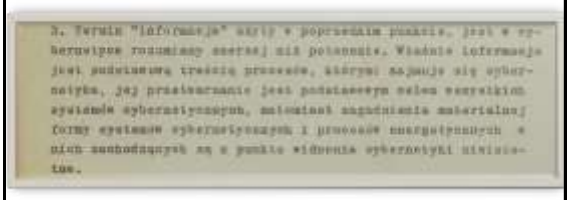
Wstęp do systemów informacyjnych

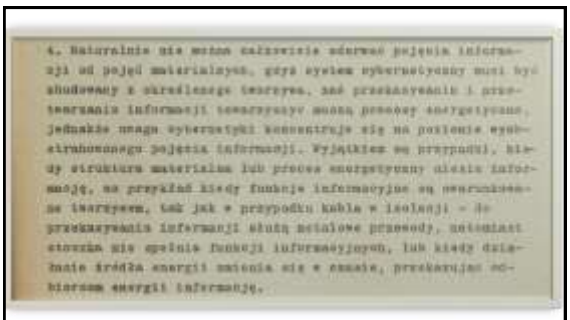
Teoria informacji

Z systemem informacyjnym wiąże się nierozerwalnie pojęcie **informacji**

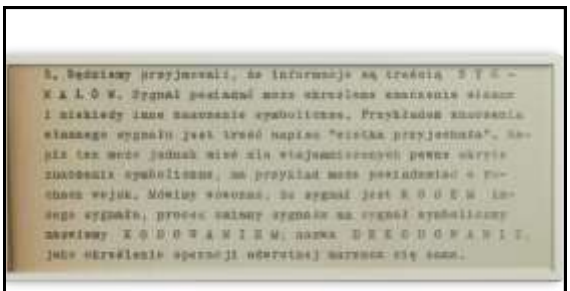
Na początek sięgniemy do kilku cytatów z mojej książki, którą napisałem w 1972 roku, a która została wydana w 1974



Proszę traktować ten zeskanowany fragment jako szacowny zabytek :-)



Proszę traktować ten zeskanowany fragment jako szacowny zabytek :-)



Proszę traktować ten zeskanowany fragment jako szacowny zabytek :-)

Jak zdefiniować i jak zmierzyć informację?

Informacja jest czynnikiem nadającą tożsamość różnym obiektom materialnym.

Rozważmy następujący przykład

Co to jest „*Pan Tadeusz*”?

Czy to jest ta oto książka?

Nie, bo czym by wtedy było każde inne wydanie?

Może więc jest to zbiór wszystkich książek z tym tytułem na okładce?

Nie, bo czym by wtedy był audiobook?

Czym by była recytacja lub inscenizacja?

Albo film?

Może więc Pan Tadeusz to zbiór myśli pewnego poety?

Nie, bo ten poeta nie żyje,
więc siedlisko jego myśli rozsypało się w proch

Czym więc jest dzieło literackie?

Informacją!

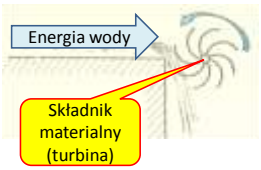
Informacja jest jednym z fundamentalnych pojęć – nie tylko biocybernetyki, ale wręcz opisu świata.

Tradycyjna nauka (fizyka, chemia, astronomia, biologia) opisywała świat biorąc za podstawę dwa główne składniki: **materię** z której zbudowane są obiekty realnego świata oraz **energię**, która powoduję, że obiekty te mogą podlegać licznym przemianom.

Jednak rozwój techniki (zwłaszcza automatyki, telekomunikacji i informatyki) zmusił do uwzględnienia w opisie świata jeszcze jednego równorzędnego bytu – właśnie **informacji**.

Aby unaocznić znaczenie informacji
w opisie świata odwołamy się
do prostego przykładu

Prosty system pozwalający prześledzić współzależności między materią, energią i informacją.

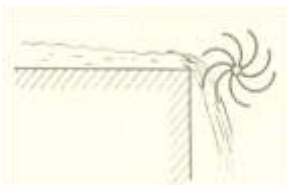


W pokazanej na rysunku sytuacji system sprawnie działa, bo wszystkie rozważane komponenty są współobecne.

Brak składnika materialnego (turbinki) powoduje, że system nie działa

Także brak składnika energetycznego (płynącej wody) spowoduje, że system nie działa

Rozważany system nie działa także w przypadku, kiedy wszystkie składniki materialne są obecne i nie brakuje składnika energetycznego (płynącej wody), ale popełniono zasadniczy błąd montażowy.



Czynnikiem, którego w tym przypadku zabrakło, jest *informacja*

Informacja ma wiele
różnych postaci

Informacją jest każdy tekst, niezależnie od tego, na czym i w jaki sposób jest zapisany.

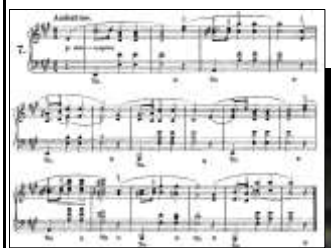

Informacją jest każdy tekst, niezależnie od tego, na czym i w jaki sposób jest zapisany.


Informacją jest każdy tekst, niezależnie od tego, na czym i w jaki sposób jest zapisany.

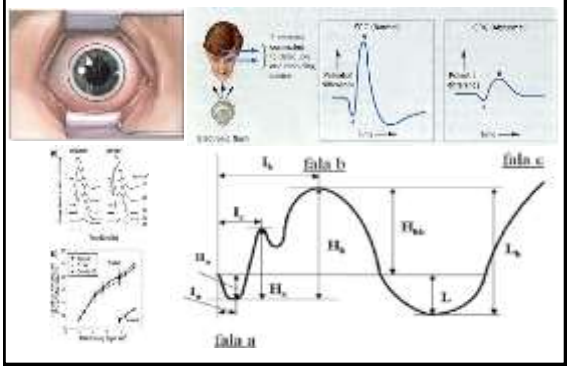

Informacją jest każdy tekst, niezależnie od tego, na czym i w jaki sposób jest zapisany.


Informacją jest tekst mówiony

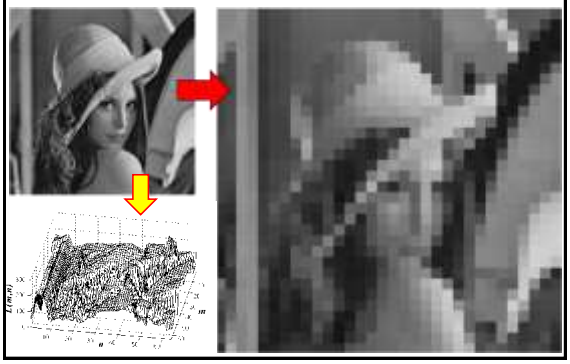

Informacją jest utwór muzyczny, zarówno zapisany w nutach jak i wykonywany



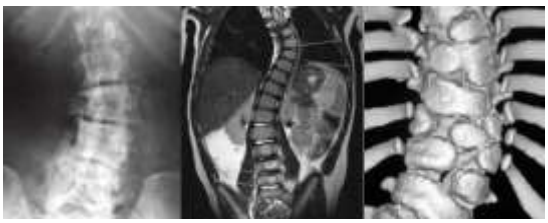

Informacją są przeróżne sygnały biomedyczne



Informacją jest też każdy obraz

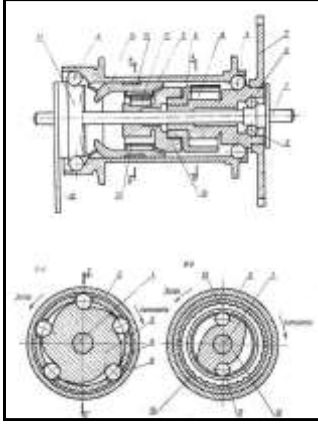


W szczególności dotyczy to obrazów medycznych



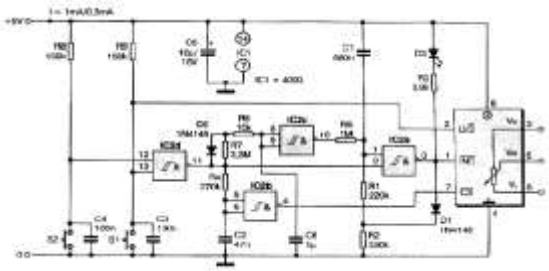
Ale nośnikiem informacji jest także każda mapa

Lub jej odpowiednik w postaci zdjęcia satelitarnego



Dla inżynierów ważnym źródłem informacji jest rysunek techniczny.

Specjalnym rodzajem informacji są schematy elektroniczne



Nośnikiem informacji może być także ubiór

Dlatego na zaproszeniach czasem podaje się tzw. *dress code* czyli sugestię, jaki ubiór jest oczekiwany

Professor
Ryszard Tadeusiewicz

On the occasion of
Moving Young Minds 2008

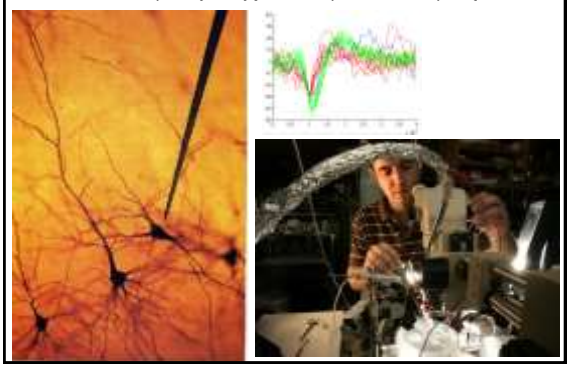
HER MAJESTY'S GOVERNMENT
in the United Kingdom of Great Britain and Northern Ireland
request the honour of your company
at a Reception at Lancaster House, St James's, SW1
on Monday 7th January 2008 at 6:00pm to 8:00pm

The Rt Hon Ed Balls MP
Secretary of State for Children, Schools and Families
will preside

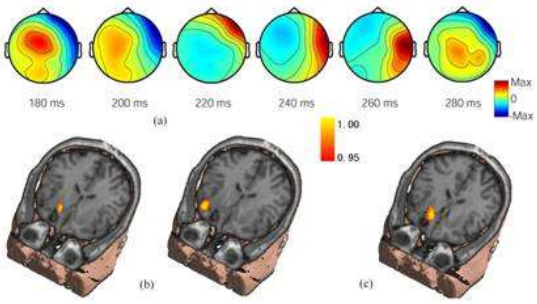
Please present this card on arrival

W biologii informacja występuje najczęściej w formie struktury określonych molekuł (animacja).

Informacją w najczystszej postaci są też sygnały komórek nerwowych rejestrujące ich aktywność elektryczną



Sygnałem jest też cała aktywność mózgu



Informacyjny charakter ma też tożsamość każdej ludzkiej istoty

Ta informacyjna tożsamość sprawia, że na każdym z tych zdjęć identyfikujemy tę samą osobę

Informacja jest istotą tożsamości

Dzięki teorii informacji wiemy, że widzimy tu jednego człowieka, a nie pięciu!

Informacja jest pozyskiwana, gromadzona, analizowana oraz wykorzystywana w wielu różnych celach.

Nas nie interesuje jednak teraz forma informacji (przedyskutowana wyżej) ani jej wartość zilustrowana na następnym slajdzie.

Koncentrujemy uwagę wyłącznie na ilości informacji.

O wartości informacji w medycynie świadczy między innymi taki żart ☺

Ale nie żartujmy, bo to jest poważna problematyka.

Skupimy się na teorii informacji.

Twórca teorii informacji Claude Shannon

Genialny pomysł Shannona polegał na tym, że zamiast próbować definiować informację wprost (co wcześniej bez powodzenia próbowali zrobić między innymi tak sławni badacze jak Harry Nyquist i Ralph Hartley) – postanowił zdefiniować miarę **braku informacji**, a samą informację wyrazić jako czynnik usuwający ten brak informacji.

Pozornie wygląda to jak próba sięgania lewą ręką do prawej kieszeni, ale okazało się, że jest to właściwa i skuteczna droga.

Wyobraźmy sobie, że otrzymaliśmy pewną wiadomość *W*.

Ile informacji wnosi ta wiadomość?

Przyglądając się samej wiadomości nie będziemy w stanie odpowiedzieć na to pytanie, bo ta sama wiadomość dla jednego odbiorcy może być bardzo cenna, a dla drugiego jest ona bezwartościowa.

Dlatego musimy odwołać do tego, czego dotyczy ta wiadomość i jakie ma ona dla nas znaczenie.

Wyobraźmy więc sobie, że interesuje nas pewne zdarzenie *Z* oraz że wiadomość *W* ma nas poinformować o tym zdarzeniu.

W takim przypadku potrzebujemy formuły matematycznej opisującej zależność *I(W/Z)* pozwalającą oszacować ilość informacji *I* jaką wnosi wiadomość *W* o zdarzeniu *Z*.

Zauważmy, że formuła *I*(*W/Z*) musi uzależniać ilość informacji zarówno od wiadomości *W* jak i od zdarzenia *Z*.

Zależność informacji od wiadomości *W* jest oczywista.

Natomiast fakt, że koncentrujemy się na zależności ilości informacji od zdarzenia *Z*, którego wiadomość dotyczy, wynika z faktu, że wiadomość *W* może zawierać dodatkowe informacje na zupełnie inne tematy.

Na przykład jeśli otrzymamy wiadomość o stanie zdrowia pacjenta **telefonicznie** to obok informacji o zdarzeniu *Z* (stan pacjenta) wiadomość *W* będzie zawierała dodatkowo informację o płci dzwoniącej osoby, o tym, czy jest zdenerwowana, a jeśli znamy głos rozmówcy - to również o jego tożsamości.

Jednak nas interesuje tylko informacja o zdarzeniu *Z*, jakim był stan zdrowia pacjenta, a wszelkie inne informacje się nie liczą, więc wyraźnie to wskazujemy w poszukiwanej formule *I*(*W/Z*).

Założenie Shannona:

Ilość informacji, jaką możemy uzyskać o jakimś zdarzeniu *Z* jest zależna od stopnia naszej **niepewności** co do tego zdarzenia.

Wyobraźmy sobie, że tę niepewność potrafimy oszacować i oznaczmy tę funkcję niepewności jako *H(Z)*.

Jeśli rozważana wiadomość *W* zdołała tę niepewność całkowicie zlikwidować – to uznamy, że ilość informacji, jaką otrzymaliśmy wraz z wiadomością *W*, będzie równa tej usuniętej niepewności:

$$I(W/Z) = H(Z)$$

Niestety nie zawsze tak bywa, że po otrzymaniu wiadomości *W* nasza niepewność co do zdarzenia *Z* zostaje zredukowana do zera.

Przeciwnie, regułą jest raczej to, że po otrzymaniu wiadomości nadal nęka nas niepewność!

Oznaczmy więc funkcją *H(Z/W)* niepewność na temat zdarzenia *Z* jaka pozostaje po otrzymaniu wiadomości *W*.

Uzasadnione jest więc przepisanie definicji informacji w ogólniejszej postaci:

$$I(W/Z) = H(Z) - H(Z/W)$$

Ze tego wzoru wynika, że decydujemy się przyjąć jako miarę ilości informacji *I(W/Z)* stopień **zmniejszenia** naszej początkowej niepewności, jaki nastąpił w wyniku odebrania wiadomość *W*.

Wzory pozwalające na wyznaczanie wartości niepewności zostaną wprowadzone nieco dalej, tu jednak możemy już „awansem” zapewnić, że każda miara niepewności jest dodatnia (lub w szczególnym przypadku zerowa), natomiast nie może być ujemna. Oznacza to, że w szczególności

$$H(Z/W) \geq 0$$

To oznacza, że przypadek opisany wzorem

$$I(W/Z) = H(Z)$$

określa nam **graniczną** (maksymalną) ilość informacji jaką możemy uzyskać w opisywanej tu sytuacji. Ta graniczna ilość informacji determinowana jest naszą początkową niepewnością *H(Z)*, natomiast na tym etapie rozważań jest niezależna od zawartości wiadomości *W*.

Wiadomość *W* jednak nie jest bez znaczenia. Zgodnie ze wzorem

$$I(W/Z) = H(Z) - H(Z/W)$$

źle zbudowana wiadomość może sprawić, że tej granicznej ilości informacji nie da się uzyskać i będziemy musieli zadowolić się ilością informacji

$$I(W/Z) \leq H(Z)$$

Warto zauważyć, że nie jest wykluczona sytuacja w której niepewność po otrzymaniu wiadomości *W* nie tylko nie zmaleje, ale przeciwnie – wzrośnie:

$$H(Z/W) > H(Z)$$

Taka sytuacja ma miejsce, gdy wiadomość *W* jest w istocie **dezinformacją**. W takim przypadku ilość informacji wnoszonej przez taką wiadomość jest ujemna ($I(W/Z) < 0$) i musimy się z tym pogodzić.

Przytoczone wyżej rozważania wskazały, że ilość informacji będziemy mogli zmierzyć (w właściwie wyznaczyć obliczeniowo) gdy znajdziemy skuteczny sposób obliczania miar niepewności *H(Z)* oraz *H(Z/W)*.

Idąc śladem pomysłów Shannona powiążemy miarę niepewności dotyczącą zdarzenia *H(Z)* z **prawdopodobieństwem** tego zdarzenia *p(Z).*

Związek *H(Z)* oraz *p(Z)* jest następujący:

Im mniejsze prawdopodobieństwo, tym większa niepewność.

Możemy to formalnie zapisać wprowadzając do rozważań dwa zdarzenia Z_1 oraz Z_2 : jedno o większym, a drugie o mniejszy prawdopodobieństwie:

$$p(Z_1) > p(Z_2)$$

Zgodnie ze sformułowanym postulatem nierówność dotycząca niepewności będzie się układała w przeciwnym kierunku:

$$H(Z_1) < H(Z_2)$$

Omawiany postulat można uzupełnić przechodząc do granicy. Graniczną wartością, do jakiej może wzrosnąć prawdopodobieństwo jest wartość 1, nadawana zdarzeniom **pewnym.**

Skoro tak, to **niepewność** zdarzenia o prawdopodobieństwie wynoszącym 1 powinna być zerowa.

To jest właśnie kolejny postulat, jaki można wysunąć pod adresem definicji tego pojęcia.

$$p(Z) = 1 \quad \leftrightarrow \quad H(Z) = 0$$

Trzeci postulat związany jest z sytuacjami, w których musimy określić miarę niepewności dla **zdarzenie złożonego.**

Załóżmy, że interesuje nas zdarzenie Z polegające na równoczesnym zajściu dwóch **niezależnych** zdarzeń. Na przykład sytuacja, w której pojawi się masowe zatrucie na wiejskim weselu i zepsuje się karetka pogotowia.

Otóż dla takiej sytuacji Shannon postulował, żeby niepewności zdarzeń składowych po prostu się sumowały.
Tak jest najprościej i najwygodniej!

Zobaczmy, co z tego wynika?

Przy zdarzeniach niezależnych prawdopodobieństwo zajścia takiego zdarzenia złożonego Z jest (jak wiadomo) równe **iloczynowi** prawdopodobieństw zdarzeń składowych Z_1 oraz Z_2 :

$$p(Z) = p(Z_1)\, p(Z_2)$$

Z postulatu Shannona wynika inna zależność:

$$H(Z) = H\left(p(Z_1)\, p(Z_2)\right) = H(Z_1) + H(Z_2)$$

Jest tylko jedna funkcja, która zamienia iloczyn na sumę:

logarytm

Zatem z postulatu

$H(Z) = H\,(p(Z_1)\,p(Z_2)\,) = H(Z_1) + H(Z_2)$

przy założeniu że

$p(Z) = p(Z_1)\,p(Z_2)$

wynika, że funkcja określająca miarę niepewności jakiegoś zdarzenia powinna być zdefiniowana jako **logarytm prawdopodobieństwa**.

Łatwo zauważyć, że przy takiej definicji spełniony jest także – niejako z automatu – postulat żeby niepewność zdarzenia pewnego

$p(Z) = 1$

wynosiła **zero**, bo logarytm jedynki ma wartość zero.

$p(Z) = 1 \quad \leftrightarrow \quad H(Z) = 0$

Odrobina kłopotu pojawia się w kontekście postulatu wyrażonego wzorami

$p(Z_1) > p(Z_2)$

$H(Z_1) < H(Z_2)$

Logarytm jest funkcją rosnącą, więc dla większych wartości argumentu (prawdopodobieństwa) będzie przyjmował większe wartości – a powinno być odwrotnie.

Ale jest na to prosta rada:

użyjemy znaku **minus** przed logarytmem

gdy wartość prawdopodobieństwa będzie rosła i w ślad za tym będzie rosła wartość logarytmu – to wartość niepewności będzie malała.

I o to chodzi!

Zatem mamy już gotową definicję miary niepewności w jej podstawowej postaci.

Oto ona:

$H(Z) = - \log_B p(Z)$

Otwarta jest jeszcze sprawa **jednostek**, w jakich będziemy te niepewność wyrażać.

Zwróćmy uwagę, że jeśli wybierzemy określone jednostki dla wyrażania niepewności – to dokładnie te same jednostki będę wyrażały miary ilości informacji.

Definicja jednostki miary niepewności (i miary ilości informacji) ma ścisły związek z symbolem *B* który pojawił się we wzorze

$$H(Z) = - \log_B p(Z)$$

jako podstawa logarytmu, którym się chcemy posłużyć.

Gdy Shannon tworzył zręby swojej teorii w powszechnym użyciu były logarytmy dziesiętne, które zastępowały współczesne kalkulatory.

W związku z tym w pierwszych pracach przyjmowano $B = 10$ i na tej podstawie stworzono jednostkę ilości informacji nazwaną *decimal information unit* - w skrócie ***dit***.

Potem do badania podstaw teorii informacji zabrali się matematycy, którzy cenią zalety logarytmów naturalnych, to znaczy takich, dla których $B = e = 2{,}7182818\ldots$

Po wprowadzeniu tej podstawy logarytmów pojawiła się inna jednostka określana jako *natural information unit* - w skrócie ***nit***.

Jednak rozwój techniki komputerowej i związana z tym rozwojem „kariera" systemu dwójkowego spowodowały, że coraz częściej zaczęto używać $B = 2$ w wyniku czego zaczęła zyskiwać na popularności jednostka określana jako *binary information unit* - w skrócie ***bit***.

Być może niektórzy z Państwa w tym momencie się zaniepokoili – wszak **bit** oznacza w informatyce coś zupełnie innego!

Jest to niepokój całkowicie nieuzasadniony,
gdyż wszystko się tu zgadza.

W technice komputerowej nazwą *bit* określana jest taka część mikroprocesora, rejestru lub pamięci cyfrowej, w której można zapisać albo 0 albo 1.

Również przy przesyłaniu informacji używa się pojęcia *bit* do określenia takiego fragmentu sygnału, który może być zerem lub jedynką.

Zastanówmy się, jaką ilość informacji (w myśl podanych wyżej definicji) może przechować lub przenieść taki element systemu informatycznego, w którym możliwe są dwa stany?

Początkowe prawdopodobieństwo, że w tym elemencie będzie zero (lub jedynka) wynosi dokładnie 0,5.

Zatem niepewność zdarzenia, jakim jest wpisanie do tego elementu takiej lub innej cyfry wynosi:

$$H = - \log_2 0{,}5 = 1$$

A zatem ilość informacji, jaka mieści się w jednym bicie wynosi dokładnie 1 bit!

Czyli te tradycyjne znaczenia słowa *bit* nie kłócą się z wyżej zdefiniowaną jednostką miary stopnia niepewności i miary ilości informacji i wszystko pięknie się zgadza!

Dygresja:

Dlaczego w informatyce wszystkie informacje wyraża się za pomocą bitów?

Potrzebna będzie odrobina wiedzy na temat pozycyjnych systemów liczenia, więc od tego zaczniemy

Pierwsze zapisy liczb były robione pismem klinowym

Starożytny rachunek za zakup zboża

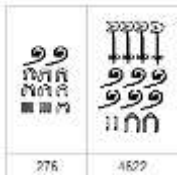
Przygoda ludzi z matematyką zaczęła się od używania systemów niepozycyjnych - **System Babiloński**

W systemie niepozycyjnym każdy symbol reprezentuje pewną (zawsze tę samą) wartość niezależnie od tego, w jaki miejscu występuje

System Egipski

1	10	100	1000	10000	100000	10^6

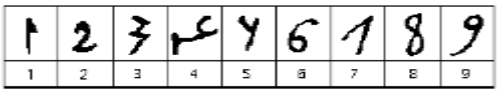
Egipskie liczebniki hieroglifowe



System Grecki

Nasze cyfry nazywamy „*arabskie*”, ale Arabowie używają innych znaków!

Nasze cyfry tak naprawdę pochodzą z Indii



Ważne jest to, że wartość reprezentowana przez cyfrę zależy od pozycji cyfry w liczbie.

Oznaczając znakiem ☺ dowolne inne cyfry w liczbie możemy symbol 3 odczytać jako wartość:

☺ ☺ ☺ ☺ ☺ ☺ 3 - trzy
☺ ☺ ☺ ☺ ☺ 3 ☺ - trzydzieści
☺ ☺ ☺ ☺ 3 ☺ ☺ - trzysta
☺ ☺ ☺ 3 ☺ ☺ ☺ - trzy tysiące
☺ ☺ 3 ☺ ☺ ☺ ☺ - trzydzieści tysięcy
☺ 3 ☺ ☺ ☺ ☺ ☺ - trzysta tysięcy
3 ☺ ☺ ☺ ☺ ☺ ☺ - trzy miliony

Wartość zależy od **pozycji** cyfry w liczbie, więc jest to system **pozycyjny**

Pozycyjne systemy liczenia (1)

- **Każda liczba całkowita $N \geq 2$ może być podstawą systemu liczenia (mówimy wówczas o systemie o podstawie *N*).**
- System dziesiętny (system o podstawie równej 10) używany na co dzień jest przykładem pozycyjnego systemu liczenia
- System binarny jest także przykładem **pozycyjnego** systemu liczenia

106

Pozycyjne systemy liczenia (2)

- Do zapisu *liczb* wykorzystywane są *cyfry*, których liczba jest taka sama, jak podstawa systemu
  - w systemie dwójkowym: 0, 1;
  - w systemie dziesiętnym: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9;
  - w systemie ósemkowym: 0, 1, 2, 3, 4, 5, 6, 7;
  - w systemie szesnastkowym: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F;
  - w systemie o podstawie N: 0, 1, ..., N - 1

107

Obliczanie wartości liczby (1)

- system dziesiętny (system o podstawie 10)

$353_{(10)} = 3 * 100 + 5 * 10 + 3 * 1 = 3 * 10^2 + 5 * 10^1 + 3 * 10^0$

$2{,}4_{(10)} = 2 * 1 + 4 * 0{,}1 = 2 * 10^0 + 4 * 10^{-1}$

$$W = \sum_i c_i \cdot 10^i$$

108

Obliczanie wartości liczby (2)

- System dwójkowy (system o podstawie 2)
  - Analizowana liczba $101,11_{(2)}$

pozycja (i)	2	1	0	-1	-2
2^i	4	2	1	0,5	0,25
cyfra	1	0	1	1	1
cyfra * 2^i	4	0	1	0,5	0,25
suma	5,75				

$$W = \sum_i c_i \cdot 2^i$$

109

Interpretacja tej samej liczby w systemie dziesiętnym i w systemie dwójkowym

1965

5
6 x 10 = 60
9 x 10 x 10 =900
1 x 10 x 10 x 10 = 1000

5 + 60 + 900 + 1000 = 1965

1965 = 11110101101

1
0 x 2 = 0
1 x 2 x 2 = 4
1 x 2 x 2 x 2 = 8
0 x 2 x 2 x 2 x 2 = 0
1 x 2 x 2 x 2 x 2 x 2 = 32, ...
1 x 2 x 2 x 2 x 2 x 2 x 2 x 2 x 2 x 2 x 2 = 1024

1 + 0 + 4 + 8 + 0 + 32 + 0 + 128 + 256 + 512 + 1024 = 1965

To, co warto zapamiętać:

Tabela 2.3. Liczby dziesiętne 1 – 16 i ich binarne odpowiedniki

Dziesiętnie:	1	2	3	4	5	6	7	8
Dwójkowo:	1	10	11	100	101	110	111	1000
Dziesiętnie:	9	10	11	12	13	14	15	16
Dwójkowo:	1001	1010	1011	1100	1101	1110	1111	10000

111

No ale dlaczego system dwójkowy?

- Bo jest naturalna reprezentacja techniczna
- Bo jest **odporny** na zakłócenia
- Bo **arytmetyka** dwójkowa jest wyjątkowo **łatwa**
- Bo jest **bliski optymalności** pod względem kosztu wykonania komputera

112

Ile kosztuje przedstawienie *L* liczb w systemie o podstawie *N*?

$$L = N^m$$

gdzie *m* jest liczbą pozycji liczby

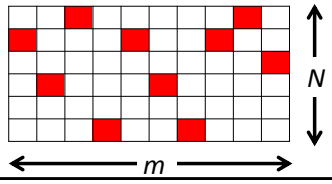
Przykład: Dla *N=10* i *m=2* można przedstawić liczby od *00* do *99*, a jest ich *100* (10^2)

113

Koszt techniczny *K* zbudowania urządzenia, które może przechowywać *m* cyfrowe liczby w systemie o podstawie *N:*

$$K = m\,N$$

Przykład:



Uwaga:

Rozważania na następnym slajdzie wymagają użycia rachunku różniczkowego.

Osoby nie znające podstaw analizy matematycznej proszone są o zamknięcie oczu i nie zwracanie uwagi na to, co się tu będzie działo.

Za chwilę wszystko wróci do normy ☺

Poszukujemy takiego *N* które zapewni *K = min* przy *L = const*

$$L = N^m$$

$$\ln L = m \ln N$$

$$m = \frac{\ln L}{\ln N}$$

$$K = m\,N = \frac{\ln L}{\ln N}\,N$$

$$\frac{dK}{dN} = \ln L \,\frac{\ln N - 1}{\ln N \,\ln N} = 0$$

ln N = 1

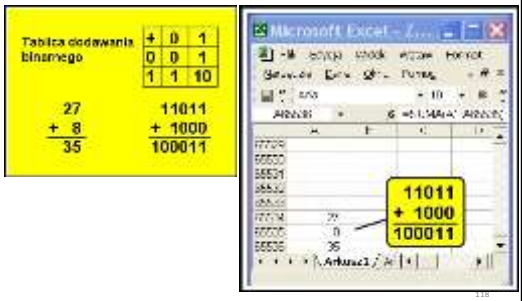
116

Optymalny system liczbowy ma podstawę

N = e = 2,718…

Najbliżej *e* jest N = 3, ale stosuje się N = 2 i też jest dobrze ☺

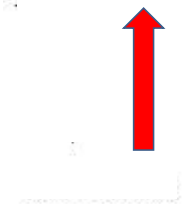
Komputer wszystkie działania wykonuje na liczbach dwójkowych, chociaż przedstawia je dziesiętnie



118

Zamiana liczb dziesiętnych na system binarny (liczba całkowita, slajd animowany)

$29_{(10)} = ?_{(2)}$



119

Zamiana liczb dziesiętnych na system binarny (ułamek)

$0{,}625_{(10)} = ?_{(2)}$

	0	625
	1	250
↓	0	500
	1	000

$0{,}625_{(10)} = 0{,}101_{(2)}$

120

Zamiana liczb dziesiętnych na system binarny (pojawiające się czasem problemy - animacja)

$0{,}35_{(10)} = ?_{(2)}$

Wniosek: ułamek dziesiętny o skończonej liczbie cyfr może wymagać ułamka binarnego o nieskończonej liczbie cyfr.

121

Prostota arytmetyki binarnej

Tabliczka dodawania:

$$1_2+1_2 = 10_2 = 2_{10}$$

To wszystko, bo zero dodane do czegokolwiek nic nie zmienia

Tabliczka mnożenia:

$$1_2*1_2 = 1_2$$

To wszystko, bo zero mnożone przez czegokolwiek daje zawsze zero

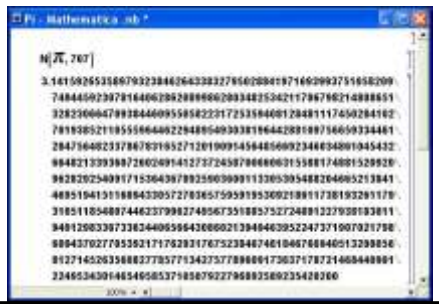
Mnożenie binarne (animacja)

```
1011   =   11
x1101  =   13
-----
```

123

Ta prostota (oraz pomysłowość elektroników) sprawiają, że obliczenia komputerowe mogą być bardzo szybkie.

Spektakularny przykład szybkości obliczeń komputerów - poniższe 707 cyfr liczby π, William Shanks w XIX wieku obliczał przez **20 lat**, komputer tysięczną część sekundy



124

Poszerzona dygresja:

Maszynowa reprezentacja informacji nienumerycznych.

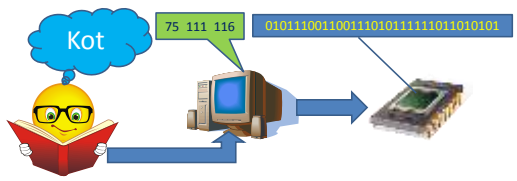
125

Reprezentacja znaków

- W systemie komputerowym każdy tekst przechowywany jest w postaci **cyfrowej**
- Jest to możliwe, ponieważ każdy znak (litera, cyfra, znak przestankowy itp.) jest zamieniany w ***kod znaku*** (będący liczbą całkowitą).
- Liczbami binarnymi są także **informacje sterujące**, określające format tekstu (na przykład wielkość i kolor czcionki, rozmiar strony, wielkość marginesu itp.)

126

Trzy sposoby widzenia tekstu



Kod ASCII

- *ASCII - American Standard Code for Information Interchange.*
  - –w wersji podstawowej - 7 bitowy (128 znaków)
  - –w wersji rozszerzonej - 8 bitowy (256 znaków)

128

Kod ASCII

Kody	Znaki
0 do 32	- znaki sterujące a w tym: 8 - wymazanie ostatniego znaku (backspace) 9 – tabulator; 10 - nowa linia; 12 - nowa strona 13 - "powrót karetki" czyli cofniecie do poczatku linii 32 - spacja czyli odstęp miedzy słowami
33 do 47	! " # $ % & ' ( ) * + , - . /
48 do 57	0 1 2 3 4 5 6 7 8 9
58 do 64	: ; < = > ? @
65 do 90	ABCDEFGHIJKLMNOPQRSTUVWXYZ
91 do 96	[\ ] ^ _ `
97 do 123	abcdefghijklmnopqrstuvwxyz
124 do 127	{|}~

Kod ASCII

kod	znak	kod	znak	kod	znak	kod	znak	kod	znak
32		51	3	70	F	89	Y	108	l
33	!	52	4	71	G	90	Z	109	m
34	"	53	5	72	H	91	[	110	n
35	#	54	6	73	I	92	\	111	o
36	$	55	7	74	J	93	]	112	p
37	%	56	8	75	K	94	^	113	q
38	&	57	9	76	L	95	_	114	r
39	'	58	:	77	M	96	`	115	s
40	(	59	;	78	N	97	a	116	t
41	)	60	<	79	O	98	b	117	u
42	*	61	=	80	P	99	c	118	v
43	+	62	>	81	Q	100	d	119	w
44	,	63	?	82	R	101	e	120	x
45	-	64	@	83	S	102	f	121	y
46	.	65	A	84	T	103	g	122	z
47	/	66	B	85	U	104	h	123	{
48	0	67	C	86	V	105	i	124	|
49	1	68	D	87	W	106	j	125	}
50	2	69	E	88	X	107	k	126	~

130

Kod ASCII – problem polskich znaków

- Znaki specyficzne dla języka polskiego (ąćęłńóśźżĄĆĘŁŃÓŚŹŻ) przypisane mają kody większe od 128 → utrudnia to proces sortowanie, wyszukiwania i porządkowania danych!!!
- Istnieją różne sposoby kodowania polskich znaków → utrudnia to wymianę dokumentów tekstowych pomiędzy różnymi systemami.

- *Podstawowe sposoby kodowania polskich znaków:*
  - **MS Windows CP 1250** (Windows Latin-2, Windows-1250) - sposób kodowania wprowadzony przez firmę Microsoft wraz z systemem Windows 3.11 PL
  - **ISO Latin-2** (ISO-8859-2, Polska Norma PN-93 T-42118) - sposób kodowania określony przez ISO, stosowany powszechnie w Internecie.

131

Polskie znaki – standardy kodowania

Znak	Windows CP-1250		ISO Latin-2 (ISO-8859-2) PN-93 T-42118	
	(dec)	(hex)	(dec)	(hex)
Ą	165	A5	161	A1
ą	185	B9	177	B1
Ć	198	C6	198	C6
ć	230	E6	230	E6
Ę	202	CA	202	CA
ę	234	EA	234	EA
Ł	163	A3	163	A3
ł	179	B3	179	B3
Ń	209	D1	209	D1
ń	241	F1	241	F1
Ó	211	D3	211	D3
ó	243	F3	243	F3
Ś	140	8C	166	A6
ś	156	9C	182	B6
Ź	175	AF	175	AF
ź	191	BF	191	BF
Ż	143	8F	172	AC
ż	159	9F	188	BC

132

Unicode (Unikod)

- ***Unikod*** (ang. *Unicode* lub *UCS - Universal Character Set*) – sposób kodowania znaków uwzględniający większości wykorzystywanych znaków w różnych językach na całym świecie.
- Znaki uwzględnione w Unikodzie podzielone zostały na:
  - *podstawowy zestaw znaków* (określany jako Basic Multilingual Plane - BMP lub Plane 0) – dla tych znaków stosowane są kody 16 bitowe;
  - *dodatkowy zestaw znaków* – stosowane są kody 32 bitowe.

133

Reprezentacja unikodów

- ***UTF*** - *Unicode Transformation Format* – metody przechowywania unikodów w pamięci komputera:
  - UTF-8 – kody znaków wchodzących w skład podstawowego zestawu ASCII zapisywane są jako wartości jednobajtowe; pozostałe kody zapisywane są na dwóch, trzech, czterech, pięciu lub sześciu bajtach (znaki o kodach zapisywanych na trzech i większej liczbie bajtów spotykane są we współczesnych językach bardzo rzadko);
  - UTF-16 – kody znaków zapisywane są na dwóch, trzech lub czterech bajtach (najczęściej wykorzystywane są znaki o kodach dwubajtowych);
  - UTF-32 – kody znaków zapisywane są na 4 bajtach.

134

Unicode jest międzynarodowym standardem zbioru znaków, który może być wykorzystany do pisania dokumentów w niemalże każdym istniejącym języku.

Wersja 4.0.1 z czerwca 2004 roku zawiera **96 447** znaków z prawie wszystkich języków na świecie.

Unicode z łatwością mieści cały alfabet łaciński, ale również pismo pochodzenia greckiego, włączając starożytne i współczesne odmiany oraz cyrylicę używaną np. w Serbii.

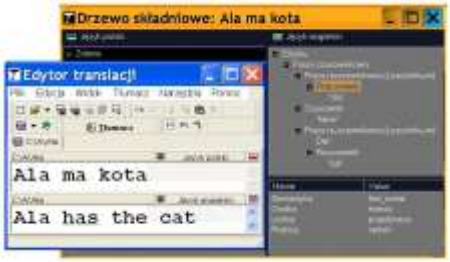
Prawdopodobnie jedna osoba na milion obywateli świata obecnie mówi językiem, który nie może być sensownie przedstawiony w Unicode.

135

W tekstach zgromadzonych w formie cyfrowej łatwo jest wyszukiwać potrzebne informacje

136

Komputerowy zapis tekstów ma także wiele innych zalet – między innymi pozwala wykonywać automatyczne tłumaczenia z jednego języka na inny



137

Koniec dygresji – wracamy do teorii informacji

Wiemy już, jak się obliczą *H(Z)*.

A co z *H(Z/W)* ?

Potrzebne jest to nam do wzoru:

$$I(W/Z) = H(Z) - H(Z/W)$$

Dokładnie tak samo, tylko bierzemy wtedy pod uwagę inne prawdopodobieństwa.

Zamiast prawdopodobieństwa apriorycznego *p(Z)* wstawiamy do odpowiedniego wzoru *prawdopodobieństwo warunkowe*

p(Z/W)

czyli prawdopodobieństwo zdarzenia *Z* po otrzymaniu komunikatu *W*.

Nadszedł moment, by wprowadzić do naszych rozważań pewną ważną nazwę.

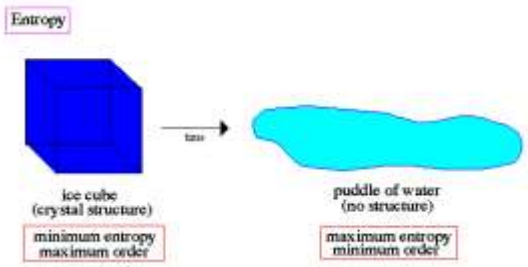
Otóż miarę niepewności *H(Z)* i wszystkie miary pokrewne z nią nazywa się **entropią.**

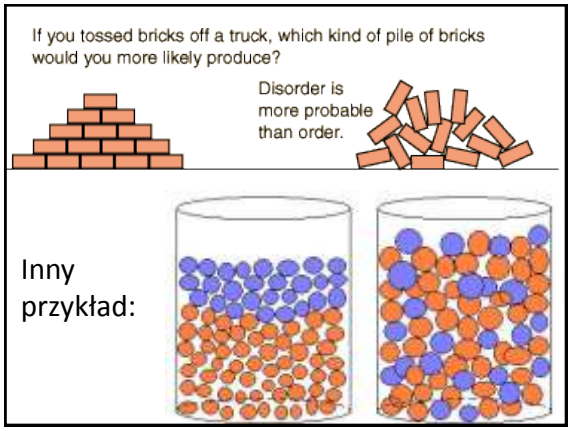
Entropia jest miarą chaosu

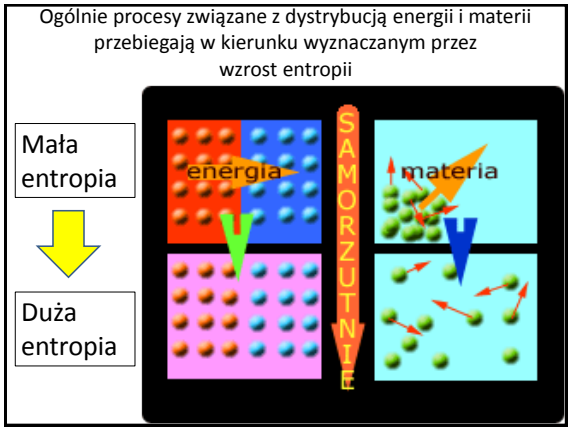
Pojęcie to wcześniej wprowadzili fizycy dla wyjaśnienia nieodwracalnego charakteru pewnych procesów termodynamicznych.

Fizycy stwierdzili, że wszystkie procesy w Przyrodzie przebiegają w taki sposób, że entropia ustawicznie rośnie.

Przykład procesu, w którym gołym okiem widać wzrost entropii







Na przykład proces spontanicznego niszczenia i rozpadu starych budowli jest nieunikniony, bo związany jest ze wzrostem entropii, która jest **przeciwieństwem informacji** (w tym przypadku zawartej w strukturze budowli).

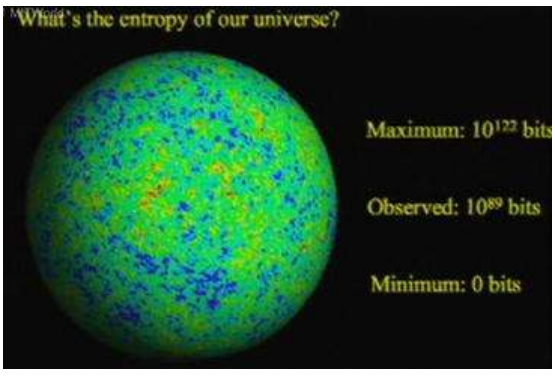
Nie obserwujemy tego, by cegły same się poukładały tworząc nowy dom bo wtedy entropia musiałaby zmaleć, a to jest niemożliwe.

Tak samo przekazywana z ust do ust wiadomość ulega zniekształceniu i zakłóceniu, natomiast nie zdarza się, by w miarę przekazywania i powtarzania wiadomości było w niej coraz więcej sensu i prawdziwej informacji.

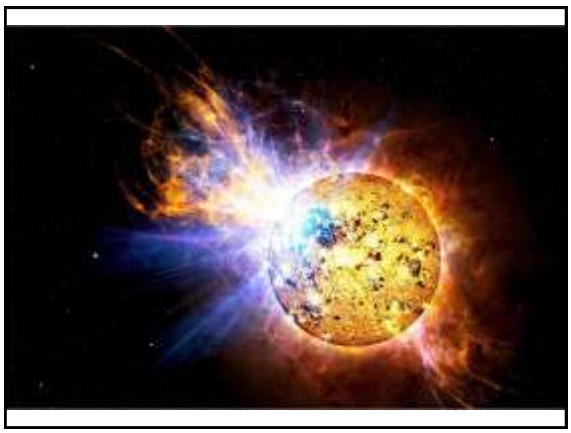
Prawo wzrostu entropii tutaj także działa!

Rozważania dotyczące entropii doprowadziły w fizyce do bardzo daleko idących wniosków, na przykład pozwoliły przewidzieć, że Wszechświat w którym teraz żyjemy, czeka zagłada.

Życiodajne światło Słońca (i innych gwiazd) wygaśnie w przyszłości na skutek wyczerpania się zasobów paliwa jądrowego w ich wnętrzach.

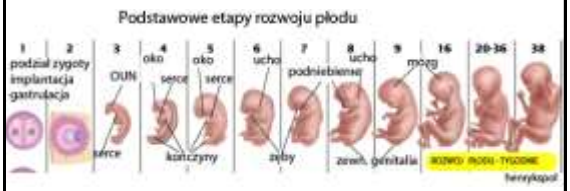


Ponieważ proces szafowania energią, który jest normą wśród gwiazd i galaktyk, jest **nieodwracalny** (bo charakteryzuje go wzrost entropii), dlatego „śmierć cieplna” całego Kosmosu jest nieunikniona.



Entropia pozwala też objaśnić wiele zjawisk w biologii, między innymi podstawowe pytania o naturę życia i śmierci.

Jedyny proces, który wydaje się podążać w kierunku wzrastającej organizacji (malejącej entropii) – to rozwój żywego organizmu



Jednak procesy wzrostu entropii biorą górę nad procesami rozwoju.

Dlatego starzejemy się i umieramy…

Animacja

Wróćmy jednak do naszego wykładu o teorii informacji

W rozważaniach przytoczonych wcześniej otrzymaliśmy oszacowanie maksymalnej ilości informacji, jakiej możemy się spodziewać w wiadomości *W* na temat zdarzenia *Z*.

$$I(W/Z) \leq H(Z)$$

Jednak oszacowanie to odwoływało się tylko do entropii zdarzenia *Z*, nie wiążąc tej maksymalnej ilości informacji z cechami samej wiadomości *W*.

To jest niekorzystne, bo na zdarzenia których dotyczą wiadomości wpływu nie mamy, natomiast na sposób formułowania wiadomości – owszem.

Możemy jednak wykorzystać fakt, że prawdopodobieństwo łączne równoczesnego pojawienia się określonej wiadomości *W* i zdarzenia *Z* można wyrazić na dwa sposoby:

$$p(WZ) = p(Z)\, p(W/Z) = p(W)\, p(Z/W)$$

$$p(WZ) = p(Z)\, p(W/Z) = p(W)\, p(Z/W)$$

Zauważmy, że we wzorze tym musieliśmy użyć **prawdopodobieństw warunkowych**, bo tu nie możemy użyć modelu

$$p(WZ) = p(W)\, p(Z)$$

obowiązującego dla zdarzeń **niezależnych.**

Zdarzenie *Z* i wiadomość o tym zdarzeniu *W* są obiektami **zależnymi** od siebie nawzajem, i to bardzo silnie zależnymi, bo na tym polega sens wiadomości.

Posługując się analogią ze wzorem

$$H(Z) = H\,(p(Z_1)\, p(Z_2)\,) = H(Z_1) + H(Z_2)$$

Obowiązującym przy założeniu

$$p(Z) = p(Z_1)\, p(Z_2)$$

możemy tożsamość

$$p(WZ) = p(Z)\, p(W/Z) = p(W)\, p(Z/W)$$

przepisać w postaci

$$H(Z) + H(W/Z) = H(W) + H(Z/W)$$

Ale ze wzoru

$$H(Z) + H(W/Z) = H(W) + H(Z/W)$$

wynika, że

$$H(Z) - H(Z/W) = H(W) - H(W/Z)$$

(Otrzymaliśmy ten wzór przenosząc na drugą stronę obie entropie warunkowe)

Zauważmy, że lewa strona tożsamości

$$H(Z) - H(Z/W) = H(W) - H(W/Z)$$

jest identyczna z wyrażeniem występującym we wzorze definicyjnym

$$I(W/Z) = H(Z) - H(Z/W)$$

W związku z tym wzór definiujący miarę ilości informacji można przepisać w postaci:

$$I(W/Z) = H(W) - H(W/Z)$$

Ten wzór mówi coś ważnego i przydatnego w praktyce:

Otóż największa ilość informacji, jaką może przenieść pewna wiadomość jest ograniczona od góry przez aprioryczną entropię tej wiadomości $H(W)$.

Składnik $H(W/Z)$ może **pogorszyć** tę maksymalną (teoretyczną) wydajność informacyjną, nie może jej natomiast zwiększyć.

$$I(W/Z) = H(W) - H(W/Z)$$

Nawiasem mówiąc przyjrzyjmy się, co w istocie oznacza składnik $H(W/Z)$.

$$I(W/Z) = H(W) - H(W/Z)$$

Jest to miara stopnia niepewności co do tego, jaka będzie wysłana wiadomość W gdy zdarzenie Z, o którym ta wiadomość ma informować, będzie już całkowicie znane.

Niezerowa wartość $H(W/Z)$ oznacza, że system generujący wiadomości szwankuje:

- nie wiadomo, co napisze dziennikarz, gdy już dokładnie widomo, co naprawdę zaszło,
- nie wiadomo jako sygnał wyśle czujnik, chociaż wiadomo, jaką wartość ma mierzony parametr *itp.*

Słowem niezerowa wartość $H(W/Z)$ to kompromitacja dla systemu informacyjnego!

Niemniej ponieważ niezerowa wartość $H(W/Z)$ może wystąpić – zapiszemy zależność:

$$I(W/Z) \leq H(W)$$

Z zależności

$I(W/Z) \leq H(W)$

wynika następujące zalecenie:

Żeby przenieść dużo informacji trzeba zapewnić możliwie dużą początkową entropię wiadomości.

W jaki sposób w praktyce spełnić to zalecenie?

Aby odpowiedzieć na to pytanie spróbujmy przeanalizować strukturę wiadomości.

Różnych form reprezentacji wiadomości jest bardzo dużo.

Na przykład może to być tekst, tabela liczb, mapa, obraz, nagranie dźwiękowe, film wideo itp.

Ale każdą z tych form można sprowadzić do jednego modelu: wiadomością jest ciąg jakichś **symboli**.

W dalszym ciągu rozważać będziemy model wiadomości tekstowej, bo taką najłatwiej będzie sobie wyobrazić.

Ale wnioski jakie sformułujemy, będą się odnosiły do wiadomości w dowolnej formie!

Zresztą dzisiaj, w dobie komputerów i Internetu, wszelkie wiadomości ostatecznie mają postać serii zer i jedynek…

Rozważmy symbole, z jakich składa się tekst – czyli po prostu litery.

Proces czytania tekstu polega na ciągu zdarzeń, a każde zdarzenie polega na odczytaniu jednej (kolejnej) litery.

Jaka jest niepewność co do tego, jaką następną literę odczytamy i jaka jest w związku z tym jej entropia?

Różne litery mają różne prawdopodobieństwa, więc musimy rozważyć szereg możliwości.

Ponumerujmy litery alfabetu kolejnymi numerami od 1 do *N* i przypiszmy im odpowiednie prawdopodobieństwa:

$$p_1,\ p_2, \ldots, p_N$$

Suma tych prawdopodobieństw musi wynosić 1, ponieważ jest **pewne**, że **jakąś** literę odczytamy:

$$\sum_{i=1}^{N} p_i = 1$$

UWAGA: Spacja (odstęp między wyrazami, puste miejsce) też jest traktowana jako litera!

Z każdą z tych liter związana jest niepewność wynikająca ze wzoru

$$H(Z) = - \log_B p(Z)$$

mamy więc zbiór wartości entropii w postaci ciągu:

$$- \log_2 p_1, - \log_2 p_2, \ldots, - \log_2 p_N$$

Uwaga: Tu przypomnienie ze statystyki:

Jeśli mamy **zmienną losową**, która może przyjmować wartości:

$$x_1,\ x_2, \ldots, x_N$$

odpowiednio z prawdopodobieństwami

$$p_1,\ p_2, \ldots, p_N$$

to wartość oczekiwaną zmiennej *X* wyznaczamy ze wzoru

$$\bar{X} = \sum_{i=1}^{N} p_i\ x_i$$

Formułę

$$\bar{X} = \sum_{i=1}^{N} p_i\ x_i$$

nazywa się niekiedy

„nadzieją matematyczną” ☺

Wracamy do oszacowania entropii pojedynczej litery w tekście

Entropia każdej kolejnej litery tekstu jest **zmienną losową** o wartościach danych formułą

$- \log_2 p_1, - \log_2 p_2, \ldots, - \log_2 p_N$

przy czym poszczególne wartości tej zmiennej losowej są przyjmowane z prawdopodobieństwami:

$p_1, p_2, \ldots, p_N$

Skoro entropia pojedynczego znaku w tekście jest zmienną losową, to dla jej prawidłowego oszacowania należy posłużyć się wzorem na **wartość oczekiwaną**.

$$\bar{X} = \sum_{i=1}^{N} p_i \, x_i$$

przy $x_i = - \log_2 p_i$

W rezultacie otrzymamy oczekiwaną wartość entropii przypadającej na jeden symbol w tekście *H(s):*

$$H(s) = -\sum_{i=1}^{N} p_i \log_2 p_i$$

Mając oszacowaną entropię *H(s)* przypadającą na jeden symbol w tekście i znając liczbę znaków *K* występujących w rozważanej wiadomości *W* możemy oszacować jej entropię

$H(W) = K\,H(s)$

To nam pozwoli oszacować, czy wiadomość ta będzie mogła przenieść wymaganą ilość informacji.

Pojawia się w tym momencie kluczowa kwestia:

Jakie właściwości powinien mieć zbiór znaków używanych do tworzenia wiadomości *W* żeby mogła ona przenosić wymaganą ilość informacji **przy użyciu możliwie niewielkiej liczby znaków**?

Zwróćmy uwagę, że każdy znak kosztuje.

Przy pisaniu na papierze jest to koszt tego kawałka strony, którą ten znak zajmuje.

Przy przechowywaniu w komputerze każdy znak zajmuje jakiś fragment pamięci.

Przy przesyłaniu siecią wysłanie każdego znaku wydłuża czas transmisji.

Zatem im więcej informacji przeniesie jeden znak – tym lepiej.

Kiedy jeden znak będzie przenosił najwięcej informacji?

Gdy jego entropia wyrażona wzorem

$$H(s) = -\sum_{i=1}^{N} p_i \log_2 p_i$$

będzie jak największa!

Jak jednak sprawić, by entropia jednego znaku była duża?

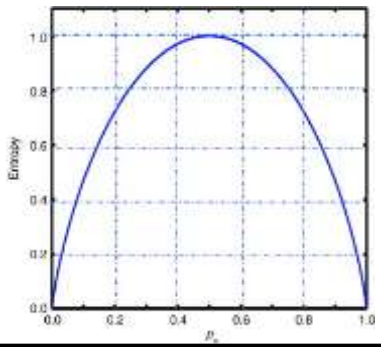
Można by to było udowodnić matematycznie, ale dowód jest długi i trudny.

Ale można też posłużyć się zdrowym rozsądkiem, zastanawiając się, kiedy nasza niepewność dotycząca tego, jaki będzie następny znak w tekście, będzie największa?

Oczywiście wtedy, kiedy każdy znak będzie się mógł pojawić…

z takim samym prawdopodobieństwem!

Dla N = 2



Jakakolwiek preferencja zmniejsza niepewność, bo gdy jedne znaki są bardziej prawdopodobne, a inne mniej – to ułatwia zadanie komuś, kto chciałby te znaki odgadywać.

Na takie nierównomierne prawdopodobieństwa „polują” szpiedzy, usiłujący złamać szyfr chroniący tajne informacje!

Dokładnie taki sam wniosek dostarcza analiza matematyczna wzoru

$$H(s) = -\sum_{i=1}^{N} p_i \log_2 p_i$$

Będzie on przyjmował wartość maksymalną gdy

$$p_1 = p_2 = \ldots = p_N$$

Gdy dodatkowo uwzględnimy zależność

$$\sum_{i=1}^{N} p_i = 1$$

to stanie się oczywiste, że dla każdego znaku musi zachodzić zależność

$$p_i = \frac{1}{N}$$

Maksymalna entropia przypadająca na jeden symbol, gdy są one wszystkie jednakowo prawdopodobne wynosi

$$H_{max}(s) = -\sum_{i=1}^{N} p_i \log_2 p_i = -\sum_{i=1}^{N} \frac{1}{N} \log_2 \frac{1}{N} = \log_2 N$$

Co ciekawe:

przypuszczenie, że możliwość przenoszenia informacji przez system kodujący wykorzystujący *N* symboli jest proporcjonalna do *log N* wyrażał już sto lat wcześniej Hartley – ale nie potrafił tego uzasadnić!

Prawo logarytmiczne pojawia się też przy analizie działania naszych zmysłów

Biolodzy już dawno odkryli zależność pomiędzy fizyczną miarą bodźca (na przykład energią dźwięku) a reakcją zmysłów (subiektywnie odczuwaną głośnością dźwięku).

Uzasadnia to prawo Webera-Fechnera

Ernst Heinrich Weber stwierdził:

Dla małych wartości bodźca B zauważalne są małe jego zmiany dB, a dla dużych B dla dostrzeżenia różnicy potrzebne są duże dB.

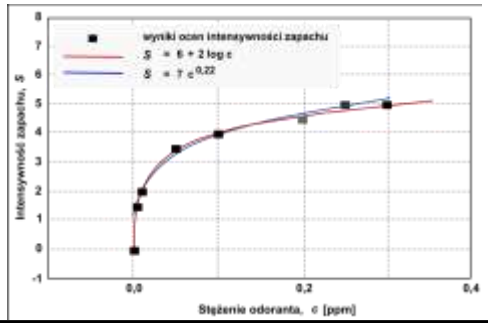
Subiektywna zmiana wrażenia dw jest proporcjonalna do względnej zmiany bodźca.

Gustav Theodor Fechner wyraził to tak:

$$dw = k\frac{dB}{B},$$

$$w = k \cdot \ln \frac{B}{B_0}$$

To prawo dotyczy wszystkich zmysłów, dla których bodziec jest mierzalny: wzroku, słuchu, węchu, siły, temperatury



Spróbujmy zbadać z punktu widzenia teorii informacji system jakim jest język

Każda litera języka przenosiłaby największą ilość informacji, gdyby wszystkie litery były tak samo prawdopodobne.

Wtedy mielibyśmy

$$H_{max}(s) = -\sum_{i=1}^{N} p_i \log_2 p_i = -\sum_{i=1}^{N} \frac{1}{N} \log_2 \frac{1}{N} = \log_2 N$$

W języku polskim, dla którego *N* wynosi 32

$H_{max}(s)$ = 5 bit/symbol

Jednak litery w języku polskim mają zróżnicowane prawdopodobieństwa, co ilustruje tabela

s_i	p_i	s_i	p_i	s_i	p_i	s_i	p_i
a	0,080	g	0,010	m	0,024	t	0,024
b	0,013	h	0,010	n	0,047	u	0,018
c	0,038	i	0,070	o	0,071	w	0,036
d	0,030	j	0,019	p	0,024	y	0,032
e	0,069	k	0,027	r	0,035	z	0,058
f	0,001	l	0,031	s	0,038	spacja	0,172

W tabeli tej nie uwzględniono polskich znaków diakrytycznych, to znaczy utożsamiono ***ą*** z ***a***, ***ć*** z ***ć***, ***ę*** z ***e*** itd. Do jednego „pudełka" zapakowano też znaki ***z, ź, ż.***

Dzięki temu można było pokazać **nierównomierność** rozkładu prawdopodobieństwa różnych liter nie tworząc nadmiernie wielkiej tabelki.

Co więcej, dokładne rozważania dotyczące wielkości niepewności określonej litery w tekście zmuszają do uwzględnienia kontekstu.

Znając wcześniejsze litery można łatwiej odgadnąć następną.

Na przykład widząc napis **KRAKÓ♥** możemy łatwo zgadnąć, że końcowe serduszko zastępuje literę **W**.

Jesteśmy tego pewni, więc prawdopodobieństwo litery **W** w tym miejscu i w tym kontekście wynosi 1.

Tymczasem prawdopodobieństwo tej samej litery bez kontekstu ma wartość zaledwie 0,036!

Jak widać, przy określaniu rzeczywistej entropii pojedynczej litery powinno się brać pod uwagę **prawdopodobieństwa warunkowe.**

Jednak wtedy zamiast prostego wzoru :

$$H(s) = -\sum_{i=1}^{N} p_i \log_2 p_i$$

musimy używać bardzo skomplikowanych formuł.

Darujmy sobie szczegóły i podajmy od razu końcowy wynik:

Otóż rzeczywista entropia jednej litery w tekście napisanym w języku polskim wynosi

$H(s)$ = 1 bit/symbol

Przypomnijmy, że

$H_{max}(s)$ = 5 bit/symbol

Wynik jest DRAMATYCZNY!

Wykorzystujemy zaledwie 20% pojemności informacyjnej naszego języka!

4/5 tego, co Państwo piszecie w zeszytach albo czytacie w książkach jest niepotrzebne!

Nasze wszystkie książki, gazety, komunikaty mogłyby być **pięć razy mniejsze!**

To samo dotyczy przechowywania tekstów na dyskach komputerów czy też przesyłania ich przez sieć.

Wszędzie redundancja polskiego języka powoduje, że płacimy pięć razy więcej!

Czy powinniśmy się martwić tym, że mamy tak kiepski język?

Zanim zaczniemy rozdzierać szaty – wprowadźmy pewną miarę, pozwalającą oceniać NADMIAROWOŚĆ różnych systemów komunikacyjnych.

Tę miarę nazywamy REDUNDANCJĄ i wyrażamy wzorem:

$$R = \frac{H_{max}(s) - H(s)}{H_{max}(s)}$$

Redundancja pozwala oszacować, jaka część tekstu komunikatów przekazywanych w danym języku jest z informacyjnego punktu widzenia niepotrzebna.

Jak łatwo sprawdzić, redundancja języka polskiego wynosi 80%.
Redundancja języka angielskiego jest mniejsza (około 72%) i dlatego ten sam komunikat w języku angielskim jest zwykle **krótszy** niż w języku polskim.

Zbadano też inne języki

Okazało się, że redundancja we wszystkich językach jest podobnego rzędu.

Najmniejsza jest w języku angielskim.

W niemieckim większa niż w angielskim, ale mniejsza niż w polskim

Ale w rosyjskim jeszcze większa niż w polskim!

Możemy to również zobaczyć porównując długość tekstu wyrażającego zachwyt po angielsku, po niemiecku, po rosyjsku i po polsku (animacja).

Czy z tego wynika, że języki słowiańskie są gorsze?

Absolutnie nie, bo największą redundancję (znacznie większą od polskiej) wykazują języku romańskie, szczególnie hiszpański.

Prawie wszystkie języki świata mają redundancję na poziomie 70 – 88%

Badano także inne systemy komunikacji międzyludzkiej

Okazało się, że w łomocie bębnów niosącym wiadomości poprzez afrykańską dżunglę redundancja jest ponad 80%!

Na przykład „język” tam-tamów

To nie może być przypadek!

Skoro **wszystkie** języki wykazują tak dużą redundancję, to być może jest ona jednak do czegoś potrzebna?

Istotnie – jest potrzebna.

Do czego?

Żeby odpowiedzieć na to pytanie wyobraźmy sobie, że wysyłamy sms i pomyliliśmy się, pisząc

MJMA

zamiast

MAMA.

Jest wysoce prawdopodobne, że odbiorca sms w ogóle nie zauważy tej pomyłki, a nawet jeśli ją dostrzeże, to bez trudu domyśli się jaki nastąpił błąd i odczyta przesłane słowo prawidłowo.

Dzięki istnieniu redundancji jesteście Państwo w stanie przeczytać i zrozumieć poniższy tekst, chociaż brakuje w nim wielu liter.

Wy_aga się czas_m od nau__ycieli, by organi_owali na_czanie, mi_o, że _rakuje materia_ów odp_wiadaj_cy_ pla_owan_m cel_m. Cz_sto imp_owizuj_ wtedy i adaptuj_ to, co maj_.

Możemy się domyślać prawidłowego brzmienia niekompletnych albo przekłamanych komunikatów właśnie ze względu na obecność redundancji!

Jeśli jednak na tej samej klawiaturze wpisywać będziemy liczbę – na przykład PIN – 6262 (zwróćmy uwagę, że są tu użyte te same klawisze co w słowie MAMA!) i popełnimy taką samą pomyłkę, jak podana wyżej (MJMA zamiast MAMA), wysyłając w konsekwencji liczbę 6562 – to odbiorca sms nie ma żadnych szans, żeby wykryć, że w przysłanej wiadomości jest błąd, a tym bardziej nie zdoła w żaden sposób odgadnąć, jaka jest prawidłowa liczba.

Praktyczne korzyści z teorii informacji:

Kompresja danych

Dzięki temu, że informacje zapisywane w różnych językach naturalnych zawierają ***redundancję*** jest możliwość zmniejszenia objętości tych informacji bez straty jej istotnych elementów.

Odwracalne zmniejszenie objętości informacji bez utraty jej treści nazywa się **kompresją.**

Formalnie **kompresja** może być rozpatrywania jako zmiana sposobu reprezentacji informacji

- *Kompresja* - przekształcenie sposobu reprezentacji informacji mające na celu zmniejszenie jej objętości w celu zmniejszenia zapotrzebowania na pamięć (wewnętrzną i zewnętrzną) przeznaczoną do jej przechowywania lub/i zmniejszenia kosztów jej przesyłania.
- *Dekompresja* - odtworzenie pierwotnego sposobu reprezentacji informacji w celu jej wygodnego wykorzystywania przez użytkownika.

234

Rodzaje kompresji

- Operacje kompresji i dekompresji najczęściej dotyczą plików, chociaż w telekomunikacji stosuje się też kompresję i dekompresję sygnałów.
- Rodzaje kompresji:
  - *bezstratna* – plik po dekompresji jest dokładnie taki sam jak plik poddany kompresji (kompresja tekstów, programów),
  - *stratna* – plik po dekompresji jest zbliżony do pliku poddanego kompresji (kompresja dźwięku, grafiki, filmów).

235

Kodowanie Huffmana jako przykład algorytmu kompresji **bezstratnej**

- Jest to metoda kompresji bezstratnej (przydatna do kompresji tekstów i programów),
- metoda ta wykorzystuje dostrzeżone różnice w częstościach występowania poszczególnych znaków w tekście,
- wszystkie znaki są kodowane binarnie, jednak krótsze kody przypisywane są znakom częściej się pojawiającym, zaś dłuższe - znakom rzadko występującym,
- metoda ta wykorzystywana jest we wszystkich popularnych programach pakujących (np. pkzip, arj, rar).

236

Przykładowe zastosowanie algorytmu Huffmana (1)

- Załóżmy, że:
  - w tekście występują tylko cztery znaki: A, B, C, D,
  - długość tekstu wynosi 1000 znaków
- *Przed kompresją:*
  - do zakodowania każdego znaku potrzebne są 2 bity:
    A – 00
    B – 01
    C – 10
    D – 11
  - do zakodowania tekstu o długości 1000 znaków potrzebny jest więc obszar pamięci o wielkości 2000 bitów.

237

Przykładowe zastosowanie algorytmu Huffmana (2)

Proces kompresji:

- wyznacza się częstości wystąpienia każdego ze znaków. Załóżmy, że są takie:
  A (45%)
  B (15%)
  C (35%)
  D (5%)

238

Przykładowe zastosowanie algorytmu Huffmana (3)

Proces kompresji:

porządkuje się elementy względem częstości występowania (od najrzadziej do najczęściej występującego)

D (5%)
B (15%)
C (35%)
A (45%)

239

Przykładowe zastosowanie algorytmu Huffmana (4)

Proces kompresji:

w kolejnych krokach pobiera się dwa najrzadziej występujące elementy, łączy w jeden i umieszcza na odpowiedniej pozycji listy

- krok 1 – połączenie elementów D i B
  (D, B) (20%)
  C (35%)
  A (45%)

240

Przykładowe zastosowanie algorytmu Huffmana (5)

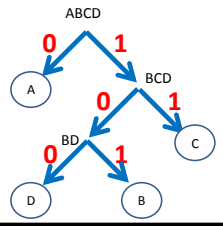
Proces kompresji:

- **krok 2 – połączenie (D, B) i C**
  - A (45%)
  - ((D, B), C) (55%)
- **krok 3 – połączenie A i ((D, B), C)**
  - (A, ((D, B), C)) (100%)

241

Przykładowe zastosowanie algorytmu Huffmana (6)

Uzyskany element (A, ((D, B), C)) przedstawia się w postaci drzewa



Kody znaków

A – 0
B – 101
C – 11
D – 100

242

Przykładowe zastosowanie algorytmu Huffmana (7)

- Długość tekstu w postaci skompresowanej
  - A: 1 * 0.45 * 1000 = 450 bitów
  - B: 3 * 0.15 * 1000 = 450 bitów
  - C: 2 * 0.35 * 1000 = 700 bitów
  - D: 3 * 0.05 * 1000 = 150 bitów
  - SUMA: 1750 bitów

A bez kodowania trzeba było mieć 2000 bitów!

243

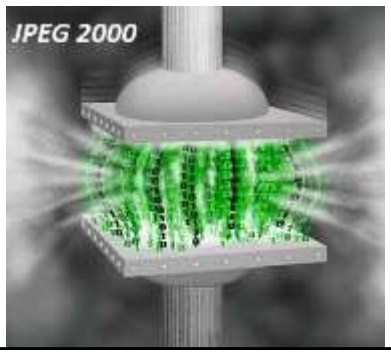
Ogromna redundancja cechuje także różne rejestrowane i przetwarzane komputerowo **sygnały.**

Dotyczy to zwłaszcza obrazów oraz nagrań wideo.

Dlatego poszukuje się wciąż nowych metod kompresji tych sygnałów, żeby łatwiej je było przesyłać albo przechowywać.

Takie powszechnie używane nazwy, jak MP3, JPEG, MPEG są w istocie nazwami algorytmów kompresji odpowiednich sygnałów.

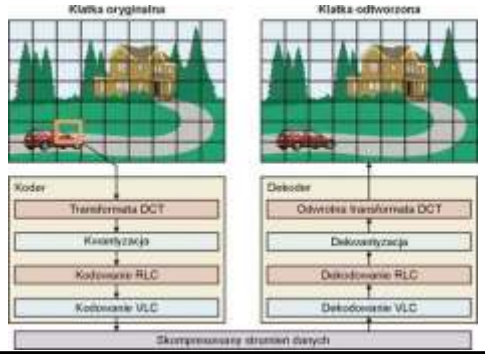
Na przykład *JPEG* jest techniką zmniejszania informacyjnej objętości obrazu



Żeby uzyskać dużą wydajność kompresji sygnałów z reguły stosuje się tu algorytmy stratne.

Wykorzystują one fakt, że percepcja człowieka nie jest doskonała i utrata niektórych subtelnych cech oryginalnego sygnału nie przeszkadza w pełnym wykorzystaniu jego zubożonej po kompresji kopii.

Kompresji poddawane są też obrazy, chociaż metoda kompresji jest wtedy inna



Na obrazie najpierw wykonywana jest tak zwana dyskretna transformacja kosinusowa DCT

$$g(0),\ldots,g(N-1)$$

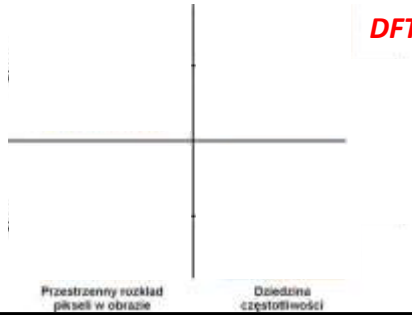
$$G(0),\ldots,G(N-1)$$

$$G(k)=\sqrt{\frac{2}{N}}\sum_{m=0}^{N-1}g(m)\cos\frac{\pi k(2m-1)}{2N}$$

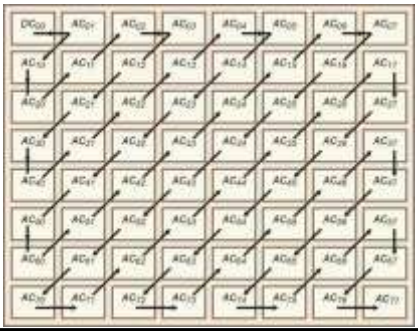
Dla *k = 1, 2, … , (N-1)*

$$G(0)=\frac{1}{\sqrt{N}}\sum_{m=0}^{N-1}g(m)$$

Zmniejszeni objętości informacji polega na tym, że za pomocą kwantyzatora usuwa się „słabe” składniki transformaty. Podczas odtwarzania otrzymuje się obraz nieco zubożony, ale zwykle niezauważalnie.



Skwantowane współczynniki DCT porządkuje się w specjalny „zygzakowaty” sposób



Taki zygzakowaty odczyt DCT powoduje, że w odczytanym kodzie pojawiają się długie łańcuchy samych zer

Objętość zapisu współczynników DCT zmniejsza się teraz metodą kodowania RLC (*Run-Length Coding*).

Metoda ta polega na tym, że zamiast wiele razy powtarzać tę samą wartość symbolu – wpisuję się ją do kodu raz, poprzedzając liczbą określającą, jak wiele powtórzeń danego symbolu jest w danym odcinku kodu.
Na przykład łańcuch symboli:

WWWWWWWWWWWWBWWWWWWWWWWWWBBBWWWWWWWWWWWW
WWWWWWWWWWWWWWBWWWWWWWWWWWWWWW

można zapisać jako:

12W1B12W3B24W1B14W

Dodatkowo do kodowania łańcuchów takich jak 12W1B12W3B24W1B14W można użyć omówionego wcześniej kodowania Huffmana, które w JPEG nazywa się VLC (Variable-Length Coding) – kodowanie ze zmienną długością słowa kodowego.

Jak pamiętamy, w metodzie tej symbole pojawiające się częściej są reprezentowane przez krótsze słowa kodowe, podczas gdy symbole pojawiające się rzadziej są kodowane przy pomocy większej liczby bitów.

Opisana wyżej metoda kompresji obrazów (JPEG) jest przykładem kompresji stratnej

Skompresowany obraz po rozkompresowaniu różni się nieco od obrazu oryginalnego.

Ale różnica ta jest zaniedbywalna.

Przykład obrazu przed i po kompresji

(a) "Woman2", Original (b) "Woman2", 0.25 bpp

Przykład obrazu przed i po kompresji

(c)"Milkdrop", Original

(d)"Milkdrop", 0.25 bpp

Przykład obrazu przed i po kompresji

(e) "Plane", Original

(f) "Plane", 0.25 bpp

Przykład obrazu przed i po kompresji

(g)"Barbara", Original

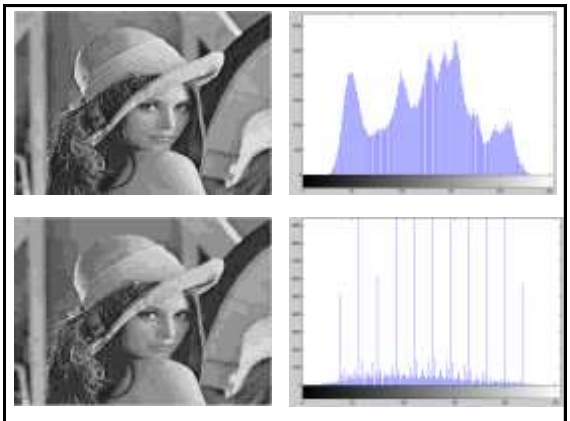
(h)"Barbara", 0.25 bpp

Przykład obrazu przed i po kompresji

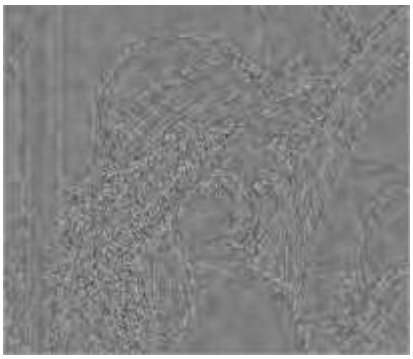
(k)"Goldhill", Original

(l)"Goldhill", 0.25 bpp

Efekty kompresji widać gołym okiem, ale jeszcze bardziej są one widoczne gdy skompresowany uprzednio obraz podda się procesowi analizy

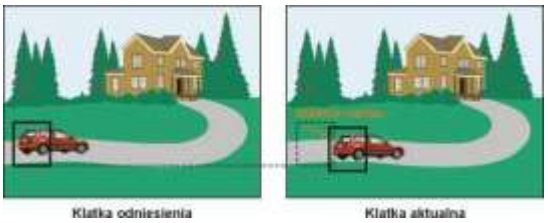


Obraz różnicowy pomiędzy obrazem oryginalnym a obrazem poddanym kompresji i dekompresji



Przy kompresji obrazów ruchomych (sekwencji wideo) dążymy do tego, żeby zakodować tylko jedną klatkę w całości, a na kolejnych klatkach przedstawiających kolejne fazy ruchu – wykryć tylko obiekty ruchome i ustalić dla nich wektor ruchu, wykorzystując obraz tła o ogólny wygląd ruchomego obiektu już zapamiętany z klatki odniesienia.

Kodowanie obiektów ruchomych



Metody kompresji służyły do tego, żeby z danych (lub sygnałów) usunąć redundancję.

Jednak redundancja bywa pożyteczna, gdy pomaga wykrywać i korygować błędy, które pojawiają się we wiadomościach.

Gdy poziom możliwych błędów jest duży – do systemu kodującego dokłada się dodatkowe elementy, żeby zwiększyć redundancję i tym samym zapewnić zwiększoną odporność na zakłócenia i przekłamania.

Najprostszy system redundancyjnego zabezpieczenia wiadomości – bit parzystości

Przykład: Binarne kodowanie liczb dziesiętnych od 0 do 7

Ten system nie ma redundancji, więc każda kombinacja 0 i 1 coś oznacza.

Jeśli więc gdzieś nastąpi przekłamanie i 0 zamieni się na 1 to będzie to nie do wykrycia!

Ma być 1 (001) a na skutek przekłamania jednego bitu jest 5 (101). Jak to wykryć?

0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

Dokładając bit parzystości powodujemy, że w każdym kodzie musi być parzysta liczba jedynek

0	0	0	0	0
1	0	0	1	1
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	0
6	1	1	0	0
7	1	1	1	1

Ma być a na Teraz skutek przekłamania jednego bitu zmieniający 1 (0011) na 5 (1011) będzie **wykryty.**

Teraz **każde** przekłamanie będzie wykryte!

Jednak nadal nie wiadomo, jaką liczbę ukrywa niepoprawny kod (1011)**?**

Dla wykrywania i **korygowania** błędów trzeba zastosować system kodowania mający jeszcze większą redundancję.

Można to zrobić bardzo wytwornie opierając się na teorii kodów Hamminga i metryce bazującej na xor.

Ale ta teoria jest trudna.

No więc zrobimy to inaczej. Jak?

A na chama!

Po prostu każdy kod (zabezpieczony przed niewykrywalnym przekłamaniem) powtórzymy dwa razy.

Teraz zmiana jednego bitu nie będzie w stanie nas wprowadzić w błąd!

1 = 00110011

Przekłamane: 10110011

Trzeba odrzucić tę część, gdzie się parzystość nie zgadza 10110011

0	0	0	0	0	0	0	0	0
1	0	0	1	1	0	0	1	1
2	0	1	0	1	0	1	0	1
3	0	1	1	0	0	1	1	0
4	1	0	0	1	1	0	0	1
5	1	0	1	0	1	0	1	0
6	1	1	0	0	1	1	0	0
7	1	1	1	1	1	1	1	1

Prawidłowy wynik został odzyskany!

Przedstawiane wyżej metody były wymyślane ad hoc.

W praktyce stosuje się metody mające porządną bazę teoretyczną.

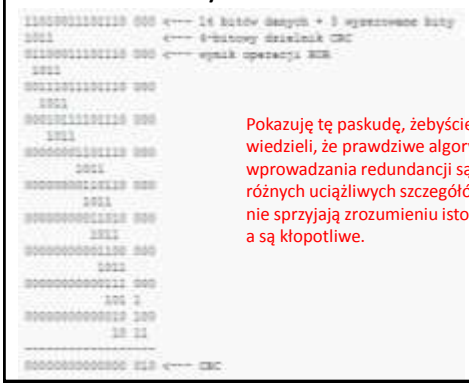
Do **wykrywania** przypadkowych błędów stosuje się najczęściej metodę **CRC**

CRC (*Cyclic Redundancy Check*) to cykliczny kod nadmiarowy wykorzystywany do wykrywania przypadkowych błędów pojawiających się podczas przesyłania i magazynowania danych binarnych.

Algorytm postępowania w celu obliczenia 3-bitowego CRC jest następujący:

- dodajemy do ciągu danych 3 wyzerowane bity,
- w linii poniżej wpisujemy 4-bitowy dzielnik CRC,
- jeżeli mamy 0 nad najstarszą pozycją dzielnika, to przesuwamy dzielnik w prawo o jedną pozycję, aż do napotkania 1,
- wykonujemy operację XOR pomiędzy bitami dzielnika i odpowiednimi bitami ciągu danych, uwzględniając dopisane 3 bity
- wynik zapisujemy w nowej linii poniżej,
- jeżeli liczba bitów danych jest większa lub równa 4, przechodzimy do kroku 2,
- 3 najmłodsze bity stanowią szukane CRC.

Przykład działania



Dlatego inne (bardzo liczne i sprytne!) praktycznie wykorzystywane algorytmy do celowego wprowadzania do ciągów kodowych wymaganej redundancji nie będą tu omawiane, ale mogą być łatwo znalezione w Internecie (hasło: kodowanie nadmiarowe).

Można by jeszcze długo mówić o wnioskach z teorii informacji, my jednak poprzestaniemy na tym, co zostało tu pokazane.