# 2.3 Kompresja informacji

## 1. Wstęp

Kompresja informacji to inaczej kompresja danych. Dokładniej, konkretna metoda kompresji danych to zawsze 2 algorytmy:

- algorytm kompresji danych
- algorytm rekonstrukcji danych

Kompresję dzielimy na 2 kategorie: ***kompresję stratną*** i ***kompresję bezstratną.*** Jeśli dane poddawane kompresji potrafimy odtworzyć, zrekonstruować w dokładnie takiej samej postaci jak dane poddawane kompresji to kompresję nazywamy bezstratną. W przeciwnym razie kompresja jest kompresją stratną.

Rodzaj kompresji zależy również od danych poddawanych kompresji. Tak więc mamy:

- kompresję plików tekstowych
- kompresję plików dźwiękowych (kompresja audio)
- kompresję obrazów statycznych
- kompresję sekwencji video czyli obrazów dynamicznych (obrazów ruchomych)

***Model probabilistyczny generacji danych*** przez źródło danych: W zagadnieniach związanych z kompresją danych bardzo ważny jest model matematyczny generacji danych. Załóżmy, że obiektami kodowanymi są litery (symbole) z pewnego alfabetu $V_1 = \{a_1, a_2, ..., a_n\}$ i znamy prawdopodobieństwo pojawienia się każdej litery tzn. podane są prawdopodobieństwa $P(a_i) = p_i$ dla $i = 1, 2, ..., n$. Jeśli przyjmiemy, że słowo nad alfabetem $V_1 = \{a_1, a_2, ..., a_n\}$ generowane jest przez źródło danych jako ciąg niezależnych $k$ losowań liter (podobnie jak w ciągu $k$ doświadczeń Bernoulliego) to otrzymujemy w naturalny rozkład prawdopodobieństwa na zbiorze $V_1^k$ wszystkich słów o długości $k$.

Ważnym parametrem charakteryzującym źródło danych jest entropia źródła. Jest to liczba definiowana wzorem:

$$H = -\sum_{i=1}^{n} p_i \log_2 p_i$$

Shannon udowodnił, że najlepszym rezultatem (przy kodowaniu binarnym) jaki można uzyskać za pomocą algorytmu kompresji bezstratnej jest taki algorytm kodowania (taki kod binarny $k : V_1 \to \{0,1\}^*$) by średnia bitów wykorzystanych do kodowania liter z alfabetu była równa entropii źródła.

Podstawowy pomysł na kompresję jest następujący. Obiekty kodowane występujące częściej (czyli o większym prawdopodobieństwie wystąpienia) powinny być kodowane krótszym słowem a obiekty występujące rzadziej (o mniejszym prawdopodobieństwie) dłuższym słowem. Takie rozwiązanie wykorzystywane jest w znanym każdemu harcerzowi kodzie zwanym „alfabetem Morse'a". Na przykład litera „e" często pojawiająca się w tekstach angielskich jest kodowana za pomocą jednej kropki (słowo o długości 1 nad alfabetem $V_2 = \{\square, -\}$).

**Jednoznaczna dekodowalność:** Dosyć ważną użyteczną własnością kodu jest jego jednoznaczna dekodowalność. Oto definicja tej własności. Niech $V_1 = \{a_1, a_2, ..., a_n\}$. Kod $k : V_1 \to V_2^*$ nazywamy jednoznacznie dekodowalnym jeżeli każdy ciąg słów kodowych daje się odkodować w dokładnie jeden sposób tzn. nie ma dwóch różnych słów $\alpha = \alpha_1\alpha_2...\alpha_m$, $\beta = \beta_1\beta_2...\beta_m$ nad alfabetem $V_1$ dających (po zakodowaniu każdej litery z $V_1$ i konkatenacji uzyskanych słów kodowych) to samo słowo tzn. nie może być

$$k(\alpha_1)k(\alpha_2)...k(\alpha_m) = k(\beta_1)k(\beta_2)...k(\beta_m)$$

***Prefiks i sufiks***. Niech $V$ będzie dowolnym ustalonym alfabetem oraz $\alpha = a_1a_2...a_n$ słowem nad alfabetem $V$, wówczas każde słowo $a_1a_2...a_k$, gdzie $1 \le k \le n$ nazywamy prefiksem słowa $\alpha = a_1a_2...a_n$ a każde słowo $a_ka_2...a_n$, gdzie $1 \le k \le n$ nazywamy sufiksem słowa $\alpha = a_1a_2...a_n$. Jednocześnie słowo puste jest sufiksem i prefiksem każdego słowa.

***Kod prefiksowy*.** Kod prefiksowy to kod w którym żadne słowo kodowe nie jest prefiksem innego słowa kodowego.

Istota rzeczy jest tu taka. Załóżmy, ze odbieramy słowo kodowe litera po literze. W kodzie prefiksowym patrząc na słowo kodowe nie mamy wątpliwości czy to już koniec słowa kodowego czy być może tylko początkowy fragment innego słowa kodowego.

Kody prefiksowe są w oczywisty sposób jednoznacznie dekodowalne.

Jeśli stosujemy kody o zmiennej długości słowa kodowego to ważnym pojęciem jest średnia długość słowa kodowego. W przypadku słów binarnych średnią długość słowa kodowego nazywamy średnią bitową kodu. Dokładniej, niech źródło danych będzie takie, że $V_1 = \{a_1, a_2, ..., a_n\}$ oraz prawdopodobieństwa $P(a_i) = p_i$ dla $i = 1, 2, ..., n$. Średnią długością kodu $k : V_1 \to V_2^*$ nazywamy liczbę

$$l = \sum_{i=1}^{n} P(a_i)l(k(a_i))$$

gdzie $l(\alpha)$ oznacza długość słowa $\alpha$

**Twierdzenie** (warunek konieczny dekodowalności kodu, nierówność Krafta-McMillana):

Jeśli $K$ jest zbiorem słów kodowych (binarnych) o długościach $l_1, l_2, ..., l_n \in N$ i kod $k : V_1 \to K$ jest jednoznacznie dekodowany to spełniony jest następujący warunek tzw. nierówność Krafta-McMillana

$$\sum_{i=1}^{n} 2^{-l_i} \le 1$$

**Twierdzenie** (warunek wystarczający istnienia kodu prefiksowego):

Jeśli $l_1, l_2, ..., l_n \in N$ spełniają nierówność

$$\sum_{i=1}^{n} 2^{-l_i} \leq 1$$

to można znaleźć dla *n* elementowego alfabetu $V_1 = \{a_1, a_2, ..., a_n\}$ kod prefiksowy $k : V_1 \to \{0,1\}^*$ o długościach słów kodowych wynoszących $l_1, l_2, ..., l_n$.

## 2. Kod Huffmana

Kod Huffmana to kod ale jednocześnie najbardziej znana, powszechnie stosowana, efektywna metoda kompresji danych. W zależności od typu kompresowanego pliku, osiąga się oszczędności w objętości danych od 20% do nawet 90%. Kod Huffmana to kod prefiksowy spełniający następujące warunki:

- obiektom kodowanym występującym częściej (mającym większe prawdopodobieństwo wystąpienia) odpowiadają krótsze słowa niż obiektom występującym rzadziej (mającym mniejsze prawdopodobieństwo wystąpienia);
- dwa najrzadziej występujące (najmniej prawdopodobne) mają słowa tej samej długości.

Będziemy dalej zakładać, że obiektami kodowanymi są litery (symbole) z pewnego alfabetu $V_1 = \{a_1, a_2, ..., a_n\}$ i znamy prawdopodobieństwo pojawienia się każdej litery tzn. podane są prawdopodobieństwa $P(a_i) = p_i$ dla $i = 1, 2, ..., n$. Ponadto używamy jako alfabetu $V_2$ zbioru binarnego $\{0,1\}$.

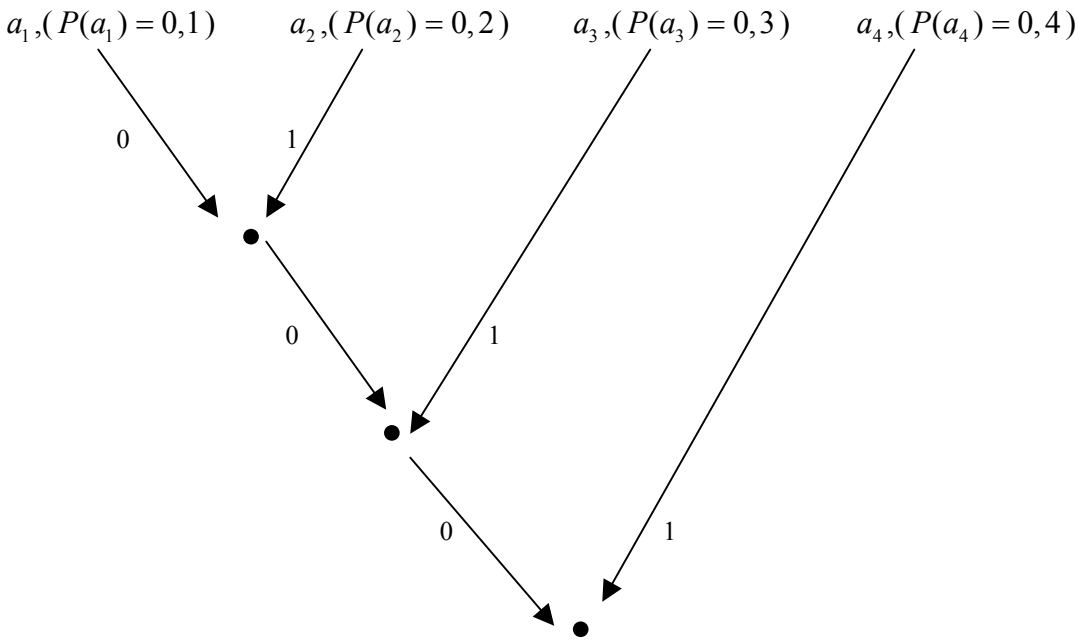
Algorytm kodowania Huffmana czyli *algorytm Huffmana* jest następujący:

- dla każdej litery tworzymy drzewo złożone tylko z korzenia i ustawiamy te drzewa w malejącym porządku prawdopodobieństwa użycia danej litery
- **while** (istnieją przynajmniej 2 drzewa)
  Z drzew $t_1$ i $t_2$ o najmniejszych prawdopodobieństwach $p_1$ i $p_2$ tworzymy nowe drzewo zawierające w korzeniu prawdopodobieństwo $p_1 + p_2$ i mające $t_1$ i $t_2$ jako lewe i prawe poddrzewo. Przypisujemy 0 każdej lewej krawędzi i 1 każdej prawej krawędzi;
- Tworzymy słowo kodowe dla każdej litery przechodząc drzewo od korzenia do liścia zawierającego prawdopodobieństwo stowarzyszone z tą literą i łącząc napotkane 0 i 1;

**Przykład:** Załóżmy, że chcemy zbudować kod Huffmana dla 4 literowego alfabetu $V_1 = \{a_1, a_2, a_3, a_4\}$ przy czym $P(a_1) = 0,1$, $P(a_2) = 0,2$, $P(a_3) = 0,3$, $P(a_4) = 0,4$. Postępują zgodnie z podanym algorytmem Huffmana tworzymy drzewo pokazane na Rys. 1. Odczytany z tego drzewa kod Huffmana jest następujący:

$a_1 \to 000$, $a_2 \to 001$

$a_3 \to 01$, $a_4 \to 1$.




**Rys. 1. Tworzenie kodu Huffmana.**

Metoda kompresji plików tekstowych oparta na wykorzystaniu kodu Huffmana jest bardzo prosta. Kodujemy kolejne litery wchodzące w skład tekstu za pomocą kodu Huffmana i tak uzyskane ciągi konkatenujemy. Warto przy tym przypomnieć, że kod Huffmana jest kodem prefiksowym, a więc jednoznacznie dekodowalnym.

## 3. Algorytmy słownikowe

Jak już wspomnieliśmy w zagadnieniach związanych z kompresją danych bardzo ważny jest model matematyczny generacji danych. Algorytmy słownikowe kompresji należą do metod kompresji wykorzystujących cechy strukturalne występujące w danych. Algorytmy słownikowe dzielimy na dwie grupy:

- algorytmy ze statycznym słownikiem
- algorytmy z dynamicznym słownikiem

Istotę algorytmów słownikowych wyjaśnia następujący przykład. Załóżmy, ze dany jest tekst złożony ze słów 4 literowych i każde słowo składa się z czterech znaków. Alfabet, w którym zapisujemy tekst, składa się z 26 małych liter i znaków przestankowych, co daje w sumie 32 znaki (5 bitów na znak). Jeśli wybierzemy ze słów 4 znakowych najczęściej spotykane słowa (np. 256 słów) i umieścimy je w słowniku, to możemy postępować tak:

Analizujemy słowa 4 znakowe (dzielimy cały tekst kompresowany na 4 znakowe bloki) jeśli dane słowo znajduje się w słowniku to przesyłamy tylko jeden bit stwierdzający fakt znalezienia słowa w słowniku i pozycję słowa w słowniku (8 bitów). W sumie więc 9 bitów. Jeśli nie znaleźliśmy słowa w słowniku to 4 znakowe słowo kodujemy tak: jeden bit stwierdzający że słowo jest spoza słownika oraz 4 słowa 5 bitowe kodujące znaki z naszego alfabetu (w sumie 21 bitów).

Efektywność takiego kodowania będzie zależeć od tego, jaki odsetek słów analizowanych znajduje się słowniku.

Jeśli prawdopodobieństwo wystąpienia wzorca ze słownika jest równe $p$, to średnia liczba bitów poświęcanych na zapis 4 znakowego fragmentu tekstu wyraża się następująco:

$$L = 9p + 21(1 - p) = 21 - 12p$$

Gdybyśmy nie stosowali powyższego algorytmu to każdy 4 znakowy fragment tekstu zabierałby 20 bitów.

Oczywiście w powyższym schemacie kodowania efekt będzie tym lepszy, im większe jest prawdopodobieństwo $p$ znalezienia wzorca w słowniku.

Popularne pakiety kompresji danych takie jak PKZip, Zip, gzip i ARJ używają algorytmów słownikowych opartych na LZ77. Inne znane słownikowe algorytmy kompresji to m.in. algorytmy LZ 77, LZ 78 (litery LZ są tu skrótem od nazwisk Lempel, Ziv), EZW itd. Powszechnie stosowanym algorytmem kompresji jest również algorytm LZW (Lempel-Ziv-Welch) należący do klasy algorytmów typu LZ 78.

Znane z systemu operacyjnego Unix polecenie compress, specyfikacja V.42.bis wykorzystywana w modemach oraz standard GIF (Graphics Interchange Format) są kolejnymi przykładami zastosowania metod słownikowych kompresji.

## 4. Kompresja plików dźwiękowych i video

Oto lista bardziej znanych metod kompresji plików dźwiękowych

- MP3
- MP3 PRO
- OGG VORBIS
- MPEG PLUS
- AAC

Z całą pewnością najczęściej stosowanym standardem kompresji plików dźwiękowych jest algorytm MP3.

Najpopularniejszymi technikami kompresji audio-video (czyli łącznej kompresji plików dźwiękowych i ruchomych obrazów) są różne odmiany standardu MPEG (od MPEG-1 przez MPEG-4 do MPEG-7).

MPEG (Moving Pictures Expert Group) to nazwa grupy naukowców zajmujących się bezpośrednio rozwojem standardu, wchodząca w skład międzynarodowej organizacji standaryzacyjnej ISO/IEC).

Prace nad standardem MPEG rozpoczęły się w 1987 roku, w instytucie Fraunhofer'a (Niemcy). Grupa około 30 naukowców przy współpracy z Uniwersytetem Erlangen rozpoczęła prace badawcze nad kompresją cyfrowego dźwięku i obrazu. Prace zakończono w 1991 roku sukcesem. Powstał standard, który nazwano MPEG-1.

## 5. Typowe programy kompresujące pliki

Najpopularniejsze formaty skompresowanych plików to ZIP oraz RAR .
**W systemie operacyjnym Unix** typowym narzędziami do kompresji (bezstratnej) plików są

| | | |
|---|---|---|
| compress | oraz | uncopress |
| gzip | oraz | gunzip |

**W systemie operacyjnym Windows** najpopularniejszym programem kompresji jest program WinZip 8.1 .

Programy do kompresji plików nazywamy archiwizerami.