Mc Millanovu větu resp. Kraftovu nerovnost.

Dostáváme tedy:

$$\sum_{i=1}^{N} 2^{-d_i} \leq 1$$

$$2^{-2} + 2^{-2} + 2^{-3} + 2^{-3} + 2^{-4} + 2^{-5} + 2^{-5} \leq 1$$

$$0.25 + 0.25 + 0.125 + 0.125 + 0.125 + 0.0625 + 0.03125 + 0.03125 \leq 1$$

$$0.875 \leq 1$$

➡ **jednoznačně dekódovatelný kód lze sestrojit**

## Příklad:

Sestrojte prefixový kód pro zdrojovou abecedu, která obsahuje 7 písmen, jsou-li dány délky jednotlivých kódových slov takto: 2, 2, 3, 3, 4, 5, 5.

***Řešení:***

2) Kód sestrojíme pomocí binárního stromu tak, že obsadíme libovolně:

**2 uzly na 2. úrovni**

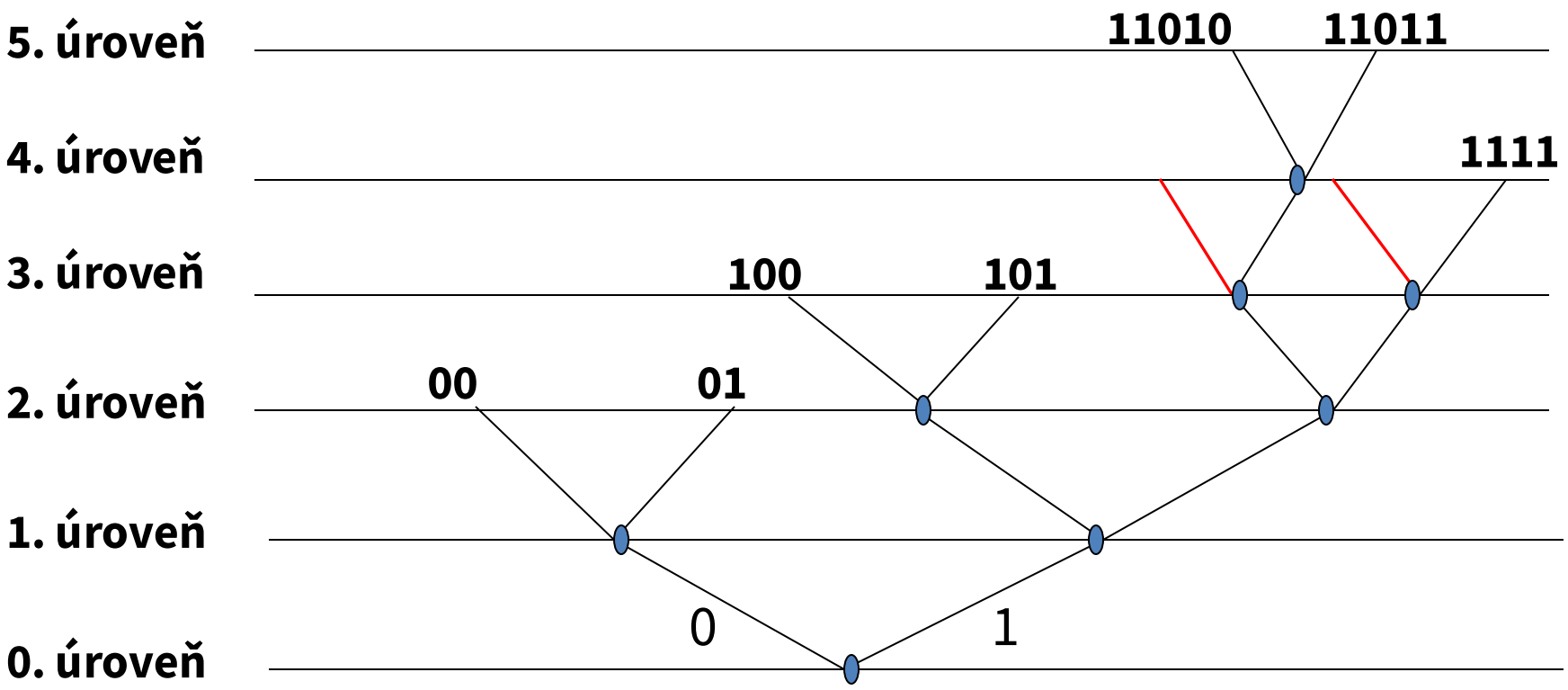
**2 uzly na 3. úrovni**

**1 uzel na 4. úrovni**

**2 uzly na 5. úrovni**

➡ **sestrojíme kódová slova, která budou tvořit jednoznačně dekódovatelný kód**

# Sestrojení jednoho z možných kódů pomocí binárního stromu




**Výsledný kód je { 00, 01, 100, 101, 1111, 11010, 11011 }.**

Doposud jsme uvažovali pouze sestavení jednoznačně dekódovatelného kódu dle známých délek kódových slov.

- Neuvažovali jsme frekvenci výskytu zdrojových znaků

**Cíl:**

- častějším zdrojovým znakům přiřadit kratší kódová slova a naopak
- Najít nejkratší kód (prefixový kód jehož průměrná délka kódového slova bude co možná nejkratší)

Uvažujme zdrojovou abecedu $\mathbf{A=\{a_1,a_2,\ldots,a_N\}}$. Označme pravděpodobnosti výskytu jednotlivých zdrojových znaků $\mathbf{\{p_1, p_2,\ldots,p_N\}}$ a délky odpovídajících kódových slov $\mathbf{\{d_1, d_2,\ldots,d_N\}}$.

Z teorie pravděpodobnosti pak vyplývá, že **průměrná (střední) délka kódového slova** je dána ve tvaru:

$$\overline{d} = d_1 \cdot p_1 + d_1 \cdot p_1 + \cdots + d_1 \cdot p_1 = \sum_{i=1}^{N} d_i \cdot p_i$$

# Příklad:

Uvažujme kód sestavený dle předchozího příkladu a pravděpodobnosti jednotlivých zdrojových znaků včetně přiřazení k jednotlivým kódovým slovům dle uvedené tabulky. Vypočítejte kolik kódových znaků musíme vyslat, chceme-li poslat zprávu obsahující 1000 zdrojových znaků.

| Zdrojový znak | $a_1$ | $a_2$ | $a_3$ | $a_4$ | $a_5$ | $a_6$ | $a_7$ |
|---|---|---|---|---|---|---|---|
| Pravděpodobnost výskytu | **0.25** | **0.05** | **0.2** | **0.15** | **0.05** | **0.1** | **0.2** |
| Kódové slovo | **00** | **01** | **100** | **101** | **1111** | **11010** | **11011** |
| Délka kódového slovo | **2** | **2** | **3** | **3** | **4** | **5** | **5** |

$$\overline{d} = \sum_{i=1}^{N} d_i \cdot p_i = (2 \cdot 0.25 + 2 \cdot 0.05 + 3 \cdot 0.2 + 3 \cdot 0.15 + 4 \cdot 0.05 + 5 \cdot 0.1 + 5 \cdot 0.2 = 3.35$$

➠ na jeden zdrojový znak připadne **3.35** kódového znaku

- **průměrná délka kódového slova.**

**Při vysílání 1000 zdrojových znaků musíme vyslat 3350 kódových znaků.**

# Příklad:

Uvažujme stejný příklad jako předchozí. Přiřaďme tentokrát krátká kódová slova častějším zdrojovým znakům. Vypočítejte kolik kódových znaků musíme vyslat, chceme-li poslat zprávu obsahující 1000 zdrojových znaků.

| Zdrojový znak | $a_1$ | $a_2$ | $a_3$ | $a_4$ | $a_5$ | $a_6$ | $a_7$ |
|---|---|---|---|---|---|---|---|
| Pravděpodobnost výskytu | **0.25** | **0.05** | **0.2** | **0.15** | **0.05** | **0.1** | **0.2** |
| Kódové slovo | **00** | **11011** | **01** | **100** | **11010** | **1111** | **101** |
| Délka kódového slovo | **2** | **5** | **2** | **3** | **5** | **4** | **3** |

$$\overline{d} = \sum_{i=1}^{N} d_i \cdot p_i = (2 \cdot 0.25 + 5 \cdot 0.05 + 2 \cdot 0.2 + 3 \cdot 0.15 + 5 \cdot 0.05 + 4 \cdot 0.1 + 3 \cdot 0.2 = 2.85$$

➠ na jeden zdrojový znak připadne **2.85** kódového znaku

- **průměrná délka kódového slova.**

**Při vysílání 1000 zdrojových znaků musíme vyslat 2850 kódových znaků.**

# EFEKTIVNÍ KÓDY

U běžného kódování známe pravděpodobnosti výskytu znaku $a_i$ zdrojové abecedy $P(a_i) = p_i$ ale neznáme délky kódových slov $d_i$.

Musíme proto hledat délky kódových slov tak, aby průměrný počet kódových znaků připadajících na jeden zdrojový znak byl minimální.

- **takové kódy nazýváme** ***EFEKTIVNÍ nebo MINIMÁLNÍ.***

V minulosti se objevilo mnoho algoritmů pro nalezení vhodné délky kódových slov, tak aby kód byl jednoznačně dekódovatelný a co možná nejkratší – co nejmenší průměrná délka kódového slova.

Prvotní myšlenka – algoritmus:

- Pro nezávislé vysílání znaků zdrojové abecedy $A=\{a_1, a_2, \dots, a_N\}$ s pravděpodobností jednotlivých znaků $P(a_i) = p_i, i=1, \dots, N,$ volíme délku kódového slova $d_i$ tak, že platí:

$$2^{-d_i} \leq p_i < 2^{-d_i+1}$$

## Příklad:

Zjistěte délku kódových slov pro znaky zdrojové abecedy s pravděpodobnostmi dle předchozího příkladu. Tzn. Zdrojová abeceda má 7 znaků s pravděpodobnostmi výskytu **0.25, 0.05, 0.2, 0.15, 0.05, 0.1, 0.2.**

Dle předchozího pravidla musí platit:

$2^{-\mathbf{2}} \leq 0.25 < 2^{-1}$ $\quad 2^{-\mathbf{5}} \leq 0.05 < 2^{-4}$ $\quad 2^{-\mathbf{3}} \leq 0.2 < 2^{-2}$

$2^{-\mathbf{3}} \leq 0.15 < 2^{-2}$ $\quad 2^{-\mathbf{5}} \leq 0.05 < 2^{-4}$ $\quad 2^{-\mathbf{4}} \leq 0.1 < 2^{-3}$

$2^{-\mathbf{3}} \leq 0.2 < 2^{-2}$

Délky kódových slov jsou: 2, 3, 3, 3, 4, 5, 5.

$$\bar{d} = \sum_{i=1}^{N} d_i \cdot p_i = (2 \cdot 0.25 + 5 \cdot 0.05 + 3 \cdot 0.2 + 3 \cdot 0.15 + 5 \cdot 0.05 + 4 \cdot 0.1 + 3 \cdot 0.2 = 3.05$$

**⇒ tento algoritmus není optimální (hledá pouze jednoznačně dekódovatelný kód)**

## Příklad:

Zjistěte délku kódových slov pro znaky zdrojové abecedy, která má 7 znaků s pravděpodobnostmi výskytu **0.22, 0.2, 0.16, 0.16, 0.1, 0.1, 0.04, 0.02.** Výsledný kód sestrojte pomocí binárního stromu.

Dle předchozího pravidla musí platit:

$$2^{-3} \leq 0.22 < 2^{-2} \qquad 2^{-3} \leq 0.2 < 2^{-2} \qquad 2^{-3} \leq 0.16 < 2^{-2}$$

$$2^{-3} \leq 0.16 < 2^{-2} \qquad 2^{-4} \leq 0.1 < 2^{-3} \qquad 2^{-4} \leq 0.1 < 2^{-3}$$

$$2^{-5} \leq 0.04 < 2^{-4} \qquad 2^{-6} \leq 0.02 < 2^{-5}$$

Délky kódových slov jsou: 3, 3, 3, 3, 4, 4, 5, 6.

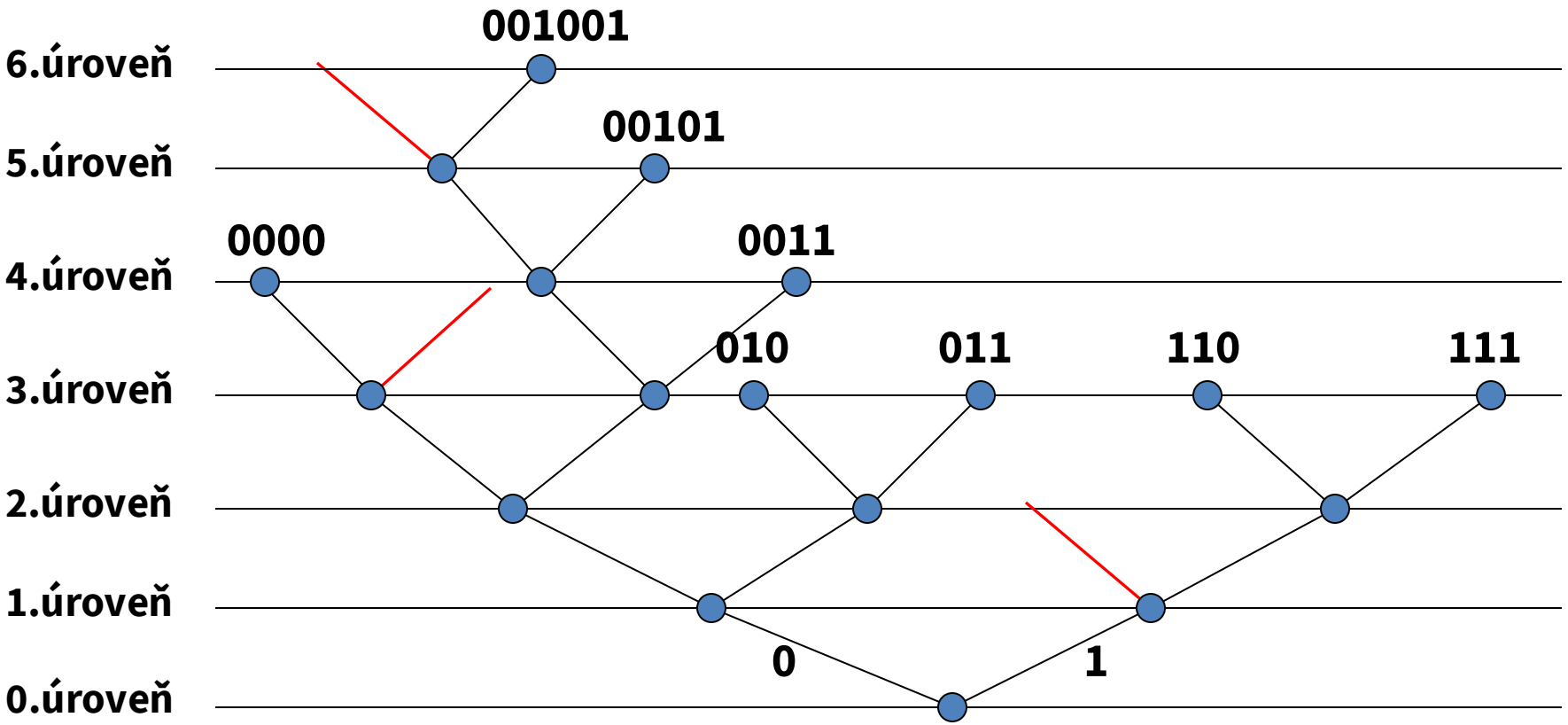
Konstrukce kódu je stejná jako u nerovnoměrných kódů – použijeme binární strom (viz následující snímek)

$$\bar{d} = \sum_{i=1}^{N} d_i \cdot p_i =$$

$$= (3 \cdot 0.22 + 3 \cdot 0.2 + 3 \cdot 0.16 + 3 \cdot 0.16 + 4 \cdot 0.1 + 4 \cdot 0.1 + 5 \cdot 0.04 + 6 \cdot 0.02) = 3.34$$

Je zřejmé, že použitý algoritmus nevede ke konstrukci efektivního kódu

- **ve stromě se nacházejí neobsazené volné uzly**



Abychom mohli porovnat jednotlivé návrhy kódů z hlediska úspornosti, byly zavedeny tyto proměnné:

**Průměrná délka kódového slova:** $$\overline{d} = \sum_{i=1}^{N} d_i \cdot p_i$$

**Efektivnost kódu:** $$\eta = \frac{H}{\overline{d}} \cdot 100\% = \frac{\sum_{i=1}^{N} p_i \cdot log_2 p_i}{\sum_{i=1}^{N} d_i \cdot p_i} \cdot 100\%$$

**Pro předcházející příklad:**

Průměrná délka kódového slova:

$$\overline{d} = \sum_{i=1}^{N} d_i \cdot p_i = (3 \cdot 0.22 + 3 \cdot 0.2 + 3 \cdot 0.16 + 3 \cdot 0.16 + 4 \cdot 0.1 + 4 \cdot 0.1 + 5 \cdot 0.04 + 6 \cdot 0.02) = \mathbf{3.34}$$

Efektivnost kódu $$\eta = \frac{H}{\overline{d}} \cdot 100\% = \frac{2.754}{3.34} \cdot 100\% = 82.5\%$$

# Shannon-Fanova metoda návrhu efektivního kódu [2]

- Shannon-Fanovo kódování je technika pro sestavení **prefixového kódu** založená na seznamu symbolů a počtech jejich výskytů (případně pravděpodobnostech).
- Metodu nezávisle na sobě publikovali v roce 1949 Claude Elwood Shannon s Warrenem Weaverem a Robert Mario Fano.
- Shannon-Fanův algoritmus lze využít i pro kompresi dat
- Je to metoda
  - statistická,
  - dvouprůchodová,
  - semiadaptivní
  - asymetrická.
- Narozdíl od Huffmanova kódování není optimalita výsledného kódu zaručena.

- Ke konstrukci je třeba použít určitý postup – **algoritmus** (viz následující snímek)

# Shannonova-Fanova metoda - algoritmus

1. Zdrojové znaky se uspořádají postupně podle pravděpodobnosti $p_i$ v klesajícím pořadí.
2. Zdrojové znaky se rozdělí na dvě podskupiny tak, aby součet pravděpodobností v obou skupinách byl přibližně stejný.
3. První skupině se přiřadí kódový znak **1** a druhé znak **0**.
4. Každou podskupinu opět rozdělíme na dvě podskupiny s přibližně stejným součtem pravděpodobností.
5. Prvním podskupinám se přiřadí znak **1** a druhým **0**. Tak získáme druhý kódový znak.
6. V dělení na skupiny pokračujeme dokud v každé podskupině nezůstane pouze jeden zdrojový znak.

## Příklad:

Zjistěte délku kódových slov pomocí Shannonovy – Fanovy metody na základě předcházejícího zadání.

**a)**

| znaky | P(ai) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| a5 | 0,22 | 0,58 | 1 | 0,22 | 1 | | | | | 11 |
| a2 | 0,20 | | 1 | 0,36 | 0 | 0,20 | 1 | | | 101 |
| a8 | 0,02 | | 1 | | 0 | 0,16 | 0 | | | 100 |
| a4 | 0,16 | 0,42 | 0 | 0,26 | 1 | 0,16 | 1 | | | 011 |
| a3 | 0,10 | | 0 | | 1 | 0,10 | 0 | | | 010 |
| a6 | 0,10 | | 0 | 0,16 | 0 | 0,10 | 1 | | | 001 |
| a7 | 0,04 | | 0 | | 0 | 0,06 | 0 | 0,04 | 1 | 0001 |
| a8 | 0,02 | | 0 | | 0 | | 0 | 0,02 | 0 | 0000 |

$d = 2, 3, 3, 3, 3, 3, 4, 4$ $\quad \overline{d} = 2.84 \quad \eta = 96.97\ \%$

**b)**

| znaky | P(ai) | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| a1 | 0,22 | | 1 | 0,22 | 1 | | | | | | | | | | | 11 |
| a2 | 0,20 | 0,42 | 1 | 0,20 | 0 | | | | | | | | | | | 10 |
| a3 | 0,16 | | 0 | | 1 | 0,16 | 1 | | | | | | | | | 011 |
| a4 | 0,16 | | 0 | 0,32 | 1 | 0,16 | 0 | | | | | | | | | 010 |
| a5 | 0,10 | | 0 | | 0 | 0,10 | 1 | | | | | | | | | 001 |
| a6 | 0,10 | 0,58 | 0 | | 0 | | 0 | 0,10 | 1 | | | | | | | 0001 |
| a7 | 0,04 | | 0 | 0,26 | 0 | 0,16 | 0 | | 0 | 0,04 | 1 | | | | | 00001 |
| a8 | 0,02 | | 0 | | 0 | | 0 | 0,06 | 0 | 0,02 | 0 | | | | | 00000 |

$d = 2, 2, 3, 3, 3, 4, 5, 5$ $\quad \overline{d} = 2.8 \quad \eta = 98.36\ \%$

# Shannonova-Fanova - zhodnocení

- Na řešení má vliv **subjektivní** rozhodnutí řešitele.
- Tato metoda tedy **nedává jednoznačný výsledek**, ani z hlediska délky kódových slov, ani z hlediska efektivnosti.
- Pokud lze soubor pravděpodobností rozdělit dvojím způsobem volíme rozdělení tak abychom dostali dvě skupiny s **rozdílnějším** počtem členů
    - **Toto vede k efektivnímu řešení**

# Huffmanova metoda návrhu efektivního kódu [1, 2, 7]

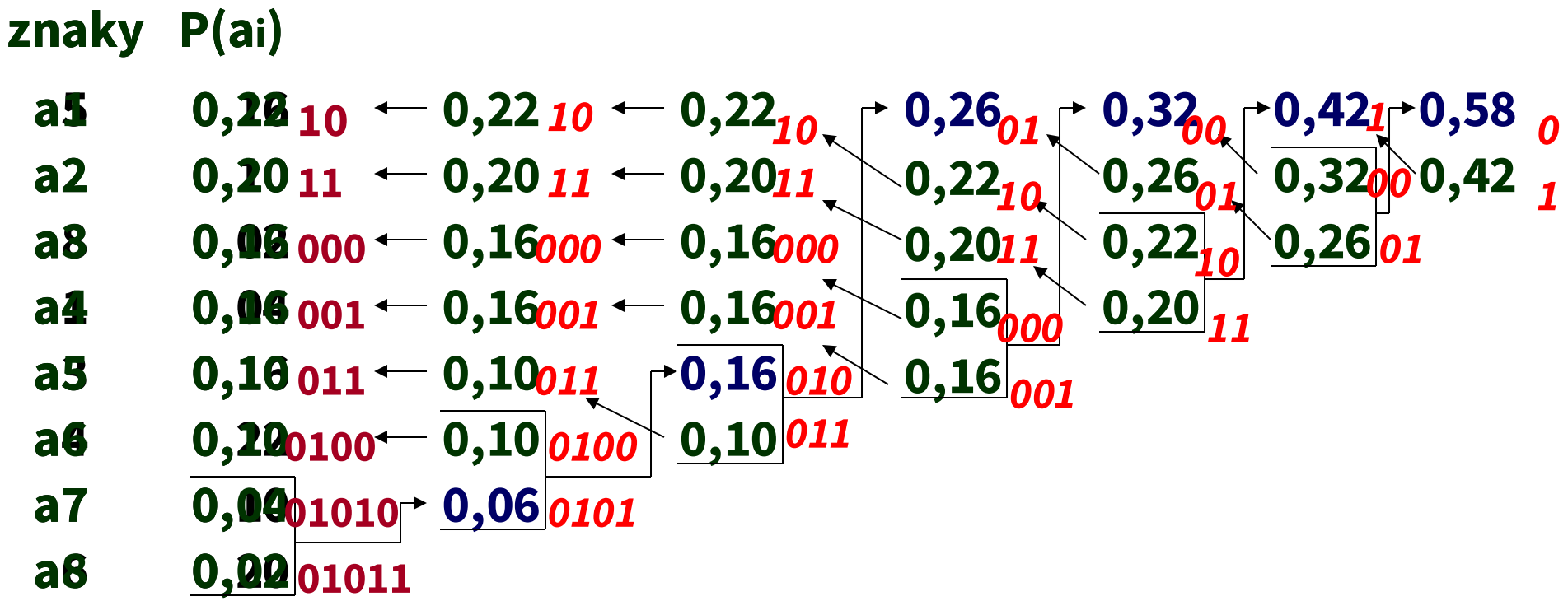
- Tato matematická metoda konstrukce minimálního kódu je složitější než předchozí metoda, **vede ale vždy ke kódu s maximální efektivnosti**.
- **David Huffman** objevil algoritmus v roce 1952. (Chtěl se vyhnout zkoušce tak se snažil vyřešit problém zadaný učitelem.)
- **Hoffmanův algoritmus kódování** je jedním z klasických algoritmů užívaných pro bezeztrátovou kompresi dat (bezeztrátové komprese PKZIP, BZIP2, PBZIP2 aj., v jistém kroku jinak ztrátové komprese JPEG, MP3).
- Prakticky existují **dva typy** Huffmanova kódování
  - *Kódování statické*
  - *Kódování adaptivní* (dynamické).
- Dále se budeme věnovat statické variantě, stěžejní myšlenka se nejdříve objevila v Shannon-Fanově kódu
- Ke konstrukci je třeba použít určitý postup – **algoritmus** (viz následující snímek)

# Huffmanova metoda - algoritmus

1. Zdrojové znaky se uspořádají postupně podle pravděpodobnosti $p_i$ v klesajícím pořadí.
2. Sečteme poslední dvě pravděpodobnosti a výsledek zařadíme podle velikosti mezi ostatní pravděpodobnosti - ***redukce*.**
3. Znovu sečteme dvě poslední pravděpodobnosti a výsledek opět zařadíme podle velikosti.
4. Sečítání pravděpodobností provádíme tak dlouho, až dojdeme k součtu **1**.
5. Posledním dvěma znakům přiřadíme kódové znaky **0** a **1**.
6. Zpětným postupem přiřazujeme jednotlivým sčítancům vždy kódové znaky **0** a **1**, dokud nepřiřadíme kódové znaky všem zdrojovým znakům.

## Příklad:

Zjistěte délku kódových slov pomocí Huffmanovy metody na základě předcházejícího zadání.




*Výsledný kód: {10, 11, 000, 001, 011, 0100, 01010, 01011}*

*d* = 2, 2, 3, 3, 3, 4, 5, 5

$\bar{d} = 2.8$ $\eta = 98.36\ \%$

# Huffmanova metoda - zhodnocení

Podrobnější úvahou je možné zjistit, že ani Huffmanova konstrukce není jednoznačná.

Pokud má více znaků stejnou pravděpodobnost, je řešení ovlivněno tím, kam se zařadí součty pravděpodobností (redukované znaky).

- jiné zařazení nemá vliv na efektivitu výsledného kódu
- ovlivní pouze pozici jedniček a nul v kódovém slově

## Příklad: (typický zkouškový)

Navrhněte kód pro zdrojovou abecedu, která obsahuje 7 symbolů a pravděpodobnosti výskytu prvních šesti symbolů jsou dány takto:

**0,4 0,2 0,05 0,1 0,15 0,04 ??.**

Použijte metodu Shannon–Fanovu i metodu Huffmanovu. Určete průměrnou délku kódového slova a efektivnost pro navržený kód. Sestrojte binární strom navrženého kódu.