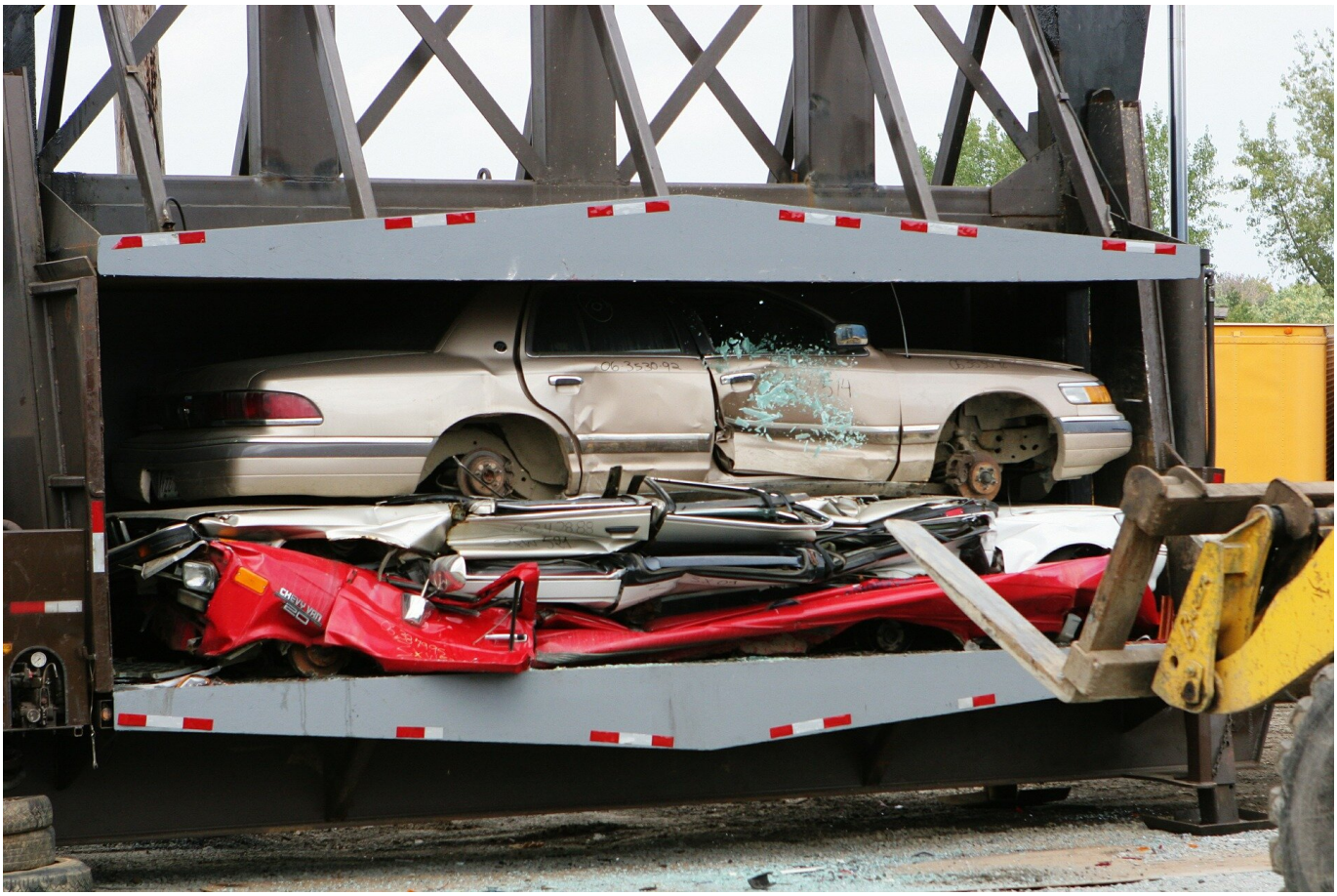
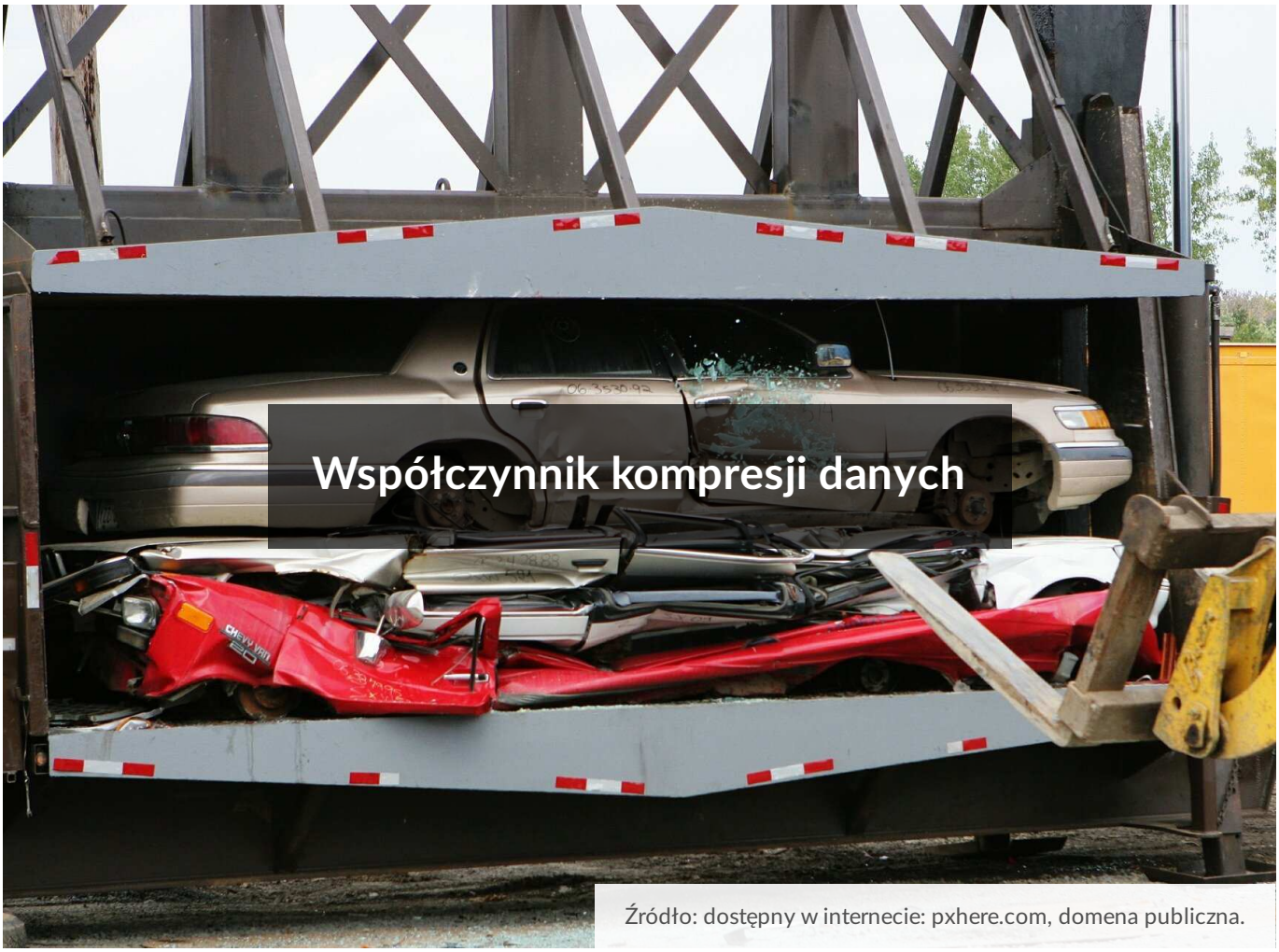


Współczynnik kompresji danych

- Wprowadzenie
- Przeczytaj
- Gra edukacyjna
- Sprawdź się
- Dla nauczyciela



Współczynnik kompresji danych

Źródło: dostępny w internecie: pxhere.com, domena publiczna.

Algorytmy kompresji towarzyszą ci każdego dnia. Kiedy wysyłasz zdjęcie za pomocą popularnego komunikatora, ten kompresuje je. W konsekwencji odbiorca otrzymuje obraz w nieco gorszej jakości, ale bez problemu może otworzyć go na urządzeniu mobilnym.

Z kompresją możesz spotkać się też, wybierając ofertę serwisu muzycznego. Zależnie od tego, ile chcesz zapłacić, serwis zaoferuje ci muzykę o odpowiednim stopniu kompresji.

Współczynnik kompresji jest miarą stosunku rozmiaru pliku pierwotnego do jego rozmiaru po zmianie sposobu zapisu informacji. W konsekwencji możemy ocenić, czy dane zostały dobrze skompresowane.

Więcej informacji dotyczących kompresji danych znajdziesz w pozostałych materiałach z serii:

- Kompresja danych,
- Kompresja stratna i bezstratna,
- Kompresja multimediów.

Więcej zadań? Zajrzyj do e-materiału: Kompresja danych – zadania maturalne.

> **Twoje cele**

- Wyjaśnisz, czym jest współczynnik kompresji danych.
- Przeanalizujesz, w jaki sposób obliczać wartość współczynnika kompresji danych na podstawie kilku różnych przykładów.
- Zastosujesz twierdzenie Shannona do obliczenia entropii.

Przeczytaj

Współczynnik kompresji danych

Współczynnik kompresji danych, zwany również **stopniem kompresji**, pozwala w sposób obiektywny ocenić skuteczność kompresji.

Pamiętajmy jednak, że nie zawsze mniejsza liczba danych po kompresji świadczy o ich jakości. M.in. dlatego istotne jest, aby skompresowane dane były jednoznacznie dekodowalne. W przypadku kompresji bezstratnej chcemy, aby dane po odkodowaniu były identyczne z wejściowymi. Natomiast w wypadku kompresji stratnej rezultat zależy od przyjętych kryteriów oraz od tego, co kompresujemy.

Dla popularnego formatu stratnej kompresji grafiki – JPEG – kryterium stanowi parametr jakości ustawiany przez użytkownika podczas kompresji. Wyraża się go w procentach, gdzie 1% to minimalna jakość (największy współczynnik kompresji), a 100% to maksymalna (najmniejszy współczynnik kompresji).

W plikach audio i wideo często stosowanym parametrem jest tzw. bitrate. Jest to liczba bitów przeznaczona na jednostkę czasu. Często używa się jednostek pochodnych, np. kilobitów na sekundę (oznaczanych kbit/s lub kbps). Im więcej bitów przeznaczymy na jedną sekundę pliku audio lub wideo, tym lepsza będzie jakość skompresowanego utworu (mniejszy współczynnik kompresji). Popularny format MP3 obsługuje bitrate od 16 kbps do 320 kbps.

W przypadku kompresji tekstu nie możemy sobie pozwolić na kompresję stratną, ponieważ tekst po dekompresji powinien odpowiadać oryginałowi. Kompresując obrazy lub filmy, możemy stosować kompresję stratną. Współczynnik kompresji dobieramy wówczas w zależności od zastosowań obrazu (filmu) i wtedy znaczenie ma to, żeby dana forma była czytelna dla człowieka. Przykładowo, jeżeli naukowcy z NASA robią z użyciem teleskopu zdjęcie nieba, to niezwykle istotne są na nim detale – w takiej sytuacji niedopuszczalna byłaby jakakolwiek kompresja stratna. Jeżeli jednak przesyłamy swoje zdjęcia na portal społecznościowy, to niewielka kompresja stratna (taka, której nie jesteśmy w stanie zauważyć na pierwszy rzut oka) jest już akceptowalna.

Dlatego też stopień kompresji jest wyznacznikiem skuteczności samej kompresji, jednak nie może w żadnym wypadku posłużyć za wyznacznik jej jakości. Dodatkowo, jeżeli porównamy współczynniki wyznaczone dla kompresji stratnej i bezstratnej, to powinniśmy zobaczyć, że kompresja stratna osiągnęła wyższy stopień kompresji.

Stopień kompresji obliczamy zgodnie z następującym wzorem:

$$\text{Stopień kompresji} = \frac{\text{Rozmiar nieskompresowanych danych}}{\text{Rozmiar danych po skompresowaniu}}$$

Porównajmy zatem rozmiar pliku ze zdjęciem poddanym kompresji. Wymiary zdjęcia to 600 x 600 pikseli.

Użyty format pliku	Rozmiar	Współczynnik kompresji
BMP (dane nieskompresowane)	1024 kB	-
PNG (kompresja bezstratna)	577 kB	1,77
JEPG (kompresja stratna, jakość kompresji 80%)	81,4 kB	12,58

We wszystkich przypadkach obraz był wyraźnie odwzorowany. Możemy przeprowadzić podobny eksperyment dla pliku audio. Kompresować będziemy krótki, 10-sekundowy utwór.

Użyty format pliku	Rozmiar	Współczynnik kompresji
WAV (dane nieskompresowane)	1740 kB	-
FLAC (kompresja bezstratna)	496 kB	3,51
MP3 (kompresja stratna)	236 kB	7,37

W przedstawionym przypadku utrata jakości utworu była niezauważalna dla ludzkiego ucha.

> **Ważne!**
>
> Kompresja stratna i kompresja bezstratna się nie wykluczają. Przykładowo, w kompresji obrazów JPEG korzysta się z kwantyzacji (potocznie można rozumieć to pojęcie jako zaokrąglania do określonej skali) oraz kodowania Huffmana. Wiele metod kompresji stratnej korzysta również z metod kompresji bezstratnej.

W omawianych przykładach posługujemy się metodami kompresji bezstratnej.

Istnieją jednak również inne kryteria pozwalające ocenić kompresję.

Claude Shannon w 1948 roku opublikował swoje „podstawowe twierdzenie dla kanałów bezszumowych". **Kanał bezszumowy** w tym kontekście oznacza, że wiadomość – pomiędzy zakodowaniem i zdekodowaniem – nie będzie podlegała żadnym błędom transmisji.

> **Twierdzenie: Podstawowe twierdzenie Shannona dla kanałów bezszumowych**
>
> Jeśli język posiada entropię h i kanał komunikacji może przetransportować C bitów na sekundę, możliwe jest zakodowanie sygnału w taki sposób, żeby transmitować $\frac{C}{h} - \mathcal{E}$

> symboli na sekundę, gdzie $\mathcal{E}$ może być dowolnie małe. Niemożliwe jest transmitowanie wiadomości o większej średniej częstości niż $\frac{C}{h}$.

Gdzie:

- Język oznacza listę liter/słów/zdań, które chcemy zakodować.
- Entropia h oznacza średnią liczbę bitów na symbol (jest to np. 8 dla kodowania ASCII).
- C oznacza przepustowość kanału komunikacji.

Przedstawione twierdzenie ma większe zastosowanie w telekomunikacji. W artykule „The Mathematical Theory of Communication" Shannon wprowadził pojęcie **entropii** oraz opisał koncepcję kompresji.

> **Ciekawostka**
>
> Artykuł z 1948 roku nazywał się „A Mathematical Theory of Communication", jednak w przedruku zmieniono mu nazwę na „The Mathematical Theory of Communication", ponieważ został uznany za bardziej przełomową pracę niż początkowo zdawało się Shannonowi.

Entropia w kontekście teorii informacji obliczana jest zgodnie z następującym wzorem:

$$H(X) = \sum_{i=1}^{n} \left(p_i \cdot \log_r \frac{1}{p_i} \right) = -\sum_{i=1}^{n} (p_i \cdot \log_r p_i)$$

Gdzie:

- X oznacza zmienną losową dyskretną o zbiorze $\{x_1, x_2, \ldots, x_n\}$;
- p_i oznacza prawdopodobieństwo wystąpienia danej zmiennej;
- r oznacza wariant, jaki przyjmujemy podczas obliczeń, jeśli pod r podstawimy 2, to wtedy obliczamy entropię dla bitów.

W teorii informacji entropią określa się średnią liczbę informacji, która przypada na pojedynczą wiadomość ze źródła. W dużym uproszczeniu „liczba informacji" – w omawianych przez nas przypadkach – oznacza pojedynczy znak, ponieważ każdy pojedynczy znak jest najmniejszą formą informacji, którą chcemy zakodować.

Przykłady obliczania entropii

Entropia wyznacza średnią długość bitów, jaką potrzebujemy do zakodowania pojedynczej informacji, czyli w omawianym przypadku **entropia to średnia długość słowa kodowego dla znaku.**

> **Przykład 1**

Spróbujmy obliczyć entropię dla wiadomości „AAAAAABBBBBBCCCCDDDD”, w której występują znaki o następujących częstościach:
A: 6
B: 6
C: 4
D: 4
Zbiór X ma następującą postać: {A, B, C, D}.
Zbiór prawdopodobieństw wygląda następująco:
$p_A = 0{,}3$
$p_B = 0{,}3$
$p_C = 0{,}2$
$p_D = 0{,}2$
Wzór na entropię przyjmuje zatem następującą postać:

$$H(X) = p_A \cdot \log_2 \frac{1}{p_A} + p_B \cdot \log_2 \frac{1}{p_B} + p_C \cdot \log_2 \frac{1}{p_C} + p_D \cdot \log_2 \frac{1}{p_D}$$

Co jest równoznaczne z:

$$H(X) = 0{,}3 \cdot \log_2 \frac{1}{0{,}3} + 0{,}3 \cdot \log_2 \frac{1}{0{,}3} + 0{,}2 \cdot \log_2 \frac{1}{0{,}2} + 0{,}2 \cdot \log_2 \frac{1}{0{,}2}$$

W celu wyliczenia przedstawionej wartości możemy napisać program albo skorzystać z arkusza kalkulacyjnego:

```
1 =0,3 * LOG(1/(0,3);2) + 0,3 * LOG(1/(0,3);2) + 0,2 * LOG(1/(0,2
```

Tym samym H(X) = 1,970951.
Oznacza to, że średnia długość informacji może wynosić minimalnie 1,970951 bitu dla kompresji bezstratnej.
W poprzednim e-materiale zakodowaliśmy tę wiadomość 2 razy, raz korzystając z kodowania Shannona, a drugi raz z kodowania Huffmana.
Słowa kodowe dla kodowania Shannona:
A: 00
B: 01
C: 100
D: 110
Dla kodowania Huffmana:
A: 00
B: 01

C: 10
D: 11

> **Efektywność kodowania**
>
> Wyrażona jest wzorem:
>
> $$\frac{H(X)}{\text{Średnia długość słowa kodowego}} \cdot 100\%$$
>
> Efektywność kodowania zbliża się tym bardziej do 100%, im mocniej zbliżamy się do niemożliwej do przekroczenia średniej długości słowa kodowego dla kodowania bezstratnego. Jeżeli jednak w wyniku kompresji udałoby się nam uzyskać wynik powyżej 100%, wtedy uzyskamy kompresję stratną.

Wiadomość ma 20 znaków (w naszym przypadku 20 informacji).
W przypadku kodowania Shannona wiadomość ma 48 bitów, czyli średnio przypada 2,4 bitu na znak.
W przypadku kodowania Huffmana wiadomość ma 40 bitów, czyli średnio przypada po 2 bity na znak.

Tym samym kodowanie Shannona w omawianym przypadku cechuje się efektywnością około 82,12%, co obliczyliśmy w następujący sposób:

$$\frac{H(X)}{\text{Średnia długość słowa kodowego}} \cdot 100\% = \frac{1,970951}{2,4} \cdot 100\% \approx 82,12\%$$

Kodowanie Huffmana w omawianym przypadku cechuje się natomiast efektywnością w okolicach 98,55%, co obliczyliśmy następująco:

$$\frac{H(X)}{\text{Średnia długość słowa kodowego}} \cdot 100\% = \frac{1,970951}{2} \cdot 100\% \approx 98,55\%$$

Stopień Kompresji dla kodowania oblicza się, używając podanego wcześniej wzoru:

$$\text{Stopień kompresji} = \frac{\text{Rozmiar nieskompresowanych danych}}{\text{Rozmiar danych po skompresowaniu}}$$

Zakładając, że wiadomość była zakodowana zgodnie z kodem ASCII, to każdy znak miałby długość 7 lub 8 bitów przed kompresją, zależnie od tego, w jaki sposób byłby przechowywany w pamięci. Jednak jeśli pamięć adresowana jest bajtowo, to każdy znak

zajmowałby 8 bitów.
A zatem wiadomość zajmowałaby 160 bitów przed zakodowaniem.
Po zakodowaniu zgodnie z algorytmem Shannona uzyskaliśmy następujący stopień kompresji:

$$\text{Stopień kompresji} = \frac{\text{Rozmiar nieskompresowanych danych}}{\text{Rozmiar danych po skompresowaniu}} = \frac{160}{48} \approx 3,33$$

Korzystając z kodowania Shannona, uzyskalibyśmy stopień kompresji wiadomości równy 3,33333....

Po zakodowaniu, zgodnie z algorytmem Huffmana, uzyskaliśmy następujący stopień kompresji:

$$\text{Stopień kompresji} = \frac{\text{Rozmiar nieskompresowanych danych}}{\text{Rozmiar danych po skompresowaniu}} = \frac{160}{40} = 4$$

W przypadku kodowania Huffmana stopień kompresji byłby równy 4.

Ważne!

Pamiętaj, żeby po kompresji wiadomość zapisać w sposób optymalny, tzn. zapisywać kilka znaków na pojedynczym bajcie. Możliwe jest to np. z zastosowaniem operacji przesunięcia bitowego. Operacja kompresji tekstu nie ma sensu, jeżeli pomimo zmiany kodowania na każdym bajcie zapisujemy tylko po jednym znaku.

Udało nam się obliczyć entropię i efektywność kodowania. Na czym jednak polega zjawisko entropii? W celu lepszego wyjaśnienia tego pojęcia skorzystamy z dwóch kolejnych przykładów.

Przykład 2

Chcemy za pomocą bitów opisać stan pogody w Krakowie.
Założymy, że do wyboru są jedynie takie opcje pogodowe:

- Pada deszcz – z prawdopodobieństwem, że pada, wynoszącym 0,25.
- Świeci słońce – z prawdopodobieństwem, że świeci, wynoszącym 0,25.
- Jest pochmurnie – z prawdopodobieństwem, że jest pochmurnie, wynoszącym 0,25.
- Jest mgliście – z prawdopodobieństwem, że jest mgliście, wynoszącym 0,25.

Informacje o tym, jaka jest pogoda, chcemy przesłać za pomocą telegrafu do Warszawy. Zakładamy bowiem, że to właśnie w stolicy składowane są informacje o pogodzie panującej we wszystkich miastach w Polsce.

Możemy więc ponumerować stany pogody od 0 do 3 i przypisać każdemu ze stanów odpowiadającą jego numerowi liczbę binarną:

- Pada deszcz: 00.
- Świeci słońce: 01.
- Jest pochmurnie: 10.
- Jest mgliście: 11.

Entropia tej wiadomości, zgodnie z podanym wcześniej wzorem, wynosi 2 bity, tzn. średnia długość przesyłanej wiadomości ma taką właśnie długość.

Załóżmy, że informacje o pogodzie przesyłamy dwa razy dziennie — rano i wieczorem. Rano przesyłamy „00", następnie wieczorem ponownie „00". Drugiego dnia przesyłamy „01", a następnie „10". I tak dalej, aż do zamknięcia stacji pogodowej. Za każdym razem długość przesyłanej wiadomości wynosi 2 bity.

Jeśli różnorodność występowania danych zjawisk pogodowych się zmieni, to wtedy kody być może zostaną zmienione. Jednak uśrednione dane podpowiadają centrali, że dotychczasowe spełniają swoją funkcje.

Entropia nie wyznacza jednak dolnego limitu długości wiadomości, a jedynie średnią długość informacji.

Przykład 3

Załóżmy, że z Krakowa przenosimy się do Dublina — miasta w którym bardzo często pada deszcz. Prawdopodobieństwo wystąpienia konkretnej pogody w Dublinie jest następujące:

- Pada deszcz: 0,5.
- Jest pochmurnie: 0,25.
- Jest mgliście: 0,125.
- Świeci słońce: 0,125.

Jeżeli dodatkowo założymy, że władze Irlandii chcą mieć na bieżąco uaktualnianą pogodę, wówczas kody będą wysyłane często. Istnieje więc ryzyko, że nie będziemy w stanie odróżnić początku i końca kolejnych kodów. Aby jednolity strumień bitów mógł zostać zdekodowany jednoznacznie, użyjemy kodowania prefiksowego.

Tym razem skorzystamy z następujących kodów. Tworzymy je zgodnie z algorytmem Huffmana. Zauważ, że żaden kod nie jest początkiem innego kodu.

- Pada deszcz: 0.
- Jest pochmurnie: 10.
- Jest mgliście: 110.
- Świeci słońce: 111.

> Załóżmy, że pogoda rzeczywiście rozkłada się zgodnie z prawdopodobieństwami. Wtedy kolejne wiadomości wyglądałyby w następujący sposób: 0 111 0 110 0 10 0 10, dla uproszczenia oddzieliliśmy kolejne informacje spacjami. Średnia długość przesłanej przez nas informacji wynosi więc 1,75 bita i rzeczywiście, jeśli policzymy entropię tej wiadomości, to będzie ona tyle wynosić.

Istnieje jednak możliwość, że entropia będzie wynosić 0. Taka sytuacja zaistnieje wtedy, gdy jedno zdarzenie ma prawdopodobieństwo 1. To znaczy, gdy informacja o stanie czegoś jest absolutnie trwała i jednoznaczna, niepodlegająca żadnym zmianom. W takim przypadku, odnosząc się do omawianych przykładów, jeśli pogoda w danym rejonie jest zawsze identyczna, to w jakim celu wysyłać wiadomości z jej aktualnym stanem?

Zgodnie ze wzorem na entropię, jeżeli prawdopodobieństwo zdarzenia wynosi 100%, to wtedy:

$$H\left(X\right) \ = \ 1 \ \cdot \ \log_2 \left(1\right) \ = \ 0$$

W praktyce jeśli prawdopodobieństwa są potęgami dwójki, to wtedy optymalna długość słowa kodowego dla danej informacji będzie równa wykładnikowi tej potęgi pomnożonemu przez minus jeden. Jeżeli więc prawdopodobieństwo wystąpienia danej litery wynosi $\frac{1}{16}$, to w formie wykładniczej ułamek ten możemy zapisać jako 2^{-4}. Optymalna długość takiego słowa wynosiłaby więc 4 bity.

> **Przykład 4**
>
> Dla wiadomości: „AAAABBBBCCCCDDFF” dane będą następujące częstości:
>
> Częstości wystąpienia znaków:
> A: 4
> B: 4
> C: 4
> D: 2
> F: 2
>
> Prawdopodobieństwa wystąpienia znaków:
> A: $\frac{1}{4} = 2^{-2}$ oczekujemy więc minimalnej długości słowa kodowego 2.
> B: $\frac{1}{4} = 2^{-2}$ oczekujemy więc minimalnej długości słowa kodowego 2.
> C: $\frac{1}{4} = 2^{-2}$ oczekujemy więc minimalnej długości słowa kodowego 2.
> D: $\frac{1}{8} = 2^{-3}$ oczekujemy więc minimalnej długości słowa kodowego 3.
> F: $\frac{1}{8} = 2^{-3}$ oczekujemy więc minimalnej długości słowa kodowego 3.
>
> Słowa kodowe uzyskane za pomocą algorytmu Huffmana:
> A: 10
> B: 11

C: 00
D: 010
F: 011
Co zgadza się z podanymi wcześniej oczekiwaniami.

Wiadomość miałaby więc postać: 10 10 10 10 11 11 11 11 00 00 00 00 010 010 011 011, co daje nam średnią długość 2,25 bitu na informacje. Można to ustalić, obliczając sumę wszystkich bitów, jakie ma końcowa wiadomość, a następnie dzieląc je przez liczbę wszystkich informacji (w tym wypadku znaków), jakie były w wiadomości pierwotnej.

Liczba bitów w zakodowanej wiadomości wynosi 36, a liczba kodowanych znaków jest równa 16. Stąd właśnie średnia długość słowa kodowego na informację wynosi 2,25 bitu. Entropię obliczymy z podanego wzoru (składniki – dla uzyskania prawdopodobieństwa tej samej wartości – zamieniono na iloczyn):

$$H(X) = -(3 \cdot 2^{-2} \cdot log_2(2^{-2}) + 2 \cdot 2^{-3} \cdot log_2(2^{-3}))$$

$$H(X) = -(3 \cdot \tfrac{1}{4} \cdot (-2) + 2 \cdot \tfrac{1}{8} \cdot (-3))$$

$$H(X) = -(-\tfrac{12}{8} - \tfrac{6}{8})$$

$$H(X) = \tfrac{18}{8} = 2.25$$

Entropia wynosi 2,25 bitu. Oznacza to, że w tym przykładzie uzyskaliśmy najlepszą możliwą do uzyskania kompresję bezstratną, ponieważ efektywność kodowania wynosi w takiej sytuacji 100%.

Jeżeli znaki kodowane były zgodnie z kodem ASCII, czyli na 7 lub 8 (w przypadku *extended ASCII*) bitach, ale zakładając, że pamięć adresowana jest bajtowo, to wiadomość przed zakodowaniem miałaby długość 128 bitów. Po zakodowaniu ma długość jedynie 36 bitów.

$$\text{Stopień kompresji } = \frac{\text{Rozmiar nieskompresowanych danych}}{\text{Rozmiar danych po skompresowaniu}} = \frac{128}{36} \approx 3,56$$

Stopień kompresji uzyskany dla tej wiadomości wynosi więc około 3,56.
Z racji tego, że efektywność kodowania wynosi 100%, jest to najmniejsza możliwa bezstratna kompresja, jaką da się uzyskać dla tej wiadomości. Przy czym zakładamy, że za pojedynczą informację przyjmiemy znak.

Jak wspominaliśmy, istnieje pewien limit kompresji, który da się osiągnąć metodami bezstratnymi. W przedstawionym przykładzie uzyskaliśmy najlepszy możliwy stopień kompresji bezstratnej przy założeniu, że jednostka informacji to pojedynczy znak.

Dla wiadomości z omawianego przykładu moglibyśmy równie dobrze stworzyć następujący słownik słów kodowych:
AAAA: 10
BBBB: 11
CCCC: 00
DD: 010
FF: 011

Słownik ten jest poprawny, jednoznacznie dekodowalny. Zakodowana zgodnie z nim wiadomość miałaby następującą postać bitową: 10 11 00 010 011, a uzyskany stopień kompresji wynosiłby około 10,67.

Wszystko się w tym przypadku zgadza i nie ma żadnego błędu. Problem w tym, że kodowaliśmy całe podciągi znaków, zamiast pojedynczych liter. Istnieje wersja kodowania Huffmana, uzyskująca większe stopnie kompresji, właśnie poprzez kodowanie całych podciągów.

Zmieniła się również entropia wiadomości, ponieważ zmianie uległa także „jednostka informacji" ze znaku na ciąg znaków.

W dwóch przykładach kodowaliśmy informacje o stanie pogody. Pogoda jest złożonym problemem, a w dużym uproszczeniu udało nam się ją sprowadzić do jedynie kilku bitów na każdy jej stan. Analogicznie duże ciągi znaków mogą być uznawane za całe słowa kodowe. Można nawet kodować całe słowa jako słowa kodowe, o ile te często się powtarzają. Nie omawialiśmy jednak przykładów, w których kodujemy więcej niż pojedyncze symbole, ponieważ jest to dużo bardziej złożone.

A zatem kodowanie Huffmana z przykładu czwartego pozwoliło nam skompresować wiadomość maksymalnie bezstratnie przy założeniu, że „jednostką informacji", jaką kodujemy, jest znak.

Załóżmy jednak, że ktoś nie wierzy, że wiadomości nie da się dalej zmniejszyć i będzie próbował zakodować jedną wiadomość wielokrotnie. Czy ma szansę na powodzenie?

Odpowiedz brzmi: to zależy.

Jeśli spróbujemy zakodować, nie zmieniając podstawowej „jednostki informacji" (tzn. jeśli zakodujemy pojedyncze znaki, a potem ponownie spróbujemy zakodować pojedyncze znaki), kompresja się nie powiedzie.

Jeśli jednak spróbujemy zakodować ponownie wiadomość, zmieniając podstawową „jednostkę informacji" na podciąg znaków, to wiadomość będzie mniejsza.
Jednakże będzie ona jednocześnie większa niż mogła by być, gdybyśmy od razu kodowali całe podciągi znaków. Wynika to z faktu, że przy każdej użytej kompresji do samej wiadomości dołączana jest w jakiejś formie książka słów kodowych.

Czyli o ile wiadomość rzeczywiście się zmniejsza zgodnie z obliczonym stopniem kompresji, o tyle do samej wiadomości należy dostarczyć jeszcze książkę kodową. Nie dołączaliśmy wielkości książki kodowej w obliczeniach, ponieważ w rzeczywistych sytuacjach jest ona pomijalnie mała.

Zakładając jednak, że próbujemy kodować ciągi bitów, to z następującej wiadomości: 10 10 10 10 11 11 11 11 11 00 00 00 00 010 010 011 011 mamy częstości:

11: 5

10: 4

00: 4

010: 2

011: 2

Stąd przedstawione ciągi bitowe, korzystając z algorytmu Huffmana, możemy zakodować w następujący sposób:

- ciąg bitów: 10 i słowo kodowe: 11
- ciąg bitów: 11 i słowo kodowe: 00
- ciąg bitów: 00 i słowo kodowe: 10
- ciąg bitów: 010 i słowo kodowe: 010
- ciąg bitów: 011 i słowo kodowe: 011

Możemy zauważyć, że najczęściej pojawiającym się ciągiem bitów jest 11, któremu algorytm przypisuje kod 00. Program kompresujący zawrze jednak w wiadomości nową książkę kodową. Za każdą próbą kompresji plik — zamiast maleć — rósłby o rozmiar nowej książki kodowej. Co więcej, samo kodowanie nie wpłynie na zmniejszenie długości ciągów. W zaprezentowanym przykładzie algorytm wygeneruje słowa kodowe o takiej samej długości jak wprowadzone ciągi bitów.

Reasumując: wielokrotna kompresja bezstratna powoduje jedynie zwiększanie rozmiaru kompresowanego pliku. Jedyną możliwością na jeszcze większe zmniejszenie rozmiaru pliku jest w tym przypadku zmiana algorytmu kompresującego. Przy czym ten też uzyska lepszy stopień kompresji, jeśli będzie kompresował oryginalny plik.

Warto jeszcze dodać, że jeśli wiadomość miałaby postać: ABCCBABCADACFBDF, to znaczy litery występowałyby w niej w kolejności losowej, to nie dałoby się pogrupować tego ciągu w sensowne podciągi, przynajmniej na pierwszy rzut oka.

> **Już wiesz**
>
> Kodowanie arytmetyczne polega na:

- zliczeniu częstości występowania danych symboli/słów/znaków,
- obliczeniu prawdopodobieństwa wystąpienia każdego z symboli i pogrupowaniu ich od największego do najmniejszego,
- kodowania wiadomości poprzez wchodzenie w coraz mniejsze przedziały wartości.

Dla wiadomości: BAB# przedziały mogą być zrealizowane graficznie w następujący sposób:

Kodowanie arytmetyczne jest ograniczone w świecie fizycznym, ale czy jest również ograniczone w świecie wirtualnym?

Odpowiedź brzmi: Tak.

Liczby po przecinku również mają swoje ograniczenia. Jeśli w komputerze spróbujemy w odpowiednim typie podzielić 1 przez 3, a następnie pomnożyć przez 3, to nie uzyskamy z powrotem jedynki.
Dzieje się tak, ponieważ wartości zmiennoprzecinkowe są zaokrąglane. Jednym ze standardów obsługujących liczby zmiennoprzecinkowe jest standard IEEE-754, który umożliwia zapisywanie małych liczb, tylko w konsekwencji tracą one swoją precyzję.

Słownik

kodowanie prefiksowe

kodowanie, w którym żaden kod nie jest prefiksem innego kodu (żaden kod nie jest początkiem innego kodu); dzięki temu możliwe jest dekodowanie ciągu kodów bez wyróżnienia końców i początków kolejnych kodów

kompresja bezstratna

metoda zapisu informacji do postaci zajmującej mniej pamięci, a jednocześnie umożliwiająca odtworzenie informacji z postaci skompresowanej do identycznej postaci początkowej

książka kodowa

zbiór słów kodowych przyporządkowanych do informacji (np. do pojedynczego znaku)

teoria informacji

nauka o przechowywaniu, przenoszeniu i zliczaniu informacji; dyscyplina ta znajduje się na pograniczu matematyki, statystyki, informatyki i elektroniki

Gra edukacyjna

Polecenie 1

Sprawdź swoją wiedzę, biorąc udział w grze.

Test

Sprawdź swoją wiedzę o współczynniku kompresji danych, biorąc udział w grze

Poziom trudności:

łatwy

Limit czasu:

7 min

Twój ostatni wynik:

-

Uruchom

Sprawdź się

Pokaż ćwiczenia:

Ćwiczenie 1

Wskaż prawdziwe zdanie.

- ◯ Współczynnik kompresji danych mówi nam, czy kodowanie jest optymalne.
- ◯ Współczynnik kompresji danych możemy obliczyć tylko dla algorytmu kompresji stratnej.
- ◯ Współczynnik kompresji oblicza się jako stosunek rozmiaru nieskompresowanych danych do rozmiaru danych skompresowanych.

Ćwiczenie 2

Oblicz współczynniki kompresji.

Dane przed zakodowaniem: 50 bitów.

Dane po zakodowaniu: 5 bitów.

Współczynnik kompresji: [ ]

Dane przed zakodowaniem: 24 bity.

Dane po zakodowaniu: 6 bitów.

Współczynnik kompresji: [ ]

Dane przed zakodowaniem: 1024 bity.

Dane po zakodowaniu: 512 bitów.

Współczynnik kompresji: [ ]

Ćwiczenie 3

Wskaż wszystkie prawdziwe zdania, odnoszące się do entropii.

- [ ] Entropia jest identyczna ze średnią długością słowa kodowego dla wszystkich kodów.
- [ ] Entropia może wyznaczać optymalną średnią ilość bitów na informację.
- [ ] Entropia wyznacza minimalną długość słowa kodowego.
- [ ] Entropia oznacza średnią liczbę informacji, która przypada na pojedynczą wiadomość ze źródła.

Ćwiczenie 4

Oblicz entropię.

Dane są prawdopodobieństwa wystąpienia danych znaków:

A: 1/2

B: 1/2

C: 1/2

D: 1/2

Entropia wynosi: _____ bity.

Dane są prawdopodobieństwa wystąpienia danych ciągów znaków:

AAAA: 1/8

BBBB: 1/8

CCCC: 1/8

ABCD: 1/8

AABB: 1/8

BBCC: 1/8

CCDD: 1/8

DDAA: 1/8

Entropia wynosi: _____ bity.

Ćwiczenie 5

Wskaż zdanie prawdziwe.

- Efektywność kodowania pomaga wyznaczyć najmniejszą liczbę bitów potrzebną do zapisania jednego słowa kodowego.
- Efektywność kodowania to stosunek rozmiaru danych przed skompresowaniem do rozmiaru danych po skompresowaniu razy 100%.
- Efektywność kodowania to stosunek entropii do średniej długości słowa kodowego razy 100%.
- Efektywność kodowania rośnie wraz ze zmniejszaniem się rozmiaru skompresowanych danych.

Ćwiczenie 6

Oblicz efektywność kodowania.

Entropia: 2

Średnia długość słowa kodowego: 4

Efektywność kodowania w procentach:

Entropia: 2

Średnia długość słowa kodowego: 2

Efektywność kodowania w procentach:

Entropia: 12

Średnia długość słowa kodowego: 48

Efektywność kodowania w procentach:

Ćwiczenie 7

Zaznacz odpowiednim kolorem zdanie, które najlepiej charakteryzuje dane pojęcie.

Entropia Efektywność Kodowania Stopień Kompresji

Wskazuje, w jakim stopniu udało nam się skompresować wiadomość.

Jest to średnia ilość bitów potrzebna do wysłania informacji.

Wskazuje nam w procentach, jak blisko ideału jest nasza kompresja.

Ćwiczenie 8

Uzupełnij tabelę o minimalną liczbę bitów, potrzebną do zakodowania informacji o podanym prawdopodobieństwie.

Prawdopodobieństwo	Minimalna liczba bitów potrzebna do zakodowania
0,25	
0,0625	
0,5	
0,125	

4 1 3 2

Dla nauczyciela

Autor: Maurycy Gast
Przedmiot: Informatyka

Temat: Współczynnik kompresji danych

Grupa docelowa:

Szkoła ponadpodstawowa, liceum ogólnokształcące, technikum, zakres rozszerzony

Podstawa programowa:

Cele kształcenia – wymagania ogólne

III. Posługiwanie się komputerem, urządzeniami cyfrowymi i sieciami komputerowymi, w tym: znajomość zasad działania urządzeń cyfrowych i sieci komputerowych oraz wykonywania obliczeń i programów.

Treści nauczania – wymagania szczegółowe

I + II. Zakres rozszerzony. Uczeń spełnia wymagania określone dla zakresu podstawowego, a ponadto:

2) wykorzystuje znane sobie algorytmy przy rozwiązywaniu i programowaniu rozwiązań następujących problemów:

a) rozkładania liczby na czynniki pierwsze,

b) wykonywania działań na liczbach w systemach innych niż dziesiętny,

III. Posługiwanie się komputerem, urządzeniami cyfrowymi i sieciami komputerowymi.

Zakres rozszerzony. Uczeń spełnia wymagania określone dla zakresu podstawowego, a ponadto:

2) dokonuje kompresji informacji, objaśnia różnice między kompresją stratną i bezstratną tekstów, obrazów, dźwięków, filmów;

Kształtowane kompetencje kluczowe:

- kompetencje cyfrowe;
- kompetencje osobiste, społeczne i w zakresie umiejętności uczenia się;
- kompetencje matematyczne oraz kompetencje w zakresie nauk przyrodniczych, technologii i inżynierii.

Cele operacyjne (językiem ucznia):

- Wyjaśnisz, czym jest współczynnik kompresji danych.
- Przeanalizujesz, w jaki sposób obliczać wartość współczynnika kompresji danych na podstawie kilku różnych przykładów.
- Zastosujesz twierdzenie Shannona do obliczenia entropii.

Strategie nauczania:

- konstruktywizm;
- konektywizm.

Metody i techniki nauczania:

- dyskusja;
- rozmowa nauczająca z wykorzystaniem multimedium i ćwiczeń interaktywnych;
- metody aktywizujące.

Formy pracy:

- praca indywidualna;
- praca w parach;
- praca w grupach;
- praca całego zespołu klasowego.

Środki dydaktyczne:

- komputery z głośnikami, słuchawkami i dostępem do internetu;
- zasoby multimedialne zawarte w e-materiale;
- tablica interaktywna/tablica, pisak/kreda.

Przebieg lekcji

Przed lekcją:

1. **Przygotowanie do zajęć.** Nauczyciel loguje się na platformie i udostępnia e-materiał: „Współczynnik kompresji danych". Nauczyciel prosi uczniów o zapoznanie się z treściami w sekcji „Przeczytaj".

Faza wstępna:

1. Nauczyciel wyświetla uczniom temat zajęć oraz cele. Prosi, by na ich podstawie uczniowie sformułowali kryteria sukcesu.
2. **Rozpoznanie wiedzy uczniów.** Uczniowie tworzą pytania dotyczące tematu zajęć, na które odpowiedzą w trakcie lekcji.

Faza realizacyjna:

1. **Praca z tekstem.** Nauczyciel wyświetla zawartość sekcji „Przeczytaj". Na forum klasy uczniowie analizują przedstawione w niej rozwiązania Przykładów 1-4.
2. **Praca z multimedium.** Nauczyciel wyświetla zawartość sekcji „Gra interaktywna". Uczniowie wspólnie odpowiadają na zawarte w niej pytania.
3. W kolejnym etapie uczniowie dobierają się w pary i wykonują ćwiczenia nr 1-8. Następnie konsultują swoje rozwiązania z inną parą uczniów i ustalają jedną wersję odpowiedzi.

Faza podsumowująca:

1. Nauczyciel wyświetla na tablicy temat lekcji i cele zawarte w sekcji „Wprowadzenie". W kontekście ich realizacji podsumowuje przebieg zajęć, a także wskazuje mocne i słabe strony pracy uczniów.
2. Na koniec zajęć nauczyciel prosi uczniów o rozwinięcie zdania: „Na dzisiejszych zajęciach nauczyłam/łem się jak...".

Praca domowa:

1. Uczniowie obliczają entropię dla podanej przez nauczyciela wiadomości.

Wskazówki metodyczne:

- Treści w sekcji „Gra edukacyjna" można wykorzystać jako materiał służący powtórzeniu materiału.